The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)

YOLOV: Making Still Image Object Detectors Great at Video Object Detection

Yuheng Shi[1], **Naiyan Wang**[2], **Xiaojie Guo**[1*]

[1] Tianjin University
[2] TuSimple
yuheng@tju.edu.cn, winsty@gmail.com, xj.max.guo@gmail.com

Abstract

Video object detection (VID) is challenging because of the high variation of object appearance as well as the diverse deterioration in some frames. On the positive side, the detection in a certain frame of a video, compared with that in a still image, can draw support from other frames. Hence, how to aggregate features across different frames is pivotal to VID problem. Most of existing aggregation algorithms are customized for two-stage detectors. However, these detectors are usually computationally expensive due to their two-stage nature. This work proposes a simple yet effective strategy to address the above concerns, which costs marginal overheads with significant gains in accuracy. Concretely, different from traditional two-stage pipeline, we select important regions after the one-stage detection to avoid processing massive low-quality candidates. Besides, we evaluate the relationship between a target frame and reference frames to guide the aggregation. We conduct extensive experiments and ablation studies to verify the efficacy of our design, and reveal its superiority over other state-of-the-art VID approaches in both effectiveness and efficiency. Our YOLOX-based model can achieve promising performance (*e.g.*, 87.5% AP50 at over 30 FPS on the ImageNet VID dataset on a single 2080Ti GPU), making it attractive for large-scale or real-time applications. The implementation is simple, we have made the demo codes and models available at https://github.com/YuHengsss/YOLOV.

Introduction

Object detection, as a key component in a wide spectrum of vision-based intelligent applications (Dalal and Triggs 2005; Felzenszwalb, McAllester, and Ramanan 2008), aims to simultaneously locate and classify objects in images. Thanks to the strong ability of Convolutional Neural Networks (CNN) (Krizhevsky, Sutskever, and Hinton 2012), numerous CNN-based object detection models have been recently proposed, which can be roughly divided into two categories, say one-stage and two-stage object detectors. Specifically, two-stage detectors first select possible object regions (proposals), and then classify these regions. The series of Region-based CNN (R-CNN) (Girshick et al. 2014; Girshick 2015; Ren et al. 2015) is the pioneer of two-stage object detectors with a variety of follow-ups (He et al. 2016; Lin et al. 2017a; Dai et al. 2016; Cai and Vasconcelos 2018; He et al. 2017; Liu et al. 2018), which remarkably boosts the accuracy of detection. Given region-level features, these detectors for still images can be easily transferred to more complicated tasks such as segmentation and video object detection. However, due to the two-stage nature, the efficiency is a bottleneck for practical use. While for one-stage object detectors, the localization and classification are jointly

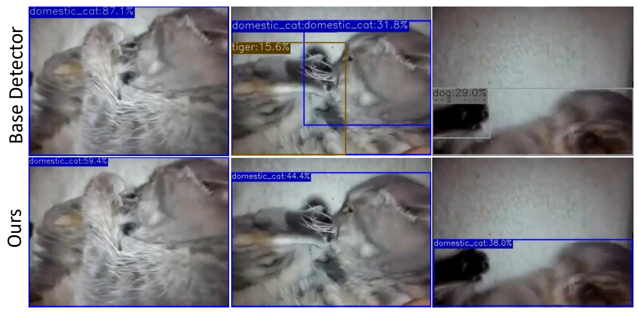

Figure 1: These frames suffer from various degradation, like motion blur and occlusion, making the base detector YOLOX fail to accurately detect the objects, while our method could.

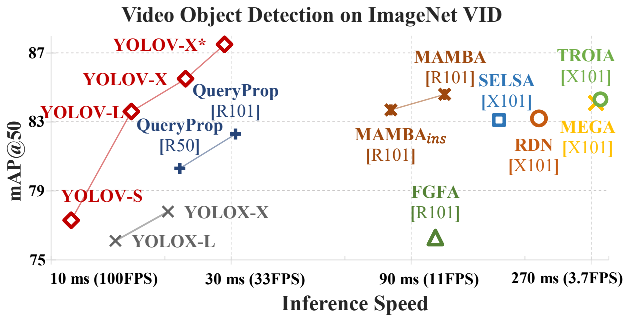

Figure 2: Performance comparison in accuracy (AP50) and speed. QueryProp and MBMBA are tested on a TITAN RTX GPU as reported in their papers, while the others are tested on a 2080Ti GPU. $*$ denotes involving post-processing.

[*]Corresponding author

and directly produced by the dense prediction from feature maps. The YOLO family (Redmon et al. 2016; Redmon and Farhadi 2017; Bochkovskiy, Wang, and Liao 2020) and SSD (Liu et al. 2016) are representatives in this group. Without resorting to region proposals as in the aforementioned two-stage approaches, the speed of one-stage detectors is superior and suitable for scenarios with real-time requirements. Though the accuracy of one-stage detectors are typically inferior at the beginning, follow-up designs (Lin et al. 2017b; Ge et al. 2021b,a; Tian et al. 2019) largely narrow the accuracy gap.

Video object detection can be viewed as an advanced version of still image object detection. Intuitively, one can process video sequences via feeding frames one-by-one into still image object detectors. Nevertheless, the temporal information across frames are unexploited, which could be the key to eliminate and reduce the ambiguity in a single image. As shown in Fig. 1, degradation such as motion blur, camera defocus, and occlusion often appears in video frames, significantly increasing the difficulty of detection. For instance, via solely looking at the last frame in Fig. 1, it is hard or even impossible for human beings to tell where and what the object is. On the other hand, video sequences can provide richer information than single still images. In other words, other frames in the same sequence can possibly support the prediction for a certain frame. Hence, *how to effectively aggregate temporal cues from different frames is crucial to the accuracy*.

In the literature, there are two main types for frame aggregation, *i.e.*, box-level and feature-level. These two technical routes can boost detection accuracy from different perspectives. Regarding box-level methods, they connect predictions of still object detectors through linking bounding boxes to form tubelets, and then refine the results in the same tubelet. The box-level methods can be viewed as post-processing, which are flexible to be applied to both one-stage and two-stage detectors. While for feature-level schemes, the features of a keyframe are enhanced by finding and aggregating similar features from other frames (*a.k.a.*, reference frames). The two-stage manner endows proposals with explicit representation from the backbone feature map extracted by Region Proposal Network (RPN) (Ren et al. 2015). Benefiting from this nature, two-stage detectors can be easily migrated to the video object detection problem. Hence, most of video object detectors are built on two-stage detectors. However, because of the aggregation module, these two-stage video object detectors further slow down, and thus hardly meet the need of real-time uses. Different from the two-stage methods, proposals are implicitly represented by features of each position in feature maps from a one-stage detector. Though, without explicit representations for objects, these features can still benefit from aggregating temporal information for the VID task. Again, as previously mentioned, the one-stage strategies usually run faster than those two-stage ones. Driven by these considerations, a natural question arises: *Can we make such region-level designs available to one-stage detectors while retaining the fast speed for building a practical (accurate and fast) video object detector*?

Contribution. This paper answers the above question via designing a simple yet effective strategy to aggregate features generated by one-stage detectors[1] (*e.g.*, YOLOX (Ge et al. 2021b) in this work to verify the primary claims). To connect the features of reference frames with those in the keyframe, we propose a feature similarity measurement module to construct an affinity matrix, which is then employed to guide the aggregation. To further mitigate the limitation of cosine similarity, an average pooling operator on reference features is customized. These two operations cost marginal computational resources with significant gains in accuracy. Equipped with the proposed strategies, our model, termed as YOLOV, can achieve a promising accuracy 85.5% AP50 on the ImageNet VID dataset with 40+ FPS on a single 2080Ti GPU (please see Fig. 2 for details) without bells and whistles, which is attractive for practical scenarios. By further introducing post-processing, its accuracy reaches higher up to 87.5% AP50 over 30 FPS.

Related Work

This section will briefly review representative approaches in object detection closely related to this work.

Object Detection in Still Images. Thanks to the development of hardware, large-scale datasets (Lin et al. 2014; Krizhevsky, Sutskever, and Hinton 2012) and sophisticated network structures (Simonyan and Zisserman 2015; He et al. 2016; Xie et al. 2017; Wang et al. 2020), the performance of object detection has continuously improved. Existing object detectors can be mainly divided into two-stage and one-stage schemes. Representative two-stage detectors such as RCNN (Girshick et al. 2014), Faster RCNN (Ren et al. 2015), R-FCN (Dai et al. 2016), and Mask RCNN (He et al. 2017). The methods in this group first select candidate regions through RPN and then extract features for the candidates through some feature extraction modules like RoIPooling (Ren et al. 2015) and RoIAlign (He et al. 2017). Finally, they conduct bounding box regression and classification through an additional detection head. There are also many successful one-stage detectors such as the YOLO series (Redmon et al. 2016; Redmon and Farhadi 2017; Bochkovskiy, Wang, and Liao 2020; Ge et al. 2021b), SSD (Liu et al. 2016), RetinaNet (Lin et al. 2017b), FCOS (Tian et al. 2019). Different from the two-stage counterparts, the one-stage detectors perform dense prediction on feature maps and directly predict the position and class probability without region proposals. One-stage detectors are usually faster than two-stage ones, owing to the end-to-end manner. However, they lack explicit region-level semantic features that are widely used for feature aggregation in video object detection. Our work attempts to explore the feasibility of aggregation over position-level features in feature map for one-stage detectors.

Object Detection in Videos. Compared to still image object detection, degradation may frequently appear in some video frames. When the keyframe is contaminated, temporal information could be useful for better detection. One branch

[1]Please note that our design is general and suitable to many detectors with different backbones.

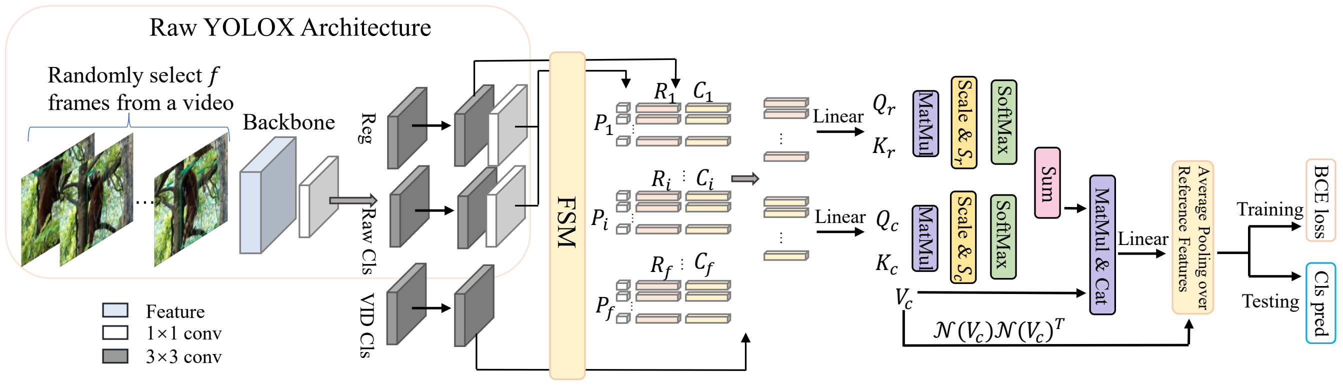

Figure 3: Framework of our design. Taking YOLOX as an example base detector, the corresponding model is termed as YOLOV. We randomly sample a number of frames from a video and feed them into the base detector to extract features. According to the predictions of YOLOX, the Feature Selection Module (FSM) picks out top k confident proposals and applies NMS on the selected proposals for further refinement. All the features from FSM are fed into our Feature Aggregation Module (FAM) for final classification. The proposed strategy can be easily applied to other base detectors.

of existing video object detectors concentrate on tracklet-level post-processing (Han et al. 2016; Belhassen et al. 2019; Sabater, Montesano, and Murillo 2020). The methods in this category try to refine the prediction results from the still image detector in consecutive frames by forming the object tubelets. The final classification score of each box is adjusted according to the entire tubelet. Another branch aims to enhance the features of keyframes, expecting to alleviate degradation via utilizing the features from (selected) reference frames. These approaches can be roughly classified as optical flow-based (Zhu et al. 2017a, 2018), attention-based (Wu et al. 2019; Deng et al. 2019; Chen et al. 2020; Gong et al. 2021; Sun et al. 2021) and tracking-based (Feichtenhofer, Pinz, and Zisserman 2017; Zhang et al. 2018b) methods. Deep feature flow (Zhu et al. 2017b) first introduces optical flow for image-level feature alignment, and FGFA (Zhu et al. 2017a) adopts optical flow to aggregate features along motion paths. Considering the computational cost of image-level feature aggregation, several attention-based methods have been developed. As a representative, SESLA (Wu et al. 2019) proposes a long-range feature aggregation scheme according to the semantic similarity between region-level features. Inspired by the relation module from Relation Networks (Hu et al. 2018) for still image detection, RDN (Deng et al. 2019) captures the relationship between objects in both spatial and temporal contexts. Furthermore, MEGA (Chen et al. 2020) designs a memory enhanced global-local aggregation module for better modeling the relationship between objects. Alternatively, TROIA (Gong et al. 2021) utilizes ROI-Align operation for fine-grained feature aggregation, while HVR-Net (Han et al. 2020) integrates intra-video and inter-video proposal relations for further improvement. Moreover, MBMBA (Sun et al. 2021) enlarges the reference feature set by introducing memory bank. QueryProp (He et al. 2022) notices the high computational cost of video object detectors and tries to speed up the process through a lightweight query propagation module. Besides the attention based methods, D&T (Feichtenhofer, Pinz, and Zisserman 2017) tries solving video object detection in a tracking manner by constructing correlation maps of different frame features. Although these approaches boost the precision of detection, they are mostly based on two-stage detectors and hence suffer from the relatively slow inference speed.

Methodology

Considering the characteristics of videos (various degradation *vs.* rich temporal information), instead of individually processing frames, how to seek supportive information from other frames for a target one (keyframe) plays a key role in boosting the accuracy of video object detection. Recent attempts (Deng et al. 2019; Chen et al. 2020; Wu et al. 2019; He et al. 2022) with noticeable improvement in accuracy corroborate the importance of temporal aggregation to the problem. However, most of the existing methods are two-stage based techniques. As previously discussed, their main drawback is relatively slow speed, compared to one-stage bases. To mitigate this limitation, we put the region/feature selection after the prediction head of one-stage detectors. In this section, we choose the YOLOX as base to present our main claims. Our proposed framework is schematically depicted in Fig. 3. Let us recall the traditional two-stage pipeline: 1) massive candidate regions are first "selected" as proposals; and 2) determine if each proposal is an object or not and which class it belongs to. The computational bottleneck mainly comes from dealing with substantial low-confidence region candidates. As can be seen from Fig. 3, our pipeline also contains two stages. Differently, *its first stage is prediction (having a large number of regions with low confidences discarded), while the second stage can be viewed as region-level refinement (taking advantage of other frames by aggregation).* By this principle, our design can simultaneously benefit from the efficiency of one-stage detectors and the accuracy gained from temporal aggregation. It is worth to emphasize that *such a small difference in design leads to a huge difference in performance.* The proposed strategy can be generalized to many base detectors such as FCOS (Tian et al. 2019) and PPYOLOE (Xu et al. 2022).

Our Design

From a possible perspective of human beings, the recognition procedure would first link *related* instances in temporal and identify which class they belong to until sufficiently *confident* cues are collected. Then, the results can be broadcasted to less confident cases. The mechanism of multi-head attention, as a key part of Transformers (Vaswani et al. 2017), seems to fit the situation well, which enhances the capability of long-range modeling. Given a sequence Z, the query, key, and value matrices are packed as Q, K and V, respectively. The self-attention can be calculated via:

$$\mathrm{SA}(Z) = \mathrm{softmax}\left(A\right) V \quad \text{with} \quad A = QK^T/\sqrt{d}, \tag{1}$$

where d is the dimension of each feature in Q (also K). Putting m self-attentions in parallel yields the multi-head attention by simply concatenating them together as follows:

$$\mathrm{MSA}(Z) = \mathrm{concat}\left(\mathrm{SA}_1(Z), \mathrm{SA}_2(Z), ..., \mathrm{SA}_m(Z)\right). \tag{2}$$

Modern two-stage based video object detectors typically obtain candidate regions for feature aggregation by RPN (Ren et al. 2015). As a representative, RelationNet (Hu et al. 2018) first introduces the above multi-head attention to still object detection task by viewing a sequence of proposals as input. The ROI-Pooling or ROI-Align operations are applied to these proposals to extract region-level features. However, one-stage detectors make dense prediction directly from feature maps. Simply transferring the region-level feature aggregation to the whole feature maps of one-stage detectors will result in intensive computational cost. To address this issue, we propose an effective strategy for selecting features that suitable for multi-head attentions.

FSM: Feature Selection Module As most of the predictions are of low confidence, the detection head of one-stage detectors is a natural and rational option to select (high-quality) candidates from the feature maps. Following the process of RPN, we first pick out top k (*e.g.*, 750) predictions according to the confidence scores. Then, a fixed quantity a of predictions (*e.g.*, $a = 30$) are chosen after the Non-Maximum Suppression (NMS) for reducing the redundancy. The features of these predictions will be collected for further refinement. In practice, we found that directly aggregating the collected features in the classification branch and backpropagating the classification loss of the aggregated features will result in unstable training. Since the weight of the feature aggregation module is randomly initialized, fine-tuning all the networks from the beginning will contaminate the pre-trained weights. To address the above concerns, we fix the weights in the base detector except for the linear projection layers in detection head. We further insert two 3×3 convolutional (Conv) layers into the model neck as a new branch, called video object classification branch, which generates features for aggregation. Then, we feed the collected features from the video and regression branches into our feature aggregation module.

FAM: Feature Aggregation Module Now we come to the step of connecting related RoIs. Let $\mathcal{F} = \{C_1, C_2, ..., C_f; R_1, R_2, ..., R_f\}$ denote the feature set selected by FSM where $C_i \in \mathbb{R}^{d_q \times a} = \left[\mathbf{c}_i^1, \mathbf{c}_i^2, ..., \mathbf{c}_i^a\right]$ and $R_i \in \mathbb{R}^{d_q \times a} = \left[\mathbf{r}_i^1, \mathbf{r}_i^2, ..., \mathbf{r}_i^a\right]$ denotes the features of the i-th frame in $\mathcal{F}$ from the video classification and regression branches, respectively. d_q refers to the feature dimension of each RoI and f refers to the number of related frames. The generalized cosine similarity is arguably the most widely used metric to compute the similarity between features or the attention weight (Wu et al. 2019; Shvets, Liu, and Berg 2019; Deng et al. 2019). Simply referring to the cosine similarity will find features most similar to the target. However, when the target suffers from some degradation, the selected reference proposals using cosine similarity are very likely to have the same problem. We name this phenomenon *the homogeneity issue*.

First, let's recall the original **QK manner** in self-attention for our method. Similarly to (Vaswani et al. 2017), the query, key, and value matrices are constructed and fed into the multi-head attention. For instance, Q_c and Q_r are respectively formed by stacking the features from the classification branch and the regression branch for all proposals in all related frames (*i.e.*, $Q_c \in \mathbb{R}^{fa \times d} = \mathrm{LP}([C_1, C_2, ..., C_f]^T)$ and $Q_r \in \mathbb{R}^{fa \times d} = \mathrm{LP}([R_1, R_2, ..., R_f]^T)$, where $\mathrm{LP}(\cdot)$ is the linear projection operator which projects the features of d_q-dim to d-dim), while the others are done similarly. By the scaled dot-product in attention, we obtain the corresponding $A_c = Q_c K_c^T/\sqrt{d}$ and $A_r = Q_r K_r^T/\sqrt{d}$.

To overcome the homogeneity issue, we further take predicted confidences from the raw detector into consideration, denoted as $\mathbf{P} \in \mathbb{R}^{2 \times fa} = \{P_1, P_2, ..., P_f\}$ with each P_i contains two scores, *i.e.*, the classification score and IoU score from the raw classification and regression heads. To fit these scores into the attention weights, we build two matrices, say $S_r \in \mathbb{R}^{fa \times fa}$ and $S_c \in \mathbb{R}^{fa \times fa}$, through respectively repeating each of two rows of $\mathbf{P}$ fa times. As a consequence, the self-attentions for the classification and regression branches turn out to be:

$$\begin{aligned} \mathrm{SA_c}(\mathcal{C}) = \mathrm{softmax}\left(S_c \circ A_c\right) V_c, \\ \mathrm{SA_r}(\mathcal{R}) = \mathrm{softmax}\left(S_r \circ A_r\right) V_c, \end{aligned} \tag{3}$$

where $\circ$ represents the Hadamard product. By this operation, the self-attention considers not only the similarity between the query and key items, but also aware of the quality of the keys. Note that, since the main purpose is to refine the classification, $\mathrm{SA_c}(\mathcal{C})$ and $\mathrm{SA_r}(\mathcal{R})$ share the same value matrix V_c. Our experiments demonstrate that replacing the original QK manner with Eq. (3) (called **affinity manner**) in multi-head attention can significantly boost the performance of video object detector. Besides, we concatenate V_c with the outputs of Eq. (3) for better preserving initial representations via:

$$\mathrm{SA}(\mathcal{F}) = \mathrm{concat}((\mathrm{SA_c}(\mathcal{C}) + \mathrm{SA_r}(\mathcal{R}), V_c). \tag{4}$$

The positional information is not embedded, because the locations in a long temporal range would not be helpful as claimed in (Chen et al. 2020).

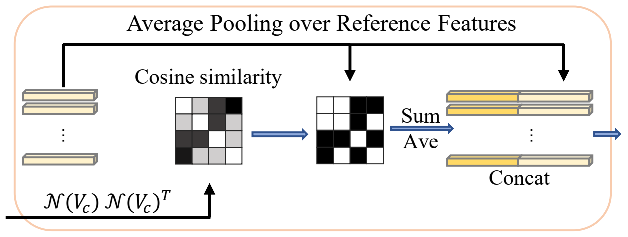
Average Pooling over Reference Features
Cosine similarity
Sum
Ave
Concat
$\mathcal{N}(V_c)\,\mathcal{N}(V_c)^T$

Figure 4: Our average pooling over reference features.

Moreover, considering the characteristic of softmax, a small part of references may hold a large portion of weights. In other words, it often overlooks the references with low weights, which limits the diversity of reference features for possible follow-up use. To avoid such risks, we introduce an **average pooling over reference features (A.P.)**. Concretely, we select all of the references with similarity scores above a threshold τ and apply the average pooling to these survivals. Note that the similarity in this work is computed via $\mathcal{N}(V_c)\mathcal{N}(V_c)^T$. The operator $\mathcal{N}(\cdot)$ means the layer normalization, which guarantees the values to be in a certain range and thus eliminates the influence of scale difference. The average-pooled features and the key features are then fed into one linear projection layer for final classification. The procedure is illustrated in Fig. 4.

Experimental Validation

Implementation Details

Similar to previous work, we also initialize our base detector from COCO pre-trained weights provided by YOLOX. Following (Zhu et al. 2017a; Wu et al. 2019; Gong et al. 2021), we combine the videos in the ImageNet VID and those in the ImageNet DET with the same classes as our training data. Specifically, the ImageNet VID (Russakovsky et al. 2015) contains 3,862 videos for training and 555 videos for validation. There are 30 categories in the VID dataset, *i.e.*, a subset of the 200 basic-level categories of ImageNet DET (Russakovsky et al. 2015). Considering the redundancy of video frames, we randomly sample 1/10 frames in the VID training set instead of using all of them. The base detectors are trained for 7 epochs by SGD with batch size of 16 on 2 GPUs. As for the learning rate, we adopt the cosine learning rate schedule used in YOLOX with one epoch for warming up and disable the strong data augmentation for the last 2 epochs. When integrating the feature aggregation module into the base detectors, we fine-tune them for 150K iterations with batch size of 16 on a single 2080Ti GPU. In addition, we use warm-up for the first 15K iterations and cosine learning rate schedule for the rest iterations. Only the linear projection layers in YOLOX prediction head, the newly added video object classification branch and the multi-head attention are fine-tuned. For the training of feature aggregation module, the number of frames f is set to 16, and the threshold of NMS is set to 0.75 for rough feature selection. While for producing final detection boxes, we alternatively set the threshold of NMS to 0.5 for retaining more confident

f_g	3	7	15	23	**31**	39
AP50 (%)	74.6	76.1	77.0	77.3	**77.3**	77.3
f_l	3	7	15	23	31	39
AP50 (%)	70.4	71.2	72.2	72.8	73.2	73.6

Table 1: Effect of the numbers of global f_g and local f_l reference frames.

a	10	20	**30**	50	75	100
AP50 (%)	76.1	77.0	**77.3**	77.3	77.4	77.4
Time (ms)	10.4	10.7	**11.3**	13.8	19.5	29.3

Table 2: Effect of the number of proposals a in FSM.

candidates. In the training phase, the images are randomly resized from 352×352 to 672×672 with 32 steps. In the testing phase, the images are uniformly resized to 576×576. The AP50 and inference speed are two metrics to reflect the performance in terms of accuracy and efficiency, respectively. Regarding the inference speed, we test all of the models with FP16-precision on a 2080Ti GPU unless otherwise stated.

Ablation Study

On the reference frame sampling strategy. Investigating frame sampling strategies to balance accuracy and efficiency is crucial for video object detectors. Several global and local sampling schemes have been discussed in previous two-stage based methods (Wu et al. 2019; Gong et al. 2021; Chen et al. 2020). As for the global sampling schemes, f_g frames are randomly selected from the whole video. With respect to the local sampling, f_l consecutive frames around the keyframe are employed. To illustrate the effect of different sampling strategies, we vary the numbers of reference frames in both global and local sampling. The numerical results are reported in Table 1. The performance of using only 3 global reference frames has already outperformed that of using 39 local reference frames, which corroborates the evidence given in (Wu et al. 2019; Gong et al. 2021). As a trade-off, we adopt the global sampling strategy with $f_g = 31$ by default for the remaining experiments according to Table 1.

On the number of proposals in a single frame. In this experiment, we adjust the quantity of most confident proposals remained for each frame a from 10 to 100 in the FSM, to see its influence on the performance. As shown in Table 2, as a increases, the accuracy continuously improves and saturates until reaching 75. As the complexity of self-attention is $O\left(n^2\right)$ with respect to the amount of input proposals, involving too many proposals per frame will dramatically increase the time cost. As a tradeoff, we adopt $a = 30$ that is much smaller than the optimal setting with 75 proposals used in the two-stage based method RDN (Deng et al. 2019).

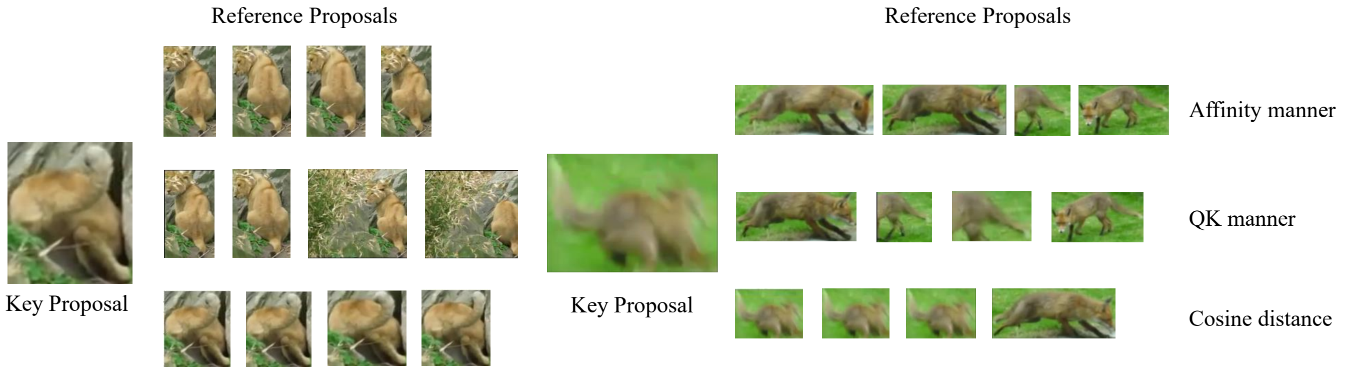

Figure 5: Visual comparison between reference proposals selected by three different methods for given key proposals. We display four reference proposals that contribute most in aggregation.

τ	0	0.2	0.5	0.65	**0.75**	0.85	1
AP50 (%)	73.1	73.9	76.1	77.1	**77.3**	77.3	76.9

Table 3: Influence of the threshold τ in average pooling over reference features.

Methods	A.M.	A.P.	Time (ms)	AP50 (%)
YOLOX-S	-	-	9.38	69.5
YOLOV-S	✓	-	10.95	$76.9_{\uparrow 7.4}$
YOLOV-S	✓	✓	11.30	$77.3_{\uparrow 7.8}$

Table 4: Effectiveness of our affinity manner (A.M.) and average pooling over reference features (A.P.).

On the threshold in average pooing of reference features. Here, we test the effect of different thresholds for average pooling over reference features. Table 3 lists the numerical results. When all the features are involved in the average pooling, *i.e.* $\tau = 0$, the AP50 is merely 73.1%. Raising to higher threshold results in better performance. When τ falls in $[0.75, 0.85]$, the accuracy keeps steady high to 77.3%. But, when $\tau = 1$, the average pooling is equivalent to only duplicating $\mathrm{SA}(\mathcal{F})$, the accuracy of which drops to 76.9%. Dynamically determining the threshold for different cases is desired and is left as our future work. For the rest of experiments, we adopt $\tau = 0.75$ as the default threshold.

On the effectiveness of FAM. To validate the effectiveness of the affinity manner (A.M.) and the average pooling over reference features (A.P.), we evaluate the performance with and without these modules. The results in Table 4 reveal that these designs can both help the feature aggregation to catch better semantic representations from one-stage detectors. Compared to YOLOX-S (69.5% AP50), the YOLOV-S only armed with A.M. gains 7.4% in accuracy. While equipping YOLOV-S with both A.M. and A.P. (complete YOLOV-S), the performance reaches 77.3% in AP50 with only about 2ms time cost in comparison with YOLOV-S. Furthermore, we also plug our design into the YOLOX-L and YOLOX-X. Tabel 5 shows the detailed comparison. The † indicates using strong augmentation (like MixUp (Zhang et al. 2018a) and Mosaic (Bochkovskiy, Wang, and Liao 2020)) when fine-tuning our version. Our YOLOV consistently outperform their corresponding baselines over 7% in AP50.

Model	Params	GFLOPs	Time (ms)	AP50 (%)
YOLOX-S	8.95M	21.63	9.4	69.5
YOLOV-S	10.28M	26.18	11.3	$77.3_{\uparrow 7.8}$
YOLOX-L	54.17M	125.90	14.8	76.1
YOLOV-L	59.45M	143.10	16.4	$83.6_{\uparrow 7.5}$
YOLOX-X	99.02M	228.15	20.4	77.8
YOLOV-X	107.26M	254.72	22.7	$85.0_{\uparrow 7.2}$
YOLOV-X†	107.26M	254.72	22.7	$85.5_{\uparrow 7.7}$

Table 5: Effectiveness of our strategy compared to bases. † indicates using strong augmentation.

Moreover, we provide two cases to intuitively exhibit the superiority of our FAM. They are the lion case with rare pose and the fox case with motion blur as shown in Fig 5. Without loss of generality, the top 4 reference proposals are listed for different feature selection modes, including the cosine similarity, QK manner in multi-head attention, and our affinity manner. As previously analyzed, the cosine manner selects proposals most similar to the key proposal but suffering the same degradation problem as the key proposal. Though the QK manner alleviates this problem, it is obviously inferior to the affinity manner. By introducing the confidence scores as guidance, our method selects better proposals and further boosts the detection accuracy.

Comparison with State-of-the-art Methods

Table 6 summarizes the detailed performance of related methods including FGFA (Zhu et al. 2017a), SELSA (Wu et al. 2019), RDN (Deng et al. 2019), MEGA (Chen et al. 2020), TROIA (Gong et al. 2021), MAMBA (Sun et al.

Method	Backbone	AP50 (%)	Time (ms)
FGFA	R101	76.3	104.2
SELSA	X101	83.1	153.8
RDN	X101	83.2	-
MEGA	R101	82.9	230.4
MEGA	X101	84.1	-
TROIA	X101	84.3	285.7
MAMBA	R101	84.6	110.3(T)
HVR	X101	84.8	-
TransVOD	R101	81.9	-
QueryProp	R50	80.3	21.9(T)
QueryProp	R101	82.3	30.8(T)
YOLOV-R50	R50	**81.4**	**17.5**
YOLOV-S	MCSP	**77.3**	**11.3**
YOLOV-L	MCSP	**83.6**	**16.3**
YOLOV-X	MCSP	**85.0**	**22.7**
YOLOV-X†	MCSP	**85.5**	**22.7**
FGFA	R101	78.4	-
SELSA	X101	83.7	-
RDN	X101	84.7	-
MEGA	X101	85.4	-
HVR	X101	85.5	-
YOLOV-S	MCSP	**80.1**	**11.3 + 6.9**
YOLOV-L	MCSP	**86.2**	**16.3 + 6.9**
YOLOV-X	MCSP	**87.2**	**22.7 + 6.1**
YOLOV-X†	MCSP	**87.5**	**22.7 + 6.1**

Table 6: Performance comparison in accuracy and speed. † indicates with strong augmentation, T means the inference time is tested on a TITAN RTX GPU as reported in corresponding papers. MCSP stands for the Modified CSP v5 backbone adopted in YOLOX. The lower part involves post-processing while the upper does not.

2021), HVR (Han et al. 2020), TransVOD (He et al. 2021) and QueryProp (He et al. 2022). Our method can achieve 85.5% AP50 with 21.1 ms per frame. With the help of a recent post-processing method REPP (Sabater, Montesano, and Murillo 2020), it reaches 87.5% AP50 spending extra 6 ms. In terms of inference efficiency, our method is much faster than the other methods. To show the improvement is not purely from changing to stronger backbone network, we also report the speed and accuracy using ResNet50.

To be specific, in the upper part of Table 6, we report the performance without any post-processing methods. YOLOV can remarkably lead in both detection accuracy and inference efficiency. For fair comparison, all the models listed in Table 6 are tested with the same hardware environment except for MAMBA and QueryProp[2]. The lower part of Table 6 reports the results of our YOLOV and the other SOTA models with post-processing. The time cost of post-processing is tested on an i7-8700K CPU.

[2]The codes and models of the methods with marks 'T' (the reported results are from their papers). '-' denotes the model is not released and the speed is not reported in orginal paper.

Model	Params	GFLOPs	AP50 (%)
PPYOLOE-S	6.71M	12.40	69.5
PPYOLOE-S+Ours	8.04M	16.73	$74.9_{\uparrow 5.6}$
PPYOLOE-L	50.35M	89.19	76.9
PPYOLOE-L+Ours	55.63M	105.49	$82.0_{\uparrow 5.1}$
FCOS	31.00M	102.84	67.0
FCOS+Ours	36.28M	120.04	$73.1_{\uparrow 6.1}$

Table 7: Effectiveness of our strategy on other base detectors.

Application to Other Base Detectors

In order to validate the generalization ability of the proposed strategy, we also try it on other widely-used one-stage detectors including PPYOLOE (Xu et al. 2022) and FCOS (Tian et al. 2019). Specifically for PPYOLOE, it has different channel numbers at different FPN levels. To achieve the multi-scale feature aggregation, we reduce the channel numbers of the detection head at different scales to be the smallest of all levels in the pre-trained model. While for FCOS, the backbone is ResNet-50 (He et al. 2016). The FPN consists of five levels in the original architecture for dealing with images with large image sizes (*e.g.*, 1333 × 800). To match the situation of the ImageNet VID, we maintain three FPN levels with the largest downsampling rate of 32. For the training procedure and other hyper-parameters settings, we simply keep them the same as those in the YOLOX. Table 7 shows that our strategy can consistently improve different base detectors by over 5% in terms of AP50. It is worth noting that searching for more suitable hyper-parameters for different base detectors could obtain better performance.

Conclusion

In this paper, we developed a practical video object detector both efficiently and effectively. To improve the detection accuracy, a feature aggregation module was designed to aggregate temporal information. For saving computational resources, different from existing two-stage detectors, we proposed to put the region selection after the one-stage prediction. This subtle change makes our detectors significantly more efficient. Experiments and ablation studies have been conducted to verify the effectiveness of our strategy, and it surpasses over previous arts. The core idea is simple and general, which can potentially inspire further research works and broaden the applicable scenarios related to video object detection.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant no. 62072327.

References

Belhassen, H.; Zhang, H.; Fresse, V.; and Bourennane, E.-B. 2019. Improving Video Object Detection by Seq-Bbox Matching. In *VISIGRAPP (5: VISAPP)*, 226–233.

Bochkovskiy, A.; Wang, C.-Y.; and Liao, H.-Y. M. 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.

Cai, Z.; and Vasconcelos, N. 2018. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 6154–6162.

Chen, Y.; Cao, Y.; Hu, H.; and Wang, L. 2020. Memory enhanced global-local aggregation for video object detection. In *CVPR*, 10337–10346.

Dai, J.; Li, Y.; He, K.; and Sun, J. 2016. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, 379–387.

Dalal, N.; and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *CVPR*, 886–893.

Deng, J.; Pan, Y.; Yao, T.; Zhou, W.; Li, H.; and Mei, T. 2019. Relation distillation networks for video object detection. In *ICCV*, 7023–7032.

Feichtenhofer, C.; Pinz, A.; and Zisserman, A. 2017. Detect to track and track to detect. In *ICCV*, 3038–3046.

Felzenszwalb, P.; McAllester, D.; and Ramanan, D. 2008. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 1–8.

Ge, Z.; Liu, S.; Li, Z.; Yoshie, O.; and Sun, J. 2021a. Ota: Optimal transport assignment for object detection. In *CVPR*, 303–312.

Ge, Z.; Liu, S.; Wang, F.; Li, Z.; and Sun, J. 2021b. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*.

Girshick, R. 2015. Fast R-CNN. In *ICCV*, 1440–1448.

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 580–587.

Gong, T.; Chen, K.; Wang, X.; Chu, Q.; Zhu, F.; Lin, D.; Yu, N.; and Feng, H. 2021. Temporal ROI align for video object recognition. In *AAAI*, 1442–1450.

Han, M.; Wang, Y.; Chang, X.; and Qiao, Y. 2020. Mining inter-video proposal relations for video object detection. In *ECCV*, 431–446.

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 preprint arXiv:1602.08465*.

He, F.; Gao, N.; Jia, J.; Zhao, X.; and Huang, K. 2022. QueryProp: Object Query Propagation for High-Performance Video Object Detection. In *AAAI*, 834–842.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask R-CNN. In *ICCV*, 2961–2969.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, L.; Zhou, Q.; Li, X.; Niu, L.; Cheng, G.; Li, X.; Liu, W.; Tong, Y.; Ma, L.; and Zhang, L. 2021. End-to-End Video Object Detection with Spatial-Temporal Transformers. In *ACM MM*, 1507–1516.

Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; and Wei, Y. 2018. Relation networks for object detection. In *CVPR*, 3588–3597.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, 1097–1105.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017a. Feature pyramid networks for object detection. In *CVPR*, 2117–2125.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017b. Focal loss for dense object detection. In *ICCV*, 2980–2988.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *ECCV*, 740–755.

Liu, S.; Qi, L.; Qin, H.; Shi, J.; and Jia, J. 2018. Path aggregation network for instance segmentation. In *CVPR*, 8759–8768.

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*, 21–37.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You only look once: Unified, real-time object detection. In *CVPR*, 779–788.

Redmon, J.; and Farhadi, A. 2017. YOLO9000: better, faster, stronger. In *CVPR*, 7263–7271.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 91–99.

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*, 211–252.

Sabater, A.; Montesano, L.; and Murillo, A. C. 2020. Robust and efficient post-processing for video object detection. In *IROS*, 10536–10542.

Shvets, M.; Liu, W.; and Berg, A. C. 2019. Leveraging long-range temporal relationships between proposals for video object detection. In *ICCV*, 9756–9764.

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

Sun, G.; Hua, Y.; Hu, G.; and Robertson, N. 2021. MAMBA: Multi-level Aggregation via Memory Bank for Video Object Detection. In *AAAI*, 2620–2627.

Tian, Z.; Shen, C.; Chen, H.; and He, T. 2019. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 9627–9636.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*, 5998–6008.

Wang, C.-Y.; Liao, H.-Y. M.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W.; and Yeh, I.-H. 2020. CSPNet: A new backbone that can enhance learning capability of CNN. In *CVPR*, 390–391.

Wu, H.; Chen, Y.; Wang, N.; and Zhang, Z. 2019. Sequence level semantics aggregation for video object detection. In *ICCV*, 9217–9225.

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated residual transformations for deep neural networks. In *CVPR*, 1492–1500.

Xu, S.; Wang, X.; Lv, W.; Chang, Q.; Cui, C.; Deng, K.; Wang, G.; Dang, Q.; Wei, S.; Du, Y.; et al. 2022. PP-YOLOE: An evolved version of YOLO. *arXiv preprint arXiv:2203.16250*.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018a. mixup: Beyond Empirical Risk Minimization. In *ICLR*.

Zhang, Z.; Cheng, D.; Zhu, X.; Lin, S.; and Dai, J. 2018b. Integrated object detection and tracking with tracklet-conditioned detection. *arXiv preprint arXiv:1811.11167*.

Zhu, X.; Dai, J.; Yuan, L.; and Wei, Y. 2018. Towards high performance video object detection. In *CVPR*, 7210–7218.

Zhu, X.; Wang, Y.; Dai, J.; Yuan, L.; and Wei, Y. 2017a. Flow-guided feature aggregation for video object detection. In *ICCV*, 408–417.

Zhu, X.; Xiong, Y.; Dai, J.; Yuan, L.; and Wei, Y. 2017b. Deep feature flow for video recognition. In *CVPR*, 2349–2358.