Video Object Detection via Object-Level Temporal Aggregation

Chun-Han Yao[1](✉), Chen Fang[2], Xiaohui Shen[3], Yangyue Wan[3], and Ming-Hsuan Yang[1,4]

[1] UC Merced, Merced, USA
{cyao6,mhyang}@ucmerced.edu
[2] Tencent, Shenzhen, China
fangchen1988@gmail.com
[3] ByteDance Research, Beijing, China
{shenxiaohui,wanyangyue}@bytedance.com
[4] Google Research, Menlo Park, USA

Abstract. While single-image object detectors can be naively applied to videos in a frame-by-frame fashion, the prediction is often temporally inconsistent. Moreover, the computation can be redundant since neighboring frames are inherently similar to each other. In this work we propose to improve video object detection via temporal aggregation. Specifically, a detection model is applied on sparse keyframes to handle new objects, occlusions, and rapid motions. We then use real-time trackers to exploit temporal cues and track the detected objects in the remaining frames, which enhances efficiency and temporal coherence. Object status at the bounding-box level is propagated across frames and updated by our aggregation modules. For keyframe scheduling, we propose adaptive policies using reinforcement learning and simple heuristics. The proposed framework achieves the state-of-the-art performance on the Imagenet VID 2015 dataset while running real-time on CPU. Extensive experiments are done to show the effectiveness of our training strategies and justify the model designs.

Keywords: Video object detection · Object tracking · Temporal aggregation · Keyframe scheduling

1 Introduction

As mobile applications prevail nowadays, increasing attention has been drawn to deploying vision models on mobile devices. With the limited computation resource on mobile or embedded platforms, it is crucial to find an appropriate tradeoff between model accuracy, processing time, and memory usage. Real-time

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58568-6_10) contains supplementary material, which is available to authorized users.

inference on videos, particularly, requires heavy computation within a glimpse of an eye. This paper focuses on object detection for videos, a fundamental task of numerous downstream applications. The performance of single-image detection has been improved considerably by the advance of deep convolutional neural networks (CNNs). However, the best way to exploit temporal information in videos remains unclear. Early methods [8,15] extend single-image detectors to videos by offline linking the per-frame predictions. The flow-guided approaches [35,39,40] calculate motion field to propagate or aggregate feature maps temporally. They are shown to be effective but computationally expensive. Recent methods [20,21,38] propagate temporal information through memory modules. By inserting recurrent units between the convolutional layers, the network can extract features based on its memory of previous frames. While they achieve the state-of-the-art performance on mobile devices, we argue that aggregation at a higher level is more efficient and generic than feature-level aggregation. In particular, we propose to aggregate information at the object/bounding-box level, *i.e.*, the output of the detection models. First, the dimensionality of object status is reduced compared to CNN feature maps. Second, it disentangles the binding between temporal aggregation and feature extraction. When adapting to a new feature extractor or distinct temporal dynamics, memory-guided feature aggregation requires model re-training as a whole. On the contrary, box predictions can be easily aggregated regardless of the semantic meaning of feature maps.

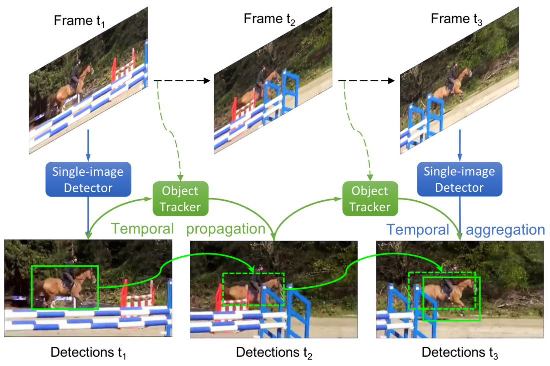


Fig. 1. Illustration of the proposed framework. Given a video sequence, we sparsely apply the detection model on certain keyframes. The object information is temporally propagated by the trackers and aggregated when the next detection is triggered.

To exploit temporal cues for object-level aggregation, tracking models come as an intuitive choice. The Detect and Track (D&T) approach [5] performs object detection and tracking with a jointly trained network, and yet its heavy computation is not applicable in real-time scenarios. For acceleration, we propose temporal aggregation modules to integrate detection and tracking information at the object level. As illustrated in Fig. 1, our framework applies detection on sparse keyframes and propagates the object status by real-time trackers. Since detection is applied sparsely, the selection of keyframes becomes crucial. We investigate the pros and cons of detection and tracking models, then propose adaptive keyframe scheduling via reinforcement learning (RL) and simple heuristics. Our experimental results demonstrate that the keyframe policies can generalize to various detection rates and produce a significant performance gain compared to fixed intervals. With the proposed aggregation modules and keyframe schedulers, we show the possibility to achieve competitive speed-accuracy tradeoffs with object-level temporal aggregation.

The contributions of this paper are as follows:

- We propose temporal aggregation modules to integrate object detection and tracking models at the object/bounding-box level.
- We present adaptive keyframe schedulers using simple heuristics and RL training with diverse video sequences.
- We conduct extensive experiments to compare our framework with the state-of-the-art methods in terms of CPU inference speed and accuracy on the Imagenet VID 2015 dataset [32].

2 Related Work

The majority of existing methods are built upon single-image object detectors. Our framework further incorporates object tracking models. Therefore, we briefly introduce the state-of-the-art object detectors and trackers, then categorize the video approaches into three groups: track-based, flow-guided, and memory-guided aggregation.

Single-Image Object Detection. One group of detection models proposes region candidates that possibly contain an object, then refines and classifies them in a second stage. They are referred to as two-stage or region-based detectors, including R-CNN [7] and its descendants [6,9,18,31]. Single-shot methods such as YOLO [29,30] and SSD [22] are proposed to improve efficiency based on a set of pre-defined anchor boxes. Bottom-up approaches [4,17,36,37] further explore the possibility of detection without anchor boxes. Several methods like Mobilenet [12,33] focus on network optimization. They are commonly combined with other detection models for acceleration on mobile devices. Since most two-stage detectors hardly run in real-time, we adopt Darknet53 [28] + YOLOv3-SPP [30] and Resnet18 [10] + CenterNet [36] as our single-image baselines, which represent the single-shot and bottom-up models, respectively.

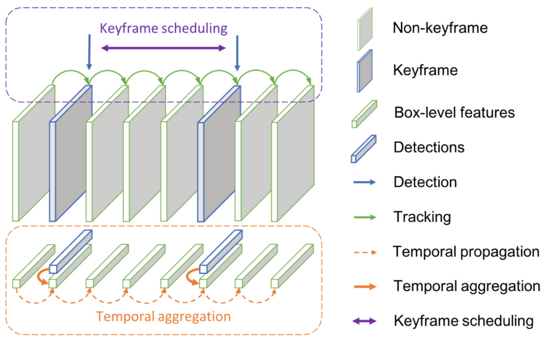
Object Tracking. Object tracking is a common vision task which aims to trace objects throughout a video sequence. Given the initial bounding box as an object template, a tracking model can estimate the current location by correlation or CNN filtering. Most existing methods assume temporal smoothness of object appearance between frames. However, deformations, occlusions, and large motions can all pose a threat to the assumption, thus making it difficult to perform association and update object status. Fortunately, our scenario does not include long-term tracking and multi-object association. The association problem is also naturally avoided since we concern about the object class and not their identity. Considering the speed requirement, we choose the Kernelized Correlation Filter (KCF) [11] and Fully-convolutional Siamese Network (SiamFC) [1] as the trackers in our experiments.

Track-Based Aggregation. As an intuitive extension from single-images to videos, one can associate the per-frame detected boxes and refine the predictions accordingly. The Seq-NMS method [8] links the bounding boxes of an object across frames, then reassigns the maximal or average confidence score to the boxes along the track. The TCN approach [15] associates the boxes via optical flow and tracking algorithms before confidence reassignment. They can both provide a 3–5% mAP gain to most single-frame baselines, but are considered offline post-processing methods. The D&T framework [5] proposes to learn detection and tracking simultaneously. Benefiting from multi-task training, it produces a considerable accuracy gain on both tasks, whereas the expensive computation limits its application on mobile devices.

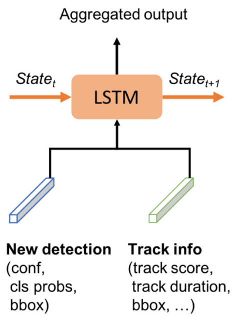
Flow-Guided Aggregation. Another group of approaches aggregates temporal information at the feature-map level via optical flow. The DFF method [40] applies expensive feature extraction on certain keyframes, then propagates the feature maps to the remaining frames. Zhu *et al.* [39] warp and average the feature maps of neighboring frames with flow-guided weighting. Wang *et al.* [35] leverage optical flow for pixel-level and instance-level calibration to boost the detection accuracy. To enable real-time inference on mobile devices, Zhu *et al.* [38] further speed up feature extraction and propagation by the Gated Recurrent Units. In spite of the acceleration efforts, most flow-guided approaches are substantially slower than other mobile-focused models. Moreover, the performance depends on the flow estimator which requires densely labeled data for training.

Memory-Guided Aggregation. Recent improvement of mobile-focused models relies on memory modules. Liu *et al.* [20] insert LSTM units between convolutional layers (ConvLSTM) to propagate temporal cues and refine feature maps. A following work by Liu *et al.* [21] proposes to balance between a small and a large feature extractor. It interleaves the extracted features by the ConvLSTM layers before the detection head. The feature extractors are trained along with the memory modules for temporal aggregation. Despite the merit of joint optimization, the binding demands intensive training when adapting to unseen domains with distinct temporal properties. We adopt a similar memory module for temporal aggregation in our framework, except that the information is aggregated at the object level. With a comparable accuracy, our method is shown to be more efficient and model-agnostic.

Adaptive Key-Frame Scheduling. Most existing methods use a fixed keyframe interval to balance between heavy and lightweight computations. Several adaptive policies are proposed for object tracking [13,34] and video segmentation [24]. For object detection, Chen *et al.* [2] select keyframes by offline measuring the difficulty of temporal propagation. Luo *et al.* [23] train a Siamese



(a) Framework overview. Our aggregation module updates the object-level features and the adaptive scheduler adjusts the keyframe interval online.



(b) Memory module for temporal aggregation.

Fig. 2. The proposed framework and aggregation module (best viewed in color).

network to switch between detection and tracking models based on image-level features. Liu *et al.* [21] propose an RL policy for model interleaving, which results in marginal performance gain. Mao *et al.* [25] and Lao *et al.* [16] focus on minimizing the delay in detecting new objects. In this work we emphasize the importance of keyframe selection and propose two adaptive schedulers: a heuristic policy and a lightweight RL model.

3 Proposed Approach

Ideally, a detection model is able to capture the presence or absence of objects. The prediction of each frame is generated independently, and thus is robust from rapid motions or scene changes. Nonetheless, we often observe temporal inconsistency in the detection results, especially when an object is slightly deformed or partially occluded. The tracking models, on the other hand, excel at exploiting temporal cues to trace an object given its initial appearance. With a more lightweight computation, they can provide accurate and stable estimations of the object location. To leverage the advantages of detection and tracking models, we propose a framework which integrates them at the object level. Specifically, detection is applied on keyframes to capture new objects and initialize trackers. The trackers update object location and propagate the box features across frames, which facilitates object association and accelerates detection. Our temporal aggregation helps stabilize the confidence scores and class probabilities by utilizing past predictions. To ensure that detection is triggered when tracking fails, we propose adaptive schedulers to determine the keyframe interval based on object status. An overview of our framework is shown in Fig. 2a.

3.1 Temporal Aggregation

A typical detection model produces a confidence/objectness score c, class probabilities $\boldsymbol{p}$, and box coordinates $\boldsymbol{b}$ for each object. Object tracking models can provide an estimate of the box coordinates $\tilde{\boldsymbol{b}}$ and a tracking score s indicating how well the current box matches the object template or how confident the estimation is. The information combined with tracking duration d are exploited as features. The selection of object-level features makes our aggregation more model-agnostic, as it applies to any object detector and tracker that yield the above output. For each feature x in $(c, \boldsymbol{p}, \boldsymbol{b}, s)$, we denote the value propagated by the trackers as $\tilde{x}$, and the aggregated output as x'.

At each keyframe, we initialize a tracker for each new object and terminate the old ones for occluded or disappeared objects. The tracked objects are associated with the current detection by thresholding the intersection-over-union (IOU) ratio. We adopt the convention in the existing methods [2,8] and set the IOU threshold as 0.5 in our experiments. Once associated, the features of the existing objects are aggregated with the new detection. As shown in Fig. 2b, we employ a memory module that takes the tracked information and new detection as inputs, updates the internal state, and produces an aggregated output. Our memory module is similar to the speed-optimized Bottleneck-LSTM proposed by Liu *et al.* [21], except that we replace convolution operations by two fully-connected layers since our inputs are object-level features instead of convolutional feature maps. We show the detail LSTM architecture in the supplementary material. Unlike the prior memory-guided methods, our aggregation module has a low-dimensional input and is only applied at sparse keyframes, thus it entails a minimal inference time.

In addition to the learning-based aggregation, we propose a simple yet effective heuristic module to mitigate the inconsistency of three predictions: 1) Box coordinates, 2) class probabilities, and 3) Confidence score. First, to produce accurate and temporally smooth box coordinates, we refine the detected boxes $\boldsymbol{b}_t$ by the tracked boxes $\tilde{\boldsymbol{b}}_t$. The coordinates are averaged based on the detection confidence and tracking score. We weight higher on the detected boxes when the new detection is more confident than tracking, and vice versa. The refinement can be expressed as:

$$\boldsymbol{b}'_t = \frac{\tilde{s}_t \tilde{c}_t \tilde{\boldsymbol{b}}_t + c_t \boldsymbol{b}_t}{\tilde{s}_t \tilde{c}_t + c_t}, \tag{1}$$

where $\tilde{c}_t$ and $\tilde{s}_t$ denote the confidence and minimal tracking score propagated from the previous keyframe to time t. Second, we aggregate the class probabilities of the previous keyframes to prevent inconsistent class predictions. The probabilities are re-weighted by the confidence scores and aggregated by cumulative average:

$$\boldsymbol{p}'_t = \frac{c_t \boldsymbol{p_t} + \gamma \sum_{i<t, i \in K} c_i \boldsymbol{p_i}}{c_t + \gamma \sum_{i<t, i \in K} c_i}, \tag{2}$$

where K is the keyframe set, and γ is the weight decay for previous predictions. Finally, the confidence score is stabilized by track-based reassignment. We keep track of the confidence scores in the keyframes and update the current confidence by the temporal maximal:

$$c'_t = \max(\tilde{c}_t, c_t). \tag{3}$$

It aims to pick up the low-confidence detection between high-confidence frames. The final detection result for frame t is given by $(\boldsymbol{b}'_t, \boldsymbol{p}'_t, c'_t)$.

3.2 Adaptive Keyframe Scheduling

Fixing the keyframe interval is a naive way to switch between lightweight and heavy and computations. However, it ignores temporal variations and fails to adapt to certain events where one model is preferred over another. If a video sequence is rather smooth, we prefer tracking over detection since it costs minimal computation. On the contrary, frequent detection is required when the object dynamics are beyond the capability of trackers, *e.g.*, large motions, deformations, and occlusions. More importantly, the selection of keyframes determines the object templates for tracker initialization, which affects tracking performance significantly. In our experiments, we observe that selecting the first frame where an object just appeared does not lead to optimal tracking results. Instead, a few frames after the first appearance typically contain more faithful and complete templates to initialize object trackers. We also find that tracking score as well as other object-level features are useful and convenient indications of the tracking and detection quality. Adding more image-level features only provides marginal information for keyframe selection.

Inspired by the above observations, we propose an adaptive policy to adjust the keyframe interval D online. The task can be formulated as an RL problem with the following state space, action space, and reward function. The state vector S is constructed from the same object-level features as leveraged in temporal aggregation. For each feature x, we encode the temporal minimal (x_{min}), maximal (x_{max}), average (x_{mean}), variance (x_{var}), and difference ($x_t - x_{t-1}$) in S to facilitate RL training. Given the state of current frame, the agent learns to predict an action a from the action space:

$$a_t = \begin{cases} 0 \text{ (trigger detection)}: & D_t = 1 \\ 1 \text{ (shorten interval)}: & D_t = \max(1, \alpha D_{t-1}) \\ 2 \text{ (fix interval)}: & D_t = D_{t-1} \end{cases}, \tag{4}$$

where α is a constant multiplier. The detection model is applied if and only if the tracking duration $d_t \geq D_t$. For the instant reward R, we calculate the average IOU ratio between the predicted boxes $\boldsymbol{b'}$ and ground truth boxes $\hat{\boldsymbol{b}}$ as follows:

$$R_t = \text{IOU}(\boldsymbol{b_t}', \hat{\boldsymbol{b_t}}). \tag{5}$$

We adopt the standard training scheme of the Asynchronous Advantage Actor-Critic (A3C) [26] model, which is shown to be effective in both continuous and discrete domains. The readers are encouraged to refer to the original paper since we omit the model details here. Considering that there is no ordering of the objects, we use PointNet [27] with 3 fully-connected layers for both the policy and value networks. The model is originally designed for point cloud data to preserve order invariance. In our work we consider each object as a point data and set a maximum number of objects according to the dataset distribution. Exceeding objects are pruned by their confidence scores.

Even though the state space and action space are low-dimensional, the diversity of video sequences makes it a challenging environment for RL training. The same action in the same state may lead to distinct rewards and next states in different videos, *i.e.*, the reward and state transition are stochastic from the perspective of the RL agent. Combining the time and accuracy constraints in the reward function is yet another critical issue. Liu *et al.* [21] add a speed reward to the accuracy reward with a specific weighting for each speed-accuracy tradeoff point. We observe that the simple weighted sum is likely to cause unstable training results. A slight difference in speed reward can easily lead to a local optimal, where the policy applies detection either most excessively or least frequently. Furthermore, the policy needs to be trained with a new reward setting whenever a new tradeoff point is desired.

We propose the following strategies to stabilize the training process and produce a more general policy. First, we mimic the concept of imitation learning by pre-training the policy network with expert supervision. Considering that exhaustive search for optimal keyframes is not feasible, a greedy algorithm is proposed to approximate an oracle policy for supervision. As described in Algorithm 1, we start from a keyframe set with fixed intervals, then iteratively replace a keyframe with the highest tracking score by a non-keyframe with the lowest tracking score. The pre-training steps using the oracle keyframes are described in the supplemental material. To train a general policy for various speed-accuracy tradeoffs, a base interval D_{base} to deviate from is randomly initialized within a range of 5 to 30 frames. The ratio between the tracking duration and base interval, d/D_{base}, is encoded in the state vector, so the agent is aware of the relative detection rate compared to the base frequency. To compensate for the varying reward distribution between diverse videos, we calculate the temporal difference of IOU scores instead of the raw values. It allows the agent to focus on policy optimization according to temporal dynamics. Finally, we add a long-term penalty at the end of an episode to constrain the policy from excessive or passive detection. The modified reward function ${R_t}'$ can be expressed as:

$$ {R_t}' = \begin{cases} R_t - R_{t-1} + \lambda\left(\frac{1}{D_{base}} - \text{mean}(\eta)\right) & \text{if } t = \text{ end of episode} \\ R_t - R_{t-1} & \text{otherwise} \end{cases}, \tag{6} $$

where η denotes the detection history and λ is weight of long-term penalty. We detail the training process in Algorithm 2.

To compare with the RL policy, we propose a heuristic scheduler by mapping the tracking score to the keyframe interval:

$$ D_t = \min(D_{t-1}, \mu\tilde{s}_t D_{base}), \tag{7} $$

Algorithm 1. Greedy Keyframe Scheduling

Inputs: video length L, base keyframe interval D_{base}, number of iterations N_{iter}
Output: K - keyframe set of size $\lfloor L/D_{base} \rfloor$

```
1: $K \leftarrow \{k | mod(k, D_{base}) = 0, 0 \leq k < L\}$
2: $R_{max} \leftarrow$ Run the framework with $K$ and compute mAP
3: for $i = 0 : N_{iter}$ do
4:     $k_{max} \leftarrow$ A keyframe with maximal tracking score
5:     $k_{min} \leftarrow$ A non-keyframe with minimal tracking score
6:     $R \leftarrow$ Run the framework with $K \not\{k_{max}\} \cup \{k_{min}\}$ and compute mAP
7:     if $R > R_{max}$ then
8:         $K \leftarrow K \not\{k_{max}\} \cup \{k_{min}\}$
9:         $R_{max} \leftarrow R$
10: return $K$
```

Algorithm 2. RL Training for Keyframe Scheduling

Definitions: timestep t, tracking duration d, keyframe interval D, episode length L, state S, action a, reward R, detection history η
Output: keyframe policy π

```
1: Pre-train $\pi$ with the oracle supervision
2: repeat
3:     Randomly initialize base detection interval $D_{base}$
4:     Sample video frames $\{I_0, I_1, ..., I_L\}$
5:     $t \leftarrow 0$, $d_0 \leftarrow 0$, $D_0 \leftarrow D_{base}$
6:     $S_0 \leftarrow$ Run detection on $I_0$ and extract object-level features
7:     $S_1 \leftarrow$ Track objects from $I_0$ to $I_1$ and propagate $S_0$
8:     for $t = 1 : L$ do
9:         $a_t \leftarrow$ arg max $\pi(S_t)$
10:        $D_t \leftarrow$ Update interval based on $a_t$ and $D_{t-1}$
11:        $d_t \leftarrow d_{t-1} + 1$
12:        if $d_t \geq D_t$ then
13:            Run detection on $I_t$ and update $S_t$
14:            Add 1 to $\eta$
15:            $D_t \leftarrow D_{base}$
16:            $d_t \leftarrow 0$
17:        else
18:            Add 0 to $\eta$
19:        $R_t \leftarrow$ Compute reward based on $S_t$ and $\eta$
20:        $S_{t+1} \leftarrow$ Track objects from $I_t$ to $I_{t+1}$ and propagate $S_t$
21:        Add $(S_t, a_t, R_t, S_{t+1})$ to reply buffer
22:    Sample batch $B$ from reply buffer
23:    Update $\pi$ with $B$ using A3C training
24: until convergence
```

where μ is a weighting parameter, $\tilde{s}_t$ is the temporally minimal tracking score since the last detection. It follows our intuition to decrease/increase the detection frequency with tracking quality. At inference time, one can easily search for the best speed-accuracy tradeoff for a specific application by setting D_{base}.

4 Performance Evaluation

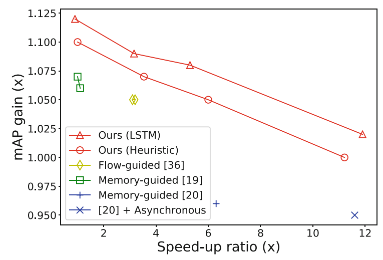
4.1 Experiment Settings

We evaluate model performance on the Imagenet 2015 VID dataset [32] with 30 object classes. Following the work by Liu *et al.* [21], the detection models are trained on the Imagenet VID, Imagenet DET [32], and COCO [19] training sets. We sample 1 per 30 frames from the VID videos and use only the relevant classes from the DET and COCO datasets, which amount to 37K frames from VID, 147K images from DET, and 43K images from COCO. For object tracking, KCF does not require any pre-training, and we directly employ the SiamFC model trained on the GOT-10k dataset [14]. The temporal aggregation module and RL policy are trained by randomly sampling video sequences of 30 frames. We set a maximum of 10 objects per frame for PointNet since most VID videos contain less than 10 objects. The sequences where no object is detected are ignored to prevent unnecessary training. For evaluation, we calculate the mAP@0.5IOU on the VID validation set. All training and evaluation are done with an input resolution of 320×320. Based on our ablation studies, we set $\gamma = 1$ for heuristic aggregation and $\alpha = 0.5, \lambda = 1, \mu = 1$ for the keyframe schedulers.

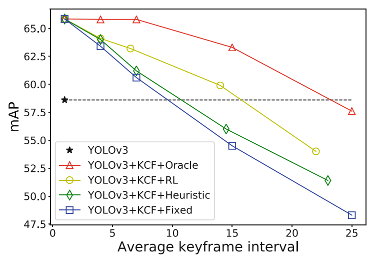
4.2 Quantitative Comparisons

We compare our framework with the single-image baselines and previous methods in Table 1. While there are numerous related works in video processing, action recognition, frame interpolation, etc., we focus on comparing methods with similar settings: online, real-time, mobile-focused, and based on single-image detection. Existing methods evaluate model performance in terms of the raw mAP and inference time. However, we find the metrics not directly comparable since they are built upon different single-image baselines and tested on different platforms. With a better baseline, one can reach higher accuracy regardless of the temporal design for videos. Considering our focus to improve from single-image detection, we report the speed-up ratio and mAP gain (both absolute and relative values) compared to each baseline. Despite the difference in baseline models, the performance of our single-image detectors lie between the baselines of the other approaches [20, 38], which makes the comparison more convincing. Combining YOLOv3-SPP and KCF at a detection interval of $D_{base} = 7$, we achieve 63.2% mAP at 31.4 fps. At $D_{base} = 15$, our framework maintains a reasonably high accuracy while having a 11.9$\times$ speed-up ratio. Note that the flow-guided approach [38] does not train the detectors on the COCO dataset, but its optical flow model requires pre-training on the Flying Chairs dataset [3].

Among all the real-time models, the memory-guided approach [21] achieves competitive accuracy based on its well-trained baseline (f_0). It can further produce a significant speed-up ratio by asynchronous inference and model quantization. Our method reaches comparable speed without any network optimization. In Fig. 3a we plot the speed-accuracy tradeoff curves, which show that the proposed framework performs favorably against the state-of-the-art methods.



(a) Relative mAP gain versus speed-up ratio of different aggregation methods. Our approach adopts YOLOv3-SPP + KCF and RL keyframe scheduler.



(b) Speed-accuracy tradeoff curves of different keyframe policies. The data points are obtained by adjusting D_{base} and averaging the detection interval.

Fig. 3. Visualization of the quantitative comparisons.

4.3 Qualitative Comparisons

In Fig. 4 we show the qualitative results on the VID validation set. The first two videos contain an object with mild motions and occlusions, causing temporal inconsistency in the per-frame detections. The proposed aggregation effectively picks up the missed detections and corrects the erroneous classification. In the third video, the rapid motions and deformations make object tracking a tougher task. Our adaptive keyframe scheduler overcomes such difficulty by triggering more frequent detection. The importance of keyframe selection can also be observed in video 2 and 3. With a better keyframe scheduler, we can initialize the object trackers with a better object template, which results in more accurate box estimations.

4.4 Speed-Accuracy Tradeoffs

To vary how frequent the expensive feature extraction is performed, Liu *et al.* [21] train an RL policy with a specific reward weighting. On the contrary, our keyframe schedulers are generic to various detection rates as they encode the base interval D_{base} in the state vector. The average detection interval can be adjusted online simply by setting D_{base}. The mAP curves of different keyframe policies are shown in Fig. 3b. Using a fixed detection interval, the mAP drops

Table 1. Performance comparison on the Imagenet VID validation set. α is the feature extractor width multiplier described in [12], and β is the flow network width multiplier. The models of [38], [20,21], and ours are evaluated on HuaWei Mate 8 phone (*), Pixel 3 phone (†), and Intel Xeon E5 CPU, respectively. We mark the models with real-time inference speed (above 30fps) in red. All of our models adopt KCF trackers and RL keyframe scheduler.

Approach	Model	mAP	Speed (fps)	mAP gain	Speed-up ratio
Flow-guided [38]	($\alpha = 0.5$)	48.6	16.4*	-	-
	($\alpha = 0.5$) + Flow ($\beta = 0.5$)	51.2	52.6*	+2.6/×1.05	×3.2
	($\alpha = 1.0$)	58.3	4.0*	-	-
	($\alpha = 1.0$) + Flow ($\beta = 1.0$)	61.2	12.5*	+2.9/×1.05	×3.1
Memory-guided [20]	($\alpha = 0.35$)	42.0	14.4†	-	-
	($\alpha = 0.35$) + LSTM	45.1	14.6†	+3.1/×1.07	×1.0
	($\alpha = 1.4$)	60.5	3.7†	-	-
	($\alpha = 1.4$) + LSTM	**64.1**	4.1†	+3.6/×1.06	×1.1
Memory-guided [21]	Non-interleaved (f_0)	63.9	4.2†	-	-
	Interleaved + Adaptive	61.4	26.6†	−2.5/×0.96	×6.3
	Interleaved + Adaptive + Async	60.7	48.8†	−3.2/ ×0.95	×11.6
Ours	CenterNet	52.0	12.1	-	-
	CenterNet + Heuristic ($D_{base} = 7$)	54.7	57.8	+2.7/×1.05	×4.8
	CenterNet + LSTM ($D_{base} = 7$)	56.4	52.3	+4.4/×**1.08**	×4.3
	CenterNet + LSTM ($D_{base} = 15$)	53.3	**95.2**	+1.3/×1.03	×7.8
	YOLOv3	58.6	6.0	-	-
	YOLOv3 + Heuristic ($D_{base} = 7$)	61.3	35.8	+2.7/×1.05	×6.0
	YOLOv3 + LSTM ($D_{base} = 7$)	63.2	31.4	**+4.6/×1.08**	×5.3
	YOLOv3 + LSTM ($D_{base} = 15$)	59.9	71.5	+1.3/×1.02	×**11.9**

linearly as the interval increases. The proposed heuristic scheduler mitigates the accuracy drop by a clear margin. With the trained RL policy, we manage to extend the keyframe interval to 15 frames and still match the performance of the single-image baseline. We also plot the oracle curve obtained by running the greedy algorithm (Algorithm 1) with $N_{iter} = 20$. It demonstrates how the performance gap from the oracle is narrowed down by our RL policy.

4.5 Ablation Studies

As discussed in the previous sections, our object-level aggregation disentangles the binding between feature maps and temporal aggregation. We demonstrate the advantage by reporting the accuracy without finetuning the detection models on the VID dataset in Table 2. Unlike feature-level aggregation, our framework provides a consistent mAP gain by simply training the aggregation module on the VID dataset and not the entire feature extractor. In Table 3a we evaluate the effectiveness of LSTM-based aggregation module and the three aggregation heuristics: box coordinate refinement, class probability re-weighting, and confidence reassignment. The confidence reassignment effectively stabilizes the detection predictions. It alone contributes to 3.5% mAP gain. The aggregation of class probabilities ($\gamma = 1.0$) and box coordinates is able to produce an additional gain of 2.7%. To validate that our framework applies to arbitrary detection and tracking models, we experiment with multiple combinations of the sub-modules. The results in Table 3b show a consistent improvement by our method with either of the detection models. With a deep object tracker (SiamFC), our framework achieves higher mAP and still runs in real-time. The RL policy further produces a considerable mAP gain while entailing minimal inference time.

Table 2. Validation with/without training the detectors on the video dataset. The models are run with KCF trackers and LSTM-based aggregation at $D_{base} = 1$. The mAP scores are calculated only on the 14 relevant classes in COCO dataset.

Model	Training data	mAP	mAP gain
CenterNet	COCO	53.1	+3.7/×1.08
	VID+DET+COCO	56.8	+4.3/×1.09
YOLOv3-SPP	COCO	62.6	+6.4/×1.12
	VID+DET+COCO	66.6	+6.6/×1.11

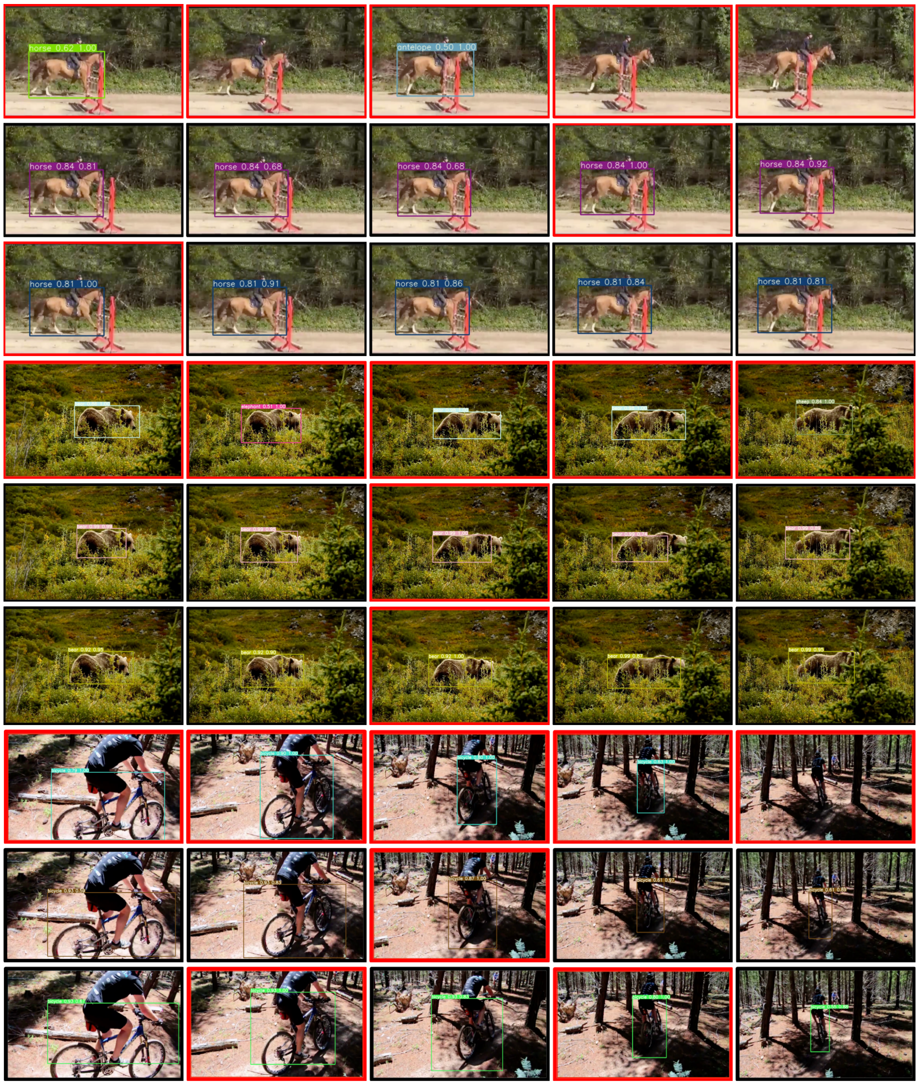


Fig. 4. Qualitative results on the Imagenet VID validation set (best viewed in color). For each video sequence, the results of per-frame detection are presented in the first row, our method (LSTM-based aggregation) with fixed keyframe interval in the second row, and adaptive scheduling (RL) in the third row. The frames where detection is applied are marked in red. Each detected object is labeled by a colored bounding box with the class name, detection confidence, and tracking score on the top. Missed detections can be observed in videos 1 and 3, and the objects are occasionally misclassified in videos 1 and 2. Our temporal aggregation effectively picks up all the missed detections and corrects the false classification. In video 3, the adaptive keyframe scheduler triggers detection more intelligently, which leads to more accurate box predictions.

Table 3. Ablation studies on temporal aggregation and keyframe scheduling.

(a) Evaluation of our heuristic and LSTM-based temporal aggregation modules. The models are run with YOLOv3-SPP + KCF at $D_{base} = 1$.

Confidence	Cls prob	Bbox	mAP
	-		58.6
✓			62.1
✓	$\gamma = 0.6$		63.8
✓	$\gamma = 0.8$		64.1
✓	$\gamma = 1.0$		64.4
✓	$\gamma = 1.0$	✓	**64.8**
	LSTM-based		**65.8**

(b) Evaluation on different module combinations. All are run with LSTM-based aggregation at $D_{base} = 7$.

Detector	Tracker	Keyframe scheduler	Speed (fps)	mAP
CenterNet	KCF	Fixed	**58.8**	53.2
		Heuristic	57.8	53.6
		RL	52.3	56.4
CenterNet	SiamFC	Fixed	45.9	55.3
		Heuristic	45.8	55.5
		RL	42.9	57.7
YOLOv3-SPP	KCF	Fixed	37.1	60.6
		Heuristic	35.8	61.2
		RL	31.4	63.2
YOLOv3-SPP	SiamFC	Fixed	23.8	62.0
		Heuristic	23.8	62.3
		RL	20.2	**63.7**

5 Concluding Remarks

We propose a framework to improve detection by tracking the objects across video frames. Temporal information is exploited and aggregated by real-time trackers, which makes the detection more consistent and efficient. Heuristic and RL policies are proposed for adaptive keyframe scheduling. As shown in the experimental results, our method achieves competitive speed-accuracy tradeoffs on the Imagenet VID 2015 dataset. Furthermore, the object-level aggregation alleviates the dependence on feature maps, making it more generic to arbitrary detection and tracking models. Currently, we initiate a tracker for each object, which linearly increases the inference time when multiple objects are being tracked. As a future work, we can accelerate our model by multi-object tracking and running the detection and tracking models in parallel.

Acknowledgement. This work is supported in part by the NSF CAREER Grant #1149783.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: Fully-convolutional siamese networks for object tracking. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9914, pp. 850–865. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48881-3_56

2. Chen, K., et al.: Optimizing video object detection via a scale-time lattice. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7814–7823 (2018)
3. Dosovitskiy, A., et al.: Flownet: learning optical flow with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2758–2766 (2015)
4. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: Centernet: keypoint triplets for object detection. arXiv preprint arXiv:1904.08189 (2019)
5. Feichtenhofer, C., Pinz, A., Zisserman, A.: Detect to track and track to detect. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3038–3046 (2017)
6. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
7. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
8. Han, W., et al.: Seq-NMS for video object detection. arXiv preprint arXiv:1602.08465 (2016)
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
11. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2014)
12. Howard, A.G., et al.: Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
13. Huang, C., Lucey, S., Ramanan, D.: Learning policies for adaptive tracking with deep feature cascades. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 105–114 (2017)
14. Huang, L., Zhao, X., Huang, K.: Got-10k: a large high-diversity benchmark for generic object tracking in the wild. arXiv preprint arXiv:1810.11981 (2018)
15. Kang, K., Ouyang, W., Li, H., Wang, X.: Object detection from video tubelets with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 817–825 (2016)
16. Lao, D., Sundaramoorthi, G.: Minimum delay object detection from video. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5097–5106 (2019)
17. Law, H., Deng, J.: CornerNet: detecting objects as paired keypoints. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11218, pp. 765–781. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_45
18. Li, Z., Peng, C., Yu, G., Zhang, X., Deng, Y., Sun, J.: Light-head R-CNN: in defense of two-stage object detector. arXiv preprint arXiv:1711.07264 (2017)
19. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
20. Liu, M., Zhu, M.: Mobile video object detection with temporally-aware feature maps. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5686–5695 (2018)

21. Liu, M., Zhu, M., White, M., Li, Y., Kalenichenko, D.: Looking fast and slow: memory-guided mobile video object detection. arXiv preprint arXiv:1903.10172 (2019)
22. Liu, W., et al.: SSD: single shot MultiBox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
23. Luo, H., Xie, W., Wang, X., Zeng, W.: Detect or track: towards cost-effective video object detection/tracking. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8803–8810 (2019)
24. Mahasseni, B., Todorovic, S., Fern, A.: Budget-aware deep semantic video segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1029–1038 (2017)
25. Mao, H., Yang, X., Dally, W.J.: A delay metric for video object detection: what average precision fails to tell. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 573–582 (2019)
26. Mnih, V., et al.: Asynchronous methods for deep reinforcement learning. In: Proceedings of the International Conference on Machine Learning, pp. 1928–1937 (2016)
27. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 652–660 (2017)
28. Redmon, J.: Darknet: open source neural networks in C (2013–2016). http://pjreddie.com/darknet/
29. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
30. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
31. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
32. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
33. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv 2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
34. Supancic III, J., Ramanan, D.: Tracking as online decision-making: learning a policy from streaming videos with reinforcement learning. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 322–331 (2017)
35. Wang, S., Zhou, Y., Yan, J., Deng, Z.: Fully motion-aware network for video object detection. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11217, pp. 557–573. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01261-8_33
36. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. In: arXiv preprint arXiv:1904.07850 (2019)
37. Zhou, X., Zhuo, J., Krahenbuhl, P.: Bottom-up object detection by grouping extreme and center points. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 850–859 (2019)
38. Zhu, X., Dai, J., Zhu, X., Wei, Y., Yuan, L.: Towards high performance video object detection for mobiles. arXiv preprint arXiv:1804.05830 (2018)

39. Zhu, X., Wang, Y., Dai, J., Yuan, L., Wei, Y.: Flow-guided feature aggregation for video object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 408–417 (2017)
40. Zhu, X., Xiong, Y., Dai, J., Yuan, L., Wei, Y.: Deep feature flow for video recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2349–2358 (2017)