Dynamic Feature Aggregation for Efficient Video Object Detection

Yiming Cui

University of Florida

Abstract. Video object detection is a fundamental yet challenging task in computer vision. One practical solution is to take advantage of temporal information from the video and apply feature aggregation to enhance the object features in each frame. Though effective, those existing methods always suffer from low inference speeds because they use a fixed number of frames for feature aggregation regardless of the input frame. Therefore, this paper aims to improve the inference speed of the current feature aggregation-based video object detectors while maintaining their performance. To achieve this goal, we propose a vanilla dynamic aggregation module that adaptively selects the frames for feature enhancement. Then, we extend the vanilla dynamic aggregation module to a more effective and reconfigurable deformable version. Finally, we introduce inplace distillation loss to improve the representations of objects aggregated with fewer frames. Extensive experimental results validate the effectiveness and efficiency of our proposed methods: On the ImageNet VID benchmark, integrated with our proposed methods, FGFA and SELSA can improve the inference speed by 31% and 76% respectively while getting comparable performance on accuracy. Codes are available at `https://github.com/YimingCuiCuiCui/DFA`.

Keywords: Video object detection · Dynamic feature aggregation.

1 Introduction

Object detection is an essential task in computer vision which aims to localize and categorize objects of interest in a single or sequence of images [3, 9, 14, 27, 31, 39, 45]. With the excellent performance of deep learning-based computer vision methods on image object detection tasks [9, 14, 27, 31], researchers have begun to extend image object detection to the more challenging video domain. Compared with still images, videos have the issues of feature degradation caused by camera jitters or fast motion that rarely happen in the image domains [3, 46], which increase the difficulty of object detection in videos. Therefore, directly applying object detectors from image domains on a frame-by-frame basis for video analysis always produces poor performance. Existing works can be divided into two directions to solve the issues caused by video feature degradation.

Since the same object always reappears in multiple frames, videos can provide rich temporal information, which provides hints for video analysis. Therefore, one

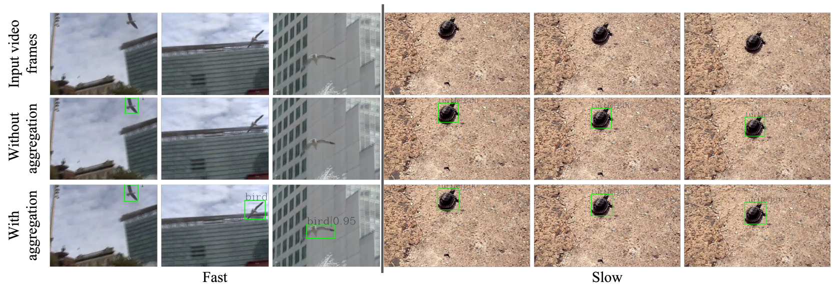


Fig. 1. Comparison of video object detection results with/without feature aggregation on objects with different motion speeds.

direction for video object detection is to exploit this temporal information with post-processing pipelines [13,20,21] where still image object detection approaches are first applied on each frame. Then those detected objects are assembled across frames with temporal hints like motion estimation or object tracking. However, those approaches are not trained end-to-end, and the detection results on a single frame and across frames cannot be optimized jointly. Therefore, if the predictions in a single frame are inaccurate, they cannot be optimized and refined during the post-processing procedure.

Another direction for video object detection is aggregating features across multiple frames to eliminate feature degradation in videos. These methods assume that frames with poor features only account for a small ratio compared with the whole video sequences. By aggregating temporal features, the performance of video object detection can be boosted. These methods can be categorized as local, global, and combinations, depending on how to aggregate features. The first sub-direction methods [35, 45, 46] exploit the local temporal information in videos to enhance the target frame features in a short time range and ignore the global information. To address this issue, the second sub-direction methods [7,11,12,30] introduce attention modules to use global temporal information. However, these methods ignore local temporal information due to GPU memory limits or computational constraints. The third sub-direction methods [3, 19] make a combination of local and global temporal information but always suffer from low inference speeds.

Though getting better performance than post-processing methods, those feature aggregation-based object detectors always have a lower inference speed. Therefore, besides focusing on the performance of video object detection, recent works [2, 18, 19, 40, 41] also design efficient frameworks to improve the inference speeds. However, these methods are designed for a specific framework and cannot be generalized to other video object detectors. What makes things worse, these approaches are always efficient during the inference process at the sacrifice of performance like accuracy or recall.

Unlike the existing efficient video object detectors, we aim to design a plug-and-play module that can be integrated into most existing methods to balance their inference speeds and performances. To achieve this, we first notice that the low inference speeds of the current feature aggregation-based object detectors are caused by their aggregation processes, which are proportional to the number of frames used for aggregation [3, 5, 45, 46]. It is natural to think whether it is necessary to always use a fixed number of frames for feature aggregation. For objects with fast motion speeds [1], feature aggregation can improve the video object detection performance. As shown in Figure 1(a), the flying bird cannot be correctly detected in multiple frames without feature aggregation. On the contrary, when the objects are with slow motion speeds, as shown in Figure 1 (b), original Faster R-CNN [27] without any feature aggregation can already detect the turtle in the current frame correctly. Therefore, using too many aggregation frames for videos with slow motion is unnecessary since the model with a few aggregation frames or even without aggregation can already perform well.

In this paper, we attempt to improve the efficiency of the current feature aggregation-based video object detectors in a simple yet effective way. We notice that there is no need to always use a fixed number of frames to aggregate features for video object detection regardless of the inputs. Therefore, we design modules to aggregate features dynamically. We first propose a vanilla dynamic feature aggregation strategy which can adaptively select frames for aggregation based on the inputs. Then, we extend the vanilla strategy to a deformable version which is more effective and reconfigurable. Finally, we introduce an inplace distillation loss to enhance the object feature representations when only a few frames are used for aggregation. Our contributions can be summarized as follows:

- To the best of our knowledge, we are the first to adaptively and dynamically aggregate features for video object detection to balance the model efficiency and performance.
- We design a vanilla dynamic feature aggregation (DFA) module and then extend it to a deformable version which can adaptively and reconfigurably enhance the object feature representations based on the input frame. Inplace distillation loss is introduced to improve the feature representations of those aggregated with fewer frames for better performance.
- Our proposed method is a plug-and-play module which can be integrated into most of the recent state-of-the-art video object detectors. Evaluated on the ImageNet VID benchmark, the performance of video object detection can be preserved with a much better inference speed when integrated with our proposed method.

2 Related Works

Still image object detection. Image object detection task aims to localize and categorize objects of interest in a still image. Current deep learning-based models

[1] For better analysis, we use the same way as FGFA [45] to categorize objects in every single frame based on their motion speeds.

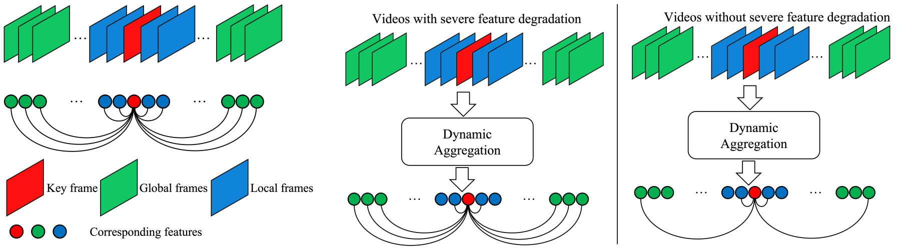
can be classified into two main directions: Two-stage object detector and one-stage object detector. Among them, R-CNN-based two-stage object detectors [1, 9, 14, 23, 27] first generate a fixed number of proposals with Region Proposal Network (RPN) [9] to localize and classify the object candidates coarsely. Then, they refine these proposals to output fine-grained predictions. To improve the inference speed of those models mentioned above, one-stage models [22,25,26,33] are introduced to predict the locations and categories of objects directly based on the extracted features from CNN without region proposals. For simplicity and generalization, our method is built based on Faster R-CNN [27], which is one of the state-of-the-art object detectors and can be easily extended to others.

Video object detection. Different from image object detection, the video object detection task must handle situations caused by motion to generate good predictions in each frame. Post-processing-based methods detect every frame separately and assemble those detected objects with various metrics like optical flow. Seq-NMS [13] assembles bounding boxes at different frames with the criteria of IoU threshold and re-ranks the linked bounding boxes. TCN [21] uses tubelet modules and applies a temporal convolutional network to embed temporal information to improve the detection across frames. Despite the simplicity, those methods are not trained end-to-end and perform poorly.

To solve the issues, feature aggregation-based methods [3, 39, 45, 46] usually enhance the object representations using the temporal information to eliminate the feature degradation caused by motions. Among them, FGFA [45] first warps the feature maps from the local adjacency frames to the keyframe based on the flow motion and then aggregates those warped features to improve the object representations for the following detection network. SELSA [39] aggregates features in a global full-sequence level. In SELSA, proposals across space-time domains with similar semantics are linked, and their features are weight-averaged for aggregation to provide richer information to handle issues like motion blur and pose changes. MANet [35] jointly aggregates object features on both pixel-level and instance-level. The pixel-level aggregation is used to model detailed motion, while the instance-level calibration is introduced to capture global motion cues. MEGA [3] takes global and local information into account where global features are first aggregated into local features. Then these global-enhanced local features are fused into the key frame for better detection performance. TF-Blender [5] improves the feature aggregation process using the temporal relations between frames. TransVOD [16] introduces the Transformer to aggregate the spatial and temporal information in a multi-head self-attention mode.

Compared with post-processing-based methods, feature aggregation-based video object detectors usually perform better with a lower inference speed. In this paper, we mainly focus on feature aggregation-based methods.

Efficient networks. Though deep learning-based methods perform better on multiple computer vision tasks, their complexities become higher, making them unsuitable for applications with constrained computational budgets but a short



(a) Existing methods

(b) Our method

Fig. 2. The framework of our proposed method, which uses dynamic aggregation strategies for video object detection tasks. (a) Current methods aggregate features with a fixed number of frames (e.g., 10) regardless of the input frames. (b) Our methods adaptively select frames for dynamic aggregation according to the input frame. For videos with severe feature degradation, 8 frames are selected to enhance the key frame representation. For videos with high qualities, 4 frames are used for aggregation for fast inference speed.

response time like mobile platforms. Therefore, recent works [17, 19, 29, 32, 40] begin to focus on how to speed up the detection process for real-time applications. Towards this goal, lightweight networks like Mobile-Net [17,29], Efficient-Net [32] and automated neural architecture search models [24,38] are introduced to take the place of heavy backbones like ResNet [15] to reduce the computation complexity for mobile applications. Besides replacing the backbones, LSTS [19] learns semantic-level spatial correspondences between neighboring frames to reduce the information redundancy in video frames to accelerate the detection process. CenterNet-HP [40] replaces two-stage detectors like Faster R-CNN [27] with one-stage model CenterNet [8] for real-time video object detection. Detection results from previous frames are propagated in the form of a heatmap to enhance the performance of the future frames. Other works [34, 37] improve the detection speeds with the help of compressed video information. Though efficient during inference, those methods usually require carefully designed modules and make great changes to the existing video object detectors, making it infeasible to generalize to other methods. Also, these methods generally have a worse performance despite high inference speeds. On the contrary, our proposed approaches are plug-and-play modules which can be easily integrated into the existing detectors to balance their efficiency and performance.

Recently, dynamic networks have been introduced, which allow selective inference paths. Slimmable networks [42–44] are models trained executable at different widths, which can be adaptive to multiple computational resources and get even better performance compared with their counterparts trained individually. For object detection, dynamic proposals are introduced for efficient inferernce [6]. In this paper, we borrow the idea from slimmable networks to make our model adaptive to different input videos and able to adjust the numbers of frames for aggregation according to the input frame for video object detection.

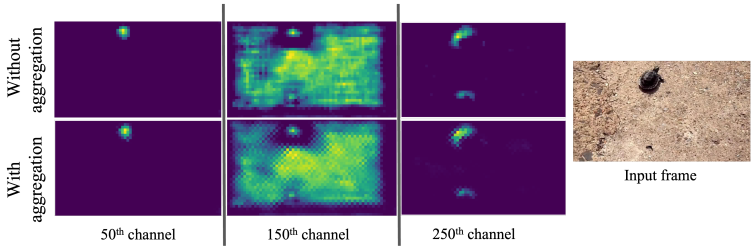


Fig. 3. Comparison of features maps with/without aggregation on objects with slow motion speeds.

3 Methodology

The key idea of our method is to replace the fixed number of frames with a dynamic size in the current feature aggregation-based video object detectors. Therefore, instead of using fixed frames, our model can adaptively choose the frames for aggregation according to the inputs, as shown in Figure 2. Our proposed method is a plug-and-play module which can be easily integrated into most feature aggregation-based video object detectors. In the following sections, we first review the current feature aggregation-based video object detectors and analyze the inefficiency of their aggregation processes. Then, we propose a vanilla dynamic aggregation strategy to adapt the model to different input frames. Next, we extend the vanilla dynamic aggregation to a deformable version which is more effective and reconfigurable. Finally, inplace distillation loss is introduced to enhance the feature representations aggregated with fewer adjacent frames.

3.1 Preliminary

The canonical feature aggregation methods [3, 7, 36, 39, 45] generally work on a fixed number of frames k, which can be summarized as: Given the current frame $\boldsymbol{I}_i$ and its neighboring frames $\boldsymbol{I}_j \in \mathcal{N}(\boldsymbol{I}_i)$, their corresponding features $\boldsymbol{f}_j$ are weighted averaged in order to aggregate the temporal feature $\Delta\boldsymbol{f}_i$:

$$\Delta\boldsymbol{f}_i = \sum_{\boldsymbol{I}_j \in \mathcal{N}(\boldsymbol{I}_i)} (w_{ij} \times \boldsymbol{f}_j), \tag{1}$$

where w_{ij} denotes the weights for aggregation and the size of $\mathcal{N}(\boldsymbol{I}_i)$ is k. Then the aggregated feature $\Delta\boldsymbol{f}_i$ is fed into a task network $\mathcal{N}_{task}$ for object detection:

$$\boldsymbol{b}_i, \boldsymbol{c}_i = \mathcal{N}_{task}(\Delta\boldsymbol{f}_i), \tag{2}$$

where $\boldsymbol{b}_i, \boldsymbol{c}_i$ denote the predicted bounding boxes and their corresponding categories in the current frame $\boldsymbol{I}_i$. However, the performance and inference speed of these feature aggregation-based models are heavily influenced by k. For example, when k decreases from 31 to 3, the inference speed of FGFA [45] can increase from 5.8 FPS to 20.6 FPS (tested on a single Titan RTX GPU) while the mAP drops from 74.6% to 72.5%.

Therefore, we wonder whether it is necessary to use so many frames (e.g., 21 frames) for feature aggregation regardless of the input videos. Figure 3 compares the feature maps with and without feature aggregation on objects with slow motion speeds. We visualize the 50-th, 150-th and 250-th feature channels in Figure 3. As shown in the figure, there is little difference between the feature maps with and without aggregation. Meanwhile, we also calculate the cosine similarity scores of the whole feature map with and without aggregation, which is 0.9819. Therefore, for objects with slow motion speeds, there is not too much improvement with feature aggregation.

3.2 Vanilla Dynamic Aggregation

In the current feature aggregation based video object detectors [10, 35, 39, 45], as described in Equation 1, w_{ij} is calculated as the cosine similarity between the neighboring feature $\boldsymbol{f}_j$ and $\boldsymbol{f}_i$, which is unrelated to k. Since k will not affect the aggregation process during inference time, it is possible to update the feature aggregation module in the current methods to a dynamic version, where an adaptive number of frames is applied to eliminate feature degradation to boost the performance of video object detection.

A simple idea to achieve this is to classify the current frames into multiple categories and determine the number of frames used for aggregation based on the categories. That is how our vanilla dynamic aggregation method comes out. In detail, we make the number of frames for feature aggregation dynamic based on the input frame: For frames where objects are with low motion speeds, fewer or even no neighboring frames are taken into account for feature aggregation. On the contrary, for those frames with severe feature degradation, more or even the whole neighboring frames are used to boost the detection performance.

To make the number of frames for feature aggregation dynamic, instead of a fixed number k, we use k_v frames for aggregation, which is determined by the current input frame $\boldsymbol{I}_i$. To achieve this, we first categorize the current frame $\boldsymbol{I}_i$ into θ categories based on the motion speeds of the objects in $\boldsymbol{I}_i$, where θ is a configurable parameter. Then, we use a function $\mathcal{S}_v\left(k, \delta\right)$ to determine k_v, where δ is an integer within the range of $(0, \theta]$ to represent the category of $\boldsymbol{I}_i$. Then, the formulation of $\mathcal{S}_v\left(k, \delta\right)$ can be represented as:

$$k_v = \mathcal{S}_v\left(k, \delta\right) = \left\lceil \delta \frac{k}{\theta} \right\rceil, \tag{3}$$

where $\lceil\rceil$ denotes the ceiling function. When θ is defined, k_v will have multiple discrete choices depending on δ. For example, when θ is chosen to be 3, $\delta = 1$ represents frames where objects are with slow motion speeds, and $k_v = \lceil \frac{k}{3} \rceil$ frames are used for feature aggregation for fast inference speed. Similarly, $\delta = 3$ means the current frame contains objects which move fast and we use $k_v = k$ frames to eliminate the feature degradation for a better performance. Given the number of frames k_v for the current frame $\boldsymbol{I}_i$, we slice a subset of the neighboring $\mathcal{N}_v\left(\boldsymbol{I}_i\right)$ from the whole range $\mathcal{N}\left(\boldsymbol{I}_i\right)$ using:

$$\mathcal{N}_v\left(\boldsymbol{I}_i\right)=\mathcal{G}\left(\mathcal{N}\left(\boldsymbol{I}_i\right), k_v\right), \tag{4}$$

where $\mathcal{G}\left(\cdot, k_v\right)$ is a sampling function to select k_v neighboring frames from k total neighboring frames. Therefore, Equation 1 will be updated as:

$$\Delta \boldsymbol{f}_i^v=\sum_{\boldsymbol{I}_j \in \mathcal{N}_v\left(\boldsymbol{I}_i\right)}\left(w_{ij}^v \times \boldsymbol{f}_j\right), \tag{5}$$

where $\Delta \boldsymbol{f}_i^v, w_{ij}^v$ represent the aggregated features of frame $\boldsymbol{I}_i$ with dynamic neighborhood $\mathcal{N}_v\left(\boldsymbol{I}_i\right)$ and the corresponding weights.

During the training and inference processes, the category of the current frame $\boldsymbol{I}_i$, denoted as δ, is predicted based on the features of $\boldsymbol{I}_i$ and its neighboring frames $\mathcal{N}\left(\boldsymbol{I}_i\right)$. In detail, $\boldsymbol{f}_i$ is first concatenated with $\boldsymbol{f}_j$ and then fed into a mini-network $\mathcal{N}_{mot}^v$ to predict the category of $\boldsymbol{I}_i$, summarized as:

$$\delta=\mathcal{N}_{mot}^v\left(\texttt{cat}\left(\boldsymbol{f}_i, \boldsymbol{f}_j\right)\right), \qquad \forall \boldsymbol{I}_j \in \mathcal{N}\left(\boldsymbol{I}_i\right) \tag{6}$$

Following FGFA [45] and MEGA [3], we measure the motion speed of an object in a frame with motion IoU, denoted as s_m, using the averaged intersection-over-union (IoU) scores with its corresponding instances in the neighboring frames. Then, we divide each frame into θ classes based on s_m to generate the ground truth category δ^{gt}. For example, when θ is set to be 3, objects are classified into slow ($s_m > 0.9$), medium ($s_m \in [0.7, 0.9]$) and fast ($s_m < 0.7$) groups, respectively. Therefore, each frame is divided into 3 categories based on the motion speeds of the objects it contains. Cross entropy loss ($\mathcal{L}_{CE}$) between δ and δ^{gt} is calculated as the loss $\mathcal{L}_{mot}^v$ to optimize the network $\mathcal{N}_{mot}^v$, as:

$$\mathcal{L}_{mot}^v=\mathcal{L}_{\mathrm{CE}}\left(\delta, \delta^{gt}\right) \tag{7}$$

3.3 Deformable Dynamic Aggregation

With our proposed vanilla dynamic aggregation method, the current video object detectors' performance and inference speed can theoretically be well balanced. However, there are two issues.

The categories to determine k_v need to be predefined during the training process and are not reconfigurable at inference time. In other words, a well-trained model is not adaptive to multiple configurations. Take $\theta = 3$, which represents objects with slow, medium, and fast motion speeds, as an example. Given a frame with the motion IoU of $s_m = 0.75$, it is always categorized as medium motion speeds, which requires $\lceil \frac{2}{3} k \rceil$ frames for aggregation. If we would like to regard the frame as slow motion speeds to use fewer frames for aggregation when computational resources are limited, we need to modify the category ranges (for example, $s_m \in [0.5, 0.7]$ for the medium group) and train a new model again. It is inconvenient and unsuitable for real-world applications when θ is set to be very large or the category ranges are switched frequently.

Moreover, experiments show that the vanilla dynamic aggregation module does not perform well in detecting objects of small sizes. Most of the existing feature aggregation-based video object detectors use Faster R-CNN [27] as the baseline method without feature pyramid networks [23]. Therefore, more frames are required to enhance the feature representations of small objects during the aggregation process. However, in the vanilla dynamic aggregation module, the sizes of objects in the frames are not considered.

To solve the issues mentioned above, we extend the vanilla dynamic aggregation module to a deformable version, which is more effective and reconfigurable. Instead of classifying the input frame $\boldsymbol{I}_i$ into θ categories, we use a function σ to project the $s \in [0, 1]$ in the range of 0 and 1, where s is a score which takes both the motion IoU $s_m \in [0, 1]$ and size $s_s \in [0, 1]$ of objects in the current frame $\boldsymbol{I}_i$ into account. Therefore, Equation 3 is updated to be:

$$k_d = \mathcal{S}_d\left(k, s\right) = \lceil \sigma\left(s\right) k \rceil = \lceil \sigma\left(s_m s_s\right) k \rceil \tag{8}$$

During the inference time, we can determine k_d by selecting σ in configure files. In real-world applications, when there are enough computational resources like applications on servers, we can choose σ, which casts $s \in [0, 1]$ in the range of 0 and 1. When there are not enough resources, like applications on cellphones or servers where partial machines are under maintenance, we can reload the configure file where a new σ projects s in a new range (e.g. $[0, 0.5]$) to use fewer frames for aggregation without the need of training a new model.

Similarly, given k_d for the current frame $\boldsymbol{I}_i$, the sampled neighboring frames $\mathcal{N}_d\left(\boldsymbol{I}_i\right)$ is represented as Equation 9 and deformable dynamic aggregation process is summarized as Equation 10.

$$\mathcal{N}_d\left(\boldsymbol{I}_i\right) = \mathcal{G}\left(\mathcal{N}\left(\boldsymbol{I}_i\right), k_d\right), \tag{9}$$

$$\Delta \boldsymbol{f}_i^d = \sum_{\boldsymbol{I}_j \in \mathcal{N}_d(\boldsymbol{I}_i)} (w_{ij}^d \times \boldsymbol{f}_j), \tag{10}$$

where $\Delta f_i^d, w_{ij}^d$ represents the enhanced features of frame $\boldsymbol{I}_i$ with dynamic neighborhood $\mathcal{N}_d\left(\boldsymbol{I}_i\right)$ generated from the deformable dynamic aggregation module and the corresponding weights.

During the training and inference processes, we use mini-networks $\mathcal{N}_{mot}^d$ and $\mathcal{N}_{size}$ to estimate the averaged motion IoU s_m and size s_s of objects in the current frame I_i, respectively. Similar to Equation 6, the above process can be represented as:

$$\begin{aligned} s_m &= \mathcal{N}_{mot}^d\left(\texttt{cat}\left(\boldsymbol{f}_i, \boldsymbol{f}_j\right)\right), \qquad \forall \boldsymbol{I}_j \in \mathcal{N}\left(\boldsymbol{I}_i\right) \\ s_s &= \mathcal{N}_{size}\left(\boldsymbol{f}_i\right) \end{aligned} \tag{11}$$

For each frame $\boldsymbol{I}_i$, we calculate the averaged bounding box area of the objects it contains as the ground truth s_s^{gt} for $\mathcal{N}_{size}$. Motion IoU ground truths s_m^{gt} are measured with the same pipeline as FGFA [45] and the vanilla dynamic aggregation module. Then, mean square error loss $\mathcal{L}_{\mathrm{MSE}}$ is applied to optimize $\mathcal{N}^d_{mot}$ and $\mathcal{N}_{size}$ as:

$$\begin{aligned} \mathcal{L}^d_{mot} &= \mathcal{L}_{\mathrm{MSE}}\left(s_m, s^{gt}_m\right) \\ \mathcal{L}_{size} &= \mathcal{L}_{\mathrm{MSE}}\left(s_s, s^{gt}_s\right) \end{aligned} \tag{12}$$

3.4 Inplace Distillation Loss

Our proposed method aims to balance the inference speed and performance of the existing feature aggregation-based video object detectors. Therefore, when it comes to the situation that k_v (k_d) is small, we would like features $\Delta\boldsymbol{f}^v_i(\Delta\boldsymbol{f}^d_i)$ aggregated with $k_v(k_d)$ frames similar to $\Delta\boldsymbol{f}_i$ aggregated with k frames. Here we borrow the idea from knowledge distillation that the performance of student models can be boosted when trained with the soft predictions of teacher models.

In our method, we treat the full model with k frames for aggregation as the teacher network and those with fewer frames $k_v(k_d) < k$ as the student models. We add an extra $\mathcal{L}_{dst}$ during the training process to ensure the features aggregated with fewer neighboring frames can perform similarly to those aggregated with the whole neighboring frames, so that the detection accuracy of objects with small/medium sizes can be improved. Here we use deformable dynamic aggregation as an example. We calculate the mean square loss ($\mathcal{L}_{\mathrm{MSE}}$) between $\Delta\boldsymbol{f}_i$ and $\Delta\boldsymbol{f}^d_i$ as:

$$\mathcal{L}_{dst} = \mathcal{L}_{\mathrm{MSE}}\left(\Delta\boldsymbol{f}_i, \Delta\boldsymbol{f}^d_i\right) \tag{13}$$

Inplace distillation loss is only applied during the training process; thus, it will not affect the inference speed.

4 Experiments

4.1 Experiment Setup.

For mini-network $\mathcal{N}^v_{mot}$ and $\mathcal{N}^d_{mot}$, a one-layer convolutional layer is used to fuse the concatenated features. Then a global average pooling operation is applied to reduce the spatial and temporal resolutions. Next, the pooled feature is fed into a 2-layer MLP for classification ($\mathcal{N}^v_{mot}$) or regression ($\mathcal{N}^d_{mot}$). The object size estimation network $\mathcal{N}_{size}$ has the same architecture as $\mathcal{N}^d_{mot}$ except that the input is $\boldsymbol{f}_i$ rather than the concatenation of $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$. For vanilla dynamic aggregation, θ is set to be 3 unless otherwise stated.

We evaluate our proposed methods on the ImageNet VID benchmark [28] as the recent state-of-the-art video object detection models [3, 7, 39, 45]. Following the widely used protocols in [3, 39, 45], we train our model on a combination of ImageNet VID and DET datasets. We implement our method mainly based on mmtracking [4][2]. The whole network is trained on 8 Tesla A100 GPUs. During the inference process, 30 neighboring frames are used for feature aggregation.

[2] There are around 2% mAP fluctuations in performance, and we take the mean after running 5 experiments.

Table 1. Performance comparison with the recent state-of-the-art video object detection approaches on ImageNet VID validation set.

	Methods	FPS	mAP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
ResNet-50	FGFA [45]	5.8	46.7	74.3	51.5	5.7	21.8	52.9
	FGFA + Vanilla DA	7.9	46.4	73.9	51.1	5.3	20.8	52.5
	FGFA + Deformable DA	7.6	46.6	74.1	51.2	6.5	21.6	52.7
	SELSA [39]	5.0	48.1	77.9	52.8	8.3	26.2	54.3
	SELSA + Vanilla DA	9.4	47.2	76.5	51.1	7.6	25.8	52.9
	SELSA + Deformable DA	8.8	47.9	77.5	52.4	8.7	26.1	53.5
	Temporal ROI Align [10]	1.5	48.1	79.0	52.1	7.0	26.2	54.1
	Temporal ROI Align + Vanilla DA	3.9	46.9	77.8	51.0	6.4	25.7	52.6
	Temporal ROI Align + Deformable DA	3.5	47.8	78.8	51.7	7.2	25.9	53.5
ResNet-101	FGFA	5.1	50.2	77.6	56.1	7.3	24.0	56.3
	FGFA + Vanilla DA	7.5	49.7	77.2	54.9	6.9	24.1	56.1
	FGFA + Deformable DA	7.1	50.1	77.5	55.8	7.9	23.8	56.1
	SELSA [39]	4.5	52.1	81.3	57.4	9.0	28.1	58.1
	SELSA + Vanilla DA	8.5	51.2	80.0	57.0	7.8	26.8	57.3
	SELSA + Deformable DA	8.0	52.0	81.0	56.8	9.1	27.9	57.8
	Temporal ROI Align [10]	1.2	51.3	82.4	56.1	10.4	28.7	56.9
	Temporal ROI Align + Vanilla DA	3.6	50.4	81.8	55.3	9.3	27.5	55.1
	Temporal ROI Align + Deformable DA	3.3	50.9	82.0	55.6	10.5	29.5	56.3

4.2 Main Results

In this section, we conduct experiments on vanilla, and deformable feature aggregation with the current video object detectors on the ImageNet VID benchmark [28]. We compare state-of-the-art feature aggregation-based video object detectors integrated with our proposed methods. The results are summarized in Table 1. For local aggregation methods like FGFA [45], our proposed dynamic aggregation can significantly improve the inference speeds while maintaining comparable performance like mAP and $AP_{0.5}$. We argue that this is because FGFA aggregates feature with local temporal neighboring frames, which share much redundant information. Therefore, removing those redundancies during the inference process will not affect the final predictions much, especially when the objects are at slow motion speeds.

For global aggregation methods like SELSA [39], and Temporal ROI Align [10], our proposed methods can still improve the inference speeds by a large margin yet at the sacrifice of performance like $AP_{0.75}$. We argue that this is because global aggregation methods select features with similar representations for aggregation, and removing several frames during aggregation may have a harmful effect on precisely localizing the bounding boxes, considering $AP_{0.75}$ drops more compared with $AP_{0.5}$. Meanwhile, we notice that compared with vanilla dynamic aggregation, the deformable version has much better performance (even better than the original model) on small object detection when

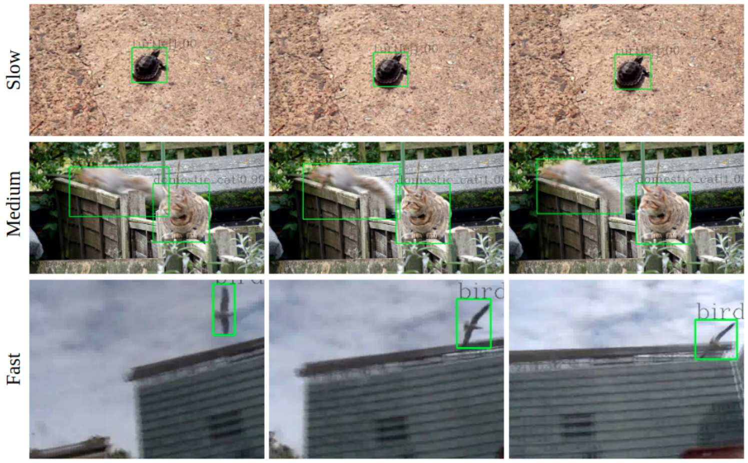


Fig. 4. Examples of FGFA [45] integrated with deformable dynamic aggregation on detecting objects with different motion speeds in the video.

taking the object sizes into account, which validates the effectiveness of the proposed modules. We argue that our methods can adaptively select the frames for aggregation, which provide adequate but not redundant information for object detection. Figure 4 shows several examples of video object detection results integrated with deformable dynamic aggregation. From the figure, our proposed methods can precisely predict the bounding boxes and categories of objects in each video frame.

4.3 Model Analysis

In this section, we conduct extensive ablation study experiments to analyze the structures and parameters of our proposed modules. By default, we use FGFA [45] with the backbone of ResNet-50 [15] as the model to conduct experiments unless otherwise stated. In this section, we mainly analyze the proposed deformable dynamic aggregation module.

Analysis of sampling function $\mathcal{G}$. We conduct experiments with deformable feature aggregation on the choices of sampling function as Table 3. "Nearest" and "Furthest" represent choosing the closest and furthest k_d frames for aggregation, while "Bin" means binning the k frames into k_d buckets and sample 1 frame from each bucket. For example, suppose the current frame is the 11^{th} of a video with 21 frames and $k_d = 7$, Table 2 shows the comparison of selected frames with different sampling functions. Besides the three sampling functions mentioned above, we also compare with random sampling results. From Table 2, "Nearest" sampling has the best performance compared with the other methods, while "Bin" sampling has a comparable result. "Furthest" and "Random" sampling

Table 2. Comparison of video object detection results with different sampling functions $\mathcal{G}$ on FGFA [45] with ResNet-50 [15] as the backbone.

Method	Nearest	Furthest	Bin
Selected Frames	8, 9, 10, 11, 12, 13, 14	1, 2, 3, 11, 19, 20, 21	2, 5, 8, 11, 14, 17, 20

Table 3. Example of selected frames with different sampling function $\mathcal{G}$.

Method	mAP	$\text{AP}_{0.5}$	$\text{AP}_{0.75}$	AP_s	AP_m	AP_l
Nearest	47.0	74.5	51.2	6.5	21.6	53.3
Furthest	45.5	72.5	50.1	5.6	20.8	51.7
Random	45.9	73.7	50.6	5.8	21.0	52.3
Bin	46.8	74.4	51.0	6.4	21.4	53.2

Table 4. Comparison of video object detection results with different mapping functions σ on FGFA [45] with ResNet-50 [15] as the backbone.

Function	FPS	mAP	$\text{AP}_{0.5}$	$\text{AP}_{0.75}$	AP_s	AP_m	AP_l
Linear ($y = 1 - x$)	7.6	46.6	74.1	51.2	6.5	21.6	52.7
Sqrt ($y = 1 - \sqrt{x}$)	7.9	46.4	74.0	51.0	6.1	21.3	52.6
Quadratic ($y = 1 - x^2$)	7.4	46.7	74.3	51.4	6.7	21.5	52.7
Learnable ($y = \texttt{MLP}(x)$)	7.1	46.9	74.4	51.5	6.9	21.9	53.1

methods have a poor performance, and we argue that this is because there are not enough effective and informative frames for aggregation when using these two strategies.

Analysis of mapping function $\boldsymbol{\sigma}$. We analyze the choice of mapping function σ in our proposed deformable feature aggregation as Table 4. We compare four different mapping functions, namely, linear, square root, quadratic and learnable function by retraining the models with the corresponding σ. From Table 4, compared with linear function, when choosing learnable networks as mapping function, the performance is the best at the sacrifice of inference speed. Square root and quadratic functions can balance the inference speed and accuracy by mapping s into different distributions.

Comparison with knowledge distillation. We also conduct experiments to compare with knowledge distillation results. We notice that FGFA aggregated with 15 frames have a similar inference speed as our proposed methods. Therefore, we use an FGFA aggregated with 15 frames to distill the knowledge from an FGFA aggregated with 30 frames and compare the results with our proposed method in Table 5. From the table, the distillation-only method is not as good as our proposed methods despite similar inference speed. Also, the model will perform worse if trained without inplace distillation loss.

Table 5. Comparison between FGFA distilled from a model aggregated with more frames and the proposed method. † means models without inplace distillation loss.

Method	FPS	mAP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
FGFA (15 frames)	7.7	46.2	73.7	50.5	5.8	20.8	52.0
FGFA (15 frames) + Distill	7.7	46.4	73.9	50.7	5.9	21.0	52.3
FGFA (30 frames)	5.8	46.7	74.3	51.5	5.7	21.8	52.9
FGFA (30 frames)+ Ours†	7.6	46.5	73.9	50.9	5.6	20.9	52.5
FGFA (30 frames)+ Ours	7.6	46.6	74.1	51.2	6.5	21.6	52.7

Table 6. Comparison between FGFA integrated with/without our proposed methods on video object detection on objects with different motion speeds.

Method	FPS	mAP	AP_{50}	AP_{slow}	AP_{medium}	AP_{fast}
FGFA	5.8	46.8	74.3	83.8	72.2	50.5
FGFA + Ours	7.6	46.5	74.2	83.7	71.8	50.3

Analysis of motion speeds. Besides object sizes, we also compare experimental results to analyze the effects on object motion speeds. Following MEGA [3], we categorize objects into slow, medium, and fast groups and calculate their corresponding accuracy as Table 6. The table shows that the performance on objects with slow motion speeds drops a little when integrated with a deformable dynamic aggregation module. However, detection accuracy on objects with medium motion speeds decreases by 0.4%. We argue that this is because situations like occlusion or rare positions are always treated as objects with medium motion speeds, and a few frames are sampled for aggregation, which are not enough to handle those cases.

5 Conclusion

Existing feature aggregation-based video object detectors usually apply a fixed number of frames to enhance objects' representations and boost performance. Therefore, the performance and inference speed are heavily influenced by the number of frames used for aggregation. In this paper, we aim to perform dynamic aggregation to the current methods to balance the performance and inference speed. We first propose vanilla dynamic aggregation and then extend to a deformable version which can adaptively and reconfigurably select frames used for feature enhancement according to the input frames. Furthermore, we introduce the inplace distillation loss to boost the performance of frames not fully aggregated. Extensive experiments on the ImageNet VID benchmark validate the effectiveness and efficiency of our proposed methods. We hope our approaches can bring some ideas to the efficient video object detection field.

References

1. Cai, Z., Vasconcelos, N.: Cascade r-cnn: High quality object detection and instance segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence p. 1–1 (2019). https://doi.org/10.1109/tpami.2019.2956516, `http://dx.doi.org/10.1109/tpami.2019.2956516`
2. Chen, K., Wang, J., Yang, S., Zhang, X., Xiong, Y., Loy, C.C., Lin, D.: 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. Chen, Y., Cao, Y., Hu, H., Wang, L.: Memory enhanced global-local aggregation for video object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10337–10346 (2020)
4. Contributors, M.: MMTracking: OpenMMLab video perception toolbox and benchmark. `https://github.com/open-mmlab/mmtracking` (2020)
5. Cui, Y., Yan, L., Cao, Z., Liu, D.: Tf-blender: Temporal feature blender for video object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8138–8147 (2021)
6. Cui, Y., Yang, L., Liu, D.: Dynamic proposals for efficient object detection. arXiv preprint arXiv:2207.05252 (2022)
7. Deng, J., Pan, Y., Yao, T., Zhou, W., Li, H., Mei, T.: Relation distillation networks for video object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7023–7032 (2019)
8. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: Centernet: Keypoint triplets for object detection. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6569–6578 (2019)
9. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision (2015)
10. Gong, T., Chen, K., Wang, X., Chu, Q., Zhu, F., Lin, D., Yu, N., Feng, H.: Temporal roi align for video object recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 1442–1450 (2021)
11. Han, L., Wang, P., Yin, Z., Wang, F., Li, H.: Exploiting better feature aggregation for video object detection. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 1469–1477 (2020)
12. Han, L., Wang, P., Yin, Z., Wang, F., Li, H.: Class-aware feature aggregation network for video object detection. IEEE Transactions on Circuits and Systems for Video Technology (2021)
13. Han, W., Khorrami, P., Paine, T.L., Ramachandran, P., Babaeizadeh, M., Shi, H., Li, J., Yan, S., Huang, T.S.: Seq-nms for video object detection. arXiv preprint arXiv:1602.08465 (2016)
14. 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)
15. 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)
16. He, L., Zhou, Q., Li, X., Niu, L., Cheng, G., Li, X., Liu, W., Tong, Y., Ma, L., Zhang, L.: End-to-end video object detection with spatial-temporal transformers. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 1507–1516 (2021)
17. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

18. Jiang, Z., Gao, P., Guo, C., Zhang, Q., Xiang, S., Pan, C.: Video object detection with locally-weighted deformable neighbors. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 8529–8536 (2019)
19. Jiang, Z., Liu, Y., Yang, C., Liu, J., Gao, P., Zhang, Q., Xiang, S., Pan, C.: Learning where to focus for efficient video object detection. In: European conference on computer vision. pp. 18–34. Springer (2020)
20. Kang, K., Li, H., Yan, J., Zeng, X., Yang, B., Xiao, T., Zhang, C., Wang, Z., Wang, R., Wang, X., et al.: T-cnn: Tubelets with convolutional neural networks for object detection from videos. IEEE Transactions on Circuits and Systems for Video Technology **28**(10), 2896–2907 (2017)
21. Kang, K., Ouyang, W., Li, H., Wang, X.: Object detection from video tubelets with convolutional neural networks. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Jun 2016). https://doi.org/10.1109/cvpr.2016.95, `http://dx.doi.org/10.1109/CVPR.2016.95`
22. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: 15th European Conference on Computer Vision, ECCV 2018. pp. 765–781. Springer Verlag (2018)
23. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
24. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.J., Fei-Fei, L., Yuille, A., Huang, J., Murphy, K.: Progressive neural architecture search. In: Proceedings of the European conference on computer vision (ECCV). pp. 19–34 (2018)
25. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
26. 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)
27. 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)
28. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
29. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)
30. Shvets, M., Liu, W., Berg, A.C.: Leveraging long-range temporal relationships between proposals for video object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9756–9764 (2019)
31. Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., Tomizuka, M., Li, L., Yuan, Z., Wang, C., et al.: Sparse r-cnn: End-to-end object detection with learnable proposals. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14454–14463 (2021)
32. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
33. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: Fully convolutional one-stage object detection. In: Proc. Int. Conf. Computer Vision (ICCV) (2019)

34. Wang, S., Lu, H., Deng, Z.: Fast object detection in compressed video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7104–7113 (2019)
35. Wang, S., Zhou, Y., Yan, J., Deng, Z.: Fully motion-aware network for video object detection. In: Proceedings of the European conference on computer vision (ECCV). pp. 542–557 (2018)
36. Wang, S., Zhou, Y., Yan, J., Deng, Z.: Fully motion-aware network for video object detection. In: Proceedings of the European Conference on Computer Vision (ECCV) (September 2018)
37. Wang, X., Huang, Z., Liao, B., Huang, L., Gong, Y., Huang, C.: Real-time and accurate object detection in compressed video by long short-term feature aggregation. Computer Vision and Image Understanding **206**, 103188 (2021)
38. Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., Keutzer, K.: Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10734–10742 (2019)
39. Wu, H., Chen, Y., Wang, N., Zhang, Z.: Sequence level semantics aggregation for video object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9217–9225 (2019)
40. Xu, Z., Hrustic, E., Vivet, D.: Centernet heatmap propagation for real-time video object detection. In: ECCV (2020)
41. Yao, C.H., Fang, C., Shen, X., Wan, Y., Yang, M.H.: Video object detection via object-level temporal aggregation. In: European conference on computer vision. pp. 160–177. Springer (2020)
42. Yu, J., Huang, T.: Autoslim: Towards one-shot architecture search for channel numbers (2019)
43. Yu, J., Huang, T.: Universally slimmable networks and improved training techniques (2019)
44. Yu, J., Yang, L., Xu, N., Yang, J., Huang, T.: Slimmable neural networks (2018)
45. 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)
46. 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)