Global Memory and Local Continuity for Video Object Detection

Liang Han, Zhaozheng Yin

***Abstract*—To deal with the challenges in video object detection (VOD), such as occlusion and motion blur, many state-of-the-art video object detectors adopt a feature aggregation module to encode the long-range contextual information to support the current frame. The main drawbacks of these detectors are three-folds: first, the frame-wise detection slows down the detection speed; second, the frame-wise detection usually ignores the local continuity of the objects in a video, resulting in temporal inconsistent detection; third, the feature aggregation module usually encodes temporal features either from a local video clip or a single video, without exploiting the features in other videos. In this work, we develop an online VOD algorithm, aiming at a balanced high-speed and high-accuracy, by exploiting the global memory and local continuity. In the algorithm, an effective and efficient global memory bank (GMB) is designed to deposit and update object class features, which enables us to exploit the support features in other videos to enhance object features in the current video frames. Besides, to further speed up the detection, we design an object tracker to perform object detection for non-key frames based on the detection results of the key frame by leveraging the local continuity property of the video. Considering the trade-off between detection accuracy and speed, the proposed framework achieves superior performance on the ImageNet VID dataset. Source codes will be released on *https://github.com/LiangHann/Global-Memory-and-Local-Continuity-for-Video-Object-Detection*.**

***Index Terms*—video object detection, global memory bank, feature aggregation, local continuity, object tracker**

I. Introduction

DUE to the advancement of deep neural networks, significant progress has been achieved on object detection in still images [1], [2], [3], [4], [5], [6], [7], [8], [9]. With the development of storage and communication technologies, video is becoming a popular media to convey more abundant information, and video-based analysis becomes pervasive nowadays, such as action recognition [10], [11], sematic segmentation [12], [13], object tracking [14], [15] and detection [16], [17], etc. Among them, video object detection (VOD), a fundamental task for numerous downstream applications such as robotics and autonomous driving, has revealed increasing importance. However, due to the deteriorated appearance caused by occlusion, motion blur, out-of-focus cameras, or rare object poses in captured videos, directly applying those image-based object detectors on a frame-by-frame basis to a video often leads to unsatisfactory performance. The temporal information inherently encoded in videos, as the rich cues of continuous object movement, can be leveraged to boost the performance of VOD [18], [19], [20].

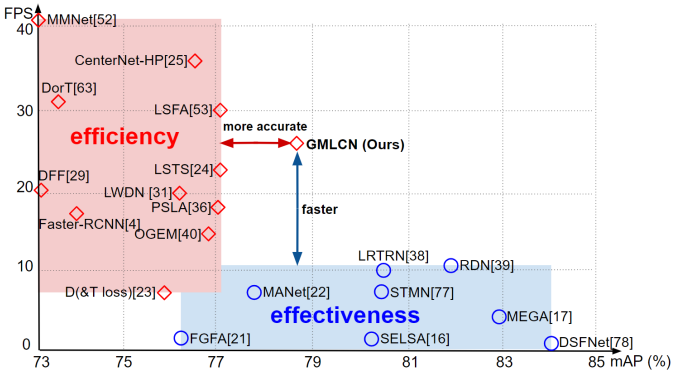


Fig. 1: Comparison with state-of-the-arts. ◇ indicates online detectors (detection using the past and current frames without using the future frame or post-processing), while ○ denotes offline detectors.

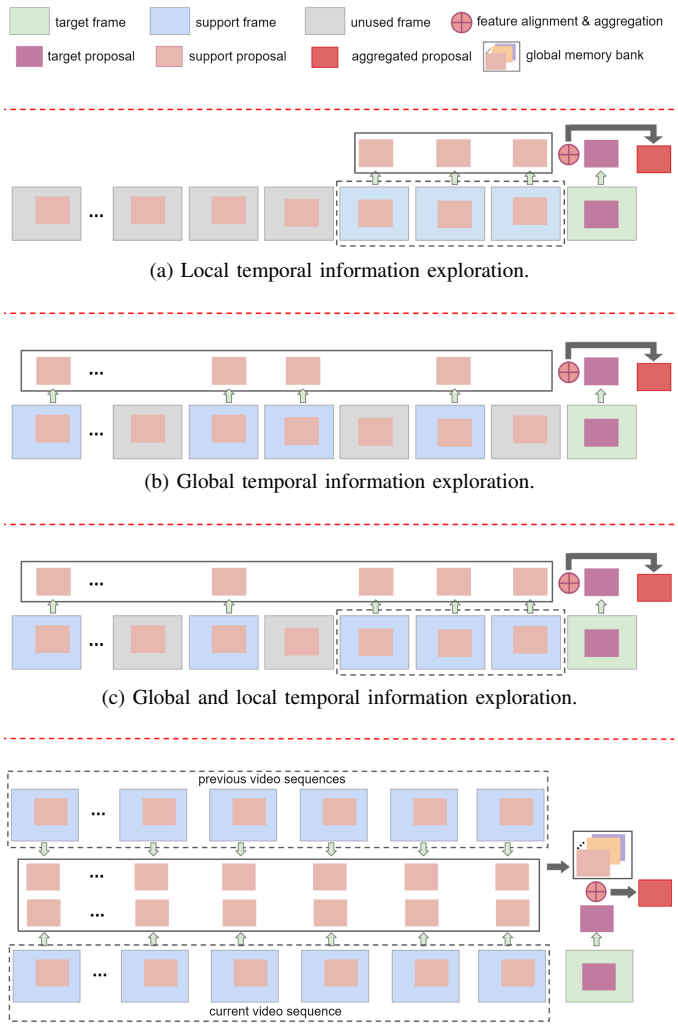
There are two metrics to consider when developing a VOD algorithm: accuracy and speed. Many recent works have been proposed to pursue the high accuracy such as FGFA [21], MANet [22], SELSA [16], and MEGA [17]. However, these algorithms have low detection speed due to their complexities. Besides, they are not online algorithms since they use forward-backward information in the temporal domain or some post processing techniques. On the other hand, faster algorithms have attracted attentions such as D&T [23], Faster-RCNN [4], LSTS [24], and CenterNet-HP [25], but they generally have lower accuracy. Our goal in this paper is to develop an *online* framework (detection using the past and current frames without using the future frame or post processing, note that here the 'online' framework is also called 'causal' framework in signal processing community), aiming at a balanced performance of high-speed and high-accuracy, as shown in Figure 1.

Our community has been actively improving the VOD in two directions. The first one aims at speeding up detection by relying on temporal information for feature propagation to avoid dense feature extraction [26], [25], [27], [28], [29], [30], [24], [31]. Unfortunately, compared with the dense feature directly extracted with networks, the quality of the propagated feature is usually degraded, which leads to less accurate detection. The other direction tries to boost the detection accuracy by aggregating the features of adjacent

Manuscript received XX XX, 2021; revised XX XX, 2022. Corresponding author: Zhaozheng Yin. Liang Han and Zhaozheng Yin have been supported by National Science Foundation via National Robotics Initiative grant CMMI-1954548 and Human Technology Frontier grant ECCS-2025929.

L. Han is with the Department of Computer Science, Stony Brook University, New York, USA (e-mail: liahan@cs.stonybrook.edu).

Z. Yin is with the Department of Computer Science and Department of Biomedical Informatics, Stony Brook University, New York, USA (e-mail: zyin@cs.stonybrook.edu).



(a) Local temporal information exploration.

(b) Global temporal information exploration.

(c) Global and local temporal information exploration.

(d) Our global memory and local continuity model.

Fig. 2: Feature aggregation comparison between our model and others. Our model is able to combine all the features in the current video and previous detected videos to perform feature aggregation. Best viewed in color.

frames to improve the target frame feature, which consists of three feature aggregation strategies: (1) local temporal feature aggregation [21], [22], [32], [33], [34], [35], [36], [37] (Figure 2 (a)), which aggregates proposal features in the target frame only with the features of a few support frames sampled from a very short time range around the target frame; (2) global temporal feature aggregation [38], [16], [39], [40], [41], [42] (Figure 2 (b)), which performs the feature aggregation for the target frame with the features of some globally sampled support frames from the whole video; and (3) both global and local temporal feature aggregation [17], [24] (Figure 2 (c)). Nevertheless, when aggregating features from support frames, it is inevitable for those approaches to perform dense feature extraction from support frames, which heavily slows down the detection. Besides, there are two more drawbacks in the feature aggregation works: 1) the local continuity information in the spatio-temporal domain is not fully exploited, which may lead to temporally inconsistent detection; 2) only the current video is used for feature aggregation, while the related object information in other videos is ignored.

Motivated by the above observations ignored by previous works, we propose the Global Memory and Local Continuity Network (GMLCN) for effective and efficient VOD. Firstly, a Global Memory Bank (GMB) is designed to store object class features, which enables to exploit the support features in other videos to enhance object features in the current video frame. The GMB further relieves the costly dense support frame feature extraction in previous works, and thus greatly speeds up the detection. Secondly, instead of performing object detection in a frame-by-frame manner which is adopted by most previous works, we propose to detect objects in a video clip each time. Each video clip consists of several consequent frames, in which the first frame is regarded as key frame and our GMB-based object detector is applied onto it, while the other frames are non-key frames and an object tracker is designed to detect objects in these non-key frames based on the detection result of the key frame by leveraging the local continuity of the video. The object tracker mitigates the temporally inconsistent detection due to motion blur or partial occlusion, and also avoids the dense feature extraction for non-key frames, which, as a side product, further speeds up the detection. The main contributions of this work are as follows:

- We design the Global Memory and Local Continuity Network (GMLCN) for Video Object Detection (VOD), which can exploit the global memory and the local continuity information to boost both the detection accuracy and speed.
- A Global Memory Bank (GMB) is designed to exploit the global memory of the videos for higher detection accuracy. Instead of storing pixel- or instance-level features as some previous works did [43], [17], [40], [44], our designed GMB stores object class features, which enables to exploit support features in the current and also other videos for feature aggregation. Besides, the GMB speeds up the detection by exempting the time-consuming support frame feature extraction and reducing the storage of the memory bank.
- An object tracker is designed to track objects from key frames to non-key frames by leveraging the local continuity of the video. Different from previous works that regress an IoU score in the tracker head to estimate the quality of the bounding box regression, in our designed object tracker, we add a classification branch to directly predict whether the tracked object disappears from the current non-key frame. The object tracker mitigates the temporally inconsistent detection, and also avoids the dense feature extraction for non-key frames, which further speeds up the detection.
- Evaluated on ImageNet VID dataset (Figure 1 and Table I), our GMLCN model achieves better accuracy compared to previous methods with the same/similar speed; compared to previous methods with the same/similar accuracy, our model is much faster with online performance.

II. Related Work

In this section, we briefly review the previous papers related to our work.

A. Still Image Object Detection

There are two main branches for still image object detection: one-stage object detector and two-stage object detector. One-stage object detectors [3], [45], [46], [5], [47], [48] directly predict the bounding box of interest based on the extracted feature map from CNN, it is usually faster but generally with worse performance than the two-stage counterpart. Two stage detectors [4], [6], [49], [7], [50], [51] usually generate region proposals with Region Proposal Network (RPN) [4] first, followed by a RoI pooling or RoIAlign pooling [7] to extract proposal features, and then each extracted proposal feature is used to perform object detection by classifying a label and regressing a bounding box. In this work, we perform two-stage object detection in videos, aiming at solving both detection accuracy and speed.

B. Video Object Detection (VOD)

There are two mainstream directions for VOD. One direction is to speed up the detection. Some works [29], [28], [26], [27], [25], [30], [24], [31] leverage the information redundancy in video frames to reduce the high feature extraction cost, some [52], [53], [54] accelerate the detection by exploiting the video compressing information, while some others adopt light-weight networks, such as MobileNet [55], [56], and Bottleneck-LSTM [57], [43]. Our work also aims at boosting the detection speed but from a different way, as we design an object tracker to perform object detection on non-key frame by tracking detected objects from key frames. Besides, a designed GMB also helps reduce the computational cost drastically by avoiding extracting dense support frame features.

The other direction is to exploit the temporal information in videos to improve the detection accuracy, which consists of three sub-directions. In the first sub-direction (Figure 2(a)), works exploit the local temporal information in videos to enhance the target frame feature. Among these methods, optical flow based feature warping [58] is widely used to propagate the features across frames nearby [21], [29], [22]. However, works in this sub-direction only exploit the temporal information between frames in a very short time range, and ignore the global temporal information. Besides, the warping does not work well for occlusion cases. To address these shortcomings, many works in the second sub-direction [38], [16], [39], [41], [42], [59] adopt the attention-based relation models [60] to exploit the long-range, global temporal information (Figure 2(b)). Limited by the GPU memory and computational cost, only some support frames in the current video is randomly selected from the long range to enhance the target frame feature. Unfortunately, the local continuity of a video is neglected in these works. Works in the third sub-direction [17], [24] try to solve these problems by exploiting both the local and global temporal information (Figure 2(c)). Inevitably, these feature aggregation works have to extract dense support frame features, which heavily slows down the detection. Besides, all these works only exploit temporal information in the current video for VOD, while ignoring the related object information in other videos. To better exploit the long-range temporal information in the video to improve the object detection accuracy, the memory mechanism is also adopted in many works to store the pixel-level [43], [61], [62], or instance-level [17] (i.e., feature of proposals generated by region proposal network), or both pixel- and instance-level temporal features [40], [44] to perform feature aggregation for the target frame. However, the storage of the memory mechanism can be very huge since the number of pixels and proposals in each frame is usually very large, which increases the computation cost of feature aggregation and thus heavily slows down the detection. Besides, these memory-based methods only exploit the temporal information in the current video with the help of their proposed memory modules, while are not able to leverage the information in other videos, since pixels or proposals of different videos usually contain different objects which can pollute the target feature rather than enhancing it when performing feature aggregation. To overcome these challenges, in our work, we design a Global Memory Bank (GMB) to store object class features instead of pixel- or instance-level features. This greatly reduces the storage of the memory bank and thus accelerate the detection. Besides, the design of GMB also enables us to exploit support features in both the current video and other videos for better feature aggregation.

Recently, some researchers also try to speed up the detection while at the same time preserve a high detection accuracy. Xu et al. [25] propose to propagate the previous reliable long-term detection in the form of heatmap instead of features to boost results of upcoming video frame based on a one-stage detector. Luo et al. [63] present a strategy of predicting whether to perform object tracking or detection for each video frame. Our work falls in this direction but in a different way. To obtain high speed and accuracy trade-off, we not only leverage the detect-or-track strategy, but also improve the design of memory bank to increase the detection accuracy while also reduce time cost.

C. Video Salient Object Detection (VSOD)

Saliency is an important cue for object detection, which has been widely exploited by many works to perform VSOD. Fan et al. [64] present a baseline model for VSOD, which is equipped with a saliency-shift-aware convLSTM to efficiently capture video saliency dynamics by learning human attention-shift behavior. Wang et al. [65] propose a fully convolutional network for VSOD, with a novel data augmentation strategy to learn diverse saliency information. Besides, similar to some VOD methods which leverage the spatial and temporal information to alleviate complicated motion and complex scene challenge in videos, many works [66], [67], [68] also exploit the spatiotemporal saliency for VSOD. VSOD aims to pop out the most salient regions in each frame of a video in the form of mask, while VOD tries to localize all the objects in a frame with bounding boxes and classify them into different classes.

D. Detection with Tracking

To accelerate the detection by avoiding computing deep features and also improve the detection accuracy by exploiting temporal consistency in a video, many video object detection

works [27], [63], [69] combine a detection module and an object tracker [70], [71], [72] in their frameworks. Inspired by this idea, we also include an object tracker in our detection framework for acceleration and accuracy improvement. However, different from these previous works that regress an IoU score in the tracker head to estimate the quality of the bounding box regression, in our designed object tracker, we add a classification branch to directly predict whether the tracked object disappears from the current non-key frame.

III. Global Memory and Local Continuity Network (GMLCN)

For models to perform accurate object detection on long-term videos with complex scenes, they are expected to not only leverage the long-range, global contextual information for object detection in the current frame, but also perform continuous and smooth object detection in the local spatial-temporal domain of videos. Keeping these motivations in mind, we propose the Global Memory and Local Continuity Network (GMLCN) for Video Object Detection (VOD).

A. Framework Overview

Figure 3 depicts the framework of the proposed GMLCN. The network takes a short video clip as input which consists of a key frame and one or more neighboring non-key frames. For the key frame, a deep backbone network (e.g., ResNet-101) is adopted to extract frame feature. Then, a Region Proposal Network (RPN) is used to generate object proposals for the key frame, followed by an RoIAlign pooling operator to extract feature for each generated object proposal. A self-attention module is employed to perform feature aggregation for these extracted proposal features with the global features reading from the Global Memory Bank (GMB). Finally, two fully connected layers are attached to each aggregated proposal feature to predict the class label and bounding box of this proposal, followed by a Non-Maximum Suppression (NMS) to remove duplicates and generate the final detection results. It is worth noting that after getting the final detection result of the key frame, the aggregated proposal features and their corresponding predicted class scores will be used to update the features in GMB by a feature writing operation. For the neighboring non-key frame, a shallow backbone network is adopted to extract coarse low-level frame feature. The low-level non-key frame feature, together with the intermediate low-level feature of the key frame generated by the deep backbone, are used to train an object tracker, which will then perform object detection for the non-key frame by tracking the detected objects from key frame to non-key frame. Different from previous two-stage video object detectors, our proposed GMLCN avoids extracting support frame features when performing object detection for the key frame with the help of the designed GMB. Besides, our GMLCN only needs to extract low-level features for the non-key frames when detecting objects in these non-key frames with the proposed object tracker, which can greatly reduce the computation cost. With the designed GMB and object tracker, our GMLCN can drastically speed up the detection while at the same time pursuing high detection accuracy.

B. Global Memory Bank

To perform object detection on video frames with deteriorated object appearances caused by partial occlusion, motion blur, etc., aggregating appearance features from other video frames is a widely-used strategy. Some works also exploit the external memory mechanism [17], [40] to increase the support features. Generally, sampling more support features or increasing the external memory storage can gain better detection result. However, this will also increase the computation cost and computer memory demand, which requires more powerful machines and even worse, slows down the detection. Keeping this challenge in mind, in this subsection, we design an efficient Global Memory Bank (GMB) to store object class features instead of storing object proposal features, which not only effectively lowers the memory demand, but also speeds up the detection.

In the GMB, a class memory feature matrix $F^{memory} \in \mathbb{R}^{(C+1)\times D}$ is restored, where C is the number of object classes in the dataset, plus one class for the background, i.e., each row of this matrix represents a feature vector for a certain class (either object class or background), and D is the dimension of each class feature vector. The feature memory matrix is initialized with the detection result of the first key frame. Suppose in the 1st key frame, N_1 object proposals are generated, each with a feature vector dimension of D. Let $F^{proposal} \in \mathbb{R}^{N_1\times D}$ denote the stacked features of these N_1 proposals, and $S \in \mathbb{R}^{N_1\times(C+1)}$ be the final predicted class score vectors of these proposals by the detection network. Each row of the feature memory matrix ($F^{memory}_{(i,\cdot)}$) is initialized by weighted-summing the proposal vectors belonging to the same class, with the predicted class scores as the summation weights. Mathematically,

$$F^{memory}_{(i,\cdot)} = \frac{(S_{(\cdot,i)})^T \cdot F^{proposal}}{\sum_{j=1}^{N_1}(S_{(j,i)})^T}, \quad i \in \{1, 2, ..., C+1\} \quad (1)$$

where $(S_{(\cdot,i)})^T$ denotes the transposed vector of the i-th column of matrix S. Please note that before initializing the memory matrix, we will first process the class score matrix S with a thresholding operation (threshold value is 0.9 in our case) to only reserve the high class scores, while the low class scores are set to 0. By doing this, only the proposals with high class scores are used to generate the class feature vectors, which can make the class feature vectors more representative and discriminated with each other.

1) Feature Reading: After the class memory feature matrix in GMB is initialized with the detection result of the first video clip, a feature reading operation is designed to read these class memory features from GMB and use them to perform feature aggregation for the following key frames. In this work, a self-attention module is adopted to implement the feature reading operation, which consists of three steps: attention weight computation, memory feature alignment, and memory feature aggregation.

Attention weight computation: Suppose for the current key frame t, N_t proposals are generated in total, and the proposal features form a feature matrix $F^{proposal} \in \mathbb{R}^{N_t\times D}$ by stacking all proposals' features together, where D is the dimension of

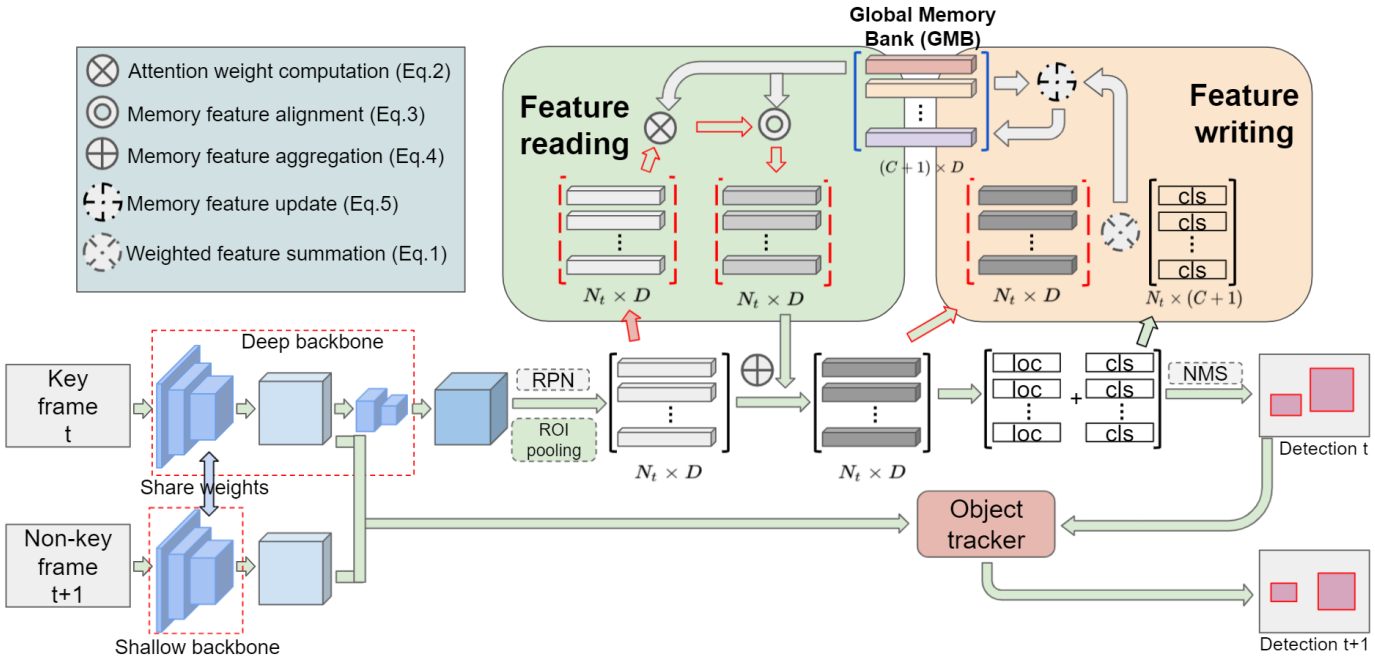


Fig. 3: Framework of the proposed GMLCN. The input is a short video clip, which consists of a key frame and one or more non-key frames. The key frame goes through a deep backbone to extract its feature, followed by a Region Proposal Network (RPN) to generate proposals and RoIAlign pooling to extract proposal features. A feature reading operation is performed to aggregate features for generated proposals with the class features in the Global Memory Bank (GMB) which consists of features for each object class and the background, and each aggregated proposal feature is used to perform the final detection, followed by a Non-maximum Suppression (NMS) to remove the duplicates. After detection, the aggregated proposal features along with their class scores are used to update the class features in GMB by a feature writing operation. The low-level features of the key frame and the non-key frame, extracted by a shallow backbone, are input to an object tracker, which detects objects for non-key frames based on the detection results of the key frame. Best viewed in color.

each proposal feature vector. The attention weights between the proposal features $F^{proposal}$ and the class memory features F^{memory} can be computed as

$$W = \phi(F^{proposal}) \cdot \psi((F^{memory})^T), W \in \mathbb{R}^{N_t \times (C+1)} \quad (2)$$

where ϕ and ψ are two embedding layers to reduce the feature dimension when computing attention weights.

Memory feature alignment: In this step, the class memory features are aligned with the computed attention weight matrix W to fit the current proposal feature

$$F^{memory}_{aligned} = \rho(W) \cdot F^{memory}, F^{memory}_{aligned} \in \mathbb{R}^{N_t \times D}, \quad (3)$$

where ρ is a row-wise softmax operation to normalize the attention weights so the aligned feature scale is unchanged.

Memory feature aggregation: In the last step, the aligned class memory features are aggregated to the current proposal features in a residual way to generate the aggregated proposal feature $\widetilde{F}^{proposal}$

$$\widetilde{F}^{proposal} = F^{proposal} + fc(F^{memory}_{aligned}), \widetilde{F}^{proposal} \in \mathbb{R}^{N_t \times D}, \quad (4)$$

where fc is a fully connected layer.

2) Feature Writing: After performing object detection for a new key frame t, the class memory features in GMB will be updated with the newly detected result. This is to guarantee that the class memory feature can contain more class feature information and thus be more robust and representative. The memory updating is implemented by a feature writing operation, which is mathematically defined as

$$F^{memory}_{new,(i,\cdot)} = \frac{F^{memory}_{old,(i,\cdot)} \cdot s_{old,(i)} + F^{memory}_{cur,(i,\cdot)} \cdot \sum_{j=1}^{N_t} S_{cur,(j,i)}}{s_{old,(i)} + \sum_{j=1}^{N_t} S_{cur,(j,i)}}$$
$$s_{new,(i)} = s_{old,(i)} + \sum_{j=1}^{N_t} S_{cur,(j,i)}, \quad i \in \{1, 2, ..., C+1\} \quad (5)$$

where F^{memory}_{new} and s_{new} are the updated class memory features and the updated vector of accumulated class scores, respectively; F^{memory}_{old} and s_{old} are the previous class memory features and the vector of accumulated class scores in GMB before updating; and F^{memory}_{cur} and S^T_{cur} are the class memory features and class score matrix computed with the new detection result (proposal features and their predicted class scores) of the current key frame t. Similar to the initialization, only the proposal features with class scores higher than a predefined threshold are used to update the GMB. Note that to update the class memory feature constantly, the class scores are also stored, along with the class memory features.

During training, the global memory bank is initialized only once (after the detection of the first key frame), but it will be updated constantly (after the detection of every new key frame) until the end of the training. This enables our GMB to update over the whole dataset, not just over a single video, which is quite different from some previous works such as [17], [40]. During inference, we use the GMB class features generated in the train process as an initialization, and writing

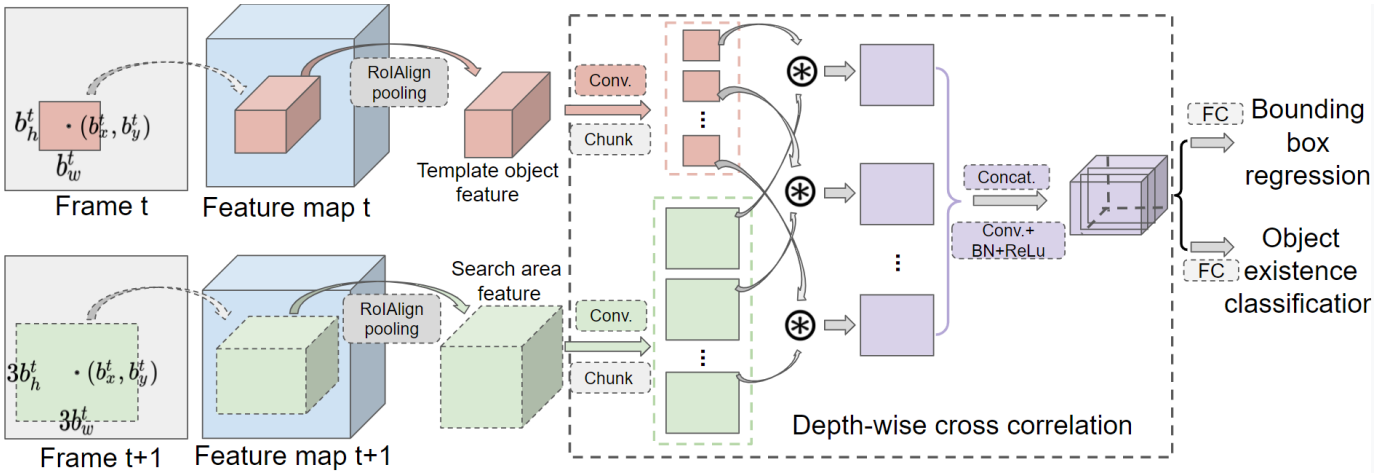


Fig. 4: Illustration of the object tracker. First, for a detected object p^t on frame t, we define its corresponding search area s^{t+1} on frame $t+1$ by fixing the box center and expanding the width and height by k times ($k = 3$ in our case). RoIAlign pooling is applied to extract regional features for the template object p^t and the search area s^{t+1}. Then, a depth-wise cross convolution module is employed to encode the object bounding box offsets by computing the channel-wise correlation between the template object feature and the search area feature, i.e., the features of each channel from the template object branch is convolved with the features of the same channel from the search area branch. Finally, the head of the tracker consisting of a classification branch and a bounding box offset regression branch is appended to regress the bounding box offsets and predict an object existence score.

Best viewed in color.

operation is performed to update the class features in GMB after performing detection for each key frame.

The benefits brought by the class memory features in GMB are multi-folds. First, the class memory bank group features at the class level, and the feature aggregation is performed from the perspective of class, which makes the support features (i.e., class memory features) more robust than the ones used in some previous works (e.g., instance feature in [40], [39], [41], [38], [16]). Second, the class memory features reduce the computation cost and machine memory demand, since the number of object classes C (in ImageNet VID dataset, $C = 30$) is usually quite smaller than the number of sampled support proposals (usually hundreds or even thousands of support proposals are sampled to perform feature aggregation in previous works). Third, the continuous update of the GMB over the whole dataset enables the detection network to aggregate features across different videos.

The detection branch for the key frame consists of a Region Proposal Network (RPN) and a detection head (DH) which predicts the class label and regresses the bounding box offset for each positive proposal. We explicitly write the objective function for RPN as follows [4]:

$$\begin{aligned} L^{RPN} &= L_{cls}^{RPN} + \lambda^{RPN} L_{loc}^{RPN} \\ &= \frac{1}{N_{cls}} \sum_i L_{bi-cls}(p_i^{RPN}, \widehat{p}_i^{RPN}) + \\ &\quad \lambda^{RPN} \frac{1}{N_{reg}} \sum_i p_i L_{loc}(\Delta_i^{RPN} - \widehat{\Delta}_i^{RPN}) \end{aligned} \tag{6}$$

In this equation, i is the index of an anchor in a mini-batch, L_{bi-cls} is the binary cross-entropy loss over two classes (object $vs.$ background), $\widehat{p}_i^{RPN}$ is the predicted probability of anchor i being an object, p_i^{RPN} is the ground-truth label which is 1 if the anchor is positive (being an object) and 0 otherwise (anchor being background), $\widehat{\Delta}_i^{RPN} = (\widehat{\Delta}_{i,x}^{RPN}, \widehat{\Delta}_{i,y}^{RPN}, \widehat{\Delta}_{i,w}^{RPN}, \widehat{\Delta}_{i,h}^{RPN})$ and $\Delta_i^{RPN} = (\Delta_{i,x}^{RPN}, \Delta_{i,y}^{RPN}, \Delta_{i,w}^{RPN}, \Delta_{i,h}^{RPN})$ are the predicted and ground-truth parameterized coordinates of the bounding box i [73] respectively, $L_{loc} = \sum_{j=x,y,w,h} L_{1,smooth}(\widehat{\Delta}_{i,j}^{RPN} - \Delta_{i,j}^{RPN})$ is the smooth L_1 loss [1].

The loss function for the detection head is [1]

$$\begin{aligned} L^{DH} &= L_{cls}^{DH} + \lambda^{DH} L_{reg}^{DH} \\ &= L_{cls}(p^{DH}, \widehat{p}^{DH}) + \\ &\quad \lambda^{DH}[c \geq 1] L_{loc}(\Delta^{DH} - \widehat{\Delta}_c^{DH}) \end{aligned} \tag{7}$$

in which $\widehat{p}^{DH} = (\widehat{p}_0^{DH}, \widehat{p}_1^{DH}, ..., \widehat{p}_C^{DH})$ and p^{DH} are the predicted discrete probability distribution over $C+1$ classes and the ground-truth class label respectively, L_{cls} is the cross-entropy loss, L_{loc} is defined the same as in Eq. 6, $\widehat{\Delta}_c^{DH}$ and Δ^{DH} are the predicted parameterized coordinates of the bounding box for a certain class c and the ground-truth parameterized coordinates of the bounding box by the detection head respectively, $[c \geq 1]$ is an indicator function which evaluates to 1 if $c \geq 1$ and 0 otherwise. By convention the catch-all background class is labeled as $c = 0$.

The MM-Net [74] also aims to distill the intrinsic characteristics of object classes by encoding images and aggregating image features into different memory slot in their designed memory module. However, compared to MM-Net, our Global Memory Bank (GMB) only stores a class representation vector in each memory slot, instead of a key-value pair which respectively denotes the memory representation and class label in the MM-Net. Besides, considering the object detection task we conducted, our GMB initializes and updates the class feature in the memory bank with proposal-level features, while

the MM-Net uses the image-level features. Moreover, different from MM-Net which encodes all the images and aggregate their features into the memory slot, our GMB initializes and updates the memory in a more picky way to guarantee the class features more representative and discriminated with each other, i.e., only the proposals that are with high predicted class scores are selected to initialize and update the memory bank, while others are ignored.

C. Object Tracker for Accelerated Detection

When performing object detection for key frames, the object detection network requires powerful feature extraction and object classification capabilities, thus the features extracted are often from deep networks, which drastically slows down the detection. Tracking could be an aid for the fast detection by predicting object bounding boxes based on the detection results in the previous frames. Instead of detecting objects from scratches, an object tracker looks for similarity between consecutive frames, and the features used for tracking are usually shallower than those for detection [69]. To further speed up the detection, we propose an object tracker to detect objects in non-key frames by predicting the object bounding box offsets from the key frame (frame t in Figure 3) to the neighboring non-key frame (frame $t+1$ in Figure 3) with frame features extracted by a shallow backbone network (e.g., feature before $conv4\_3$ of ResNet-101).

Figure 4 illustrates the process of the designed object tracker. For a detected object p^t on frame t, we denote its bounding box b^t with a 4-dimension vector $(b_x^t, b_y^t, b_w^t, b_h^t)$, where b_x^t and b_y^t are the two coordinates of the box center, b_w^t and b_h^t are the width and height of the bounding box, respectively. Then, for this object, we define its corresponding search area s^{t+1} on frame $t+1$ by fixing the box center and expanding the width and height by k times ($k=3$ in our experiments), i.e., the search area can be represented by $(b_x^t, b_y^t, 3\cdot b_w^t, 3\cdot b_h^t)$. RoIAlign pooling is applied to extract regional features for the template object p^t and the search area s^{t+1}. To keep the same pooled feature scales, the pooled feature size of the search area is also k times bigger than the feature size of template proposal. After that, a depth-wise cross convolution module [72], [69] is employed to encode the proposal location offsets by computing the channel-wise correlation between the template object feature and the search area feature, as shown in the **Depth-wise cross correlation** module of Figure 4. More specifically, the template object feature and the search area feature generated by the RoIAlign pooling operation first pass two non-shared convolutional layers to suit the tracking task, then these two adjusted features with the same number of feature channels are chunked along the feature channel dimension to generate channel-wise features. After that, the features of each channel from the template object branch is convolved with the features of the same channel from the search area branch to perform the cross-correlation channel by channel, followed by a $conv-bn-relu$ block to fuse these different channel-wise outputs. Finally, the head of the tracker consisting of a classification branch and a bounding box offset regression branch is appended behind the depth-wise correlation output. The classification branch is a fully connected layer to classify whether there exists an object in the search area, while the bounding box offset regression branch is another fully connected layer to regress the bounding box offsets. The classification is to deal with the case that an object disappears from the neighboring non-key frames due to exiting or occlusion.

The object tracker is supposed to predict the object bounding box offsets between two adjacent frames and an object existence score to indicate whether an object still exists in the neighboring non-key frame. During training, the ground truth objects are used to train the designed object tracker. The regressing target $\Delta^{t+1} = (\Delta_x^{t+1}, \Delta_y^{t+1}, \Delta_w^{t+1}, \Delta_h^{t+1})$ is determined by the ground truth bounding box of the object to be tracked $\overline{b}^t = (\overline{b}_x^t, \overline{b}_y^t, \overline{b}_w^t, \overline{b}_h^t)$ on frame t and the ground truth bounding box of the tracked object $\overline{g}^{t+1} = (\overline{g}_x^{t+1}, \overline{g}_y^{t+1}, \overline{g}_w^{t+1}, \overline{g}_h^{t+1})$ on frame $t+1$, which is defined as

$$\begin{aligned} \Delta_x^{t+1} &= \frac{\overline{g}_x^{t+1} - \overline{b}_x^t}{\overline{b}_x^t}, \quad \Delta_y^{t+1} = \frac{\overline{g}_y^{t+1} - \overline{b}_y^t}{\overline{b}_y^t}, \\ \Delta_w^{t+1} &= ln\frac{\overline{g}_w^{t+1}}{\overline{b}_w^t}, \quad \Delta_h^{t+1} = ln\frac{\overline{g}_h^{t+1}}{\overline{b}_h^t}. \end{aligned} \tag{8}$$

The ground truth label for the existence score is defined as: if an object exists in the search area of the neighboring non-key frame, the label is 1, otherwise, the label is 0. During inference, if the predicted existence score is too small (less than 0.5 in our case), the object is considered to disappear from the non-key frame. For a detected object p^t with bounding box $b^t = (b_x^t, b_y^t, b_w^t, b_h^t)$ in frame t, the predicted bounding box $b^{t+1} = (b_x^{t+1}, b_y^{t+1}, b_w^{t+1}, b_h^{t+1})$ of the tracked object p^{t+1} in frame $t+1$ can be inferred through the bounding box offsets $\widehat{\Delta}^{t+1} = (\widehat{\Delta}_x^{t+1}, \widehat{\Delta}_y^{t+1}, \widehat{\Delta}_w^{t+1}, \widehat{\Delta}_h^{t+1})$ regressed by the well-trained object tracker, which is computed as

$$\begin{aligned} b_x^{t+1} &= \widehat{\Delta}_x^{t+1}\cdot b_x^t + b_x^t, \quad b_y^{t+1} = \widehat{\Delta}_y^{t+1}\cdot b_y^t + b_y^t, \\ b_w^{t+1} &= exp(\widehat{\Delta}_w^{t+1})\cdot b_w^t, \quad b_h^{t+1} = exp(\widehat{\Delta}_h^{t+1})\cdot b_h^t. \end{aligned} \tag{9}$$

Similar to the objective loss of the RPN, the tracking loss is defined as

$$\begin{aligned} L^{track} &= L_{cls}^{track} + \lambda^{track} L_{loc}^{track} \\ &= L_{bi-cls}(p^{track}, \widehat{p}^{track}) + \\ &\quad \lambda^{track}[p_{track} \geq 0.5] L_{loc}(\Delta^{t+1} - \widehat{\Delta}^{t+1}) \end{aligned} \tag{10}$$

where $\widehat{p}^{track}$ is the predicted existence probability, p^{track} is the defined ground truth label for the existence score, L_{bi-cls} is the binary cross-entropy loss over two classes (existing $vs.$ no existing), $[p_{track} \geq 0.5]$ is an indicator function which evaluates to 1 if $p_{track} \geq 0.5$ and 0 otherwise.

IV. Experiments

The proposed video object detection network, Global Memory and Local Continuity Network (GMLCN), is evaluated in this section. First, we briefly describe the implementation details of the network, the dataset and metric used for evaluation, and the training & testing settings. Then, the proposed GMLCN is compared with state of the art to demonstrate

its superiority. After that, ablation studies are performed to evaluate the proposed global memory bank and object tracker in GMLCN. Finally, we conduct some experiments to make further analyses on the proposed modules.

A. Network Implementation

Backbone network: ResNet-101 [75] is adopted as the deep backbone network to extract features for key frames, and the subnet of ResNet-101 (i.e., network before $conv4\_3$) is used as the shallow backbone network.

Region feature extraction network: Region Proposal Network (RPN) [4] is applied on the feature extracted by $conv4$ of ResNet-101 to obtain the object proposals for the key frames. Totally 9 anchors with 3 different scales and 3 different aspect ratios are leveraged in RPN. During both training and inference stages, we first extract 6000 proposals with the highest objectness scores for each key frame, then Non-Maximum Suppression (NMS) is performed on these proposals with IoU threshold of 0.7 to finally get 300 proposals. RoIAlign pooling followed by a fully connected layer is applied on the $conv5$ feature to extract RoI feature for each object proposal.

B. Dataset and Evaluation Metric

The proposed framework is trained with an intersection of the ImageNet DET and VID datasets [76] by taking their shared 30 object classes, with the same training and validation split settings as [21]. After training, the framework is evaluated on the VID validation dataset with all 30 classes. The widely-used mean average precision (mAP)@IoU=0.5 is adopted to evaluate the detection accuracy, and the Frames Per Second (FPS) is used to measure the detection speed.

C. Training and Testing

The proposed model is trained end-to-end by simultaneously optimizing the detection loss and tracking loss. We first initialize the backbone network with the pre-trained weights on ImageNet classification, then all modules in the GMLCN are trained and optimized simultaneously. A total of 120k iterations are performed to train the model with a SGD optimizer. Batch size is set to 4 with each GPU holds one minibatch. We use an initial learning rate of $2e^{-4}$, which is divided by 10 after 80k iterations. In both training and testing, the video frames are resized to be with the shorter dimension of 600 pixels.

D. Comparison with State of the Art

To evaluate the effectiveness of our proposed model, we compare it with the state-of-the-arts, and summarize the results in Table I.

The comparison is performed under the fair circumstance that all models are with the same backbone ResNet-101. As our proposed GMLCN is an online VOD method, we first compare with some models that can perform online detection. From Table I, we can conclude that our proposed method outperforms these previous online detection models considering the accuracy and speed trade-off. Specifically,

Method	Online	mAP (%)	Runtime (FPS)	Device
Faster-RCNN [4]	✓	73.8	17.7	TESLA V100
DFF [29]	✓	73.1	20.3	TITAN X
D (& T loss) [23]	✓	75.8	7.8	TITAN X
DorT [63]	✓	73.4	31.0	TITAN X
MMNet [52]	✓	73.0	41.0	TITAN X
LWDN [31]	✓	76.3	20.0	TITAN X
OGEM [40]	✓	76.8	14.9	TITAN X
PSLA [36]	✓	77.1	18.7	TITAN X
CenterNet-HP [25]	✓	76.7	37.0	-
LSTS [24]	✓	77.2	23.0	TITAN X
LSFA [53]	✓	77.2	30.0	TITAN X
GMLCN (Ours)	✓	**78.6**	**25.2**	TESLA V100
FGFA [21]	✗	76.3	1.3	TITAN X
MANet [22]	✗	77.6	7.8	TITAN X
STSN [32]	✗	78.9	-	-
STMN [77]	✗	80.5	8.1	TITAN X
RDN [39]	✗	81.8	10.6	TITAN V
SELSA [16]	✗	80.3	1.7	TESLA V100
LRTRN [38]	✗	80.6	10.0	TITAN X
MEGA [17]	✗	82.9	4.2	TESLA V100
HVR [59]	✗	83.2	-	-
DSFNet [78]	✗	84.1	1.1	TESLA V100
D & T + Viterbi [23]	✗	79.8	5.5	TITAN X
FGFA + Seq-NMS [21]	✗	78.4	1.2	TESLA V100
MANet + Seq-NMS [22]	✗	80.3	4.6	TITAN X
STSN + Seq-NMS [32]	✗	80.4	-	-
RDN + BLR [39]	✗	83.8	-	-
CenterNet-HP + Seq-NMS [25]	✗	78.4	34	-

TABLE I: Comparison with state-of-the-arts on ImageNet VID validation set. 'X+Y' means post-processing strategy Y is employed on method X.

when comparing with the baseline detector Faster-RCNN, our GMLCN achieves a much better detection both on accuracy (+4.8% mAP) and speed (+7.5 FPS). Our model is also much better than the detection with tracking baseline, D (& T loss) [23] model, and some other state-of-the-art, e.g., PSLA [36] and LSTS [24], both on accuracy and speed. Compared with DorT [63], and the latest state-of-the-art, LSFA [53] and CenterNet-HP [25], our proposed GMLCN can achieve a better detection accuracy (+5.2mAP%, +1.4mAP%, and +1.9% mAP, respectively) while maintaining a real-time detection. Moreover, if we further accelerate the detection of our GMLCN by changing the key frame interval (i.e., Table III), we can see that our GMLCN can achieve better detection performance than DorT [63] and LSFA [53] in form of both speed and accuracy. Besides, our model obtains a better result both on accuracy and speed compared to another latest state-of-the-art, LSTS [24].

Then, we make a comparison with some offline video object detectors. When comparing with FGFA [21] (76.3% mAP) and MANet [22] (78.1% mAP), which both aggregate features based on optical flow, our method greatly improves the detection speed with a higher detection accuracy. The self-attention based methods, e.g., SELSA [16] (82.7% mAP), LRTRN [38] (81.0% mAP), MEGA [17] (82.9%)), obtain better detection accuracy, however, the detection speed of these methods is usually very slow as dense features from multiple support frames are needed to perform feature aggregation. The DSFNet [78] achieves the highest detection accuracy among all of these methods by performing feature aggregation both in the pixel level and instance level. However, this accuracy boosting strategy makes the time cost for detecting object in each frame quite high (i.e., only about 1 frame per second), which is far from real-time detection. The detection speed

	GMLCN W/O GMB	GMLCN W/O Tracker	GMLCN
mAP (%) overall	72.0	80.8	78.6
mAP (%) slow	82.4	87.5	87.1
mAP (%) medium	68.1	78.9	76.0
mAP (%) fast	46.3	60.2	54.4
FPS	28.1	15.7	25.2

TABLE II: Ablation study on proposed modules. mAP slow/medium/fast/overall represent the detection precision for objects with slow/medium/fast motion and all objects, respectively.

of RDN in [39] is improved compared to DFSNet [78]. Unfortunately, it is still not real-time.

When some post-processing strategies are adopted, most of these methods gain more or less on detection precision (mAP). However, the post-processing will further decelerate the detection speed. Moreover, with post-processing, only offline detection can be performed.

It is worth noting that in this work, we aim at designing an online and real-time video object detector, which can achieve a good balance between high speed and high accuracy. MEGA [17], which achieves the best detection precision, is neither online nor real-time (only 4.2 FPS). When compared with the state-of-the-art online and real-time detector CenterNet-HP [25], our proposed detector achieves a better detection precision ($\sim$ +2%mAP) while reserving a real-time and online inference.

In conclusion, the evaluation results on the ImageNet VID dataset (Figure 1 and Table I) show that our GMLCN model achieves better detection accuracy compared to previous methods with the same/similar speed, and achieves much faster detection speed with online performance compared to previous methods with the same/similar accuracy.

E. Ablation Study on Proposed Modules

In this subsection, we perform some ablation studies to evaluate the effectiveness of the proposed modules. The evaluation results are summarized in Table II.

First, we delete the feature aggregation with GMB from the proposed GMLCN to evaluate the designed GMB ("GMLCN W/O GMB" in Table II), i.e., the faster-RCNN is used to perform object detection for the key frames in the video, while object tracker conducts the object detection for the non-key frame. On the one hand, the overall detection accuracy drops drastically (-6.6% mAP overall). Moreover, the accuracy drops for all the objects with different motion speeds. This shows that our designed GMB is beneficial for detection accuracy improvement by enhancing the target proposal feature with feature aggregation, which makes the target proposal feature more representative and discriminative. On the other hand, the detection speed is only slightly increased (+2.9 FPS) by delecting the feature aggregation with GMB. This demonstrates that the designed GMB only introduces a little computational cost. In conclusion, the designed GMB is both effective and efficient for VOD.

Then, we evaluate the proposed object tracker by removing it from our GMLCN ("GMLCN W/O Tracker" in Table II), i.e., all the video frames are regarded as key frames. The table shows that the detection heavily slows down (-9.8 FPS) when tracking is not adopted. This is reasonable because deep features need to be extracted for all the frames if the object tracker is deleted, which is quite time-costly. Unfortunately, the object tracker harms the detection accuracy (-2.2% mAP overall). If we take a closer look at the detection accuracy, we can see that the accuracy of objects with slow motion only drops slightly (-0.4% mAP), while the accuracy of objects with fast motion is greatly influenced by the object tracker (-5.8% mAP). The reason is that objects with slow motion usually have small bounding box offsets and small pose variation between consecutive video frames, thus this local continuity makes it easy for tracking. However, for objects with fast motion, the bounding box offsets between consecutive frames are usually much larger. Moreover, fast-moving objects are more likely to have drastic shape and pose variation in neighboring frames, and are easier to be occluded. These all challenge the object tracker by disobeying the assumption of local continuity.

We further qualitatively demonstrate the effectiveness of the proposed GMB and object tracker by presenting a detection sample in Figure 5. In this example, the input video clip consists of one key frame and one non-key frame. Three consequent frames in a video are visualized. The first column and the last column shows two key frames, while the middle column is a non-key frame. The first row shows the detection result of deleting the feature aggregation with GMB from the proposed GMLCN, from which we can see that some objects are missed (the white car in the first and second frames). Besides, the classification confidence scores of the detected objects are low. The reason is that without aggregating features from GMB, the extracted features of some target objects are not very representative and informative, especially for target objects with occluded or unclear appearance. The second row is the detection result of removing object tracker from our proposed GMLCN. In this case, all the frames are regarded as key frames and the object detection is performed in a frame-by-frame manner. This row shows that without leveraging the object tracker, the detection in two consequent frames (the first and second frame, which are the key frame and non-key frame in an input video clip) is not always continuous. The last row shows the detection result of our proposed GMLCN, which shows that we can obtain continuous detection for consequent frames in an input video clip when the proposed object tracker is included in the detection framework. Besides, with the GMB included in the detection framework, detected objects can be correctly labeled with higher confidence.

F. Further Analysis of Global Memory Bank

In our experiments, the GMB is updated both in the training stage and the testing stage. To see how the GMB evolves during training and testing, we perform experiments in which the update of GMB is disabled in various stages, and the results are summarized in Figure 6.

(a) GMB is disabled: We first disable the GMB in our proposed GMLCN, i.e., we delete the GMB from the GMLCN pipeline. In this case, the detection accuracy is only 72.0%mAP (column 'GMLCN W/O GMB' in Table II).

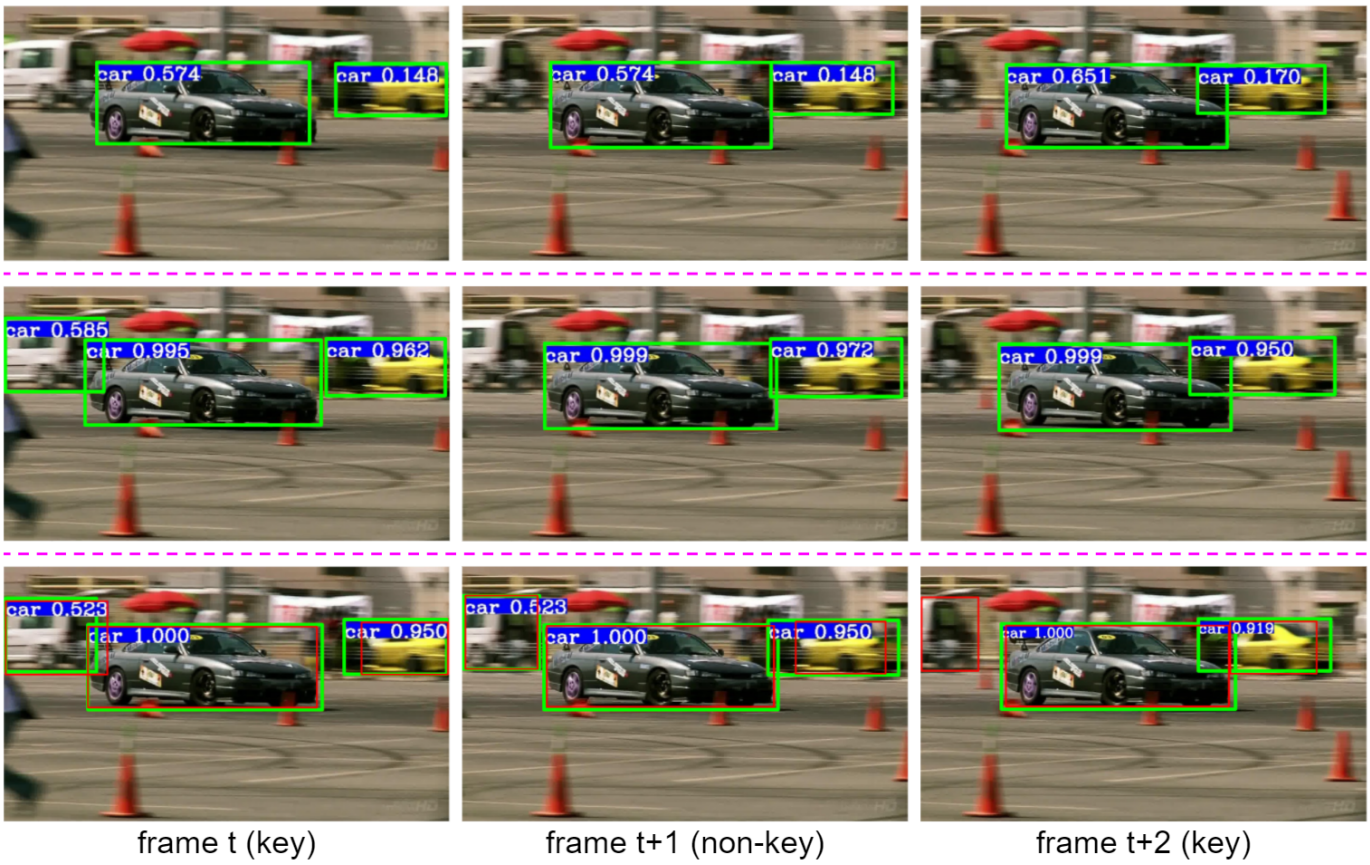


Fig. 5: Visualization for qualitative ablation study. First row: result of GMLCN W/O GMB. Second row: result of GMLCN W/O tracker. Third row: result of our proposed GMLCN. Ground truth is shown with red box in the third row.

(b) GMB is updated only in the current testing video: Then, we include the designed GMB in our GMLCN, but the update of GMB is only enabled in the current testing video. Specifically, during testing, for each video, the GMB is initialized with the detection result of the first key frame in this video and updated with the detection results of the following key frames. When detecting the next testing video, the GMB will be emptied, and initialized and updated with the information of this new video. Compared with deleting GMB from the GMLCN, the detection accuracy of only updating GMB in the current testing video increases by +3.9%mAP (75.9%mAP). This comparison result demonstrates that the object detection can benefit from leveraging the temporal information in the current video.

(c) GMB is updated in all testing videos: During testing, the GMB is initialized only once with the detection result of the first key frame in the first testing video, and continuously updated with all the following key frames in all the testing videos. Compared to initializing and updating GMB in one testing video, updating GMB in all tesing videos improves the detection accuracy by +2.4%mAP (78.3%mAP). The reason is that when we initialize GMB in each testing video and update it with the information in this single video, the stored class memory feature in GMB will not be so informative and stable because of the limited representation information from support frames, especially for the first few frames of each video. When GMB is updated with all the testing videos, the stored class memory features in GMB are much more informative and stable to conduct feature aggregation for key frames in the testing videos, except for the first few frames in the first testing videos. This comparison verifies our claim that exploiting the support object information in other videos can benefit the object detection accuracy in the current video.

(d) GMB is updated only in all training videos: In the training stage, the GMB is initialized only once with the detection result of the first key frame in the first training video, and then it is continuously updated with all the following key frames in all the training videos. During testing, the GMB built in the training stage is used for feature aggregation, while no update is performed on the trained GMB. The detection accuracy of updating GMB with all training videos is 78.4%mAP. Updating GMB with all training videos achieves comparative detection accuracy with updating GMB with all testing videos. This is because the training dataset is reasonably comprehensive (there are 3862 training videos and 555 testing videos in the dataset), the aggregated and stored class memory features in the GMB after training are already very robust.

(e) GMB is updated in all training and testing videos: Different from (d) where GMB is updated only with all training videos, in this scenario, the designed GMB is updated with all the training and testing videos. Concretely, after

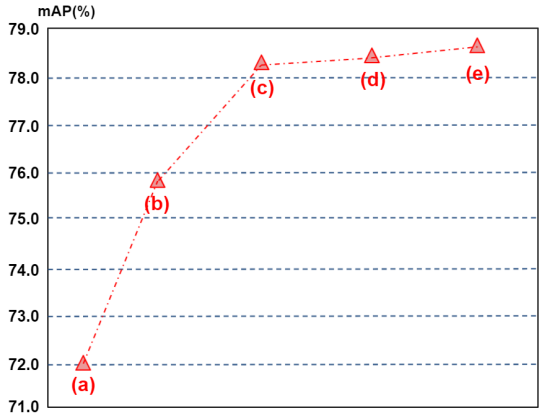


Fig. 6: Analysis on the updating strategy of GMB. (a): GMB is disabled, (b): GMB is updated only in the current testing video, (c): GMB is updated in all testing videos, (d): GMB is updated only in all training videos, (e): GMB is updated in all training and testing videos.

	$M = 1$	$M = 2$ (recursive)	$M = 2$ (direct)
mAP (%) overall	78.6	77.7	77.3
mAP (%) slow	87.1	86.8	86.7
mAP (%) medium	76.0	73.9	73.1
mAP (%) fast	54.4	49.4	46.9
FPS	25.2	32.4	32.4

TABLE III: Ablation studies on the number of non-key frames in an input video clip (M) and tracking strategies on multiple continuous frames (recursive and direct).

building the GMB in the training stage in (d), instead of fixing the class memory features in GMB, the features are updated with all the testing key frames. Compared to updating GMB only with training vidoes, further updating GMB with testing videos brings detection accuracy improvement of +0.2%mAP (78.6%mAP). This also demonstrates our claim that collecting object features from more videos can further boost the detection accuracy.

Note that we do not use the forgetting mechanism in the feature writing (Eq. 5), because we want to collect as much as possible object information in each object class feature in the GMB to make the features robust.

G. Further Analysis of Object Tracker

The input of our proposed GMLCN is a video clip consisting of a key frame and one or more non-key frames. The number of non-key frames in the input video clip can be an important hyper-parameter to affect the detection. Besides, different tracking strategies also make different influences on the detection.

First, we conduct experiments to study the influence of the number of non-key frames, M, in an input clip. From Table III we can see that when we double the number of non-key frames in an input clip, the detection speed is improved (+7.2 FPS), because averagely we save more time on feature extraction. However, the detection accuracy is negatively influenced ($\sim$ -1% mAP), especially for the objects with fast motion. The reasons are the same as we explained in the last paragraph of Sec.IV-E.

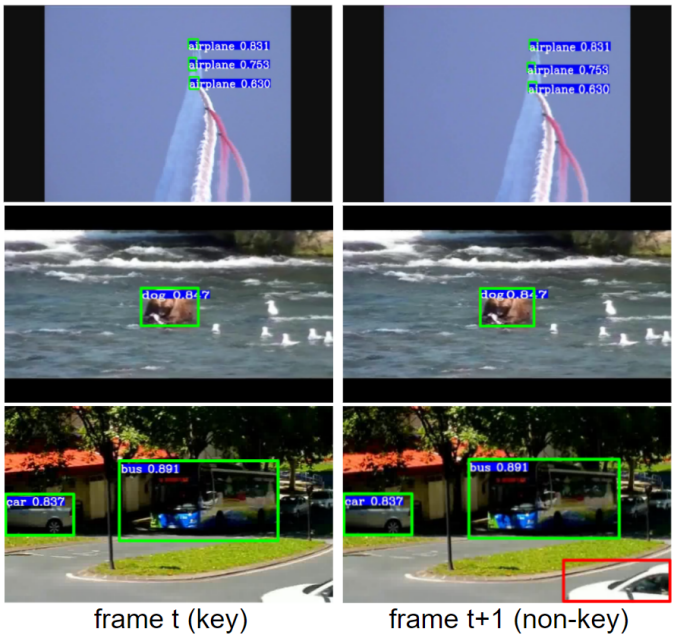


Fig. 7: Failure case analysis. First row: miss detection of small objects. Second row: wrong classification label. Third row: miss detection of newly appeared object.

Then, different tracking strategies are investigated. Suppose there are one key frame t and two non-key frames $t+1$ and $t+2$ in the input video clip. When performing the tracking on multiple continuous non-key frames with the designed object tracker, there are two different options, (1) **recursive**: we recursively track the objects from frame t to frame $t+1$, and then from frame $t+1$ to frame $t+2$; (2) **direct**: we directly track the objects from frame t to frame $t+1$ and $t+2$, separately. Table III presents the comparison between these two different tracking strategies, from which we can conclude that the recursive tracking performs better than the direct tracking on detection accuracy, and the superiority increases with the motion speed of the objects. The reason is that the bounding box offsets and pose variation of fast-moving objects between frame t and frame $t+2$ are much larger than those between frame t and frame $t+1$ and between frame $t+1$ and frame $t+2$.

H. Failure Case Analysis

There are mainly three different kinds of failure cases for our proposed GMLCN. (1) The GMLCN fails to detect some very small objects in the video frames (as shown in the first row of Fig. 7), which is a typical drawback of most RPN-based two-stage detectors. (2) Some objects with confusing appearance features are given wrong class labels, as shown in the second row of Fig. 7. The reason is that the appearance feature of the bear in these two frames is similar to the one of a dog. Without exploiting the surrounding context information, the model is with high possibility to wrongly classify the bear. (3) A newly appeared object (e.g., the car bounded with the red box in the third row of Fig. 7) in the non-key frame is miss detected. This is because the proposed GMLCN can not

detect this car in the key frame as only a very small part of this car is appeared, and when we detect objects in the following non-key frame with object tracker, the tracker is not able to track this car. This is a drawback of our designed GMLCN, which we will try to improve in our future work.

V. Conclusion

We propose a Global Memory and Local Continuity Network (GMLCN) for video object detection (VOD). The designed Global Memory Bank (GMB) in GMLCN deposits and updates object class features, enabling us to exploit the support features in other videos to enhance object features in the current video frames. Besides, to further speed up the detection, we design an object tracker to perform object detection for non-key frames based on the detection result of the key frame by leveraging the local continuity property of the video. Experiments demonstrate the efficiency and effectiveness of the proposed GMLCN for VOD both on detection accuracy and speed.

References

[1] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[2] J. Li, Y. Wei, X. Liang, J. Dong, T. Xu, J. Feng, and S. Yan, "Attentive contexts for object detection," *IEEE Transactions on Multimedia*, vol. 19, no. 5, pp. 944–954, 2016.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "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.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[6] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, pp. 379–387, 2016.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[8] C. Chen and Q. Ling, "Adaptive convolution for object detection," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3205–3217, 2019.

[9] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, pp. 213–229, Springer, 2020.

[10] F. Angelini, Z. Fu, Y. Long, L. Shao, and S. M. Naqvi, "2d pose-based real-time human action recognition with occlusion-handling," *IEEE Transactions on Multimedia*, vol. 22, no. 6, pp. 1433–1446, 2019.

[11] M. Moniruzzaman, Z. Yin, Z. H. He, R. Qin, and M. Leu, "Human action recognition by discriminative feature pooling and video segmentation attention model," *IEEE Transactions on Multimedia*, 2021.

[12] Z. Qiu, T. Yao, and T. Mei, "Learning deep spatio-temporal dependence for semantic video segmentation," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 939–949, 2017.

[13] W. Ge, X. Lu, and J. Shen, "Video object segmentation using global and instance embedding learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16836–16845, 2021.

[14] K. Chen and W. Tao, "Learning linear regression via single-convolutional layer for visual object tracking," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 86–97, 2018.

[15] Z. Wang, L. Zheng, Y. Liu, Y. Li, and S. Wang, "Towards real-time multi-object tracking," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 107–122, Springer, 2020.

[16] H. Wu, Y. Chen, N. Wang, and Z. Zhang, "Sequence level semantics aggregation for video object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9217–9225, 2019.

[17] Y. Chen, Y. Cao, H. Hu, and L. Wang, "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.

[18] Q. Xie, O. Remil, Y. Guo, M. Wang, M. Wei, and J. Wang, "Object detection and tracking under occlusion for object-level rgb-d video segmentation," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 580–592, 2017.

[19] P. Tang, C. Wang, X. Wang, W. Liu, W. Zeng, and J. Wang, "Object detection in videos by high quality object linking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 5, pp. 1272–1278, 2019.

[20] H. Sabirin and M. Kim, "Moving object detection and tracking using a spatio-temporal graph in h. 264/avc bitstreams for video surveillance," *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 657–668, 2012.

[21] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, "Flow-guided feature aggregation for video object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 408–417, 2017.

[22] S. Wang, Y. Zhou, J. Yan, and Z. Deng, "Fully motion-aware network for video object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 542–557, 2018.

[23] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3038–3046, 2017.

[24] Z. Jiang, Y. Liu, C. Yang, J. Liu, P. Gao, Q. Zhang, S. Xiang, and C. Pan, "Learning where to focus for efficient video object detection," in *European Conference on Computer Vision*, pp. 18–34, Springer, 2020.

[25] Z. Xu, E. Hrustic, and D. Vivet, "Centernet heatmap propagation for real-time video object detection," in *European Conference on Computer Vision*, pp. 220–234, Springer, 2020.

[26] K. Chen, J. Wang, S. Yang, X. Zhang, Y. Xiong, C. Change Loy, and D. Lin, "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.

[27] C.-H. Yao, C. Fang, X. Shen, Y. Wan, and M.-H. Yang, "Video object detection via object-level temporal aggregation," in *European conference on computer vision*, pp. 160–177, Springer, 2020.

[28] X. Zhu, J. Dai, L. Yuan, and Y. Wei, "Towards high performance video object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7210–7218, 2018.

[29] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2349–2358, 2017.

[30] C. Hetang, H. Qin, S. Liu, and J. Yan, "Impression network for video object detection," *arXiv preprint arXiv:1712.05896*, 2017.

[31] Z. Jiang, P. Gao, C. Guo, Q. Zhang, S. Xiang, and C. Pan, "Video object detection with locally-weighted deformable neighbors," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8529–8536, 2019.

[32] G. Bertasius, L. Torresani, and J. Shi, "Object detection in video with spatiotemporal sampling networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 331–346, 2018.

[33] K. Kang, W. Ouyang, H. Li, and X. Wang, "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.

[34] K. Kang, H. Li, T. Xiao, W. Ouyang, J. Yan, X. Liu, and X. Wang, "Object detection in videos with tubelet proposal networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 727–735, 2017.

[35] K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang, *et al.*, "T-cnn: Tubelets with convolutional neural networks for object detection from videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2896–2907, 2017.

[36] C. Guo, B. Fan, J. Gu, Q. Zhang, S. Xiang, V. Prinet, and C. Pan, "Progressive sparse local attention for video object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[37] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, and T. Mei, "Single shot video object detector," *IEEE Transactions on Multimedia*, vol. 23, pp. 846–858, 2020.

[38] M. Shvets, W. Liu, and A. C. Berg, "Leveraging long-range temporal relationships between proposals for video object detection," in *Proceedings*

of the IEEE International Conference on Computer Vision, pp. 9756–9764, 2019.

[39] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, and T. Mei, "Relation distillation networks for video object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7023–7032, 2019.

[40] H. Deng, Y. Hua, T. Song, Z. Zhang, Z. Xue, R. Ma, N. Robertson, and H. Guan, "Object guided external memory network for video object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6678–6687, 2019.

[41] L. Han, P. Wang, Z. Yin, F. Wang, and H. Li, "Exploiting better feature aggregation for video object detection," in *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1469–1477, 2020.

[42] L. Han, P. Wang, Z. Yin, F. Wang, and H. Li, "Class-aware feature aggregation network for video object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[43] M. Liu, M. Zhu, M. White, Y. Li, and D. Kalenichenko, "Looking fast and slow: Memory-guided mobile video object detection," *arXiv preprint arXiv:1903.10172*, 2019.

[44] G. Sun, Y. Hua, G. Hu, and N. Robertson, "Mamba: Multi-level aggregation via memory bank for video object detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 2620–2627, 2021.

[45] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[46] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[47] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.

[48] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636, 2019.

[49] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[50] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3588–3597, 2018.

[51] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, 2018.

[52] S. Wang, H. Lu, and Z. Deng, "Fast object detection in compressed video," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7104–7113, 2019.

[53] X. Wang, Z. Huang, B. Liao, L. Huang, Y. Gong, and C. Huang, "Real-time and accurate object detection in compressed video by long short-term feature aggregation," *Computer Vision and Image Understanding*, vol. 206, p. 103188, 2021.

[54] P. K. Kim, H. H. Kim, T. W. Kim, Y. K. Cha, *et al.*, "Compressed video stream based object detection," in *CS & IT Conference Proceedings*, vol. 9, CS & IT Conference Proceedings, 2019.

[55] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[56] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[57] M. Liu and M. Zhu, "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.

[58] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2758–2766, 2015.

[59] M. Han, Y. Wang, X. Chang, and Y. Qiao, "Mining inter-video proposal relations for video object detection," in *European Conference on Computer Vision*, pp. 431–446, Springer, 2020.

[60] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[61] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Video object segmentation using space-time memory networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9226–9235, 2019.

[62] L. Hu, P. Zhang, B. Zhang, P. Pan, Y. Xu, and R. Jin, "Learning position and target consistency for memory-based video object segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4144–4154, 2021.

[63] H. Luo, W. Xie, X. Wang, and W. Zeng, "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.

[64] D.-P. Fan, W. Wang, M.-M. Cheng, and J. Shen, "Shifting more attention to video salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8554–8564, 2019.

[65] W. Wang, J. Shen, and L. Shao, "Video salient object detection via fully convolutional networks," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 38–49, 2017.

[66] Z. Liu, X. Zhang, S. Luo, and O. Le Meur, "Superpixel-based spatiotemporal saliency detection," *IEEE transactions on circuits and systems for video technology*, vol. 24, no. 9, pp. 1522–1540, 2014.

[67] Z. Liu, J. Li, L. Ye, G. Sun, and L. Shen, "Saliency detection for unconstrained videos using superpixel-level graph and spatiotemporal propagation," *IEEE transactions on circuits and systems for video technology*, vol. 27, no. 12, pp. 2527–2542, 2016.

[68] X. Zhou, Z. Liu, C. Gong, and W. Liu, "Improving video saliency detection via localized estimation and spatiotemporal refinement," *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 2993–3007, 2018.

[69] Y. Lyu, M. Y. Yang, G. Vosselman, and G.-S. Xia, "Plug & play convolutional regression tracker for video object detection," *arXiv preprint arXiv:2003.00981*, 2020.

[70] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*, pp. 850–865, Springer, 2016.

[71] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8971–8980, 2018.

[72] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4282–4291, 2019.

[73] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "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.

[74] Q. Cai, Y. Pan, T. Yao, C. Yan, and T. Mei, "Memory matching networks for one-shot image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4080–4088, 2018.

[75] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[76] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[77] F. Xiao and Y. Jae Lee, "Video object detection with an aligned spatial-temporal memory," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 485–501, 2018.

[78] L. Lin, H. Chen, H. Zhang, J. Liang, Y. Li, Y. Shan, and H. Wang, "Dual semantic fusion network for video object detection," in *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1855–1863, 2020.