Spatio-temporal Prompting Network for Robust Video Feature Extraction

Guanxiong Sun[1, 2], Chi Wang[1], Zhaoyu Zhang[1], Jiankang Deng[2, 3], Stefanos Zafeiriou[3], Yang Hua[1]

[1]Queen's University Belfast   [2]Huawei UKRD   [3]Imperial College London

{gsun02, cwang38, zzhang55,Y.Hua}@qub.ac.uk, {j.deng16,s.zafeiriou}@imperial.ac.uk

Abstract

Frame quality deterioration is one of the main challenges in the field of video understanding. To compensate for the information loss caused by deteriorated frames, recent approaches exploit transformer-based integration modules to obtain spatio-temporal information. However, these integration modules are heavy and complex. Furthermore, each integration module is specifically tailored for its target task, making it difficult to generalise to multiple tasks. In this paper, we present a neat and unified framework, called Spatio-Temporal Prompting Network (STPN). It can efficiently extract robust and accurate video features by dynamically adjusting the input features in the backbone network. Specifically, STPN predicts several video prompts containing spatio-temporal information of neighbour frames. Then, these video prompts are prepended to the patch embeddings of the current frame as the updated input for video feature extraction. Moreover, STPN is easy to generalise to various video tasks because it does not contain task-specific modules. Without bells and whistles, STPN achieves state-of-the-art performance on three widely-used datasets for different video understanding tasks, i.e., ImageNetVID for video object detection, YouTubeVIS for video instance segmentation, and GOT-10k for visual object tracking. Codes are available at https://github.com/guanxiongsun/STPN

1. Introduction

Video understanding is a fundamental research direction in the field of computer vision. It plays an important role in many real-world applications, such as autonomous driving [64, 32], video surveillance [34, 5], and sports analysis [56, 25]. However, a significant challenge in this field is the deterioration of video frames due to motion blur, occlusion, and deformation, which makes it difficult to extract relevant information.
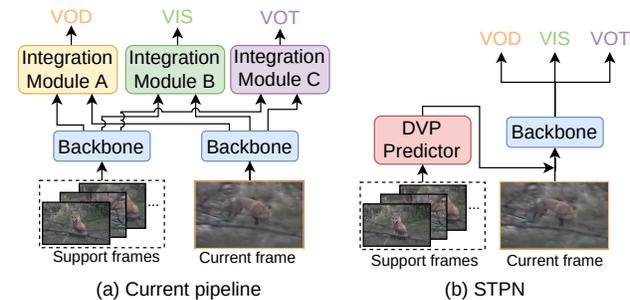

Figure 1. Comparisons between pipelines of (a) existing methods and (b) the proposed Spatio-temporal Prompting Network (STPN). Existing methods introduce complex and **task-specific** integration modules (▢ ▢ ▢) *after* backbone networks (▢). In contrast, STPN is a **unified** framework for multiple tasks. A lightweight dynamic video prompt (DVP) predictor (▢) generates a set of DVPs to adjust input *before* backbone networks (▢). *Best viewed in colour.*

To overcome this challenge, inspired by the great success of transformers in many computer vision tasks [49, 54, 27, 19, 4], researchers explore various transformer-based integration modules to alleviate the information loss on the deteriorated video frames. For example, in video object detection, transformer-based feature aggregation methods [13, 7, 57, 43] are investigated to enhance the feature of proposals [40] in the detection head. In video instance segmentation, deteriorated frames usually cause wrong instance associations in a sequence, so 3D mask decoders [55, 22, 8] are introduced to learn associations of instance masks in an end-to-end manner. For visual object tracking, traditional correlation filters [26, 47] meet their limitations in severely degraded frames. Therefore, transformer-based integration modules [11, 62, 6, 10] are proposed to better capture complicated correlations between the template region and the search region. To conclude, the summarised pipeline of recent methods is shown in Figure 1 (a). A backbone network extracts spatially-only features from the current frame and

the support frames in the same video. Then, different integration modules integrate spatial-only features on multiple frames to obtain spatio-temporal features. Finally, the spatio-temporal features are used for the targeted video understanding task.

While the current pipeline achieves good performance, there are still two major limitations of transformer-based integration modules. Firstly, these integration modules are involved as an extra component after the backbone network, leading to increased complexity and additional computational costs. Secondly, each integration module is tailored specifically for the target task and thus cannot be generalised to multiple video tasks. Hence, we put forward the following question: can we remove the complex integration modules and directly obtain spatio-temporal information in backbone networks?

To answer this question, inspired by recent prompting techniques [24, 23, 28], in this paper, we present a neat and unified framework, called Spatio-Temporal Prompting Network (STPN). Instead of using complex integration modules, STPN simplifies the current pipeline for video understanding by introducing spatio-temporal information into the backbone network, as shown in Figure 1 (b). Specifically, given a video frame, we propose a dynamic video prompt (DVP) predictor to generate several video prompts according to support frames. Then, the predicted DVPs are prepended to the patch embeddings of the current frame as the updated input. Finally, a vision transformer backbone network extracts video features using the updated input for future video understanding tasks. It is worth noting that the DVP predictor is a lightweight structure and introduces only a small number of extra parameters, e.g., 0.11M. Moreover, STPN can be easily adapted to different video understanding tasks, since it does not contain task-specific modules, and all modifications happen before the backbone network.

In summary, our key contributions are: (1) We present the Spatio-Temporal Prompting Network (STPN) that can extract robust video features on deteriorated video frames. STPN simplifies the current pipeline for video understanding and is easy to generalise to different video understanding tasks. (2) To the best of our knowledge, we are the first to explore promoting techniques for robust video feature extraction on the task of video object detection (VOD), video instance segmentation (VIS), and visual object tracking (VOT). (3) Without bells and whistles, STPN achieves state-of-the-art performance on three widely-used video understanding benchmarks, i.e., ImageNet-VID [41] for VOD, YouTube-VIS [63] for VIS, and GOT-10k [21] for VOT.

2. Related Work

In this section, we first review the state-of-the-art (SOTA) methods for related video understanding tasks. Then, we introduce the visual prompting techniques.

Video Object Detection (VOD). SOTA VOD methods use attention modules to aggregate features of support frames onto the current frame and thus enhance the feature quality of the current frame. For example, RDN [13] and SELSA [57] utilise relation modules to aggregate proposal features from local frames and global frames, respectively. These methods use sample memory architectures to store features on support frames, e.g., a sliding window or a memory queue, which limit the "diversity" of support features. Therefore, MAMBA [43] proposes a memory bank architecture that can partially update and sample support features and thus achieve a good speed-accuracy trade-off. More recent approaches TransVOD [17] and TDViT [45] introduce transformers into the field and further improve the performance of VOD.

Video Instance Segmentation (VIS). The goal of VIS is to simultaneously segment and track all object instances in the video sequence. There are two categories of approaches: Firstly, *per-frame* approaches conduct instance segmentation on each frame independently and then associate instance masks by post-processing. MaskTrackRCNN [63] adds a tracking prediction head on MaskRCNN [16] for instance association. More recently, some approaches [60, 20, 18] use transformers [49, 4] to reason the relationships between instances. Secondly, *per-clip* approaches [55, 59, 8, 22] directly predict 3D masks for each instance in a video. VisTR [55] proposes a transformer encoder-decoder structure to obtain spatio-temporal features, which are then used in an instance sequence matching module to get object tracks. IFC [22] uses memory tokens inside of the transformer encoder module to transfer temporal information between input frames.

Visual Object Tracking (VOT). Object tracking is one of the most fundamental research topics in computer vision and has a long history. Siamese-based methods [2, 30, 29, 53, 67] consider tracking as a template-matching problem by extracting correlation feature maps of the target and the search region via two-head siamese networks. Recently, transformers have been introduced to perform attention-based correspondence modelling and these methods [6, 51, 10, 62] achieve SOTA performance. Among these trackers, MixFormer [11] is a unified framework established solely on transformers for feature extraction, correspondence modelling, and result generation.

Prompting. The concept of prompting is originally proposed in natural language processing (NLP). Prompting refers to designing instructions and prepending them to the input so that the pre-trained language models [38, 14, 3] can "understand" the task. Recently, prompting techniques

Meaning	Symbol
The i-th transformer layer	$\mathbf{L}_i$
Collection of the output embeddings from $\mathbf{L}_i$	$\mathbf{E}^i$
Number of embeddings in $\mathbf{E}^i$	n_i
Dimension of embeddings in $\mathbf{E}^i$	d_i
Collection of dynamic video prompts	$\mathbf{P}$
Number of dynamic video prompts	N_P

Table 1. List of notations.

have been explored in many computer vision tasks. A major trend [35, 36, 24] is to transfer pre-trained vision-language models [15, 37, 39] to the task of video recognition, e.g., video classification and action recognition. Another popular research direction is parameter-efficient transfer learning via prompt tuning. For example, VPT [23] and UPT [65] tune a small number of parameters in the input space of a model and can achieve good performance comparable to the full fine-tuning scheme on a variety of recognition tasks.

3. Preliminaries

In this section, we define notations necessary for subsequent demonstrations. I_t denotes the current frame and t is the current time step in the video. I_{sup} denotes the support frames used to predict video prompts and aid in the feature extraction process. Specifically, $I_{sup} = \{I_\tau\}_{t-S(K/2)}^{t+S(K/2)}$, where K denotes the number of support frames (assumed to be an even number) and S denotes the frame interval in the time dimension between support frames.

Transformers encoders [27, 33, 58] are used to extract feature embeddings from video frames. A transformer encoder contains a patch embedding layer and L transformer layers. The input frame is first divided into n non-overlapping patches $\{\mathbf{x}_j \in \mathbb{R}^{3 \times h \times w} | j \in \mathbb{N}, 1 \leq j \leq n\}$, where h and w are the height and width of the patches. Each patch is then embedded into patch embeddings, denoted as $\mathbf{E} \in \mathbb{R}^{n_0 \times d_0}$. Other notations are summarised in Table 1.

4. Methodology

4.1. Overview

The overview of our framework is shown as in Figure 2 (a). There are two stages in the framework: the predicting stage and the prompting stage. The goal of the predicting stage is to generate a set of prompts that vary according to spatio-temporal information. We first select K support frames around I_t and then pass the support frames I_{sup} to the patch embedding layer and the transformer encoder to extract image embeddings, called support embeddings. The support embeddings are then passed to the dynamic video prompt (DVP) predictor network as the input. The DVP predictor outputs N_P dynamic video prompts, denoted as

$\mathbf{P}$, which contain the spatio-temporal information of the current video. Details of the DVP predictor network are illustrated in §4.2.

The second stage is the prompting stage whose goal is to extract spatio-temporal features of the current frame I_t using $\mathbf{P}$. Specifically, $\mathbf{P}$ is prepended to the patch embeddings of the current frame I_t. Then, the mixed embeddings are passed into the transformer encoder to produce the spatio-temporal embeddings. Finally, the spatio-temporal embeddings are used for different video understanding tasks, e.g., video object detection, video instance segmentation and visual object tracking. More details of the prompting stage are presented in §4.3.

4.2. Predicting Stage

Given the current frame I_t, we first sample K support frames around I_t as support frames. The support frames are then passed into the patch embedding layer and the transformer encoder:

$$\mathbf{E}_\tau^i = \mathbf{L}^i(\mathbf{E}_\tau^{i-1}) \quad \mathbf{E}_\tau^i \in \mathbb{R}^{n_i \times d_i}, i = 1, 2, \ldots, L. \quad (1)$$
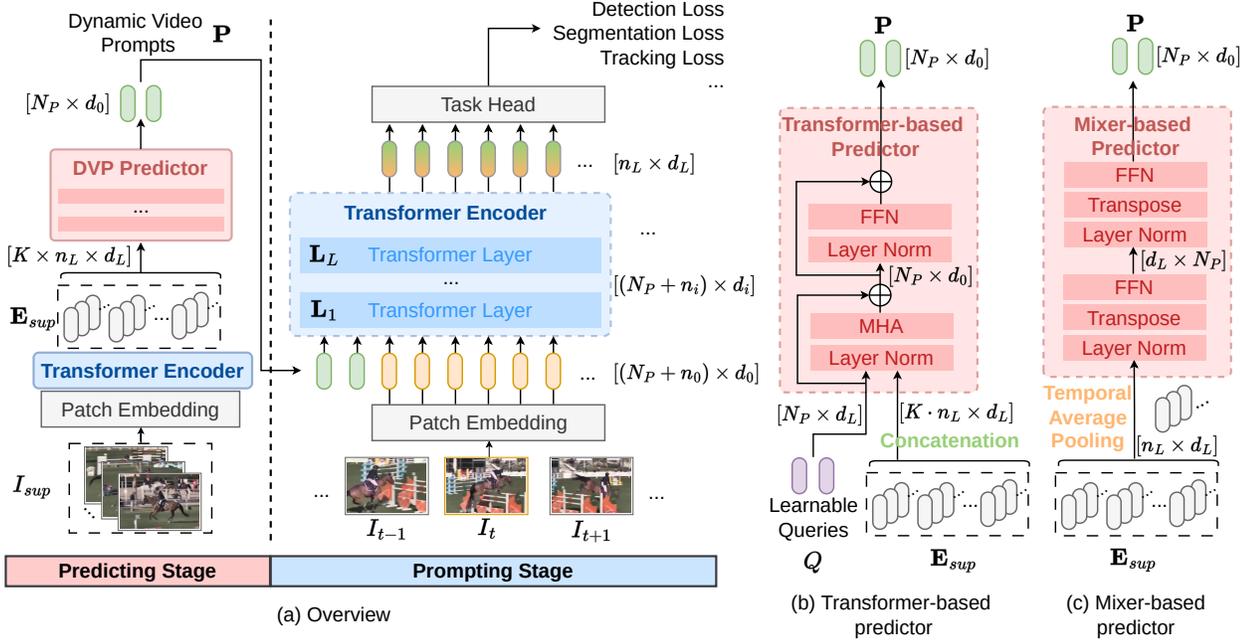
Therefore, the collection of support embeddings is denoted as: $\mathbf{E}_{sup} = \{\mathbf{E}_\tau^{\mathbf{L}} \in \mathbb{R}^{n_L \times d_L} | \tau \in \mathcal{T}\}$, where $\mathcal{T}$ is the collection of sampled time steps around the current frame, $\mathcal{T} = [t - S(K/2), ..., t - S, t + S, ..., t + S(K/2)]$. For example, if $K = 6$ and $S = 2$, three support frames are sampled from the past of the current time step t, and three are sampled from the future, with a time interval of 2 between each sampled frame in both directions. Given $\mathbf{E}_{sup}$, we design two predicting networks, i.e., transformer-based predictor and mixer-based predictor, to obtain dynamic video prompts $P \in \mathbb{R}^{N_P \times d}$.

Transformer-based Predictor. This design is inspired by the object decoder in DETR [4] where a set of randomly initialised object queries are used to retrieve object features on the image embeddings. As depicted in Figure 2 (b), we introduce a set of learnable prompt queries $Q \in \mathbb{R}^{N_P \times d_L}$ to gather spatio-temporal information from the support embeddings. Specifically, a multi-head attention (MHA) [49] module takes the support embeddings and the prompt queries as input, and produces the aggregated embeddings. The process is formulated as follows:

$$K = \text{Concat}(\mathbf{E}_{sup}) \quad K \in \mathbb{R}^{K \cdot n_L \times d_L}, \quad (2)$$

$$\hat{Q} = \text{MHA}(\text{LN}(Q), \text{LN}(K)) + Q \quad \hat{Q} \in \mathbb{R}^{N_P \times d_L}, \quad (3)$$

where $\text{Concat}(\cdot)$ indicates the concatenation operation and LN denotes the LayerNorm layer. The output embeddings are then added back to the prompt queries and passed to the LayerNorm layer. The normalised output embeddings are further transformed by a simple feed-forward network

Figure 2. (a) An overview of our approach. In the predicting stage, a set of dynamic visual prompts (DVPs) **P** is generated by the DVP predictor that takes support embeddings $\mathbf{E}_{sup}$ on support frames I_{sup} as the input. Then, in the prompting stage, predicted DVPs are prepended with the patch embeddings of the current frame to extract spatio-temporal embeddings via a transformer encoder which contains L transformer layers. Finally, different task heads take the spatio-temporal embeddings and output final results for various general video tasks, e.g., video object detection, video instance segmentation, and visual object tracking. (b) Details of the transformer-based predictor. (c) Details of the Mixer-based predictor.

(FFN) with a residual connection that makes the dynamic video prompts:

$$\mathbf{P} = \mathrm{FFN}(\mathrm{LN}(\hat{Q})) + \mathrm{LN}(\hat{Q}) \quad \mathbf{P} \in \mathbb{R}^{N_P \times d}. \quad (4)$$

Mixer-based Predictor. Inspired by the MLP-Mixer [46] (shortened as Mixer), we design a Mixer-based predictor to transform support embedding into dynamic video prompts. The details of a Mixer-based predictor are shown in Figure 2 (c). Unlike the transformer-based predictor whose first step is to concatenate all support embeddings, we conduct average pooling in the temporal dimension to obtain the average support embeddings $\overline{\mathbf{E}}_{sup}$ as follows:

$$\overline{\mathbf{E}}_{sup} = \mathrm{AvgPool}(\mathbf{E}_{sup}) \quad \overline{\mathbf{E}}_{sup} \in \mathbb{R}^{n_L \times d_L}, \quad (5)$$

where AvgPool denotes the average pooling across the time dimension. Then, the average support embedding is normalised by a LayerNorm layer and further transposed and passed into a feed-forward network (FFN) to obtain the hidden feature h with N_P channels. Finally, the hidden feature h is used to generate N_P dynamic video prompts with d channels. These two steps can be formulated as follows:

$$h = \mathrm{FFN}((\mathrm{LN}(\overline{\mathbf{E}}_{sup}))^T) \quad h \in \mathbb{R}^{d_L \times N_P}, \quad (6)$$

$$\mathbf{P} = \mathrm{FFN}((\mathrm{LN}(h)^T) \quad \mathbf{P} \in \mathbb{R}^{N_P \times d}, \quad (7)$$

where LN and $(\cdot)^T$ denote the LayerNorm layer [1] and the transpose layer, respectively.

4.3. Prompting Stage

Given the predicted dynamic video prompts **P**, we concatenate **P** together with the patch embeddings of the current frame $\mathbf{E}_t$ as the input of the transformer encoder. The details of this process are shown in the prompting stage (right part) of Figure 2 (a). More concretely, the process of applying **P** and extracting the spatio-temporal feature of the current frame I_t can be formulated as:

$$\left[\mathbf{Z}_t^1, \mathbf{E}_t^1\right] = \mathbf{L}^1(\mathrm{Concat}(\mathbf{P}, \mathbf{E}_t)) \quad (8)$$

$$\left[\mathbf{Z}_t^i, \mathbf{E}_t^i\right] = \mathbf{L}^i(\mathrm{Concat}(\mathbf{Z}^{i-1}, \mathbf{E}_t^{i-1})) \quad i = 2, 3, ..., L \quad (9)$$

$$y_t = \mathrm{Head}(\mathbf{E}_t^L), \quad (10)$$

where $\mathrm{Concat}(\cdot)$ indicates the concatenation operation and $\mathbf{Z}_t^i \in \mathbb{R}^{N_P \times d_i}$ represents the extra embeddings produced by adding **P** in the input embeddings. The weights of the transformer encoder are shared in the predicting and prompting stages. Finally, the output embeddings $\mathbf{E}_t^L$ are passed into the head network for different video tasks.

STPN Shallow *v.s.* **STPN Deep.** Following the protocols in VPT [23], we design two STPN variants, i.e., STPN shal-
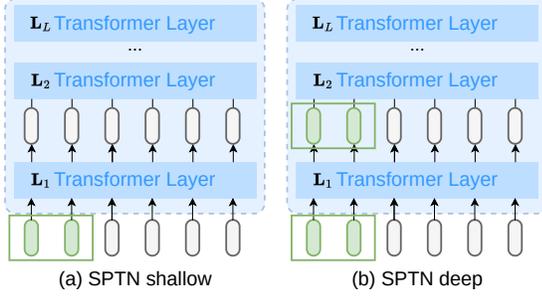
Figure 3. A comparison between the (a) STPN shallow and the (b) STPN deep. The green rounded rectangles denote predicted DVPs. In (a), one set of DVPs is prepended to the patch embeddings before the **first** layer of the transformer encoder. In (b), L sets of DVPs are predicted and then prepended to the input embeddings of **all** L transformer layers in the transformer encoder.

low $v.s.$ STPN deep, according to how many sets of dynamic video prompts (DVPs) are generated to adjust the transformer encoder. For example, in the shallow version, one set of DVPs is predicted and prepended only before the first transformer layer. In the deep version, L sets of DVPs are predicted via L separate FC layers, and each set of DVPs is prepended to the corresponding transformer layer. Formally, the DVPs of the i-th layer are denoted as $\mathbf{P}^i \in \mathbb{R}^{N_P \times d_i}$ and the feature extraction process Equation (9) is changed to:

$$\left[\ldots, \mathbf{E}_t^i\right] = \mathbf{L}^i(\text{Concat}(\mathbf{P}^{i-1}, \mathbf{E}_t^{i-1})) \quad i = 2, 3, ..., L. \quad (11)$$

The difference between the shallow and deep variants is shown in Figure 3.

5. Experiments

We first conduct experiments and compare SPTN with other state-of-the-art methods on three general video understanding tasks, i.e., video object detection (VOD), video instance segmentation (VIS) and visual object tracking (VOT). Then, in §5.3, we conduct ablation studies on how different design choices affect the overall performance, which helps to gain a deeper understanding of SPTN.

5.1. Settings

Datasets. For VOD, the ImageNetVID dataset [41] is used. It contains 3,862 training and 555 validation videos. Following previous approaches [43, 57, 66], we add overlapped 30 classes of the ImageNet DET dataset into the $train$ set. Specifically, we sample 15 frames from each video in the VID $train$ and at most 2,000 images per class from the DET $train$. For VIS, we train our models and report results on the YouTube VIS 2019 dataset [63]. The dataset contains 2,238 training, 302 validation, and 343 test videos, which covers 40 object categories. For VOT, we

choose the GOT10k [21] dataset, because it is a large-scale dataset (10,000 training videos and 180 test videos) with more than 1.5 million manually labelled bounding boxes, enabling stable training and evaluation of trackers.

Evaluation Metrics. For video object detection and video instance segmentation, we report detailed results in the COCO [31] evaluation format. Specifically, we report the average precision (AP) metric which computes the average precision over ten IoU thresholds $[0.5 : 0.05 : 0.95]$ for all categories. Meanwhile, the COCO evaluation contains other important metrics, e.g., AP_{50} and AP_{75} that are calculated at IoU thresholds of 0.50 and 0.75, and AP_S, AP_M, AP_L that are calculated on different object scales, i.e., small, medium and large. The metrics for detection and segmentation are noted with box and $mask$, respectively. For visual object tracking, we report the widely used average overlap (AO) and the success rate (SR). AO denotes the average overlaps between all ground truth and estimated bounding boxes, while SR measures the percentage of successfully tracked frames where the overlaps exceed a threshold, e.g., $SR_{0.5}$ and $SR_{0.75}$.

Transformer Encoders. Our method is compatible with different transformer backbones. For demonstration, we conduct experiments on two well-known transformer backbones, i.e., Swin [33] and CvT [58]. Swin is used for video object detection and video instance segmentation. CvT is used for visual object tracking. The details of the backbones are illustrated in the supplementary material.

Training and Inference. All reported results are obtained with Python 3.7 and PyTorch 1.8.1 on Tesla V100 GPUs. We follow the protocols in TransVOD [17], Mask2Former [9], and MixFormer [11] to set the hyper-parameters, e.g., learning rate, data augmentations, optimiser, etc., for three tasks, respectively. More details about the hyper-parameter settings are described in the supplementary material.

5.2. Comparisons with SOTA Methods

Video Object Detection. Table 2 shows the results of STPN and state-of-the-art methods. Firstly, in the AP50 metric, using a simple FasterRCNN [40] base detector with the Swin-T [33] backbone, STPN achieves 85.0% of AP50, which makes a 1.3% improvement over the best competitor TransVOD [17]. Furthermore, when changing the FasterRCNN base detector to a multi-frame aggregation method SELSA [57], STPN increases further to 86.5% of AP50. When using a more strict AP metric, STPN with FasterRCNN and SELSA achieve remarkable improvements of +6.5/6.9% over the best competitor EOVOD [44], respectively. Finally, when changing to stronger backbones, STPN with the Swin-B backbone achieves further improved

Model	Base Detector	Backbone	AP	AP50	AP75	AP_S	AP_M	AP_L
CHP [61]	CenterNet	R101	-	76.7	-	-	-	-
EOVOD [44]	FCOS	R101	54.1	79.8	59.5	10.5	28.3	60.1
RDN [13]	FasterRCNN	R101	53.4	81.8	60.1	8.5	27.4	59.6
TransVOD [17]	Deformable DETR	R101	-	82.0	-	-	-	-
MEGA [7]	FasterRCNN	R101	53.2	82.9	59.2	9.1	29.4	59.1
TDViT [45]	FasterRCNN	SwinT	50.9	79.9	55.7	9.1	26.9	57.2
SELSA* [57]	FasterRCNN	SwinT	52.3	82.3	58.2	11.4	28.8	57.9
TransVOD [17]	Deformable DETR	SwinT	-	83.7	-	-	-	-
STPN	FasterRCNN	SwinT	60.6	85.0	**68.8**	12.1	33.7	66.8
STPN	SELSA	SwinT	**60.9**	**86.5**	68.6	**13.3**	**34.8**	**67.1**
MEGA [7]	FasterRCNN	ResNeXt101	-	84.1	-	-	-	-
MAMBA [43]	FasterRCNN	ResNeXt101	-	86.7	-	-	-	-
TransVOD [17]	Deformable DETR	SwinB	-	90.0	-	-	-	-
STPN	FasterRCNN	SwinB	63.6	86.1	71.6	12.1	33.8	70.3
STPN	SELSA	SwinB	**65.2**	**90.6**	**73.5**	**14.1**	**37.7**	**71.8**

Table 2. Comparison with state-of-the-art methods on ImageNet VID for video object detection. ∗ represents our re-implementation.

Model	Backbone	AP	AP50	AP75	FPS
VisTR [55]	R101	40.1	64.0	45.0	57.7
CrossVIS	R101	36.6	57.3	39.7	35.6
MaskProp	R101	42.5	-	45.6	-
SeqFormer	R101	49.0	71.1	55.7	64.6
IFC [22]	R101	44.6	69.2	49.5	**89.4**
IDOL [60]	R101	50.1	73.1	56.1	26.0
VITA [18]	R101	51.9	75.4	57.0	-
M2F-VIS [8]	SwinT	51.5	75.0	56.5	-
MinVIS [20]	SwinT	51.9	75.1	58.2	27.1
$STPN_{MinVIS}$	SwinT	**54.8**	**78.7**	**61.0**	26.5

Table 3. Comparison with state-of-the-art methods for video instance segmentation on the YouTube VIS 2019 dataset.

Model	Backbone	AO	SR0.5	SR0.75
SiamRPN++ [29]	R101	51.7	61.6	32.5
SiamR-CNN [50]	R101	64.9	72.8	59.7
PrDiMP [12]	R101	63.4	73.8	54.3
TrDiMP [52]	R101	67.1	77.7	58.3
TREG [10]	R101	66.8	77.8	57.2
TransT [6]	R101	67.1	76.8	60.9
STARK [62]	R101	68.8	78.1	64.1
MixFormer[11]	CVT-22K	70.7	80.0	67.8
MixFormer [11]	CVT-1K	71.2	79.9	65.8
$STPN_{MixFormer}$	CVT-22K	71.8	81.6	**69.0**
$STPN_{MixFormer}$	CVT-1K	**72.7**	**81.9**	67.7

Table 4. Comparison with state-of-the-art tracking methods on the GOT-10k dataset.

performance of 65.2% AP. Compared to TransVOD, STPN with SELSA makes a 0.6% improvement in AP50.

Video Instance Segmentation. We compare STPN with state-of-the-art methods in Table 3. Using the M2F-VIS [8] as the baseline segmentation method, STPN improves performance by large margins with 2.9/3.6% of AP/AP50, respectively. Compared with SOTA methods with similar backbone complexities, for example, Swin-T or ResNet-101, STPN achieves the best accuracy. Specifically, STPN improves the performance of the best competitor VITA [18] by 2.9/3.3% of AP/AP50, respectively.

Visual Object Tracking. Table 4 compares STPN with state-of-the-art methods. We use MixFormer [11] as our baseline. Specifically, when CVT-22k and CVT-1k are used

as the backbone, STPN improves the AO of MixFormer by +1.1% and +1.5%, respectively. STPN achieves the best performance compared to other SOTA methods on the GOT-10K dataset.

These experimental results demonstrate that STPN is easy to generalise to various video understanding tasks and achieves remarkable performance across all three different video tasks.

5.3. Ablation Study

We investigate the effect of different designs of our approach on the overall performance. All experiments in §5.3 are conducted on the ImageNet VID dataset for video object detection.
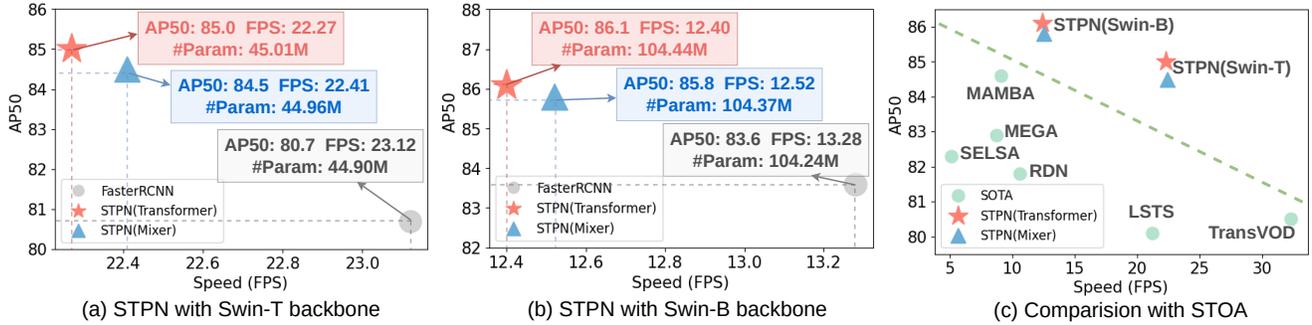
Figure 4. Comparisons between the transformer-based and the Mixer-based DVP predictors.
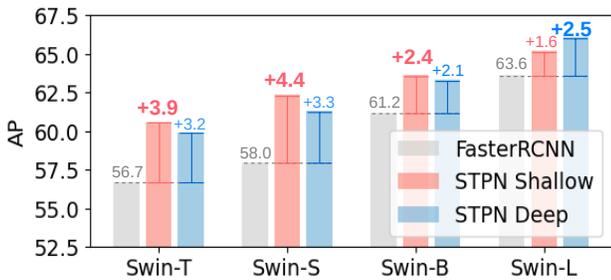

Figure 5. Comparisons of STPN deep and STPN shallow on transformer encoders with different scales.

(a)	S	1	2	4	8	16	32
	AP	58.7	59.6	60.1	**60.6**	60.5	60.5

(b)	K	1	3	5	7	9	11
	AP	52.6	53.5	58.9	**60.6**	60.6	60.7
	FPS	22.4	22.4	22.3	**22.3**	21.7	20.5

(c)	N_P	3	5	7	9	11	13
	AP	60.2	60.3	**60.6**	60.6	60.5	60.6
	FPS	22.9	22.7	**22.3**	21.6	21.0	19.9

Table 5. Effect of hyper-parameters in STPN: the temporal stride S, the number of support frames K, and the number of dynamic video prompts N_P. Different choices of hyper-parameters are listed in rows highlighted in grey.

DVP Predictor Designs. We compare the effects of DVP predictor designs on different baselines. Figure 4 (a) shows the results on the Swin-T backbone. The FasterRCNN baseline achieves 80.7% AP50, runs at 23.12 FPS, and has 44.90M parameters. Using the transformer-based predictor and the Mixer-based predictor, STPN achieves 85.0/84.5% AP50 and runs at 22.27/22.41 FPS and has 45.01/44.96M parameters, respectively. These results show that the transformer-based predictor can achieve better accuracy, while the Mixer-based predictor has less complexity and faster speed. Experiments on the Swin-B backbone are shown in Figure 4 (b) and the experimental results indicate the same conclusion.

Efficiency of STPN. Compared with other state-of-the-art methods, STPN with both predictor designs actually achieves a significantly improved speed-accuracy trade-off, as shown in Figure 4 (c). These results demonstrate that the pipeline of STPN is very efficient and is not sensitive to different predictor designs. We choose the transformer-based predictor as our default setting since it achieves a slightly higher accuracy with a comparable fast speed.

STPN Shallow *v.s.* **STPN Deep.** As shown in the Figure 5, we conduct experiments on Swin-T/S/B/L variants. The

first finding is that both STPN shallow and STPN deep can continuously improve performance on all Swin variants, indicating that STPN has good compatibility with backbone scales. Another finding which contradicts our intuition is that surprisingly STPN shallow achieves better performance than STPN deep on three Swin variants. We believe the reason is that STPN shallow has fewer parameters and thus is easier to optimise in the training process. In contrast, the Swin-L variant is much more complex and thus needs more adjustments in the intermediate transformer layers. By default, we use STPN shallow.

Support Frame Sampling. We use two hyper-parameters S and K to control the sampling of support frames. Table 5 (a) and Table 5 (b) show the effects of choosing different values of S and K, respectively. Firstly, we fix the value of K as 7, so 15 surrounding support frames are used in total. Then, we vary the temporal stride S from 1 to 32. As shown in Table 5 (a), S=8 achieves the best performance. Afterwards, with S fixed as 8, we verify the effect of different K with respect to both performance (AP) and speed (FPS). Specifically, when K=7, STPN achieves a good speed-accuracy trade-off with 60.6% AP and 22.3
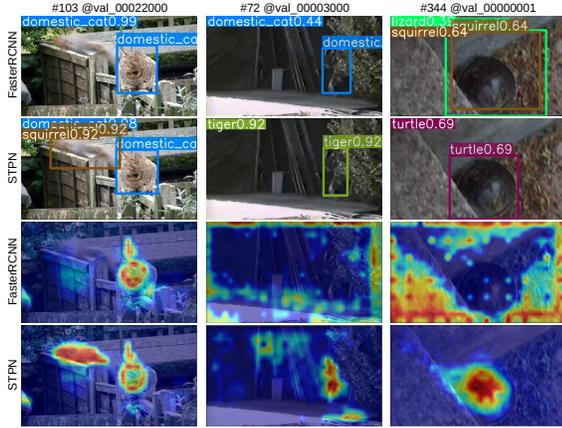
Figure 6. Bounding boxes and Grad-CAM visualisations of Faster-RCNN and STPN. Each column shows a video frame from the ImageNet VID validation set.
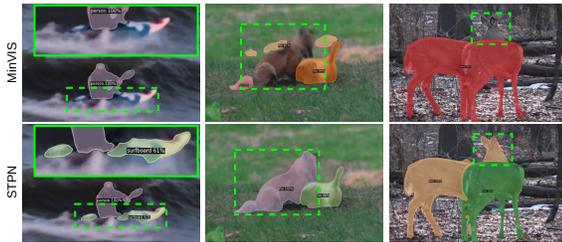

Figure 7. Instance mask visualisations of MinVIS [20] and STPN. Each column shows a video frame from the YouTube VIS dataset. MinVIS fails to generate accurate masks in low-quality frames.

FPS. Therefore, we choose $S = 8$ and $K = 7$ as default.

Number of dynamic video prompts. To explore the effect of the number of dynamic video prompts N_P, we show the performance and speed results by varying this number from 3 to 13 in Table 5 (c). The best speed-accuracy trade-off is obtained when N_P is 7. In particular, once N_P is greater than 7, AP is less affected by the change of N_P. Thus, we use 7 as the default value of N_P.

5.4. Qualitative Results

Grad-CAM [42] is used to visualise the "concentration" areas of the FasterRCNN baseline and STPN in some challenging scenarios, for example, motion blur, occlusion, and both. As shown in Figure 6, FasterRCNN either fails to detect the blurry object or generates false positive detections, while STPN can produce accurate detections. In the first column, the squirrel is very blurry because of its fast movement. As a result, FasterRCNN cannot detect the squirrel, and thus the Grad-CAM does not show much attention to the region of the squirrel. On the contrary, STPN successfully detects the squirrel, and Grad-CAM

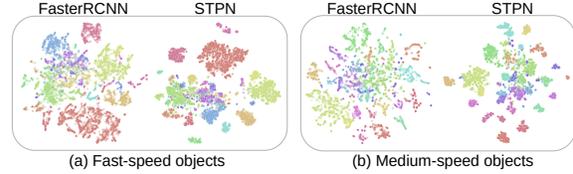(a) Fast-speed objects   (b) Medium-speed objects

Figure 8. t-SNE visualisation of the object features produced by FasterRCNN and STPN on (a) fast speed and (b) medium speed objects in the ImageNet VID dataset.

shows a good concentration map in the squirrel region.

Instance Mask Visualisation. Figure 7 shows the mask segmentation results of the state-of-the-art method MinVIS [20] and the STPN in deteriorated frames. For example, in the first and the second columns, MinVIS fails to generate instance masks for the surfboard and the fox because of motion blur but our STPN can generate accurate instance masks in these frames. The third column shows a case of occlusion. The left deer is occluded by the right one, so the MinVIS fails to generate a mask in the head area of the left dear. Besides, it cannot distinguish the two deer and generate one instance mask. In contrast, STPN works well for generating the mask of the occluded deer and distinguishing the two instances of deer.

t-SNE [48] is a statistical method for visualising high-dimensional data in 2D maps and is usually used to demonstrate the discriminative abilities of different models in classification tasks. We visualise the object features extracted by FasterRCNN and STPN using t-SNE. Figure 8 (a) and (b) show the results on the objects with the fast and medium moving speeds, respectively. We can see that STPN has better clustering results than the FasterRCNN baseline, demonstrating that STPN can improve the discriminative ability of features. Details of how to define different moving speeds and how to generate object features are illustrated in the supplementary material.

6. Conclusion

To the best of our knowledge, we are the first to explore promoting techniques for robust video feature extraction. We present a neat and unified framework, named Sptio-Temporal Prompting Network (STPN). STPN simplifies the current pipeline for video understanding tasks. Moreover, STPN is easy to generalise to various video tasks. We conduct extensive experiments and report STPN's superior performance on three widely-used video benchmarks: ImageNet VID for video object detection, YouTube for video instance segmentation, and GOT-10K for visual object tracking. We hope our work can inform future research on robust video understanding.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[2] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCV*, 2016. 2

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020. 2

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 2, 3

[5] Jianguo Chen, Kenli Li, Qingying Deng, Keqin Li, and S Yu Philip. Distributed deep learning model for intelligent video surveillance systems with edge computing. *Industrial Informatics*, 2019. 1

[6] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, 2021. 1, 2, 6

[7] Yihong Chen, Yue Cao, Han Hu, and Liwei Wang. Memory enhanced global-local aggregation for video object detection. In *CVPR*, 2020. 1, 6

[8] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. *arXiv preprint arXiv:2112.10764*, 2021. 1, 2, 6

[9] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 5

[10] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Target transformed regression for accurate tracking. *arXiv preprint arXiv:2104.00403*, 2021. 1, 2, 6

[11] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *CVPR*, 2022. 1, 2, 5, 6

[12] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *CVPR*, 2020. 6

[13] Jiajun Deng, Yingwei Pan, Ting Yao, Wengang Zhou, Houqiang Li, and Tao Mei. Relation distillation networks for video object detection. In *ICCV*, 2019. 1, 2, 6

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 2

[15] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. *NeurIPS*, 2020. 3

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 2

[17] Lu He, Qianyu Zhou, Xiangtai Li, Li Niu, Guangliang Cheng, Xiao Li, Wenxuan Liu, Yunhai Tong, Lizhuang Ma, and Liqing Zhang. End-to-end video object detection with spatial-temporal transformers. In *ACMMM*, 2021. 2, 5, 6

[18] Miran Heo, Sukjun Hwang, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. Vita: Video instance segmentation via object token association. In *NeurIPS*, 2022. 2, 6

[19] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, 2018. 1

[20] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 2, 6, 8

[21] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE transactions on pattern analysis and machine intelligence*, 43(5):1562–1577, 2019. 2, 5

[22] Sukjun Hwang, Miran Heo, Seoung Wug Oh, and Seon Joo Kim. Video instance segmentation using inter-frame communication transformers. *NeurIPS*, 2021. 1, 2, 6

[23] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022. 2, 3, 4

[24] Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *ECCV*, 2022. 2, 3

[25] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 1

[26] Hamed Kiani Galoogahi, Ashton Fagg, and Simon Lucey. Learning background-aware correlation filters for visual tracking. In *ICCV*, 2017. 1

[27] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 3

[28] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 2

[29] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 2, 6

[30] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018. 2

[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5

[32] Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi. Computing systems for autonomous driving: State of the art and challenges. *IEEE Internet of Things Journal*, 8(8):6469–6486, 2020. 1

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 3, 5

[34] Rashmika Nawaratne, Damminda Alahakoon, Daswin De Silva, and Xinghuo Yu. Spatiotemporal anomaly detection using deep learning for real-time video surveillance. *Industrial Informatics*, 16(1):393–402, 2019. 1

[35] Bolin Ni, Houwen Peng, Minghao Chen, Songyang Zhang, Gaofeng Meng, Jianlong Fu, Shiming Xiang, and Haibin Ling. Expanding language-image pretrained models for general video recognition. In *ECCV*, 2022. 3

[36] Junting Pan, Ziyi Lin, Xiatian Zhu, Jing Shao, and Hongsheng Li. St-adapter: Parameter-efficient image-to-video transfer learning for action recognition. In *NeurIPS*, 2022. 3

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 3

[38] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019. 2

[39] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021. 3

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 5

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 2, 5

[42] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 8

[43] Guanxiong Sun, Yang Hua, Guosheng Hu, and Neil Robertson. Mamba: Multi-level aggregation via memory bank for video object detection. In *AAAI*, 2021. 1, 2, 5, 6

[44] Guanxiong Sun, Yang Hua, Guosheng Hu, and Neil Robertson. Efficient one-stage video object detection by exploiting temporal consistency. In *ECCV*, 2022. 5, 6

[45] Guanxiong Sun, Yang Hua, Guosheng Hu, and Neil Robertson. Tdvit: Temporal dilated video transformer for dense video tasks. In *ECCV*, 2022. 2, 6

[46] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021. 4

[47] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, 2017. 1

[48] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 8

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2, 3

[50] Paul Voigtlaender, Jonathon Luiten, Philip H. S. Torr, and Bastian Leibe. Siam R-CNN: visual tracking by re-detection. In *CVPR*, 2020. 6

[51] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, 2021. 2

[52] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, 2021. 6

[53] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 2

[54] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 1

[55] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *CVPR*, 2021. 1, 2, 6

[56] Fei Wu, Qingzhong Wang, Jiang Bian, Ning Ding, Feixiang Lu, Jun Cheng, Dejing Dou, and Haoyi Xiong. A survey on video action recognition in sports: Datasets, methods and applications. *Multimedia*, 2022. 1

[57] Haiping Wu, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Sequence level semantics aggregation for video object detection. In *ICCV*, 2019. 1, 2, 5, 6

[58] Haiping Wu, Bin Xiao, Noel C. F. Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *ICCV*, 2021. 3, 5

[59] Junfeng Wu, Yi Jiang, Song Bai, Wenqing Zhang, and Xiang Bai. Seqformer: Sequential transformer for video instance segmentation. In *ECCV*, 2022. 2

[60] Junfeng Wu, Qihao Liu, Yi Jiang, Song Bai, Alan Yuille, and Xiang Bai. In defense of online models for video instance segmentation. In *ECCV*, 2022. 2, 6

[61] Zhujun Xu, Emir Hrustic, and Damien Vivet. Centernet heatmap propagation for real-time video object detection. In *ECCV*, 2020. 6

[62] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, 2021. 1, 2, 6

[63] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 2, 5

[64] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE access*, 8:58443–58469, 2020. 1

[65] Yuhang Zang, Wei Li, Kaiyang Zhou, Chen Huang, and Chen Change Loy. Unified vision and language prompt learning. *arXiv preprint arXiv:2210.07225*, 2022. 3

[66] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017. 5

[67] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, 2018. 2