Pattern Recognition 131 (2022) 108847

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Temporal feature enhancement network with external memory for live-stream video object detection

Masato Fujitake[a,*], Akihiro Sugimoto[b]

[a] Dept. of Informatics, The Graduate University for Advanced Studies, SOKENDAI, Tokyo, 101–8430, Japan
[b] National Institute of Informatics, Tokyo 101–8430, Japan

ARTICLE INFO

Article history:
Received 8 February 2021
Revised 16 May 2022
Accepted 12 June 2022
Available online 13 June 2022

MSC:
68T10

Keywords:
Video object detection
Video analysis
Object detection

ABSTRACT

This paper proposes a method exploiting temporal context with an attention mechanism for detecting objects in real-time in a live streaming video. Video object detection is challenging and essential in practical applications such as robotics, smartphones, and surveillance cameras. Although methods have been proposed to improve the accuracy or run-time speed by exploiting temporal information, the trade-off between them tends to be ignored. We thus focus on the trade-off between accuracy and speed, and propose a method to improve the accuracy by aggregating the past information from a lightweight feature extractor with an attention mechanism. Evaluations on the UA-DETRAC and ImageNet VID datasets demonstrate our model's superior performance to state-of-the-art methods on live streaming real-time object detection.

1. Introduction

Object detection in images or video is one of the fundamental problems in computer vision. In particular, object detection in videos has been of more interest since it has a wide range of applications, including robotics, vehicles, smartphone, and surveillance systems. In practical scenarios, it is crucial to be able to cope with live stream videos in which frames flow one after another and to be able to perform object detection in real-time.

Thanks to deep convolutional neural networks (CNNs), still-image object detectors [1–6] provide reasonably high-performances in detection in recent years. However, directly applying these detectors to videos faces new challenges such as motion blur, occlusion, out-of-focus, and compression artifacts.

Therefore, video object detectors [7–11] have been actively studied to improve the detection performance by utilizing temporal consistency over the video. Some methods [9,12] have been proposed to stabilize detection by utilizing not only past and present frames but also future ones.

In the case of live streaming videos, however, future information cannot be utilized. Improving the still-image detectors with geometrical constraints [13] or the cascade strategy [14] has been proposed for frame-by-frame object detection. Such approaches improve the accuracy while they are limited to run in real-time. Some methods have been proposed to deal with real-time and live-stream video using recurrent neural network [15,16] and Flow [10] information. However, due to the emphasis on speed, the accuracy has been neglected there. There are few studies from the perspective of real-time and live streaming, which is necessary for practical applications.

To achieve accurate real-time object detection in live-stream videos, we aim to generate enriched feature maps by aggregating coarse feature maps in previous frames, which are extracted using a lightweight feature extractor, by using an attention mechanism effectively. To this end, we propose an encoder-decoder based network, Temporal Feature Enhancement Network (TFEN), that utilizes (i) spatial information from coarse spatial features and (ii) temporal information available from the live stream of video data. The encoder consists of recurrent convolutional units dealing with both spatial and temporal features. The decoder has the external memory to store feature maps generated in the past to utilize temporal information and exports densely aggregated feature maps using attention weights. In this way, TFEN enriches coarse features with spatial and temporal information so that the trade-off between accuracy and speed is considerably enhanced. We evaluate TFEN on the UA-DETRAC dataset [17] and the ImageNet VID dataset [18] using MobileNetV2 [19] as the feature extractor and Cascade R-CNN as the object detector. Experimental results demonstrate that TFEN performs in real-time while keeping comparable accuracy with state-of-the-art. Fig. 1 and Fig. 10 show some detec-

* Corresponding author.
E-mail addresses: fujitake@nii.ac.jp (M. Fujitake), sugimoto@nii.ac.jp (A. Sugimoto).

https://doi.org/10.1016/j.patcog.2022.108847
0031-3203/

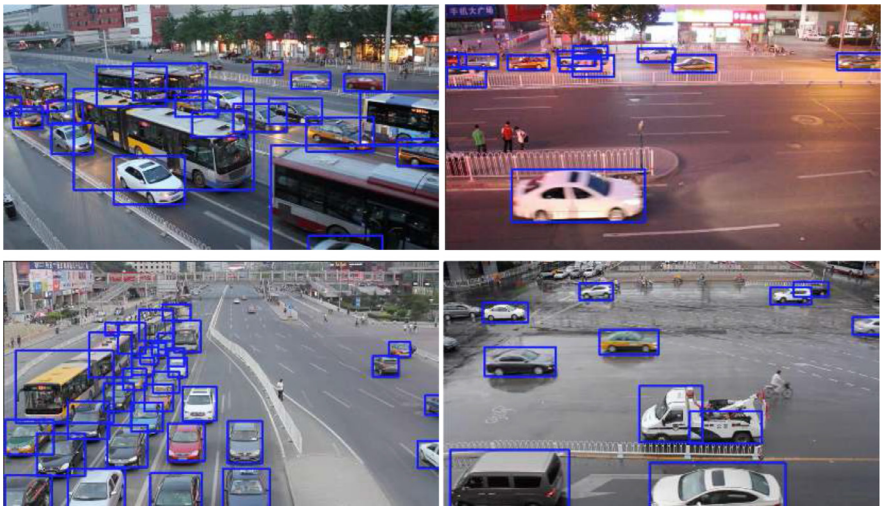


Fig. 1. Examples of the detection results by TFEN on the UA-DETRAC. A bounding box is plotted if its confidence score is larger than 0.4.

tion results by TFEN on the UA-DETRAC and the Imagenet VID. We see that TFEN successfully detects most of the objects in different scenes, especially even when heavy and partial occlusions occur, also in a rare pose.

Additionally, unlike other methods [8–10] that utilize optical flow to warp feature maps across neighboring frames, TFEN entirely relies on only appearance information from image feature extractors. The architecture of TFEN is thus simple to design. It enables us to efficiently optimize its loss function because we do not suffer from any disturbance caused by differences between appearance features and optical flow features.

The rest of this paper is organized as follows. We briefly review the related work in Section 2. Then, we present the details of our proposed method in Section 3. Section 4 discuss our experiments. Section 5 draws the conclusion. We remark that this paper extends the work reported in [20]. Our main extensions in this paper are deepening the discussion on related work and adding more experiments to demonstrate the effectiveness of TFEN thoroughly.

2. Related work

Object detection is the task of estimating the location and category of objects appearing in a given image or video. The pipeline of object detection algorithms can be roughly divided into the feature extraction stage and the detection stage. The former generates feature maps from images or videos, while the latter utilizes the features for detection. Numerous studies to improve accuracy have been reported from the above two perspectives.

From the viewpoint of feature extraction, they focus on obtaining good feature maps from still images or videos by improving backbones or aggregating features from neighboring frames. In terms of the detection stage, the research focuses on detecting objects from the obtained feature maps precisely. Some works exploit cascade the strategy, geometric information, and tracking mechanism to boost their detection accuracy.

Since this work focuses on generating “good” feature maps from a video, we briefly review works for the detection stage in Section 2.1 first, and then thoroughly survey methods for the feature extraction stage in Section 2.2.

2.1. Detection stage

Before CNNs have dominated recent progress in object detection, deformable part-based models [21] successfully detect target objects by finding object parts and combining their spatial information. Since CNNs [19,22,23] have shown their great performance in image recognition, the majority of work on still-image object detectors use CNNs. Moreover, research has been developed in two broad ways to detect objects well using feature maps from CNNs. One is the region-based two-stage approach containing proposal and classification steps, popularized by R-CNN [1] and its families [2,4,24,25]. The other is the detector based one-step approach [3,5,26], directly predicting boxes at fixed anchor positions. This approach includes SSD [3], YOLO [27], and YOLOv3 [5].

In addition to the still-image detector approach, several methods have been proposed for object detection in videos. Evolving Boxes (EB) [14] improves its region proposal network with a cascade strategy, refining object boxes. GP-FRCNN [13], on the other hand, proposes geometric proposals for Faster R-CNN [25], whereby they re-rank the geometric object proposals with an approximate geometric estimate of the scene to remove false positives. Foreground Gating and Background Refining Network (FG-BR Net) [28] incorporates the background subtraction method to ignore a non-interested region efficiently for false-positive elimination. TCNN [12,29,30] proposed tubelet proposals to improve detection accuracy considering temporary consistency within a tracklet. D&T [7] proposed to jointly learn ROI tracker along with detector where the tracker is also exploited to link the cross-frame boxes. Other methods [31] have also been proposed to boost the weak confidence of detection by using the detection results of neighboring frames as the post-processing step. Although these methods have contributed to improving accuracy, they have not achieved real-time processing.

2.2. Feature extraction stage

It is vital in the object detection task to extract “good” feature maps from images and frames. CNN-based detectors popularly use a classifier developed in image classification task such as VGG [32] and ResNet [22] to extract feature maps for detection [1,3,24]. Since a feature extractor based only on image classification may not provide detailed features, feature extractors for video object detection have also been proposed to capture the detection target more clearly [33,34]. It is sometimes difficult to obtain clear feature maps in video, in particular, live-stream video, due to blur, motion blur, and rare poses. Therefore, methods have been proposed to utilize feature maps of neighboring frames in either an offline manner or a live manner.

The offline methods deal with how to aggregate feature maps from past and future frames to refine them to improve detec-

tion accuracy. FGFA [9] exploits flow information to have pixel-to-pixel correspondence among nearby frames including the past and the future. MANet [35] further utilizes flow information to capture object motion. They have shown the importance of feature maps from nearby frames; however, they require additional networks and training costs for flow information. STSN [36], on the other hand, uses deformable convolutions across time to align the features from the adjacent frames. STMN [37] computes the correlation between neighboring frames and introduces a memory module to aggregate their features. RDN [38] and SELSA [39] focus on the object relations to align object features for feature aggregation. MEGA [40] proposed to aggregate feature maps globally in a video as well as locally. These methods achieve high detection accuracy; however, they use a computationally expensive feature extractor and future information, which impede real-time and live streaming detection.

The methods for live stream video are similar to the offline methods in that they utilize feature maps of past frames for detection in the current frame, but the main difference is whether or not they run lively in real time. DFF [8] and Flow-guided [41] accelerate detection by running the detector on sparse key-frames and using optical flow to generate the remaining feature maps. LSTM-SSD [15], Memory-guided [42] and TSSD(-OTA) [16,43] propagate feature maps across frames with convolutional recurrent neural networks. Although their models are faster than existing methods, the performance is much degraded.

In contrast to the above mentioned works, we focus on enriching feature maps from coarse ones extracted using a lightweight feature extractor to achieve both real-time speed and high-performance in detection accuracy. To this end, we address the issue by efficiently aggregating the coarse feature maps obtained from a lightweight feature extractor using the attention mechanism from the past to the present to enrich feature maps. Very recently, VOD-MT [44] was proposed to improve the feature map using an attention mechanism similar to our method. In VOD-MT, feature maps are collected for the current frame and multiple past frames with an attention mechanism frame by frame. The crucial difference of our proposed method from VOD-MT is that our method collects features by computing which frames in the past to focus on in a one-shot manner, allowing us to aggregate feature maps even faster.

3. Proposed model

3.1. Architecture

Figure 2 shows the overall architecture of our proposed TFEN at previous time $t-1$ and current time t where TFEN is connected to the feature extractor and the object detector. The skip connection is employed from the feature extractor to the object detector to retain information from the feature extractor. We denote by F_t the feature maps extracted from the feature extractor at time t, which is fed to TFEN. As the lightweight feature extractor, we employ MobileNetV2 because its computational cost is low.

Our proposed TFEN receives extracted feature maps F_t's and enriches them to pass to the object detector (see the rectangle area in pink in Fig. 2). It consists of the spatiotemporal encoder and the temporal attention decoder having the external memory.

For the frame at time t, the spatiotemporal encoder creates temporally-aware feature maps $\tilde{F}_t$ using recurrent convolutional neural networks similarly to [15]. It exploits the spatial attention module BAM [45] and ConvGRU [46]. BAM refines feature maps F_t by inferring simple spatial and channel attention, while ConvGRU uses spatiotemporal information for temporally-aware feature maps.

The temporal attention decoder, on the other hand, utilizes both the current time feature maps F_t and the external memory which stores the temporally-aware feature generated in the past m frames: $\{\tilde{F}_i\}_{t-m+1\le i\le t} := \{\tilde{F}_t, \tilde{F}_{t-1}, \ldots \tilde{F}_{t-m+1}\}$. Then it outputs densely aggregated feature maps according to the attention coefficient calculated in the decoder.

We adopt a residual learning scheme [22] to facilitate the gradient flow. The aggregated feature maps (called enhanced feature maps) are fed to the object detector. We detail the encoder and the decoder in the following subsections.

3.2. Spatiotemporal encoder

Our encoder is designed for extracting spatiotemporal information from feature maps F_t's coming from the feature extractor. As shown in Fig. 3, it consists of BAM [45] and ConvGRU [46]. BAM is a simple and effective attention module, which infers an attention map along two separate pathways: channel and spatial. ConvGRU is recurrent convolutional units that are able to deal with both spatial and temporal features.

First, for given input feature maps $F_t \in \mathbb{R}^{C\times H\times W}$ at time t, BAM infers spatial attention maps $M(F_t) \in \mathbb{R}^{C\times H\times W}$ where C, H, W denote the number of channels, the horizontal and vertical sizes of feature maps, respectively. The refined feature maps $F_t{}'$ are computed as

$$F_t{}' = F_t + F_t \otimes M(F_t), \tag{1}$$

where $\otimes$ denotes the element-wise multiplication. We introduce the compressibility value p to reduce the channels of F_t' by applying the 1×1 convolution operation to have $F_t'' \in \mathbb{R}^{pC\times H\times W}$. We then feed F_t'' into ConvGRU to store temporal information with hidden state. See [46] for details on ConvGRU. The output of ConvGRU

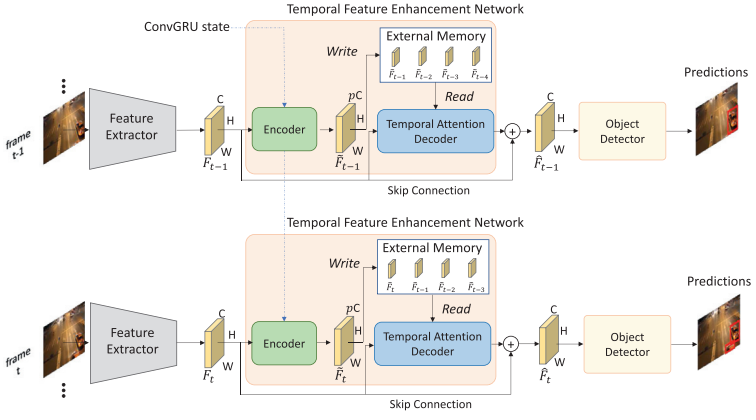


Fig. 2. Architecture of our proposed TFEN.

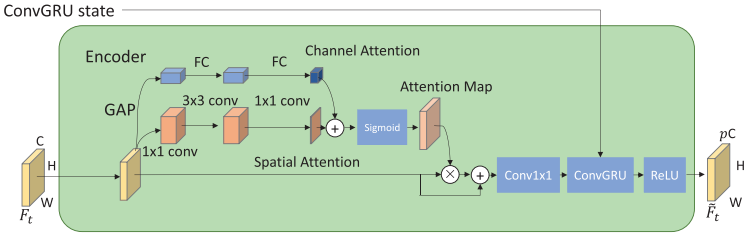


Fig. 3. Architecture of the spatiotemporal encoder.

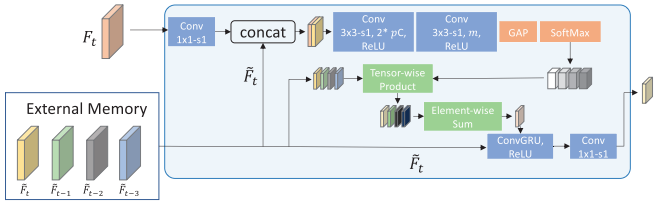


Fig. 4. Architecture of the temporal attention decoder.

is fed into the Rectified Linear Unit (ReLU) to output temporally-aware feature maps $\tilde{F}_t$ which are saved in the external memory.

3.3. Temporal attention decoder

The temporal attention decoder is the most important, and its architecture is shown in Fig. 4. Our temporal attention operation is similar to dense feature aggregation [7]. At the time t, the decoder performs multiple dense feature maps aggregation by summing all the temporally-aware feature maps $\{\tilde{F}_i\}_{t-m+1\leq i\leq t}$ based on soft attention weights[1]. The weights determine which time of temporally aware feature maps should be focused.

Soft attention weights for time are calculated through the tensor computed from $\tilde{F}_t$ and current time feature maps F_t. The 1×1 convolution is first applied to F_t to adjust its size, and then its output is concatenated with $\tilde{F}_t$ in the channel direction. Transforming operation with the stacked convolution layers and ReLU is applied, and then global average pooling (GAP) [47] and the softmax function are applied to have soft attention weights for time. The output channel of the first convolution layer and the second one depicted in Fig. 4 are pC and m, respectively.

The computed soft attention weights are used for tensor-wise products with all $\tilde{F}_i$ in $\{\tilde{F}_i\}_{t-m+1\leq i\leq t}$ stored in the external memory. Then, the element-wise summation of tensors and $\tilde{F}_t$ are fed to ConvGRU followed by ReLU. The hidden state of ConvGRU is initialized by $\tilde{F}_t$. The 1×1 convolution operation is next applied to adjust the channels of feature maps to export enhanced feature maps $\hat{F}_t$, which are forwarded to the object detector.

3.4. External memory

Our external memory consists of a data buffer and a set of *Write* and *Read* operations to access. The data buffer stores the past temporally-aware feature maps $\{\tilde{F}_i\}_{t-m+1\leq i\leq t}$ where m is the number of frames to be stored.

The data structure inside the memory uses a first-in-first-out queue. Therefore, older temporally-aware feature maps are pushed out over time as new ones are written to the memory. With the *Write* operation, the latest temporally-aware feature maps are enqueued into the buffer after the oldest maps are discarded. The *Write* operation allows the decoder to access all the tensors.

3.5. Loss function

Since all the modules described above are differentiable, TFEN can be trained in an end-to-end manner. We follow the Cascade R-CNN loss proposed in [4] for multi-stage classification and bounding box regression. This is because we employ the conventional cascade R-CNN as the object detector in our experiments.

At each stage, the detector head predicts the classification score and bounding box regression offset for all sampled RoIs. The overall loss function takes the form of multi-task learning:

$$L=\sum_{s=1}^{S}(L^s_{\text{loc}}+L^s_{\text{cls}}), \tag{2}$$

where L^s_{loc} and L^s_{cls} are the losses of the bounding box predictions and classification prediction at stage s, and S is the total number of multi-stages. We follow [4] and set $S=3$.

4. Experiments

4.1. Datasets and evaluation metrics

We evaluated the proposed model on two datasets. One is the UA-DETRAC dataset [17] for surveillance, and the other is the ImageNet VID dataset [18] for natural scenes. The UA-DETRAC dataset [17] was published as a large-scale benchmark for vehicle detection in video. It offers challenges such as smaller object sizes and higher resolution frames. The ImageNet VID dataset [18], on the other hand, provides various challenges for general purposes such as motion blur and occlusion.

4.1.1. Datasets

UA-DETRAC dataset [17] contains 100 video sequences corresponding to more than 140,000 frames of real-world traffic scenes. More than 1.2 million vehicles are labeled with bounding boxes in this dataset. The videos are taken at 24 different locations in Beijing and Tianjin in China and recorded at 25 fps, with the resolution of 960×540 pixels. There are 60 videos in the training set and 40 videos in the test set.

ImageNet VID dataset [18] is one of the large-scale benchmarks for video object detection, and it contains 3862 training videos and 555 validation videos with annotated bounding boxes of 30 classes. ImageNet VID consists of a variety of videos collected from the Internet. Hence, it is suitable for a robust object detection task for a wide range of video resolutions and frame-rates.

4.1.2. Evaluation metrics

The evaluation metric for the UA-DETRAC dataset follows the average precision (AP) score proposed in the PASCAL VOC challenge [48] and use the IoU threshold of 0.7. Note that while the

[1] Although the objects can be spatially displaced between frames, the impact of displaced objects can be negligible in short-term aggregation. Indeed, we used $m=4$ in our experiments, meaning about 120 *msec*. Moreover, the aggregated feature map is used as a state of ConvGRU and, thus, the effect of displaced objects across frames on the feature map is indirect and insignificant.

PASCAL VOC challenge uses the IoU threshold of 0.5, the UA-DETRAC dataset requires object detection at a more precise location. For ImageNet VID dataset's evaluation, the detection accuracy is measured by the mean average precision (mAP) at the IoU threshold of 0.5.

4.2. Implementation details

We employed MobileNetV2 [19] as the feature extractor and cascade R-CNN [4] as the object detector. We re-implemented the cascade R-CNN network with the pre-trained MobileNetV2 in PyTorch [49] and regarded it as our baseline model.

Each of the datasets provides only two sets: a training set and a test (or validation) set; we split the training set into mini-training and mini-validation sets at the ratio of 8 to 2. For data augmentation, a random horizontal flip was adopted during training.

In order to train TFEN, we use a multistep training strategy. Namely, we first fine-tune our baseline model to the dataset domain as a static image detector. In the next step, we initialize the weights of the feature extractor and the object detector with the weights of the fine-tuned baseline model while we randomly initialize the weights of the feature enhancement network. We then train all the weights together in an end-to-end manner.

In the first step, we fine-tuned our baseline model on all the 60 videos in UA-DETRAC training set for the UA-DETRAC dataset. For ImageNet VID dataset, ImageNet DET dataset [18] was employed as training assistance. The 30 categories in VID dataset are a subset of the 200 categories in the DET dataset. Therefore, following practicals [7,50], we trained the model with VID and DET (only using the data from the 30 VID classes). We trained it in 36 epochs using asynchronous gradient descent with 0.9 momentum, 0.0005 wt decay, in a batch size of 4 images on 2 GPUs for both the dataset. The initial learning rate was 0.005 and 0.01 on UA-DETRAC dataset and ImageNet VID dataset, respectively. And we decreased the rate by 0.1 after 18 and 30 epochs.

In the second step, we fine-tuned the model injected TFEN in a temporal manner. The initial learning rate was set to 0.001 for both datasets, and we decreased it in the same way as the baseline model. For the data augmentation, we adapted the random horizontal flip for the UA-DETRAC dataset. Following the common data augmentation in the ImageNet VID dataset, we adapted random sample crop, random horizontal flip, and photometric distortions as in [3,42] for the ImageNet VID dataset.

We used a PC with Intel 3.9GHz Xeon W-2123 CPU, NVIDIA RTX 2080 Ti GPU with 11 GB Memory, and 64 GB of RAM. The experiments are executed with cuDNN v7.6 and CUDA 10.1. Our proposed TFEN ($m = 4$) runs in 29.11 and 29.02 fps on UA-DETRAC and ImageNet VID datasets, respectively.

4.3. Model design parameter analysis

We first experimentally investigated the optimal channel compressibility and the number of frames stored in the external memory in terms of the trade-off between speed and accuracy. This allows us to fix the parameters not determined through training TFEN.

4.3.1. Bottleneck dimension

We analyzed the impacts of the ConvGRU output channel dimension on accuracy and speed. In this experiment, we changed the compressibility p, which defines the number of output channels of feature maps in the spatiotemporal encoder, from 1.0 to 0.1. We remark that we fixed the number m of frames in the external memory to four and used FP16 due to our GPU memory constraint. We also remark that the processing speed of models using FP16 tends to be faster than FP32 since the computation can be done efficiently using Tensor Core units on GPUs.

Table 1
AP v.s. FPS under different compressibility p of the output channel dimension on UA-DETRAC dataset.

p	1.0	0.7	0.5	0.3	0.1
AP[%]	84.12	82.77	82.40	76.53	73.42
FPS	40.92	42.53	43.96	46.13	49.32

Table 2
mAP v.s. FPS under different compressibility p of the output channel dimension on ImageNet VID dataset.

p	1.0	0.7	0.5	0.3	0.1
mAP[%]	69.4	69.1	68.9	67.4	65.1
FPS	41.43	43.02	44.96	46.65	49.87

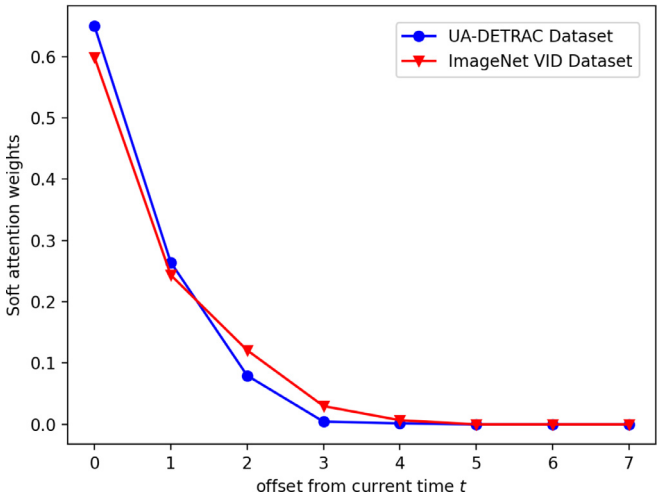


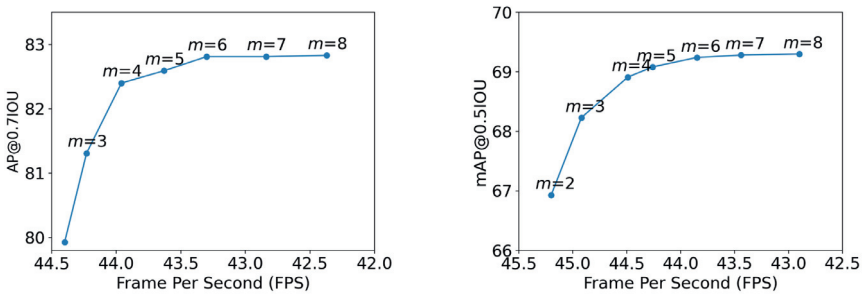
Fig. 5. Soft attention weights used in the temporal decoder.

Tables 1 and 2 show the impacts on accuracy and speed under different p on the UA-DETRAC and ImageNet VID datasets. We observe that the accuracy is decreased by compressing the feature map while the processing time and the model capacity are reduced. We also see that the accuracy remains almost constant up to $p = .5$, then drops. This is applied to both datasets. Accordingly, we confirm that when aggregating feature maps from the past, it is unnecessary to use the actual feature maps obtained from the frames and that it is possible to reduce the weight to some extent. From this experiment, we may conclude that the compressibility p controls the trade-off between detection speed and accuracy of TFEN and that $p = .5$ is the best choice.

4.3.2. Number of frames in attention decoder

We evaluated the number m of frames to be stored in the external memory. Since four frames are maximum using FP32 to accommodate in our GPU memories, we used FP16 in this experiment so that we can accommodate up to 8 frames. We changed m from 2 to 8 and computed the accuracy (AP or mAP) and fps. We also computed soft attention weights to see temporally-aware feature maps of which frames are really focused on to derive the enhanced feature maps.

Fig. 6 illustrates accuracy and fps under different m on the UA-DETRAC and ImageNet VID datasets, respectively. Fig. 5 shows the average of soft attention weights when $m = 8$. We note that the horizontal axis shows the offset from the current frame, meaning



(a) UA-DETRAC dataset (b) ImageNet VID dataset

Fig. 6. Accuracy v.s. FPS under different number m of frames to be stored in the external memory.

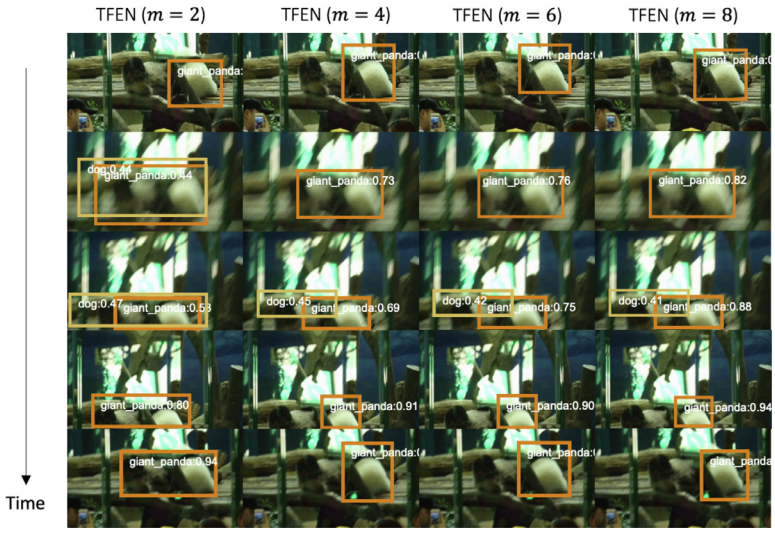


Fig. 7. Example detection results of Baseline and TFEN ($m = 2, 4, 6, 8$) for frames with lots of blur on ImageNet VID dataset. A bounding box is plotted if its confidence score is larger than 0.4.

that 0 indicates the current feature map, and 7 indicates the feature map of the last frame in the external memory.

We see that from Fig. 6 the accuracy tends to be improved by increasing the number of frames and is saturated with $m = 6$ on both the datasets. The run-time speed, on the other hand, decreases as m increases. We may conclude that $m = 4, 5$ or 6 is a good compromise as the trade-off between accuracy and speed.

Fig. 5 shows that the weight for the current frame is most significant, which is represented as 0 in the horizontal axis, and weights for the last 3 and 4 frames are dominant. It also shows that even if we store eight frames in the external memory, the frames that are really used in the computation are the last 3 or 4 frames.

Figure 7 visualizes some detection results by TFEN ($m = 2, 4, 6, 8$) where lots of blur is present due to object or camera motion. We can see that the detection's confidence and location become more stable by aggregating over longer periods ($m = 8$ is better than $m = 4, 6$, for example). However, when comparing the improvement between $m = 2$ and $m = 4$ with that between $m = 4$ and $m = 8$, we see that the improvement between $m = 4$ and $m = 8$ is smaller and is not significant.

The above observation indicates that storing more than five frames in the external memory results in just taking run-time while it does not contribute to improving accuracy much. Accordingly, we can conclude that in practice, $m = 4$ is the best choice in terms of accuracy and speed.

4.4. Quantitative comparison with state-of-the-art methods

We set $m = 4$ and $p = 0.5$ according to the above experimental results and compared the average precision (AP) and the mean average precision (mAP) of TFEN with the state-of-the-art methods.

4.4.1. Comparison on UA-DETRAC

Table 3 shows the performance comparison on UA-DETRAC. We trained and evaluated TSSD [16] with the UA-DETRAC dataset from the official code. We re-implemented VOD-MT by ourselves based on [44] (referred to as VOD-MT⋆) because the code is not publicly available. We remark that most of the published works on UA-DETRAC benchmarks are for still-image object detection. We also remark that CSP, RD^2, ExtendNet, IMIVD-TF, and MYOLO are not available as published papers up to now, and thus we used their reported scores and regard them just as reference scores.

SpotNet performs best among the methods; however, it utilizes a heavy backbone (namely, Hourglass-104 [54] and cannot run in real-time. Table 3 shows that TFEN ranks at the top among all methods except for SpotNet on the easy and sunny subsets. We also see that TFEN outperforms VOD-MT⋆ by large margins in accuracy and speed. MSVD_SPP achieves better accuracy on almost all subsets, while TFEN achieves comparable accuracy with more than three times faster processing speed. The gap between TFEN and the other methods except for TSSD is significant in speed but

Table 3
Comparison of AP scores [%] on UA-DETRAC under various environmental conditions. Bold faces are the top performance on each subset. Methods in the first block are for still images. The methods in the second block are for videos. The bottom block lists unpublished methods and thus shows just the reference scores. (* is tested by ourselves; ⋆ is our own implementation.)

Method	Backbone	Overall	Easy	Medium	Hard	Cloudy	Night	Rainy	Sunny	FPS	GPU
DPM [21]	-	25.70	34.42	30.29	17.62	24.78	30.91	25.55	31.77	0.17	N/A
ACF [51]	-	46.35	54.27	51.52	38.07	58.30	35.29	37.09	66.58	0.67	N/A
R-CNN [1]	-	48.95	59.31	54.06	39.47	59.73	39.32	39.06	67.52	0.10	Tesla K40
CompACT [52]	-	53.23	64.84	58.70	43.16	63.23	46.37	44.21	71.16	0.22	Tesla K40
Faster R-CNN [25]	VGG-16 [32]	58.45	82.75	63.05	44.25	62.34	66.29	45.16	69.85	11	Titan X
GP-FRCNN [13]	VGG-M [32]	76.57	91.79	80.85	66.05	85.16	81.23	68.59	77.20	4	Tesla K40
EB [14]	VGG-16 [32]	67.96	89.65	73.12	54.64	72.42	73.93	53.40	83.73	11	Titan X
YOLOv3-SPP [34]	Darknet-53 [5]	84.96	95.59	89.95	75.34	88.12	88.81	77.46	89.46	6–7	Titan Xp
MSVD_SPP [33]	Darknet-53 [5]	85.29	96.04	89.42	76.55	88.00	88.67	78.90	88.91	9–10	Titan Xp
FG–BR_Net [28]	ResNet-18 [22]	79.96	93.49	83.60	70.78	87.36	78.42	70.50	89.89	10	Tesla M40
SpotNet [53]	Hourglass-104 [54]	**86.80**	**97.58**	**92.57**	**76.58**	**89.38**	**89.53**	**80.93**	**91.42**	N/A	GTX 1080 Ti
3D-DETNET [55]	Darknet-Conv23 [55]	53.30	66.66	59.26	43.22	63.30	52.90	44.27	71.26	26	N/A
TSSD* [16]	VGG-16 [32]	57.16	81.06	62.07	43.14	57.59	63.87	44.98	67.73	**32**	RTX 2080 Ti
VOD-MT⋆ [44]	VGG-16 [32]	67.22	82.81	74.36	55.29	71.43	66.79	64.16	70.82	14	RTX 2080 Ti
TFEN	MobileNetV2 [19]	82.42	97.40	88.90	72.18	87.54	82.41	72.32	90.78	29	RTX 2080 Ti
CSP [56]	ResNet-50 [22]	77.67	93.65	83.67	64.54	89.66	86.81	61.39	80.63	4	Tesla K40
RD^2 [57]	N/A	85.35	95.80	89.84	76.64	89.67	86.59	78.17	90.49	N/A	Tesla P40
ExtendNet [57]	N/A	83.59	95.46	88.75	73.36	86.89	85.05	76.75	90.77	45	Titan X
IMIVD-TF [57]	N/A	85.67	96.32	91.17	75.45	87.02	88.93	80.60	89.69	1	N/A
MYOLO [57]	N/A	83.50	95.15	88.18	73.99	88.58	83.38	77.06	88.37	7	N/A

Table 4
Performance comparison with state-of-the-art end-to-end models for live-streaming videos on ImageNet VID validation set.

Method	Components				Performances	
	Backbone	Feature Aggregation?	Attention?	RNN?	mAP	FPS (Device)
TCNN [29]	DeepID [58]+Craft [59]	✓			61.5	N/A (N/A)
D&T [7]	ResNet-101 [22]				78.7	8 (Titan X)
LSTM-SSD [15]	MobileNetV1 [60]			✓	54.4	15 (Pixel 2)
Memory-guided [42]	MobileNetV2 [19]			✓	61.4	24 (Pixel 3)
Flow-guided [41]	MobileNetV1 [60]	✓		✓	61.2	13 (Mate 8)
TSSD(-OTA) [43]	VGG-16 [32]		✓	✓	65.4	21 (Titan X)
VOD-MT [44]	VGG-16 [32]	✓	✓	✓	71.0	18 (N/A)
TFEN ($m = 4$)	MobileNetV2 [19]	✓	✓	✓	68.9	29 (2080 Ti)
TFEN ($m = 6$)	MobileNetV2 [19]	✓	✓	✓	69.2	28 (2080 Ti)
TFEN ($m = 4$) w/ SSD	VGG-16 [32]	✓	✓	✓	70.6	25 (2080 Ti)

not in accuracy. TSSD also runs in real-time, but its performance is far worse than TFEN.

Fig. 8 shows precision-recall curve comparison. As with TFEN, Faster R-CNN and EB have a common point as a two-stage detector; however, we see that TFEN draws a better curve. We also observe that TFEN keeps high-level precision, even though the recall becomes higher.

4.4.2. Comparison on ImageNet VID

Table 4 shows the performance comparison with other object detection methods for live-streaming videos on ImageNet VID. We present models of TFEN ($m = 4, 6$) trained using FP32. TFEN($m = 6$) was trained with half of the batch size for memory allocation[2] In terms of accuracy, D&T, which employs the tracking operation, surpasses all methods; however, it cannot run in real-time because of the high cost of simultaneous detection and tracking and the heavy backbone. Some methods using ResNet-101 [22] or VGG [32] as the backbone achieve higher mAP, but using such heavy backbones is not suitable in the context of real-time object detection in live-streaming video.

We see that methods exploiting MobileNetV1 [60] or MobileNetV2 [19] as the backbone realize real-time object detection thanks to the lightness of the backbone; however, they tend to achieve lower mAP compared to methods having richer feature extractors. In contrast, TFEN achieves the highest accuracy among the methods using MobileNet (either V1 or V2), beating the second place by about 7 points. This is because TFEN exploits feature aggregation, attention mechanism, and recurrent neural networks altogether, and, furthermore, their combination brings the improvement of accuracy.

We see that although VOD-MT [44] outperforms TFEN, its backbone is heavier than that of TFEN. As a result, VOD-MT runs at only 18 fps while TFEN does at 29 fps. To fairly compare TFEN with VOD-MT, we used VGG-16 [32] as the backbone and SSD as the detector for TFEN, resulting in the difference from VOD-MT is the only feature aggregation module (the last line in Table 4). We see that TFEN processes more than 7 fps faster than VOD-MT under the same backbone and detector while achieving comparable accuracy[3]. This faster processing time comes from the one-shot manner aggregation of TFEN. We also see that a lightweight feature extractor, MobileNetV2, speeds up the processing time even more. We confirm that TFEN is more suitable for applications where run-time speed is essential.

[2] We confirm that when using FP32, the performances in accuracy and run-time speed at $m = 4$ and $m = 6$ are almost same like the case where we use FP16.

[3] Since VOD-MT [44] does not provide device information, we instead evaluated the runtime of VOD-MT⋆ with 2080 Ti under fair conditions, resulting in 14 fps. Even if we use only the spatio-temporal feature aggregation module of VOD-MT⋆, the runtime is 22 fps.

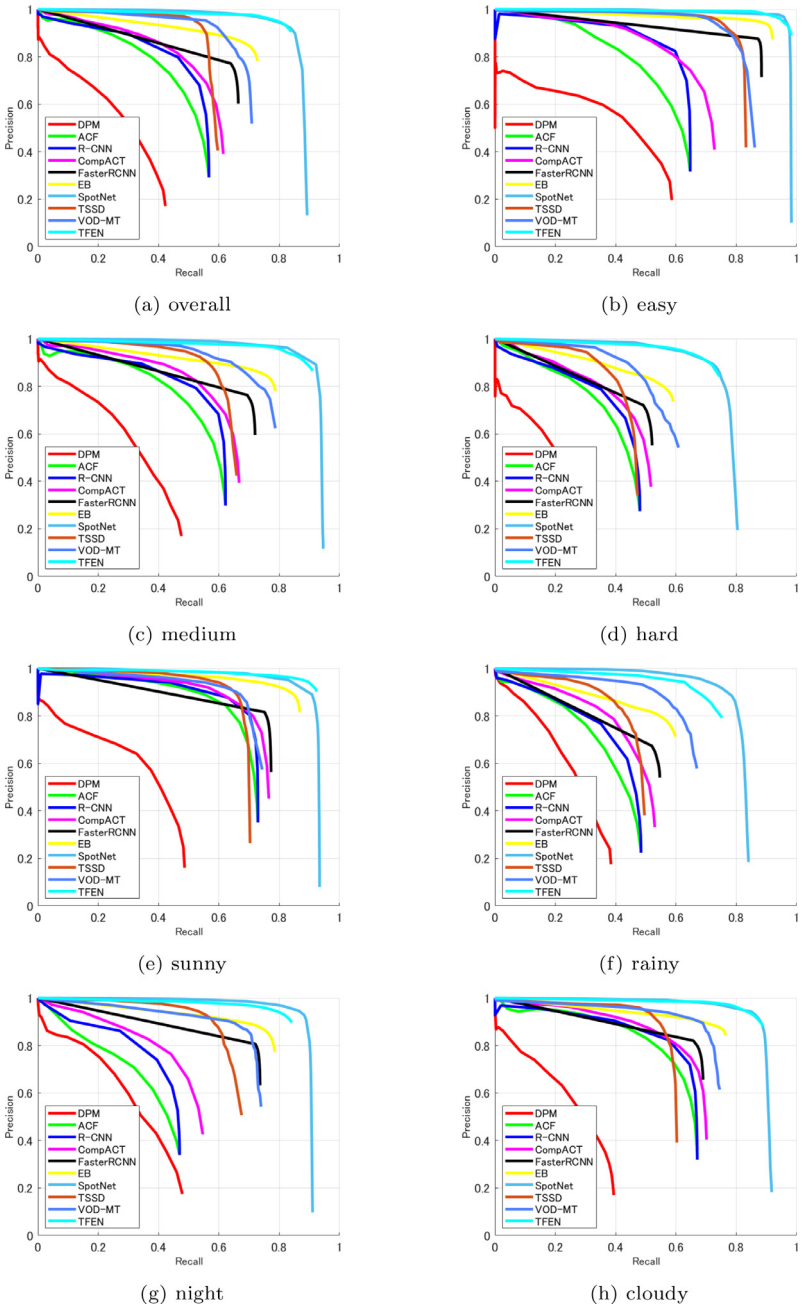


Fig. 8. Comparison of precision-recall curves on each subset of UA-DETRAC test set. We show results by the default models provided by the dataset providers [17] and results by SpotNet, TSSD, and VOD-MT.

4.5. Qualitative comparison

Figure 9 shows detection results obtained by TFEN, the baseline model (MobileNetV2 based Cascade R-CNN), and TSSD along a sequence of frames on UA-DETRAC dataset. It also shows ground truth. The video clip shows that both TFEN and TSSD successfully detect the occluded car behind the bus while the baseline model fails. This confirms the importance of using temporal information because the baseline model does not use temporal information at all.

Figure 10 illustrates some visualized examples of the detection result by the baseline and TFEN. First, we can see that the baseline model is not stable in detection, as it sometimes incorrectly labels the target with another class or outputs multiple bounding boxes even when the target is not moving much. On the other hand, the detection results by TFEN are more stable, and the confidence level tends to be higher. This is thanks to incorporating time-series information in TFEN.

4.6. Ablation study

We evaluated the effectiveness of each component in TFEN to show its necessity on both datasets. We removed each component of TFEN one by one from the complete model to have ablation models. They are the model w/o TAD (temporal attention decoder), model w/o SK (skip connection), model w/o SA (spatial attention), and model w/o TF (temporally-aware feature maps). Note that the baseline model corresponds to the model dropping TFEN. Performances of the ablation models and the baseline model are illustrated in Table 5.

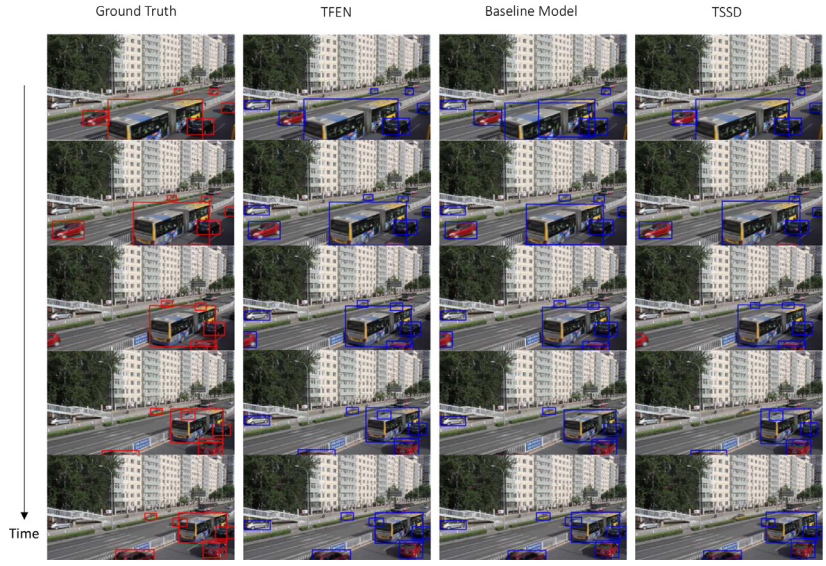


Fig. 9. Example detection results of TFEN, Baseline Model, and TSSD on UA-DETRAC dataset.

Table 5
Performance [%] of ablation models.

Method	Components					UA-DETRAC				Imagenet VID
	Video	Temporal Attention Decoder	Skip Connection	Spatial Attention	Temporally-aware Feature maps	Overall	Easy	Medium	Hard	mAP
(a) baseline model	–	–	–	–	–	73.39	90.92	79.28	60.33	64.1
(b) model w/o TAD	✓	–	✓	✓	✓	79.26	95.96	85.83	67.42	67.8
(c) model w/o SK	✓	✓	–	✓	✓	72.53	91.26	78.57	59.24	64.3
(d) model w/o SA	✓	✓	✓	–	✓	80.93	97.17	86.08	66.44	68.5
(e) model w/o TF	✓	✓	✓	✓	–	79.22	95.06	84.77	65.46	67.7
(f) (complete) TFEN	✓	✓	✓	✓	✓	82.42	97.40	88.90	72.18	68.9

4.6.1. Temporal attention decoder

The model w/o TAD (Table 5 (b)) shows the ablation results of replacing the temporal attention decoder with a standard decoder without an attention mechanism. This replacing decoder consists of a simple stacked ConvGRU and ReLU, which receives only current-frame feature maps and has no external memories; the model w/o TAD thus cannot exploit past feature maps except the hidden state. The performance of the model w/o TAD drops 3.16 points of AP and 1.3 points of mAP for the overall subset and mAP, respectively. This demonstrates the effectiveness of the temporal attention decoder.

4.6.2. Skip connection

Table 5 (c) and (f) show that applying skip connection between the feature extractor and the object detector brings consistent gains on all the sets. Moreover, Table 5 (a) and (c) reveal that the model w/o SK has lower scores than the baseline model in overall, medium, and hard subsets on the UA-DETRAC dataset. On the ImageNet VID dataset, the model w/o SK slightly outperforms the baseline model but achieves the lowest score among all the ablation models. We conjecture that the skip connection helps gradient flowing through TFEN and is an essential part of TFEN.

4.6.3. Spatial attention

Table 5 (d) and (f) show that the spatial attention mechanism used in the spatiotemporal encoder brings additional improvements 1.49%, 0.23%, 2.82%, 5.74% in AP on overall, easy, medium, hard subsets, respectively. We also see 0.4 point improvements on ImageNet VID. This suggests that the spatial attention mechanism is useful for all the subsets, but especially for the hard subset. We confirm that the hard subset tends to contain dense vehicles or small vehicles. Therefore, we may conclude that the spatial attention mechanism removes useless features around objects and makes feature maps easier to handle in the temporal attention decoder.

4.6.4. Temporally-aware feature maps

To verify the necessity of temporally-aware feature maps, we store the feature maps without temporal information in the external memory instead of the temporally-aware feature maps, which is the model of w/o TF. We remark that the model w/o TF uses the encoder only to create attention weights. Table 5 (e) and (f) show that temporal information in the external memory gives additional improvements 3.20%, 2.34%, 4.13%, 6.72%, 1.2% on overall, easy, medium, hard subsets, and ImageNet VID, respectively. This suggests that the temporal information stored in the external memory is useful for all the sets. We thus confirm that both the feature aggregation with temporal attention mechanism and temporal-aware feature maps are necessary for improving the detection accuracy.

4.6.5. Attention based feature aggregation

We validated our introduced attention mechanism in feature aggregation by replacing it with other aggregation ways on the UA-DETRAC dataset. Table 6 shows the results. Our proposed attention mechanism for feature aggregation, which is dynamic weighting, is

Fig. 10. Example detection results of TFEN and Baseline on ImageNet VID dataset where our model outperforms the baseline model. The upper row of each sequence corresponds to the baseline model, and the lower one corresponds to the proposal model, TFEN. A bounding box is plotted if its confidence score is larger than 0.4.

Table 6
Effectiveness of temporal attention mechanism on UA-DETRAC dataset.

Feature aggregation ways	AP[%]	FPS
Attention (ours)	82.42	29.11
Weighted averaging	79.81	31.32
Cos similarity	81.04	27.85

denoted by attention. Weighted averaging [15] is a static weighting way for feature aggregation where the weights of frames are determined so that the previous and current frames are weighted 1:3, and all other frames are set to 1. Cos-similarity [9] is another dynamic weighting way where the weights of the current and past frames are dynamically computed using cosine similarity and the softmax function.

We see from Table 6 that the dynamic weighting ways are more accurate than the static weighting as the weights are changed for each video and that our introduced attention is most beneficial. We also observe that attention is superior in terms of run-time. This is because it predicts the weights in a one-shot manner, not calculating weights one by one.

4.6.6. Feature maps enhancement

We finally visualize the feature maps before and after TFEN (F_t and $\hat{F}_t$) in Fig. 11 to illustrate how feature maps are enriched. Yellow means higher activation values, whereas dark Mazarin indicates negligible feature activation. We observe that compared with F_t, $\hat{F}_t$ shows stronger responses near regions where vehicles exist, even highly occluded vehicles. We thus see that our feature maps enhancement is practical and improves the detection accuracy.

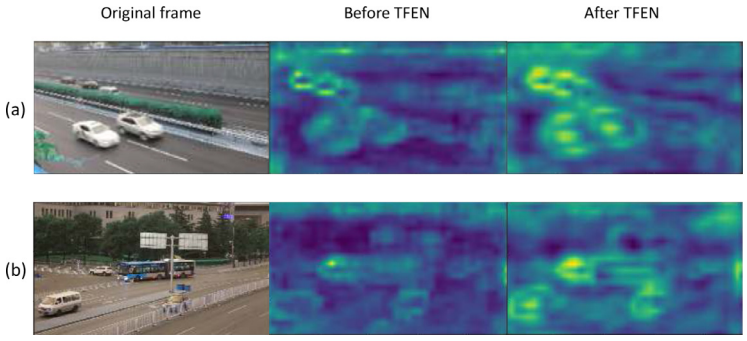


Fig. 11. Feature activation before and after TFEN. For visualization, we summed up the feature maps in the channel direction, then mapped their values to the interval of 0–255, and up-sampled the feature channel by the bi-linear interpolation.

5. Conclusion

We presented the temporal attention network with the external memory called TFEN for real-time object detection in live-streaming video. TFEN exploits the spatial and temporal information available to enrich feature maps extracted using a lightweight feature extractor. While the memory-based approach has been used only for offline video object detectors, TFEN deals with frames in the memory with an efficient attention mechanism to realize online object detection in live-streaming videos. Compared with attention-based methods that aggregate relevant frames to focus on frame by frame, TFEN utilizes temporally-aware feature maps to efficiently compute attention weights in a one-shot manner, which leads to real-time object detection. We confirmed that TFEN achieves real-time performance while keeping comparable accuracy with state-of-the-art methods using publicly available datasets. TFEN demonstrates the attention module's clear benefits based on the external memory and achieves a considerably enhanced trade-off between accuracy and speed.

Our method computes attention weights for aggregation through the features of the current frame and demonstrates reasonable performance in accuracy and speed. However, there will be other ways to compute the weights. Exploring further efficient and effective aggregation weights is left for future work.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2022.108847.

References

[1] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2014, p. 580587.
[2] J. Dai, Y. Li, K. He, J. Sun, R-FCN: object detection via region-based fully convolutional networks, in: Proceedings of International Conference on Neural Information Processing Systems, 2016, p. 379387.
[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S.E. Reed, C.-Y. Fu, A.C. Berg, Ssd: single shot multibox detector, in: Proceedings of European Conference on Computer Vision, volume 9905, Springer, 2016, pp. 21–37.
[4] Z. Cai, N. Vasconcelos, Cascade r-CNN: Delving into high quality object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6154–6162.
[5] J. Redmon, A. Farhadi, Yolov3: an incremental improvement, arXiv preprint arXiv:1804.02767 (2018).
[6] K. Shuang, Z. Lyu, J. Loo, W. Zhang, Scale-balanced loss for object detection, Pattern Recognit 117 (2021) 107997.
[7] C. Feichtenhofer, A. Pinz, A. Zisserman, Detect to track and track to detect, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3057–3065.
[8] X. Zhu, Y. Xiong, J. Dai, L. Yuan, Y. Wei, Deep feature flow for video recognition, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4141–4150.
[9] X. Zhu, Y. Wang, J. Dai, L. Yuan, Y. Wei, Flow-guided feature aggregation for video object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 408–417.
[10] X. Zhu, J. Dai, L. Yuan, Y. Wei, Towards high performance video object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7210–7218.
[11] H. Deng, Y. Hua, T. Song, Z. Zhang, Z. Xue, R. Ma, N. Robertson, H. Guan, Object guided external memory network for video object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 6678–6687.
[12] K. Kang, H. Li, T. Xiao, W. Ouyang, J. Yan, X. Liu, X. Wang, Object detection in videos with tubelet proposal networks, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 889–897.
[13] A. Sikandar, G. Fabio, Geometric proposals for faster r-CNN, in: Proceedings of IEEE International Conference on Advanced Video and Signal-Based Surveillance, 2017, pp. 1–6.
[14] L. Wang, Y. Lu, H. Wang, Y. Zheng, H. Ye, X. Xue, Evolving boxes for fast vehicle detection, in: Proceedings of IEEE International Conference on Multimedia & Expo, 2017, pp. 1135–1140.
[15] M. Liu, M. Zhu, Mobile video object detection with temporally-aware feature maps, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5686–5695.
[16] X. Chen, Z. Wu, J. Yu, Tssd: Temporal single-shot detector based on attention and lstm, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2018, pp. 1–9.
[17] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, S. Lyu, UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking, Comput. Vision Image Understanding 193 (2020) 102907.
[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, Imagenet large scale visual recognition challenge, Int J Comput Vis 115 (3) (2015) 211–252.
[19] M. Sandler, A.G. Howard, M. Zhu, A. Zhmoginov, L.C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.
[20] M. Fujitake, A. Sugimoto, Temporal feature enhancement network with external memory for object detection in surveillance video, in: International Conference on Pattern Recognition, 2020, pp. 7684–7691.
[21] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, Cascade object detection with deformable part models, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2241–2248.
[22] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[23] P. Singh, P. Mazumder, V.P. Namboodiri, Context extraction module for deep convolutional neural networks, Pattern Recognit 122 (2022) 108284.
[24] R. Girshick, Fast r-CNN, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, p. 14401448.
[25] S. Ren, K. He, R. Girshick, J. Sun, Faster r-CNN: towards real-time object detection with region proposal networks, IEEE Trans Pattern Anal Mach Intell 39 (6) (2017) 1137–1149.
[26] J. Redmon, A. Farhadi, Yolo9000: Better, faster, stronger, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6517–6525.
[27] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real–time object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.
[28] Z. Fu, Y. Chen, H. Yong, R. Jiang, L. Zhang, X.-S. Hua, Foreground gating and background refining network for surveillance object detection, IEEE Trans. Image Process. 28 (12) (2019) 6077–6090.

[29] K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang, W. Ouyang, T-CNN: Tubelets with convolutional neural networks for object detection from videos, IEEE Trans. Circuits Syst. Video Technol. 28 (10) (2018) 2896–2907.
[30] K. Kang, W. Ouyang, H. Li, X. Wang, Object detection from video tubelets with convolutional neural networks, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 817–825.
[31] W. Han, P. Khorrami, T.L. Paine, P. Ramachandran, M. Babaeizadeh, H. Shi, J. Li, S. Yan, T.S. Huang, Seq-NMS for video object detection, arXiv preprint arXiv:1602.08465 (2016).
[32] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Proceedings of the International Conference on Learning Representations, 2015.
[33] K.-J. Kim, P.-K. Kim, Y.-S. Chung, D.-H. Choi, Multi-scale detector for accurate vehicle detection in traffic surveillance data, IEEE Access 7 (2019) 78311–78319.
[34] K.-J. Kim, P.-K. Kim, Y.-S. Chung, D.-H. Choi, Performance enhancement of YOLOv3 by adding prediction layers with spatial pyramid pooling for vehicle detection, in: Proceedings of IEEE International Conference on Advanced Video and Signal-Based Surveillance, 2018, pp. 1–6.
[35] S. Wang, Y. Zhou, J. Yan, Z. Deng, Fully motion-aware network for video object detection, in: Proceedings of European Conference on Computer Vision, Springer, 2018, pp. 542–557.
[36] G. Bertasius, L. Torresani, J. Shi, Object detection in video with spatiotemporal sampling networks, in: Proceedings of European Conference on Computer Vision, Springer, 2018, pp. 331–346.
[37] F. Xiao, Y.J. Lee, Video object detection with an aligned spatial-temporal memory, in: Proceedings of European Conference on Computer Vision, Springer, 2018, pp. 485–501.
[38] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, T. Mei, Relation distillation networks for video object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 7023–7032.
[39] H. Wu, Y. Chen, N. Wang, Z. Zhang, Sequence level semantics aggregation for video object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9217–9225.
[40] Y. Chen, Y. Cao, H. Hu, L. Wang, Memory enhanced global-local aggregation for video object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 10337–10346.
[41] X. Zhu, J. Dai, X. Zhu, Y. Wei, L. Yuan, Towards high performance video object detection for mobiles, arXiv preprint arXiv:1804.05830 (2018).
[42] M. Liu, M. Zhu, M. White, Y. Li, D. Kalenichenko, Looking fast and slow: memory-guided mobile video object detection, arXiv preprint arXiv:1903.10172 (2019).
[43] X. Chen, J. Yu, Z. Wu, Temporally identity-aware SSD with attentional LSTM, IEEE Trans Cybern 50 (6) (2020) 2674–2686.
[44] K. Jaekyum, K. Junho, L. Byeongwon, Y. Seungji, J.W. Choi, Video object detection using objects motion context and spatio-temporal feature aggregation, in: International Conference on Pattern Recognition, 2020, pp. 1604–1610.
[45] J. Park, S. Woo, J.-Y. Lee, I.-S. Kweon, BAM: Bottleneck attention module, in: Proceedings of British Machine Vision Conference, 2018, pp. 1–14.
[46] N. Ballas, L. Yao, C. Pal, A. Courville, Delving deeper into convolutional networks for learning video representations, arXiv preprint arXiv:1511.06432 (2015).
[47] M. Lin, Q. Chen, S. Yan, Network in network, in: Proceedings of the International Conference on Learning Representations, 2014.
[48] M. Everingham, S.M.A. Eslami, L. Van Gool, C.K.I. Williams, J. Winn, A. Zisserman, The pascal visual object classes challenge: aretrospective, Int J Comput Vis 111 (1) (2015) 98–136.
[49] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, in: Proceedings of International Conference on Neural Information Processing Systems, 2019, pp. 8026–8037.
[50] L. Galteri, L. Seidenari, M. Bertini, A.D. Bimbo, Spatio-temporal closed-loop object detection, IEEE Trans. Image Process. 26 (3) (2017) 1253–1263.
[51] P. Dollár, R.D. Appel, S.J. Belongie, P. Perona, Fast feature pyramids for object detection, IEEE Trans Pattern Anal Mach Intell 36 (8) (2014) 1532–1545.
[52] Z. Cai, M. Saberian, N. Vasconcelos, Learning complexity-aware cascades for deep pedestrian detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, p. 33613369.
[53] H. Perreault, G.-A. Bilodeau, N. Saunier, M. Héritier, Spotnet: Self-attention multi-task network for object detection, in: Proceedings of Conference on Computer and Robot Vision, 2020, pp. 230–237.
[54] H. Law, J. Deng, Cornernet: Detecting objects as paired keypoints, in: Proceedings of European Conference on Computer Vision, Springer, 2018, pp. 734–750.
[55] S. Li, F. Chen, 3d-DETNet: a single stage video-based vehicle detector, in: Third International Workshop on Pattern Recognition, volume 10828, SPIE, 2018, pp. 60–66.
[56] W. Liu, S. Liao, W. Ren, W. Hu, Y. Yu, High-level semantic feature detection: A new perspective for pedestrian detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 5182–5191.
[57] S. Lyu, M.-C. Chang, D. Du, W. Li, Y. Wei, M. Del Coco, P. Carcagnì, A. Schumann, B. Munjal, D.-H. Choi, et al., Ua-detrac 2018: Report of avss2018 & iwt4s challenge on advanced traffic monitoring, in: Proceedings of IEEE International Conference on Advanced Video and Signal-Based Surveillance, 2018, pp. 1–6.
[58] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, C.-C. Loy, X. Tang, DeepID-net: Deformable deep convolutional neural networks for object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2403–2412.
[59] B. Yang, J. Yan, Z. Lei, S.Z. Li, Craft objects from images, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 6043–6051.
[60] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).

Masato Fujitake received the Master degree in Electrical, Electronic and Communications Engineering Technology from the Shibaura Institute of Technology, Tokyo, Japan, in 2019. He is currently pursuing a Ph.D. degree in computer science with SOKENDAI, Japan. His current research interests include deep learning and video analysis.

Akihiro Sugimoto received the Dr. Eng. in mathematical engineering from the University of Tokyo. He is currently a full professor at the National Institute of Informatics, Tokyo. He has been an associate editor of International Journal of Computer Vision since 2014. He also served several top-tier conferences including IEEE International Conference on Computer Vision (ICCV), European Conference on Computer Vision (ECCV), Asian Conference on Computer Vision (ACCV), International Conference on Pattern Recognition (ICPR), and International Conference of 3D Vision (3DV) as an area chair, a program chair, and a general chair. He is a regular reviewer of international conferences/journals in computer vision, AI, and pattern recognition. He has published more than 150 peer-reviewed journal/international conference papers. He received Best Paper Awards from the Information Processing Society of Japan in 2001 and from the Institute of Electronics, Information and Communication Engineers (IEICE) in 2011. He is a member of IEEE and ACM.