Tube-Link: A Flexible Cross Tube Framework for Universal Video Segmentation

Xiangtai Li[1]   Haobo Yuan[1]   Wenwei Zhang[1]
Guangliang Cheng[2,3]   Jiangmiao Pang[4]   Chen Change Loy[1✉]

[1] S-Lab, Nanyang Technological University   [2] University of Liverpool   [3] SenseTime Research   [4] Shanghai AI Lab

{xiangtai.li, ccloy}@ntu.edu.sg

Abstract

Video segmentation aims to segment and track every pixel in diverse scenarios accurately. In this paper, we present Tube-Link, a versatile framework that addresses multiple core tasks of video segmentation with a unified architecture. Our framework is a near-online approach that takes a short subclip as input and outputs the corresponding spatial-temporal tube masks. To enhance the modeling of cross-tube relationships, we propose an effective way to perform tube-level linking via attention along the queries. In addition, we introduce temporal contrastive learning to instance-wise discriminative features for tube-level association. Our approach offers flexibility and efficiency for both short and long video inputs, as the length of each subclip can be varied according to the needs of datasets or scenarios. Tube-Link outperforms existing specialized architectures by a significant margin on five video segmentation datasets. Specifically, it achieves almost 13% relative improvements on VIPSeg and 4% improvements on KITTI-STEP over the strong baseline Video K-Net. When using a ResNet50 backbone on Youtube-VIS-2019 and 2021, Tube-Link boosts IDOL by 3% and 4%, respectively. Code is available at https://github.com/lxtGH/Tube-Link.

1. Introduction

The success of the Detection Transformer (DETR) [3] has inspired recent works [8, 64, 9, 47] to develop universal architectures for addressing all image segmentation tasks using the same architecture, also known as universal image segmentation. In video segmentation, the Video Panoptic Segmentation (VPS) task involves segmenting and tracking each pixel in input video clips [20, 52, 18], unifying the Video Semantic Segmentation (VSS) [30] and Video Instance Segmentation (VIS) [60] tasks. To minimize specialized architecture design for each task, recent studies [25, 21] follow a universal approach to video seg-
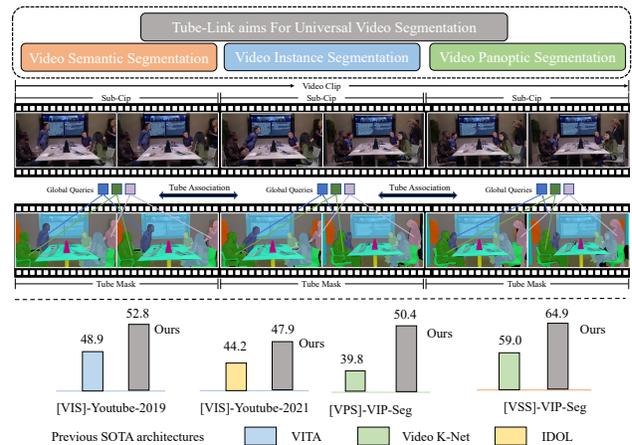

Figure 1: Tube-Link takes subclips as inputs and links the resulting tubes in a near online manner. Our design embraces flexibility, efficiency, and temporal consistency, making it suitable for various video segmentation tasks, including VSS, VIS, and VPS. Notably, our method outperforms the best-specialized architectures for these tasks on multiple datasets. Best viewed in color.

mentation by solving sub-tasks of VPS in a single unified framework. These methods typically use an end-to-end set prediction objective and successfully address multiple tasks without modifying the architecture and loss.

While recent studies have demonstrated promising results, there are still several issues with VPS models and universal video segmentation methods. One major challenge is the lack of exploration of VPS for arbitrary scenes and video clip lengths. To address this gap, Miao *et al.* [29] have introduced a more challenging benchmark, named VIPSeg, which features long videos and diverse indoor and outdoor scenes. This new dataset presents new challenges to existing VPS methods [20, 29, 25, 37], such as increased occlusions and appearance changes in diverse scenarios. Another issue with universal methods [25, 21] is that they cannot achieve comparable results to recent Transformer-based VIS methods [57, 55], which raises the question of whether we can design a universal video segmentation method to

avoid these specialized designs.

To gain a better understanding of the limitations of current solutions for video segmentation, we examine the existing methods and categorize them into two groups based on how they process input video clips: *online* and *near-online*. The former [20, 25, 57, 60, 54] performs video segmentation at the frame level, while the latter [37, 52, 21, 50, 8, 19, 55] processes clip-wise inputs and directly obtains tube-wise masks. However, there are trade-offs in deploying either approach. Although online methods offer great flexibility, they struggle to use temporal information effectively and thus compromise segmentation performance. On the other hand, near-online methods achieve better segmentation quality, but they cannot handle long video clips, and most approaches are only validated in VIS tasks, which have fewer instances and simpler scenes.

In this study, we introduce Tube-Link, a universal video segmentation framework that combines the benefits of both online and near-online methods. The framework follows a common input and output space for video segmentation tasks, where a long clip input is split into multiple subclips. Each subclip contains several frames within a temporal window, and the output is a spatial-temporal mask that tracks the target entity. Our framework is compatible with contemporary methods such as Mask2Former-VIS [7], where each global query encodes the same tracked entity, and the global queries perform cross-attention with spatial-temporal features in the decoder directly.

In particular, we propose several key improvements to the Mask2Former-VIS meta-architecture. **First**, we extend the instance query into an entity query (either thing or stuff) to improve temporal consistency for both thing and stuff segmentation, thus generalizing Mask2Former-VIS into a universal segmentation architecture. **Second**, we improve the modeling of cross-tube relationships from two different aspects through temporal consistency learning and temporal association learning. For the former, we design a simple link head with self-attention layers that links global queries across tubes to enforce segmentation consistency across tubes. For the latter, we generalize previous frame-level contrastive learning into tube-level and learn temporal association embeddings with a temporal contrastive loss. Unlike previous works [25, 57, 32] that only learn from two adjacent frames, we consider multiple frames to learn cross-tube consistency. The learned embeddings are then used to perform tube mask matching, which is much more effective than previous counterparts [25] in complex video scenarios. **Third**, with the flexibility of window size and learned association embeddings, we show that one can enlarge the subclip size to improve temporal consistency and inference efficiency, even when trained with fewer subclip inputs.

Our approach is a simple yet flexible framework that outperforms specialized architectures across various video seg-

mentation tasks. We evaluate Tube-Link on three video segmentation tasks using six datasets (VIP-Seg [29], KITTI-STEP [52], VSPW [30], YouTube-VIS-2019/2021 [60], OVIS [36]). We demonstrate that, for the first time, our single architecture performs on par or better than the most specialized architectures on five video benchmarks. In particular, as shown in Fig. 1, using the same ResNet-50 backbone, Tube-Link outperforms recently published works Video K-Net [25] 4% VPQ on KITTI-STEP, 13% VPQ, and 8% STQ on VIP-Seg, VITA [16] and IDOL [57] on YouTube-VIS-2019 by 3% mAP. We also outperform TubeFormer [21] on VPSW by 1.7% mIoU, and on YouTube-VIS-2019 by 5.3% mAP.

2. Related Work

Specialized Video Segmentation. VSS aims to predict a class label for each pixel in a video. Recent approaches [42, 70, 30, 43, 44] model the temporal consistency or acceleration using methods such as optical flow warping or spatial-temporal attention. VIS [60] extends instance segmentation into video, aiming to segment and track each object simultaneously. Several methods [1, 26, 67, 38, 23] link instance-wise features in the video. Recent studies [50, 19, 55, 57, 61, 11] have extended DETR into VIS, proposing better ways to fuse different queries along the temporal dimension. However, these methods cannot be directly transferred to complex scenes [52, 29] due to the limited instances and simpler scenes used in their training. VPS aims to generate instance tracking IDs and panoptic segmentation results across video clips. Kim *et al.* [20] mainly focus on short-term tracks, using only six frames for each clip in Cityscapes video sequences. STEP [52] proposes a Segmentation and Tracking Quality (STQ) metric that decouples the segmentation and tracking error, along with long sequence VPS datasets. Recently, VIP-Seg [29] proposed a more challenging dataset containing various scenes, scales, instances, and clip lengths. However, current solutions [20, 54, 25] for VPS mainly focus on online or near-online approaches, which have difficulty adapting to general tasks (VSS, VIS) or handling the complex scenarios in VIP-Seg dataset.

Universal Architectures For Segmentation. Recent studies [47, 64, 8, 62, 24, 63] adopt mask classification architectures with an end-to-end set prediction objective for universal segmentation, achieving better results than specialized models. In video segmentation, two representative works are Video K-Net [25] and TubeFormer [21]. The former unifies the video segmentation pipeline via kernel tracking and linking, while the latter adopts a near-online approach and obtains tube-level masks with cross-attention along the tube features and queries. However, both works fail to replace specialized models, as their performance on specific tasks or datasets is still worse than the best-specialized ar-

chitecture (they perform worse in VIS). Our proposed Tube-Link is the first generalized architecture that outperforms existing state-of-the-art specialized architectures on three video segmentation tasks. In comparison to [21], Tube-Link further explores cross-tube association, which is critical in long and complex scenes.

Video Object Detection and Tracking. Object tracking is a crucial task in VPS, and many works adopt the tracking-by-detection paradigm [2, 22, 59, 68, 35]. These methods divide the task into two sub-tasks, where objects are first detected by an object detector and then associated using a tracking algorithm. Recent works [33, 12] also perform clip-wise segmentation or tracking [41]. However, the former only focuses on single-object mask tracking, while the latter considers global tracking via clip-level matching. Our proposed Tube-Link naturally handles both settings yet provides additional segmentation masks as outputs. In Video Object Detection, the literature [69, 10, 5, 58, 66] has seen the broad usage of information across multiple frames. Our work is related to TransVOD [66], which uses a local temporal window. However, in these studies, learning correspondences and links across tubes are not explored, partly due to the nature of ImageNet-VID dataset [40], which does not require object tracking and segmentation.

3. Methodology

3.1. Preliminary and Motivation

In this subsection, we begin by introducing a unified notation for universal video segmentation (VSS, VIS, and VPS), and then demonstrate how this task can be modeled through linking the tracked short tube masks. After that, we further motivate tube-wise matching by comparing it against conventional frame-wise matching.

Universal Video Segmentation Formulation. We denote a video clip input as $V \in \mathbb{R}^{T \times H \times W \times 3}$, where T represents the frame number and $H \times W$ are the spatial size. The video clip is annotated with segmentation masks. The masks of a particular entity can be linked along the time dimension to form a tube. The annotations are denoted as $\{y_i\}_{i=1}^{G} = \{(m_i, c_i)\}_{i=1}^{G}$, where the G is the number of ground truth, each tube mask $m_i \in \{0, 1\}^{T \times H \times W}$ does not overlap with each other, and c_i denotes the ground truth class label of the i-th tube. The background is assigned with value 0, and the foreground masks are assigned to be 1 in each tube mask m_i. VPS requires temporally consistent segmentation and tracking results for each pixel. Specifically, a model makes predictions on a set of video clips $\{\hat{y}_i\}_{i=1}^{N} = \{(\hat{m}_i, \hat{p}_i(c))\}_{i=1}^{N}$, where $\hat{m}_i \in [0, 1]^{T \times H \times W}$ denotes the predicted tube, and $\hat{p}_i(c)$ denotes the probability of assigning class c to a clip $\hat{m}_i$ belonging to a predefined category in a set C. The number of entities is given by N, which includes countable thing classes and countless stuff

Table 1: **Exploration experiment on tube-wise matching.** Youtube-VIS: mAP. VIP-Seg:VPQ. We directly use pre-trained models by changing the input to two consecutive frames.

Method	Youtube-VIS-2019	Youtub-VIS-2021	VIP-Seg
Min-VIS [17]	47.4	44.2	-
Min-VIS + tube matching	48.8 (+1.4)	45.5 (+1.3)	-
Video K-Net [25]	-	-	26.1
Video K-Net + tube matching	-	-	27.6 (+1.5)

classes. In particular, i is the tracking ID for the thing class. When $N = C$ and C only contain stuff classes, VPS turns into VSS. If $\{\hat{y}_i\}_{i=1}^{N}$ can overlap and C only contains the thing classes, VPS turns into VIS. We use such notations for universal video segmentation.

Video Segmentation as Linking Short Tubes. To segment a video clip V, we divide it into a set of L smaller sub-clips: $\{v_i\}_{i=1}^{N}$, where $v_i \in \mathbb{R}^{n \times H \times W \times 3}$, and $n = T/L$ with n representing the window size along the temporal dimension. The window size is flexible and can be adjusted according to the dataset. By taking subclips as inputs, we obtain shorter segmentation tubes than the whole clip. We perform tracking and linking across nearby tubes. Each predicted tube is represented as $\{\hat{y}_i^t\}_{i=1}^{N} = \{(\hat{m}_i^t, \hat{p}_i(c)^t)\}_{i=1}^{N}$, where t is the index of each small tube, $\hat{m}_i^t \in [0, 1]^{n \times H \times W}$ represents the segmentation masks, and $\hat{p}_i(c)^t$ is the corresponding category. The final video prediction is obtained by linking each tube, $\{\hat{y}_i\}_{i=1}^{N} = \text{Link}(\{\hat{y}_i^t\}_{i=1}^{N})_{t=1,..L}$. In our formulation, tracking is only performed across different tubes, and the segmentation within each tube is assumed to be consistent. Note that the key to achieving temporally consistent segmentation is the design of function Link.

Motivation of Tube-wise Matching. Existing video segmentation methods [57, 17, 25] often perform instance association via frame-wise matching. This approach ignores local temporal information and can lead to occlusion errors. In this study, we propose to perform tube-wise matching using global queries based on their corresponding trackers. To motivate our approach, we modify the input of two representative works, Min-VIS [17] and Video K-Net [25], by replacing the single frame input with a subclip input. Each subclip contains two frames. Without additional re-training or computation costs, we observe consistent improvements on three video segmentation datasets and two video segmentation tasks, VIS and VPS, as shown in Table 1. These findings suggest that cross-tube information is worth exploring for achieving universal segmentation. Therefore, we propose to model the Link function as tube-wise matching and linking, and term our framework Tube-Link.

3.2. Tube-Link Framework

We first introduce our extension to Mask2Former-VIS, which will serve as a baseline. Then, we present two improvements, including cross-tube Temporal Contrastive
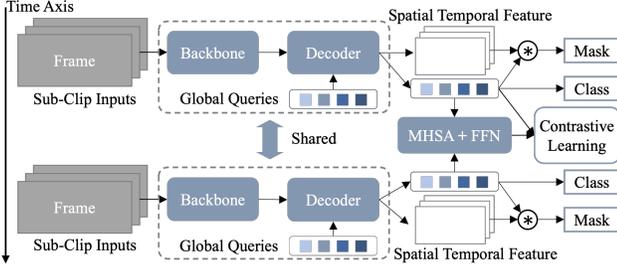
Figure 2: Our proposed Tube-Link framework. Given the sub-clips as input, we first use global queries to perform within-tube spatial-temporal learning to obtain the tube masks and labels. Then we link the object queries with cross-tube contrastive learning and self-attention.

Learning (TCL) and Cross-Tube Linking (CTL), to better model cross-tube relations. The training process of our method is illustrated in Fig. 2.

Mask2Former-VIS Extension as Baseline. Following [50, 7], given a subclip v_i, we first employ Mask2Former-VIS [7] to extract the spatial-temporal feature $\mathbf{F}_i \in \mathbb{R}^{n \times C \times H \times W}$. Mask2Former uses multiscale features and a cascaded decoder to perform cross-attention. Thus, we denote the layer index l to indicate the layer number. The *global queries*, $\mathbf{Q}_{l-1} \in \mathbb{R}^{N_q \times C}$, perform masked cross-attention between $\mathbf{F}_i$ and $\mathbf{Q}_i$ as follows,

$$\mathbf{Q}_l = \mathrm{softmax}(\mathbf{M}_{l-1} + \mathrm{MLP}(\mathbf{Q}_{l-1})\mathbf{K}_l^T)\mathbf{V}_l + \mathbf{Q}_{l-1}, \tag{1}$$

where N_q is the query number and is set to 100 by default, $\mathbf{M}_{l-1} \in \mathbb{R}^{n \times H \times W}$ is the binarized output of the resized tube-level mask prediction from the previous stage, following Cheng *et al.* [8]. Instead of considering only thing masks as in previous works [50, 55], we jointly process both thing and stuff masks, shown in Equation (1). MLP denotes linear layers to transform the object query, while $\mathbf{K}_l$ and $\mathbf{V}_l$ represent the spatial-temporal features transformed from $\mathbf{F}_i$, where $\mathbf{K}_l = \mathrm{Key}(\mathbf{F}_i)$ and $\mathbf{V}_l = \mathrm{Value}(\mathbf{F}_i)$, Key and Value are linear functions as in the common attention design. In the implementation, F_i is sampled from the multiscale feature output following Mask2Former, and we use the feature with the highest resolution for simple formulation purposes.

Within each tube, the query index is naturally the tracking ID for each object with no tracking association within the tube. This process is shown in the dash box area of the Fig. 2.

Cross-Tube Temporal Contrastive Learning. Recent studies [25, 57, 32] have demonstrated the effectiveness of contrastive learning in video segmentation. However, all of these studies perform learning on *only two adjacent frames*, which is not suitable for tube-level matching. To capture a larger temporal context, we propose cross-tube temporal
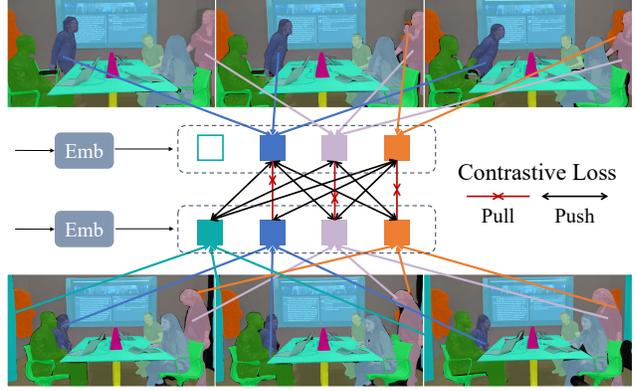

Figure 3: Illustration of the proposed cross-tube temporal contrastive learning. The input queries are first sent to the Emb. Then cross-tube contrastive learning is performed. The same instance across different frames is indicated in the same color. We use spatial-temporal ground truth masks to perform positive/negative assignments to each query embedding.

contrastive learning. While temporal contrastive learning is not new, our study is the first attempt to perform contrastive learning at the tube level.

As shown in Fig. 2, given a pair of subclips v_i and v_j as inputs, we first perform tube-level inference to obtain the global queries, $\mathbf{Q}_i$ and $\mathbf{Q}_j$, corresponding to two tubes. Note that both queries already encapsulate spatial-temporal information through Equation (1). We randomly select two subclips from the neighborhood of all subclips, assuming that they contain corresponding objects (i.e., objects with the same IDs). We then add an extra lightweight embedding head Emb after each global query to learn the association embedding, which is implemented through several fully-connected layers. Following Li *et al.* [25], we use a mask-based assignment for contrastive learning.

Different from previous frame-wise methods, we propose a tube-wise label assignment strategy to form contrastive targets. Recall that our query embedding encodes information from more than one single frame, we use spatial-temporal masks (the same instance masks within the tube) for mask-based assignment, as shown in Fig. 3. Specifically, we define a query embedding as positive to one object if its corresponding *tube mask* has an IoU higher than α_1, and negative if the IoU is lower than α_2. We set α_1 and α_2 as 0.7 and 0.3, respectively. We use a sparse set of matched global queries for learning, where the query indices are assigned from ground truth tube masks.

We assume that there are X matched queries from $\mathbf{Q}_i$ and Y matched queries from $\mathbf{Q}_j$ as contrastive targets, where both X and Y are *much fewer than* all queries N.

Table 2: System-level comparison between Tube-Link with related approaches [25, 21, 57].

Property and Settings	Video K-Net [25]	TubeFormer [21]	IDOL [57]	Our Tube-Link
Online	✓	✓	✓	✓(n=1)
Nearly Online		✓		✓
VIS	✓	✓	✓	✓
VSS	✓	✓		✓
VPS	✓	✓		✓
Mulitple Frames	✓		✓	✓
Frame Query Matching	✓		✓	
Mask Mathicing		✓		
Global Query Matching				✓

The cross-tube temporal contrastive loss is written as:

$$L_{\text{track}} = -\sum_{\mathbf{y}^+} \log \frac{\exp(\mathbf{x} \cdot \mathbf{y}^+)}{\exp(\mathbf{x} \cdot \mathbf{y}^+) + \sum_{\mathbf{y}^-} \exp(\mathbf{x} \cdot \mathbf{y}^-)}, \quad (2)$$

where $\mathbf{x}$, $\mathbf{y}^+$, $\mathbf{y}^-$ are query embeddings of tube pairs, their positive targets, and negative targets, which are sampled from $\mathbf{Q}_i$ and $\mathbf{Q}_j$, respectively, as illustrated in Fig. 2. The loss pulls positive embeddings close to each other and pushes the negative away, as shown in Fig. 3. In addition, following previous work [32, 25], we also adopt L2 loss as an auxiliary loss to regularize the global query association process.

$$L_{\text{track\_aux}} = \left(\frac{\mathbf{x} \cdot \mathbf{y}}{||\mathbf{x}|| \cdot ||\mathbf{y}||} - b \right)^2, \quad (3)$$

where b is 1 if there is a match between the two samples, and 0 otherwise. Compared with previous works [25, 57] for contrastive learning, our loss considers additional temporal information, thus achieving a much better result. Experiments are reported in Sec. 4.3.

Cross-Tube Linking. Apart from the improved supervision at the tube level, we take a further step to link tubes via their global queries, $\mathbf{Q}_i$ and $\mathbf{Q}_j$ for both training and inference. This encourages the interactions among tubes along the temporal dimension. We adopt a Multi-Head Self Attention (MHSA) layer with a Feed Forward Network (FFN) [46] to learn the correspondence among each query to obtain the updated queries, allowing full correlation among queries. This process is shown as follows:

$$\mathbf{Q}_j^f = \text{FFN}(\text{MHSA}(\text{Query}(\mathbf{Q}_j), \text{Key}(\mathbf{Q}_i), \text{Value}(\mathbf{Q}_i))). \quad (4)$$

In this way, the information from the i-th tube is propagated to the j-th tube via the affinity matrix. The linked output $\mathbf{Q}_j^f$ is employed as the input to the embedding head Emb, shown in the *middle* of Fig. 2 and the left of the Fig. 3.

Relation with Previous Works. We summarize the differences and properties of Tube-Link with previous methods in Table 2. Compared to Tubeformer [21], Tube-Link explores cross-tube relationships from a global query matching perspective, while Tubeformer adopts simple mask matching. A detailed experimental comparison can be found in

Sec. 4. Compared to IDOL [57] and Video K-Net [25], our method explores multiple-frame information and global query matching, which is more robust for complex scenes and temporal consistency. If $n = 1$, Tube-Link degenerates into the naïve online method. It is noteworthy that since we process the video by sampling n frames as input, we can obtain a much faster inference speed than online approaches via more efficient GPU memory usage and parallelization (see Sec. 4.4).

3.3. Training and Inference of Tube-Link

Training and Loss Function. In addition to the tracking loss in Equation (2), we also use tube-wise segmentation loss. Specifically, we obtain the tube masks by stacking the mask of the same instance from different frames. This establishes a one-to-one mapping between the predicted tube-level mask and the ground-truth tube-level mask based on the masked-based matching cost [3, 8]. The tube-level masks are obtained from global queries $\mathbf{Q}$ like Mask2Former [8]. The final loss function is given as $L = \lambda_{cls}L_{t\_cls} + \lambda_{ce}L_{t\_ce} + \lambda_{dice}L_{t\_dice} + \lambda_{track}L_{track} + \lambda_{aux}L_{track\_aux}$. Here, L_{t_ce} is the Cross Entropy (CE) loss for tube-level classification, and L_{t_ce} and L_{t_dice} are tube-level mask Cross Entropy (CE) loss and Dice loss [31, 49] for segmentation, respectively. L_{track} and L_{track_aux} are the tracking losses. For VSS, we remove the L_{track} term.

Simplicity. We intend to keep a simple framework, and thus we ***do not*** use any extra tricks or auxiliary losses employed in previous works, such as extra semantic segmentation loss [47], tube-level mask copy and paste [21], joint image dataset and video dataset pre-training [55, 57], etc.

Inference. Taking VPS as an example, we use Tube-Link to generate tube-level panoptic segmentation masks from each input. We use the query embeddings from the learned embedding head Emb as association features and feed them as input to Quasi-Dense Tracker [32] in a near-online manner. Note that we only track the preserved instance masks from the panoptic segmentation maps, and the matching process is performed at *tube-level* between two global queries. Unlike previous studies [53, 51], we do not use extra motion cues to help track across subclips. Instead, we only use simple query feature matching across subclips, which saves time for track matching compared to online methods. An advantage of Tube-Link is its flexible inference via different subclip size settings n. Enlarging n can improve both inference speed and performance on most datasets. For complex and high-resolution inputs, we decrease n by only utilizing information within the temporal vicinity. We perform detailed ablations in Sec. 4.3. We will provide more details on inference for other tasks (VIS, VSS) in the appendix.

4. Experiments

4.1. Experimental Settings

Dataset. We conduct experiments on five video datasets: VIPSeg [29], VSPW [30], KITTI-STEP [52], and YouTube-VIS-19/21 [60]. We mainly conduct experiments on VIPSeg due to its scene diversity and long-length clips. The training, validation, and test sets of VIPSeg contain 2,806/343/387 videos with 66,767/8,255/9,728 frames, respectively. Although VSPW and VIPSeg share the same video clips, the training details are different since they are different tasks. Please refer to the *supplementary material* for other datasets.

Evaluation Metrics. For the VPS task, we adopt two metrics: VPQ [20] and STQ [52]. The metric STQ contains geometric mean of two items: Segmentation Quality (SQ) and Association Quality (AQ), where $STQ = (SQ \times AQ)^{\frac{1}{2}}$. The former evaluates the pixel-level tracking, while the latter evaluates the pixel-level segmentation results in a video clip. For the VSS task, the Mean Intersection over Union ($mIoU$) and mean Video Consistency (mVC) [30] are used for reference. For the VIS task, mAP is adopted.

Implementation Details and Baselines. We implement our models in PyTorch [34] with the MMDetection toolbox [4]. We use the distributed training framework with 16 V100 GPUs. Each mini-batch has one image per GPU. Following previous work, we use the image baseline pre-trained on COCO dataset [27]. ResNet [15], STDC [13], and Swin Transformer [28] are adopted as the backbone networks, which are pre-trained on ImageNet, and the remaining layers adopt the Xavier initialization [14]. For the detailed settings of other datasets, pretraining, and fine-tuning, please refer to the *supplementary material*. To further verify the effectiveness of our approach, we build a stronger baseline by unifying Video K-Net with Mask2Former, where we replace the image encoder with Mask2Former. We term it Video K-Net+. We denote the extended Mask2Former-VIS for VPS as Mask2Former-VIS+.

4.2. Benchmark Results

[VPS] Results on VIPSeg. We present the results of our Tube-Link method compared to previous works on the VIPSeg dataset in Tab. 3. Our approach outperforms Video K-Net[25] (under the same backbone) with 12%-15% VPQ and 7%-10% STQ improvements, respectively. Notably, our method with Swin-base [28] backbone achieves new state-of-the-art results. We also evaluate our method using a lightweight backbone [13] for more efficient inference on video clips, and it achieves even better results than all previous methods with a larger ResNet50 backbone. These results demonstrate the effectiveness of our approach in exploiting temporal information. Benefiting from the joint

Table 3: **Results on VIPSeg-VPS [29] validation dataset.** We report VPQ and STQ for reference. Following Miao *et al.* [29], we report VPQ scores at different window sizes (1, 2, 4, 6). We report the results obtained from either an efficient or a strong backbone for comparison.

Method	backbone	VPQ^1	VPQ^2	VPQ^4	VPQ^6	VPQ	STQ
VIP-DeepLab [37]	ResNet50	18.4	16.9	14.8	13.7	16.0	22.0
VPSNet [20]	ResNet50	19.9	18.1	15.8	14.5	17.0	20.8
SiamTrack [54]	ResNet50	20.0	18.3	16.0	14.7	17.2	21.1
Clip-PanoFCN [29]	ResNet50	24.3	23.5	22.4	21.6	22.9	31.5
Video K-Net [25]	ResNet50	29.5	26.5	24.5	23.7	26.1	33.1
Video K-Net+ [8, 25]	ResNet50	32.1	30.5	28.5	26.7	29.1	36.6
Video K-Net [25]	Swin-base	43.3	40.5	38.3	37.2	39.8	46.3
Tube-Link	STDCv1	32.1	31.3	30.1	29.1	30.6	32.0
Tube-Link	STDCv2	33.2	31.8	30.6	29.6	31.4	32.8
Tube-Link	ResNet50	41.2	39.5	38.0	37.0	39.2	39.5
Tube-Link	Swin-base	54.5	51.4	48.6	47.1	50.4	49.4

Table 4: **Results on the YouTube-VIS datasets.** We report the mAP metric. † adopt COCO video pseudo labels. Axial means using the extra Axial Attention [48]. Our method does not apply these techniques for simplicity.

Method	Backbone	YTVIS-2019	YTVIS-2021
VISTR [50]	ResNet50	36.2	-
TubeFormer [21]	ResNet50 + Aixal	47.5	41.2
IFC [19]	ResNet50	42.8	36.6
SeqFormer [55]	ResNet50	47.4	40.5
Mask2Former-VIS [7]	ResNet50	46.4	40.6
IDOL [57]	ResNet50	46.4	43.9
IDOL [57] †	ResNet50	49.5	-
VITA [16] †	ResNet50	49.8	45.7
Min-VIS [17]	ResNet50	47.4	44.2
Tube-Link	ResNet50	52.8	47.9
SeqFormer [55]	Swin-large	59.3	51.8
Mask2Former-VIS [7]	Swin-large	60.4	52.6
IDOL [57]	Swin-large	61.5	56.1
IDOL [57]	Swin-large †	64.3	-
VITA [16] †	Swin-large	63.0	57.5
Min-VIS [17]	Swin-large	61.6	55.3
Tube-Link	Swin-large	64.6	58.4

Table 5: **Results on VSPW-VSS validation set.** mVC_c means that a clip with c frames is used.

VPSW	Backbone	mIoU	mVC_8	mVC_{16}
DeepLabv3+ [6]	ResNet101	35.7	83.5	78.4
TCB(PSPNet) [30, 65]	ResNet101	37.5	86.9	82.1
Video K-Net (Deeplabv3+) [25, 6]	ResNet101	37.9	87.0	82.1
Video K-Net (PSPNet) [25, 65]	ResNet101	38.0	87.2	82.3
MRCFA [44]	MiT-B5	49.9	90.9	87.4
CFFM [43]	MiT-B5	49.3	90.8	87.1
TubeFormer [21]	Axial-ResNet50x64	63.2	92.1	88.0
Tube-Link	ResNet50	42.3	86.8	83.2
Tube-Link	Swin-large	59.7	90.3	88.4

inference of subclips, our method achieves a much faster inference speed, as shown in Fig. 4.

[VIS] Results on YouTube-VIS-2019/2021. In Tab. 4, we compare our method with state-of-the-art VIS methods on the YouTube-VIS 2019 and 2021 datasets. Our method achieves a 3.0% and 2.2% mAP gain over VITA [16] when using the ResNet50 backbone. Furthermore, compared with the Mask2Former-VIS baseline [7], our method achieves

Table 6: **Results on VIP-Seg-VSS validation set**. mVC_c means that a clip with c frames is used.

VPSW	Backbone	mIoU	mVC_8	mVC_{16}
Video K-Net (Deeplabv3+) [25, 6]	ResNet101	38.3	88.0	83.1
Video K-Net (PSPNet) [25, 65]	ResNet101	39.0	88.2	84.2
Mask2Former [8]	ResNet50	38.4	87.5	82.5
Video K-Net+ [8, 25]	Swin-base	57.2	90.1	87.8
Tube-Link	ResNet50	43.4	89.2	85.4
Tube-Link	Swin-base	62.3	91.4	89.3
Tube-Link	Swin-large	64.9	92.4	89.9

Table 7: **Results on the KITTI val set.** OF refers to an optical flow network [45].

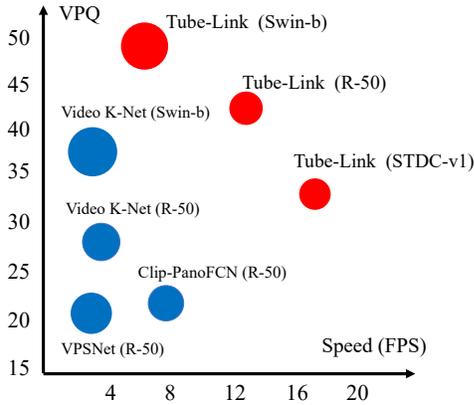| KITTI-STEP	Backbone	OF	STQ	AQ	SQ	VPQ
P + Mask Propagation	ResNet50	✓	0.67	0.63	0.71	0.44
Motion-Deeplab [52]	ResNet50		0.58	0.51	0.67	0.40
VPSNet [20]	ResNet50	✓	0.56	0.52	0.61	0.43
TubeFormer-DeepLab [21]	ResNet-50 + Axial		0.70	0.64	0.76	0.51
Video K-Net [25]	ResNet50		0.71	0.70	0.71	0.46
Video K-Net [25]	Swin-base		0.73	0.72	0.73	0.53
Tube-Link	ResNet50		0.68	0.67	0.69	0.51
Tube-Link	Swin-base		0.72	0.69	0.74	0.56

Figure 4: Tube-Link also achieves the best accuracy and speed trade-off on VIP-Seg dataset. FPS is measured on RTX GPU.

4-5% mAP gains on the two datasets with different backbones. Our method also outperforms the previous near-online method TubeFormer [21] by 5-6% in terms of mAP on the two VIS datasets.

[VSS] Results on VSPW and VIP-Seg. We further conduct experiments on VSPW dataset [30] for VSS to demonstrate the generalization of Tube-Link. As shown in Tab. 5, our method achieves over 4% mIoU improvement compared to the Mask2Former baseline. Under the same ResNet101 backbone, our method achieves the best results. Using the Swin base backbone, our method achieves about 3.7% mIoU gains over Video K-Net+ with consistent improvements on mVC. Our method with a lightweight backbone achieves comparable results to DeepLabv3+ with ResNet101, but with about four times faster inference speed (shown in Fig. 6). Without using any additional techniques,

our method also outperforms recent methods specifically designed for VSS [43, 44]. In Tab. 6, we also compare the video semantic segmentation methods in recent VIPSeg datasets with higher-resolution images. Compared with previous state-of-the-art methods, our approaches also achieve state-of-the-art results.

[VPS] Results on KITTI STEP. We further validate our method on KITTI STEP [52] and report the results in Tab. 7. Our method achieves 0.51 VPQ with the ResNet50 backbone, setting a new state-of-the-art result *without* using temporal attention or optical flow warping. When using a strong Swin-base [28] backbone, our method still achieves better results than Video K-Net [25] by 3% VPQ and comparable results on STQ. It is worth noting that one can further improve the performance of Tube-Link by employing a better tracker design.

4.3. Ablation Study and Visual Analysis

Improvements over Strong VPS Baseline. In Tab. 8a, we demonstrate the effectiveness of each component proposed in Sec. 3.2. The first row shows the results of the frame matching baseline. After adopting the tube matching, we obtain a gain of 1.6% VPQ_{th} and 2.1% on VPQ, even without any specific tracking design, which results in the same observation as shown in Tab. 1. Thus, we use Mask2Former-VIS+ (T, T=2) as our baseline by default, which achieves a strong starting point of 34.5 VPQ. VPQ_{th} refers to the VPQ for the thing class. This result shows the effectiveness of the naïve framework. The addition of TCL further boosts performance, with a gain of 3.5% on VPQ_{th} and 1.7% on VPQ. Furthermore, adding CTL, which makes the association more consistent, improves VPQ_{th} by 1.5%.

Ablation on Temporal Contrastive Loss. We also compare our TCL design with previous works that use dense queries [32] or sparse queries [25] for matching. Both settings use only one frame, while our subclip size is two. As shown in Tab. 8b, our method achieves the best results since tube matching encodes more temporal information. In particular, we observe 3.0% VPQ improvements compared to the strong Video K-Net baseline.

Ablation on Association Target Assignment. In Table 8c, we show the results of the ablation study on building association targets. We find that using a tube-level mask achieves the best results. Using the mask from one of the input subclips leads to inferior results. This is because the ground truth masks of a single frame are not aligned with the input global queries, where the global queries are learned from multiple frames using Equation (1).

Effect of Sub-clip Size for Training. In Tab. 8d, we investigate the impact of subclip size on training. Tube-Link becomes an online method when the subclip size is 1. As shown in the table, enlarging the subclip size improves the performance. We also examine overlapping during sam-

Table 8: **Ablation studies and comparative analysis on VIPSeg validation set with the ResNet50 backbone.**

(a) Ablation Study on Each Component.

baseline	TCL	CTL	VPQ$_{th}$	VPQ
Mask2Former-VIS+ (F)	-	-	29.4	32.4
Mask2Former-VIS+ (T)	-	-	31.0	34.5
	✓	-	34.6	36.8
	✓	✓	35.1	37.5

(b) Design Choices of TCL.

Method	VPQ	STQ
Dense Query [32]	30.2	30.1
Sparse Query [25]	34.5	35.1
Global Query(Ours)	37.5	36.5

(c) Association Target Assign.

Method	VPQ	STQ
All-Masks [32]	30.1	29.2
GT-Mask [25]	35.6	35.9
Tube-Mask	37.5	36.5

(d) Input Sub-clip Size with Tube Window Size of 2 as Input.

Clip Size	STQ	VPQ	VPQ$_{th}$
T=1	34.5	35.6	30.2
T=2	36.5	37.5	35.1
T=2(ovl)	35.9	37.3	35.0
T=3	36.4	37.0	35.3

(e) Tube-Window for Inference with Input Sub-clip Size 2 for Training.

Window Size	STQ	VPQ	VPQ$_{th}$
W=2	36.5	37.5	35.1
W=4	39.2	39.0	38.2
W=6	39.5	39.2	38.9
W=8	38.3	38.5	37.3

(f) Tracking Choices with the Default Setting of Tab.(d).

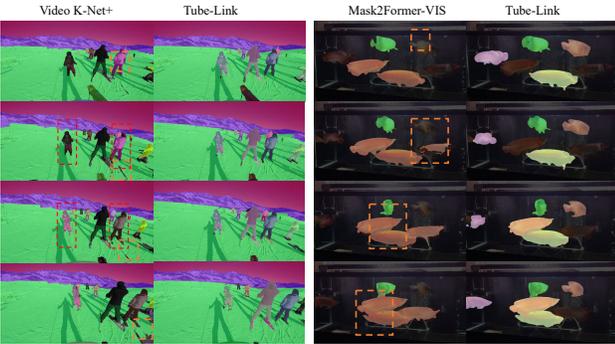| Settings	STQ	VPQ	VPQ$_{th}$
Extra Tracker [51, 53]	33.9	36.6	34.1
RoI Features [32]	34.5	35.9	34.5
Query Embedding [25]	33.1	36.0	33.0
Our Tube embedding	36.5	37.5	35.1

Figure 5: Comparison results on VIP-Seg and YuoTube-VIS. Our method achieves consistent segmentation (shown in orange boxes) and better tracking results (shown in red boxes).
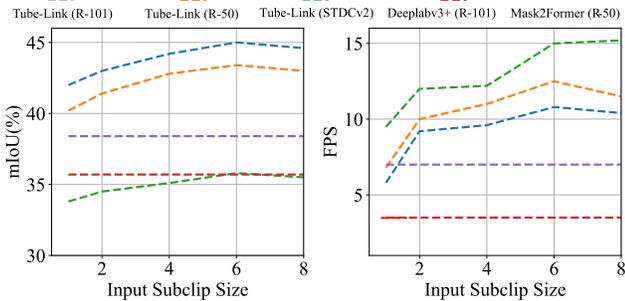

Figure 6: Efficiency Analysis of Tube-Link. Left: Segmentation results (mIoU) of VSPW with different subclip sizes. Right: Inference speed (FPS) with different subclip sizes.

pling, denoted as ovl, where two input subclips overlap at one frame. As shown in Tab. 8d, enlarging the subclip size to 2 achieves significant improvement. However, we find that either frame overlapping or using a larger subclip size ($T = 3$) does not bring extra gains. Adding more frames

does not benefit temporal association learning, since most instances are similar within a subclip. Moreover, using more frames is not memory-friendly during training. Thus, the subclip size is set to 2. We can enlarge the size for inference, as shown in Tab. 8e.

Effect of Sub-clip Size for Inference. During training, the subclip size is limited due to memory constraints, but we can expand it during inference to improve the performance. For instance, we use a subclip size of 2 during training and increase it to 6 during inference. Tab. 8e shows that enlarging the subclip size for inference improves the performance considerably for all three metrics: STQ, VPQ, and VPQ$_{th}$. However, when the subclip size is further increased to 8, the performance drops because the global queries are not designed to handle larger subclips. Increasing the subclip size can also speed up the inference process by utilizing the full GPU memory, as demonstrated in Fig. 6.

Different Tracking Choices. In Tab. 8f, we compare different tracking approaches used in previous studies [32, 25, 53] with our Tube-Link. Our Tube-Link only uses the learned tube-level embedding for the association. We find that the default tube embedding works best in our framework, without requiring any association embedding head or RoI crop operation on the VIPSeg dataset.

4.4. Visualization and More Analysis

GFLops and Parameter Analysis. Compared with Mask2Former baseline, we only add one Emb head and one self-attention layer, introducing only 2.2% GFLops and 1.4% extra parameters with 720×1280 input.

Speed and Accuracy with Different Input Subclip Size. As shown in Table 8e, adding more frames improves the VPS results. To further analyze the speed-accuracy trade-off, we present a detailed comparison of different methods on the VSPW dataset in Fig. 6. The left plot shows that enlarging the subclip size also improves the VSS results.

The right plot illustrates that increasing the subclip size improves the single-frame baseline by 1.25-1.5% for various backbones. Both performance and speed reach a plateau when the size increases to 6. The experiment justifies our choice of using an input subclip size of 6 for inference.

Visual Improvements on Baselines. In Fig. 5, we present the visual comparison with several strong baselines (Video K-Net+ and Mask2Fomer-VIS) in VPS and VIS settings. The results are randomly sampled from a long clip. We achieve better results on both segmentation and tracking. More visual examples can be found in the supplementary material.

5. Conclusion

We present Tube-Link, a simple yet flexible universal video segmentation framework. Our key insight is to perform cross-tube matching rather than cross-frame matching. By equipping the Mask2Former architecture with the proposed cross-tube learning, Tube-Link achieves new state-of-the-art results in all three major video segmentation tasks (VSS, VIS, VPS) using one unified architecture.

Limitation and Future Work. Tube-Link is pre-trained on image datasets and requires re-training for each new video dataset. In the future, we hope to develop a model that can be trained only once to unify universal image and video segmentation tasks. One potential solution can be training our Tube-Link with the merged image and video datasets using CLIP [39, 56] text embedding to unify labels. Then, we can build a universal image/video model for various scenes.

Broader Impact. Our work pushes the boundary of video panoptic segmentation algorithms in a simple, flexible, and efficient way. The proposed framework provides a unified and general solution for dense video segmentation, which has the potential to greatly simplify and expedite model development in various real-world applications that heavily rely on video input, including autonomous driving and robot navigation.

References

[1] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *CVPR*, 2020. 2

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1, 5

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint*, 2019. 6

[5] Kai Chen, Jiaqi Wang, Shuo Yang, Xingcheng Zhang, Yuanjun Xiong, Chen Change Loy, and Dahua Lin. Optimizing video object detection via a scale-time lattice. In *CVPR*, 2018. 3

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 6, 7

[7] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. *arXiv preprint*, 2021. 2, 4, 6

[8] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 1, 2, 4, 5, 6, 7

[9] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021. 1

[10] Jiajun Deng, Yingwei Pan, Ting Yao, Wengang Zhou, Houqiang Li, and Tao Mei. Relation distillation networks for video object detection. In *ICCV*, 2019. 3

[11] Henghui Ding, Chang Liu, Shuting He, Xudong Jiang, and Chen Change Loy. Mevis: A large-scale benchmark for video segmentation with motion expressions. In *ICCV*, 2023. 2

[12] Henghui Ding, Chang Liu, Shuting He, Xudong Jiang, Philip HS Torr, and Song Bai. MOSE: A new dataset for video object segmentation in complex scenes. In *ICCV*, 2023. 3

[13] Mingyuan Fan, Shenqi Lai, Junshi Huang, Xiaoming Wei, Zhenhua Chai, Junfeng Luo, and Xiaolin Wei. Rethinking bisenet for real-time semantic segmentation. In *CVPR*, 2021. 6

[14] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010. 6

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[16] 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

[17] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 3, 6

[18] Juana Valeria Hurtado, Rohit Mohan, Wolfram Burgard, and Abhinav Valada. Mopt: Multi-object panoptic tracking. In *CVPR Workshops*, 2020. 1

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

[20] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 1, 2, 6, 7

[21] Dahun Kim, Jun Xie, Huiyu Wang, Siyuan Qiao, Qihang Yu, Hong-Seok Kim, Hartwig Adam, In So Kweon, and Liang-Chieh Chen. Tubeformer-deeplab: Video mask transformer. In *CVPR*, 2022. 1, 2, 3, 5, 6, 7

[22] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. Learning by tracking: Siamese cnn for robust target association. In *CVPR Workshops*, 2016. 3

[23] Xiangtai Li, Hao He, Yibo Yang, Henghui Ding, Kuiyuan Yang, Guangliang Cheng, Yunhai Tong, and Dacheng Tao. Improving video instance segmentation via temporal pyramid routing. *T-PAMI*, 2022. 2

[24] Xiangtai Li, Shilin Xu, Yibo Yang, Guangliang Cheng, Yunhai Tong, and Dacheng Tao. Panoptic-partformer: Learning a unified model for panoptic part segmentation. In *ECCV*, 2022. 2

[25] Xiangtai Li, Wenwei Zhang, Jiangmiao Pang, Kai Chen, Guangliang Cheng, Yunhai Tong, and Chen Change Loy. Video k-net: A simple, strong, and unified baseline for video segmentation. In *CVPR*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[26] Huaijia Lin, Ruizheng Wu, Shu Liu, Jiangbo Lu, and Jiaya Jia. Video instance segmentation with a propose-reduce paradigm. *ICCV*, 2021. 2

[27] 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. 6

[28] 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. *ICCV*, 2021. 6, 7

[29] Jiaxu Miao, Xiaohan Wang, Yu Wu, Wei Li, Xu Zhang, Yunchao Wei, and Yi Yang. Large-scale video panoptic segmentation in the wild: A benchmark. In *CVPR*, 2022. 1, 2, 6

[30] Jiaxu Miao, Yunchao Wei, Yu Wu, Chen Liang, Guangrui Li, and Yi Yang. Vspw: A large-scale dataset for video scene parsing in the wild. In *CVPR*, 2021. 1, 2, 6, 7

[31] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, 2016. 5

[32] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, 2021. 2, 4, 5, 7, 8

[33] Kwanyong Park, Sanghyun Woo, Seoung Wug Oh, In So Kweon, and Joon-Young Lee. Per-clip video object segmentation. In *CVPR*, 2022. 3

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: an imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 6

[35] Lorenzo Porzi, Markus Hofinger, Idoia Ruiz, Joan Serrat, Samuel Rota Bulo, and Peter Kontschieder. Learning multi-object tracking and segmentation from automatic annotations. In *CVPR*, 2020. 3

[36] Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip Torr, and Song Bai. Occluded video instance segmentation: A benchmark. *IJCV*, 2022. 2

[37] Siyuan Qiao, Yukun Zhu, H. Adam, A. Yuille, and Liang-Chieh Chen. Vip-deeplab: Learning visual perception with depth-aware video panoptic segmentation. *CVPR*, 2021. 1, 2, 6

[38] Zheyun Qin, Xiankai Lu, Xiushan Nie, Yilong Yin, and Jianbing Shen. A graph matching perspective with transformers on video instance segmentation. In *CVPR*, 2022. 2

[39] 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. 9

[40] 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. *IJCV*, 2015. 3

[41] Woo Sanghyun, Park Kwanyong, Wug Oh Seoung, So Kweon1 In, and Lee Joon-Young. Tracking by associating clips. In *ECCV*, 2022. 3

[42] Evan Shelhamer, Kate Rakelly, Judy Hoffman, and Trevor Darrell. Clockwork convnets for video semantic segmentation. In *ECCV*, 2016. 2

[43] Guolei Sun, Yun Liu, Henghui Ding, Thomas Probst, and Luc Van Gool. Coarse-to-fine feature mining for video semantic segmentation. In *CVPR*, 2022. 2, 6, 7

[44] Guolei Sun, Yun Liu, Hao Tang, Ajad Chhatkuli, Le Zhang, and Luc Van Gool. Mining relations among cross-frame affinities for video semantic segmentation. *ECCV*, 2022. 2, 6, 7

[45] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 7

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 5

[47] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021. 1, 2, 5

[48] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *ECCV*, 2020. 6

[49] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. Solo: Segmenting objects by locations. In *ECCV*, 2020. 5

[50] 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. 2, 4, 6

[51] Zhongdao Wang, Hengshuang Zhao, Ya-Li Li, Shengjin Wang, Philip HS Torr, and Luca Bertinetto. Do different tracking tasks require different appearance models? *NeurIPS*, 2021. 5, 8

[52] M. Weber, J. Xie, M. Collins, Yukun Zhu, P. Voigtlaender, H. Adam, B. Green, A. Geiger, B. Leibe, D. Cremers, Aljosa Osep, L. Leal-Taixé, and Liang-Chieh Chen. Step: Segmenting and tracking every pixel. *NeurIPS*, 2021. 1, 2, 6, 7

[53] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017. 5, 8

[54] Sanghyun Woo, Dahun Kim, Joon-Young Lee, and In So Kweon. Learning to associate every segment for video panoptic segmentation. In *CVPR*, 2021. 2, 6

[55] Junfeng Wu, Yi Jiang, Song Bai, Wenqing Zhang, and Xiang Bai. Seqformer: Sequential transformer for video instance segmentation. In *ECCV*, 2022. 1, 2, 4, 5, 6

[56] Jianzong Wu, Xiangtai Li, Shilin Xu, Haobo Yuan, Henghui Ding, Yibo Yang, Xia Li, Jiangning Zhang, Yunhai Tong, Xudong Jiang, Bernard Ghanem, and Dacheng Tao. Towards open vocabulary learning: A survey. *arXiv pre-print*, 2023. 9

[57] 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. 1, 2, 3, 4, 5, 6

[58] Fanyi Xiao and Yong Jae Lee. Video object detection with an aligned spatial-temporal memory. In *ECCV*, 2018. 3

[59] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *ICCV*, 2019. 3

[60] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 1, 2, 6

[61] Shusheng Yang, Xinggang Wang, Yu Li, Yuxin Fang, Jiemin Fang, Wenyu Liu, Xun Zhao, and Ying Shan. Temporally efficient vision transformer for video instance segmentation. In *CVPR*, 2022. 2

[62] Qihang Yu, Huiyu Wang, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. k-means mask transformer. In *ECCV*, 2022. 2

[63] Haobo Yuan, Xiangtai Li, Yibo Yang, Guangliang Cheng, Jing Zhang, Yunhai Tong, Lefei Zhang, and Dacheng Tao. Polyphonicformer: Unified query learning for depth-aware video panoptic segmentation. *ECCV*, 2022. 2

[64] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation. In *NeurIPS*, 2021. 1, 2

[65] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 6, 7

[66] Qianyu Zhou, Xiangtai Li, Lu He, Yibo Yang, Guangliang Cheng, Yunhai Tong, Lizhuang Ma, and Dacheng Tao. Transvod: End-to-end video object detection with spatial-temporal transformers. *TPAMI*, 2022. 3

[67] Feng Zhu, Zongxin Yang, Xin Yu, Yi Yang, and Yunchao Wei. Instance as identity: A generic online paradigm for video instance segmentation. In *ECCV*, 2022. 2

[68] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *ECCV*, 2018. 3

[69] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017. 3

[70] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *CVPR*, 2017. 2