End-to-End Video Instance Segmentation via Spatial-Temporal Graph Neural Networks

Tao Wang[1], Ning Xu[2], Kean Chen[1], and Weiyao Lin[1]*
[1]Shanghai Jiao Tong University, Shanghai, China
[2]Adobe Research,[†] San Jose, USA
wang_tao1111@sjtu.edu.cn, nxu@adobe.com, ckadashuaige@sjtu.edu.cn, wylin@sjtu.edu.cn

Abstract

*Video instance segmentation is a challenging task that extends image instance segmentation to the video domain. Existing methods either rely only on single-frame information for the detection and segmentation subproblems or handle tracking as a separate post-processing step, which limit their capability to fully leverage and share useful spatial-temporal information for all the subproblems. In this paper, we propose a novel graph-neural-network (GNN) based method to handle the aforementioned limitation. Specifically, graph nodes representing instance features are used for detection and segmentation while graph edges representing instance relations are used for tracking. Both inter and intra-frame information is effectively propagated and shared via graph updates and all the subproblems (*i.e. *detection, segmentation and tracking) are jointly optimized in an unified framework. The performance of our method shows great improvement on the YoutubeVIS validation dataset compared to existing methods and achieves 36.5% AP with a ResNet-50 backbone, operating at 22 FPS.*

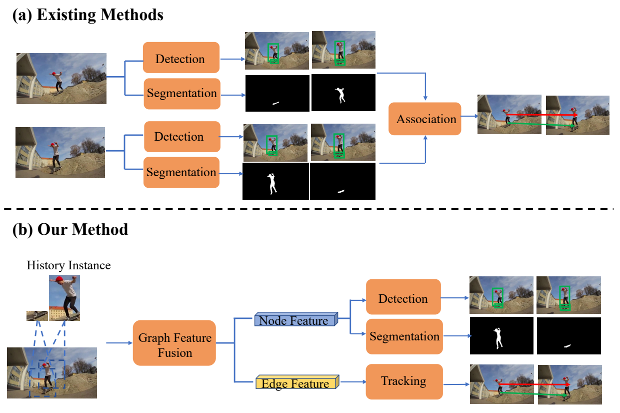


Figure 1. Comparison between existing methods and our methods. (a) Most existing methods solves VIS subproblems separately and ignore temporal feature fusion. (b) In our method, we use graph neural network to model the instance-level relations and fuse spatial-temporal features. Meanwhile, detection, segmentation and tracking results are jointly predicted based on graph features.

1. Introduction

Video instance segmentation (VIS) is a challenging and fundamental task in computer vision. In the image domain, instance segmentation needs to simultaneously detect and segment object instances [10], while in the time domain, the video instance segmentation [32] is much more challenging since it also requires accurate tracking of objects across the entire videos.

Existing VIS methods typically adopt two different strategies to tackle the instance tracking task. The first strategy is to adopt the tracking-by-detection framework which first predicts candidate detection and segmentation frame by frame with a sophisticated image instance segmentation model, and then associates these candidates by classification or re-identification to generate mask sequences [5, 18, 32], as shown in Figure 1 (a). Its performance heavily relies on the qualities of image-level instance segmentation, the property of similarity metrics and the association strategy. The other strategy is to predict clip-level instance masks by propagate instance masks from central frame to the whole video clip and merge these clip-level sequences to generate video-level results [2]. Such propagation process usually relies on some heuristics and thus a lot of post-processing and refinement operations are necessary.

In addition, both the strategies have one common limitation that their tracking is a separate step from detection and segmentation modules, forbidding useful information shared across different tasks. For example, a newly de-

*Corresponding Author, Email: wylin@sjtu.edu.cn

[†]The paper is supported in part by the following grants: National Key Research and Development Program of China Grant (No.2018AAA0100400), National Natural Science Foundation of China (No. 61971277), and Adobe Gift Funding.

tected instance associated with an existing sequence whose class prediction label is "dog" should be very helpful to predict the class of that instance, meaning that better tracking results should contribute to better detection or segmentation. Vice versa, improved detection and segmentation should also contribute to improved tracking. However, existing methods cannot easily leverage such benefits.

Another common limitation of existing methods is that they ignore inter-frame and intra-frame instance relation information. The inter-frame instance relations refer to the relationship between instances of different frames while the intra-frame instance relations refer to the relationship between instances within the same frame. Such inter and intra-frame relations usually contain rich spatial-temporal information which is useful for all VIS tasks. However, many existing methods directly detect and segment instance on single frame features. They [2, 5, 18, 32] also associate and track each new candidate with existing instance sequences independently, without having a global view from other candidate instances. A few recent methods [1, 2] have noticed the problem but they directly fuse the features from adjacent frames at the entire frame level, which could cause inaccurate information propagation and thus affect the accuracy negatively. In addition, they only leverage such information for detection and segmentation, but not for tracking.

We believe that both sharing information among different subtasks and extract inter and intra-frame information are crucial for the VIS task. Therefore, in this paper we propose a novel framework to achieve the two points at the same time, which is illustrated in Figure 1 (b). Given a pair of reference frame and target frame, our method first builds a graph neural network (GNN) to connect the two frames where nodes represent instance candidates while edges represent instance relations. Then spatial-temporal features can be obtained via graph message passing. Our detection and segmentation branches which are inspired by recent anchor-free image detection and segmentation methods [36, 22] leverage the updated node features to predict detection and segmentation of the target frame. While the tracking branch takes as input the updated edge features and performs binary classification to predict associations. Through iterative GNN updates, the tracking information contained in graph edges and the detection/segmentation information contained in graph nodes are also shared. Our model is trained end-to-end and during inference, it is applied iteratively over pairs of consecutive frames to obtain detection, segmentation and tracking results simultaneously.

Another novelty of our method is the segmentation branch. Given that the GNN features are high-level which may not contain enough shape information which are more useful for mask prediction, we propose a novel mask-information propagation module to warp the low-level instance shape features of the reference frame to the target frame. By combining the warped features with the shape feature of the target frame, our mask prediction head can achieve better segmentation results.

We train and evaluate our method on YoutubeVIS dataset and our model with a ResNet-50 backbone achieves 36.5% AP which is superior than most existing methods. Our method is also very efficient given its unified framework, which operates at a 22 FPS speed.

In summary, our method has four main contributions:

- We present a novel and efficient GNN-based framework for VIS to simultaneously detect, segment and associate instances.
- We propose to leverage inter-frame and intra-frame features via feature aggregation in GNN, which is proved to be effective for all VIS subtasks.
- We propose a mask-information propagation module to fuse the historical shape information for more accurate mask prediction.
- We evaluate our method on the benchmark dataset and achieve competitive results compared to existing methods.

The rest of our paper is organized as follows. In Section 2, we briefly introduce related tasks and state the difference between our method and existing VIS methods. In Section 3 we describe our approach in detail. Experiments results are presented in Section 4 and we conclude the paper in Section 5.

2. Related Work

Image Instance Segmentation: The aim of instance segmentation is to group pixels into different object instances to get classes, segmentation masks and unique identities of objects[10]. Most of the instance segmentation can be divided into two-stage methods and one-stage methods. Two-stage methods always generate detection bounding boxes first and predict masks on these boxes. Mask R-CNN [11] is still the mostly widely used two-stage framework in image instance segmentation and video instance segmentation, which generate object proposal by region proposal network and use two seperate branches to predict bounding boxes and segmentation masks on ROI features obtained from proposal areas on the feature map. Mask Scoring R-CNN [13] designs a MaskIOU branch to evaluate the accuracy of mask, rather than directly using the classification score in Mask R-CNN. PANet [25] use bottom-up path augmentation, adaptive feature pooling, and fully-connected fusion to improve the accuracy of segmentation. In one-stage instance segmentation methods, masks are directly predicted from images without generating a large

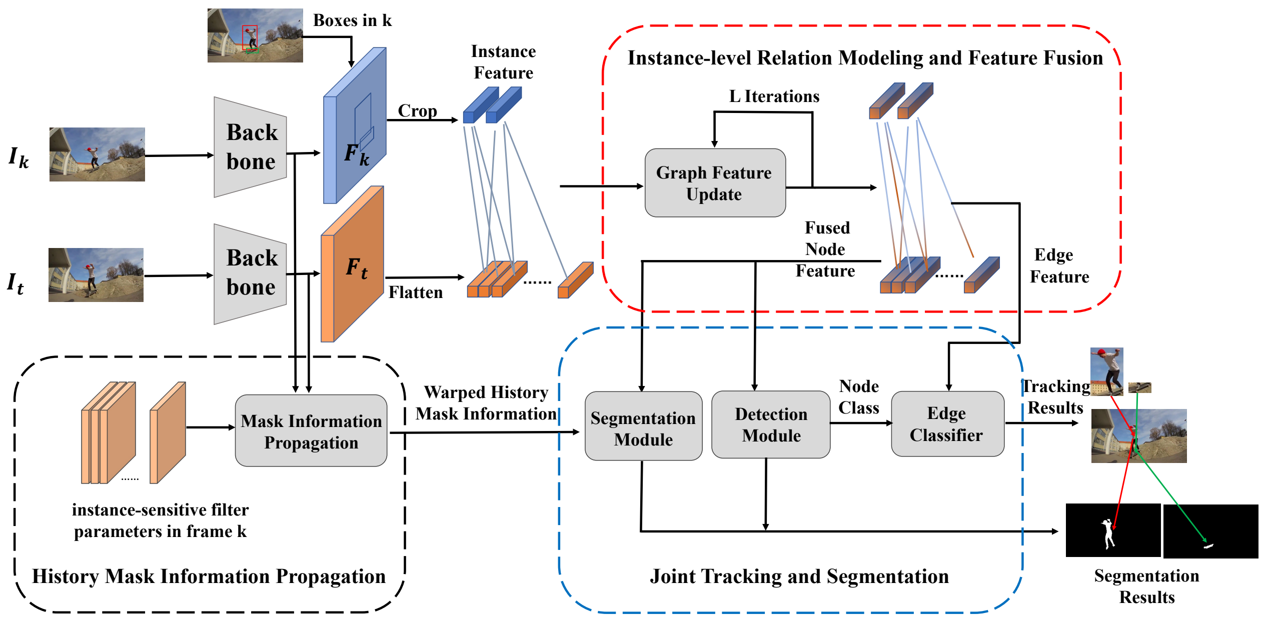


Figure 2. The overview of our framework. Given a reference frame I_k and a target frame I_t, our method first construct a graph between the two frames and perform message passing to obtain aggregated spatial-temporal features for the target frame, which is shown in the red box. Then, our method predicts the detection and segmentation results of the target frame based on graph node features and predict tracking associations based on graph edge features, which is shown in the blue box. In addition, our segmentation branch has a mask-information propagation module (shown in the black box) to propagate the instance shape information of the reference frame to the target frame, which can improve the segmentation accuracy. All the modules are trained end-to-end and our method handles VIS detection, segmentation and tracking in an unified framework. For detailed structure of individual modules, please refer to Figure 3 and 4.

amount of proposals first. PolarMask [30] uses 16 rays to represent a mask and add these representation on FCOS [23] framework to achieve instance segmentation. CondInst [22] introduces conditional convolution into FCOS and uses dynamic instance-sensitive convolution filters to represent the masks. YOLACT [3] uses a group of templates and coefficients to represent masks and designs a fully convolutional network to predict them.

Video instance segmentation: The video instance segmentation task is more challenging than video object segmentation[16], which requires classifying, segmenting instances in each frame and linking the same instance across frames [32]. It can be applied in many other computer vision fields, including human behavior analysis[17] and object position estimation[34]. The baseline method MaskTrack R-CNN [32] adds a embedding branch on Mask R-CNN [11] and uses a historical instance feature queue to associate instance masks across frame. Maskprop [2] propagates masks in previous frames to predict their positions and shapes in the later frames. VisTR [27] introduces transformers and uses queries on encoding features to obtain instance sequence from video clips. SipMask [5] designs a novel image instance segmentation method and uses the same association strategy of MaskTrack R-CNN [32] to link instance masks across frames. STEm-Seg [1] models video clips as 3D space-time volume and generate instance sequences by clustering pixels based on space-time embedding. We found that most of these methods tracks and segments each instance separately, without extracting and utilizing relation information between instances, while our method use instance-level relations extracted by graph neural network to guide the feature fusion and learn joint detection, segmentation as well as tracking in one framework, which is not only more efficient but also can improve the accuracy of all VIS subproblems.

Graph Neural Networks for Relation and Tracking: Graph neural network was first proposed to process data with a graph structure using neural networks [9]. In GNN, a graph with nodes and edges relating each other is constructed and features on nodes and edges will be updated based on relations. GNN has already been used in object detection and multi object tracking to improve the accuracy. In object detection, GNN is often used to fuse the features of related objects or proposals to increase the detection accuracy [26, 6, 21, 31]. As for MOT, most of the methods formulates data association as an edge classification problem using GNNs [4, 14, 15, 28], where each node represents a detection or a sequence of an object and each edge denotes the similarity between detection and sequences. After aggregating node features and edge features based on object relations, the final association results can be obtained by classifying the edges. However, in MOT, tracking is usually a separate step from other tasks such as detection. While in our method graph is leveraged to fuse spatial-temporal features which are useful for not only tracking but also other tasks such as detection and segmentation .

3. Approach

To simultaneously detect, segment and associate instances in an unified framework and leverage useful inter-frame and intra-frame features, we introduce GNN to model the instance-level relations. Specifically, the feature maps of the current frame are aggregated with instance features of the previous frame via the GNN (Sec. 3.2) [26]. After some iterations of graph convolution and updates, the aggregated node features of the current frame will be used to predict detection (Sec. 3.3) and segmentation results (Sec. 3.4), while edge features will be used to predict tracking results (Sec. 3.5). In addition, to better leverage the mask predictions of previous frame, we introduce another novel module (Sec. 3.4) to warp instance-sensitive mask filters of previous frame to the current frame to obtain more accurate mask prediction. The whole framework of our method is shown in Figure 2.

3.1. Problem Definition

During training, given a pair of frames I_k and I_t of a video at time k, t, and existing instance sequences at k including bounding box detections denoted as $B_k = \{B_k^1, B_k^2, ..., B_k^m\}$ and masks denoted as $M_k = \{M_k^1, M_k^2, ..., M_k^m\}$ where m is the number of existing instances, we aim at obtaining a set of instance candidates at frame t with detections $B_t = \{B_t^1, B_t^2, ..., B_t^n\}$ and masks $M_t = \{M_t^1, M_t^2, ..., M_t^n\}$ where n is the number of new instances which can be different from m. In addition, we need to assign the n instances to the existing m instance sequences or initialize unmatched candidates to new instance sequences. During inference, we iterate the above process over pairs of consecutive frames (*i.e.* I_{t-1} and I_t) to obtain the prediction of a whole video sequence. Our method is also able to handle the case where an existing instance sequence is occluded at frame I_{t-1} but reappears at frame I_t, which will be detailed in Sec. 3.6.

3.2. Spatial-Temporal Feature Fusion via GNN

Different from images, video contains rich information across different frames and different spatial locations. Given the feature maps $F_k, F_t \in \mathcal{R}^{H\times W\times D}$ of frame k and t extracted by a CNN encoder (*e.g.* ResNet-50), our method aims to produce an aggregated feature map $\hat{F}_t$ for frame t which fuses both inter-frame (F_k) and intra-frame (F_t) features for better prediction of the detection and segmentation subproblems than using the single-frame feature

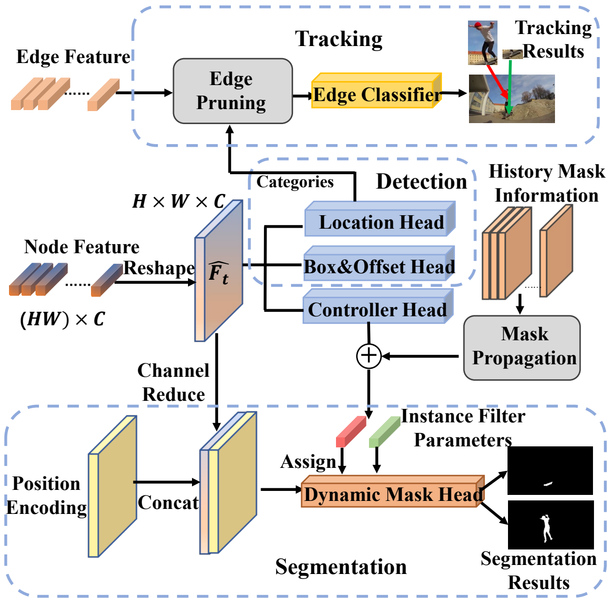


Figure 3. The detailed structure of our detection, segmentation and tracking branches.

F_t. In addition, we also want to predict tracking results simultaneously. All the goals are achieved by a GNN which is explained as follows.

Graph Construction: Our graph $G(V, E)$ consists of a node set V and an edge set E. The node definition is different for frame k and t since at frame k we already know the instance candidates while at frame t we do not. Therefore, for frame k we define each instance candidate as a node. The node feature is defined as the cropped feature of F_k by ROIAlign [11] and further downsampled and flattened to one D-dimensional feature vector (as shown in Fig. 2). For frame t, inspired by anchor-free image detection methods such as CenterNet [36], we define each pixel location (x, y) of F_t as a node and $F_t(x, y) \in \mathcal{R}^D$ as its node feature. Despite the different definition, we use h_i^v to denote the feature of node i in set V.

An edge e_{ij} in set E represents an undirected connection between node i and j and h_{ij}^e denotes the edge feature which is initialized as $|h_i^v - h_j^v|$. Our graph is sparsely connected with the following two criteria. First, only nodes across frames can be connected (as shown in Fig. 2). Second, for an instance node at frame k whose center is (x, y), it is only connected to the nodes at frame t whose distances to (x, y) is smaller than a threshold (*e.g.* a $w \times w$ window around the center).

There are three motivations for our choice. The first reason is the memory and computation concern since a dense graph will cost too many resources [26]. Secondly, our method uses edge features to predict tracking results and thus only cross-frame edges are meaningful. Thirdly, the displacement of an instance between nearby frames is usually local. It also should be noted that although our graph only has inter-frame connections, the iterative graph updates can still aggregate useful intra-frame features into the fused feature map $\hat{F}_t$, which will be explained next.

Graph Feature Update: We follow the mechanism of previous message passing networks [8, 4] to update our graph. Specifically, at each iteration, the node and edge features are propagated and updated in the following order: 1) edge features are updated by its two endpoints, 2) node features are updated by its connected edges, the process of which can be represented as follows.

$$h_{ij}^{e(l)} = N_e([h_{ij}^{e(l-1)}, h_i^{v(l-1)}, h_j^{v(l-1)}]) \tag{1}$$

$$h_i^{v(l)} = \sum_{j|e_{ij}\in E} N_v([h_{ij}^{e(l)}, h_i^{v(l-1)}]) \tag{2}$$

where N_e and N_v denote two learnable functions which are two multilayer perceptrons (MLP) in our experiments. $[\cdot]$ denotes the concatenation operation. After total L iterations, the updated node feature at frame t are reshaped to $\mathbb{R}^{H\times W\times D}$ to create the aggregated feature $\hat{F}_t$ which will be used to predict detection and segmentation results while the updated edge features will be used to predict tracking results.

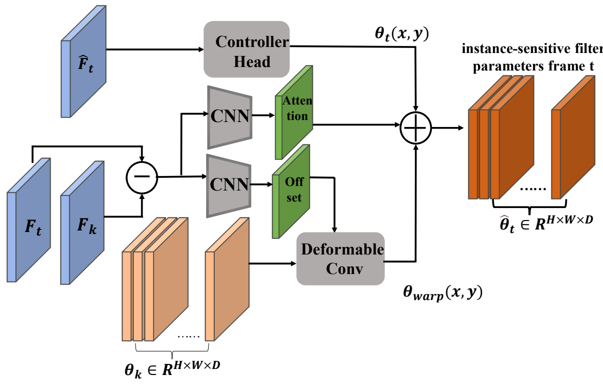
It is worth noting that as long as the graph update iterations $L \geq 2$, the updated node and edge features will contain both inter-frame and intra-frame features even though our graph has no intra-frame connections. Since a node will receive the information of its nearby nodes of the same frame after their connected common node of the other frame aggregates all their information after the first iteration. Such design enables our graph to be sparsely connected and thus requires less computation resources while still manage to extract useful spatial-temporal features.

3.3. Detection With Node Features

We use the aggregated spatial-temporal feature map $\hat{F}_t$ for the detection branch. We follow the anchor-free image detection method CenterNet [36] to predict object categories and locations by finding their center points and bounding box sizes. Specifically, $\hat{F}_t$ is fed into a three-convolutional head to obtain the location heat map, size map and coordinate-refinement map. The location heat map estimates instance center points and their categories. The size map estimates bounding box sizes. The refinement map helps refine the locations of center points. The loss of our detection branch follows that in CenterNet [36] which consists of the center-point loss, size loss and offset loss:

$$L_{\text{det}} = L_{\text{center}} + \lambda_{\text{size}} L_{\text{size}} + \lambda_{\text{offset}} L_{\text{offset}} \tag{3}$$

where λ_{size} and λ_{offset} balances the three losses.



$\hat{F}_t$
Controller Head
$\theta_t(x,y)$
instance-sensitive filter parameters frame t
CNN
Attention
CNN
Offset
F_t
F_k
Deformable Conv
$\theta_{warp}(x,y)$
$\hat{\theta}_t \in R^{H\times W\times D}$
$\theta_k \in R^{H\times W\times D}$

Figure 4. The structure of our warping module within the segmentation branch. The module uses deformable convolution [7] to warp historical instance convolution kernels and fuse them with those predicted by the controller head to obtain final instance-sensitive filter parameters.

3.4. Segmentation Branch

Inspired by recent image instance segmentation method [22] which predicts dynamic convolution filters for each instance candidate, our segmentation branch also has a controller head to predict such instance-sensitive convolution filters $\theta_t \in \mathcal{R}^{H\times W\times D}$. The convolution filter $\theta_t(x,y) \in \mathcal{R}^D$ at an instance location (x,y) can then be used as the 1×1 convolution kernels in the mask head (*i.e.* a fully-convolutional network (FCN)) to predict the corresponding instance mask (as shown in Figure 3). One difference is that our controller head leverages the spatial-temporal features $\hat{F}_t$ which contains richer information than that of a single frame.

Furthermore, in our setting, the instance-sensitive kernels θ_k at the reference frame k which represent historical instance mask and shape information are also available, which could be useful to refine the segmentation results at frame t. Therefore we propose another novel module to warp θ_k to the current frame based on the feature differences of the two frames. Specifically, as shown in Figure 4, given the two feature maps F_k and F_t, their difference is fed into a CNN to estimate a group of offsets. The offsets are reshaped to two 3×3 filters for x and y axis and then used as deformable convolution parameters. The input of the deformable convolution is θ_k while the output is the warped kernels θ_{warp} for the current frame. The final instance-sensitive kernels $\hat{\theta}_t$ for frame t is the combination of the two kernels θ_t and θ_{warp} via the attention mask. Our warping module shares a similar spirit to the alignment operation in MaskProp [2] but has two major differences. First, our warping operates at the feature level while MaskProp warps at the mask level. Second, our method simultaneously warps all instance filters while MaskProp warps one instance mask at a time, and thus is less efficient than ours.

Finally, to predict an instance mask at location (x, y), our mask head takes as inputs the merged instance-sensitive kernel $\hat{\theta}_t(x, y)$, the spatial-temporal feature map $\hat{F}_t$ as well as the position encoding map $P_{x,y}$ [22] as follows.

$$M_{x,y} = \text{MaskHead}([\hat{F}_t^d, P_{x,y}]; \hat{\theta}_t(x, y)) \tag{4}$$

where $[\cdot]$ denotes the concatenate operation, $\hat{F}_t^d$ is generated from $\hat{F}_t$ by 1×1 convolution to reduce the channel dimension to 8. $P_{x,y}$ is a two-channel position map which encodes the relative x-axis and y-axis offsets from (x, y) to the other locations. Our MaskHead operation is the same as [22]. In addition, we define the loss function for the segmentation branch L_{mask} as the dice loss [19] between predicted instance masks and ground truth masks. It can be seen that our segmentation branch leverages two types of spatial-temporal information, *i.e.* both the aggregated semantic feature via GNN and the historical shape-related kernels from previous frame.

3.5. Tracking With Edge Features

Our tracking branch utilizes the graph edges to associate existing tracks or initialize new tracks. However, since many graph nodes at frame t are classified as background in the detection branch, it is redundant to still keep these edges and thus they are removed prior to the tracking prediction. Moreover, some graph nodes at frame t have different class predictions with their connected graph nodes at frame k. These are also meaningless edges since no instance can change its class. Therefore we also remove this type of edges from the graph. The remaining edges are kept for tracking prediction. Specifically, for each remaining edge e_{ij}, its final edge feature $h_{ij}^{e(L)}$ is fed into a binary classifier to predict the association relationship between node i at frame k and node j at frame t. A positive prediction indicates the two nodes belong to the same instance and vice versa. Therefore the loss for our tracking branch is defined as:

$$L_{\text{edge}} = \frac{-1}{|E_{\text{prune}}|} \sum_{e_{ij} \in E_{\text{prune}}} y_{ij} \log(\hat{y}_{ij}) + (1 - y_{ij}) \log(1 - \hat{y}_{ij}) \tag{5}$$

where $\hat{y}_{ij}$ is the prediction for edge e_{ij} and E_{prune} denotes the pruned edge set after removing the redundant edges.

A positive prediction indicates that the node j is the extension of node i in current frame and thus should be associated to the same track. For those instances with multiple matched edges, our method only keeps the edges with the highest classification score and remove the remaining ones. While if an instance of frame t is not matched to any existing sequence after edge classification, a new track based on this instance will be initialized.

3.6. Training and Testing

Training: We randomly select two frames I_k and I_t from a video sequence as the reference and target frame, where the time gap between k and t should be smaller than 5. We use the ground truth detections of the reference frame to extract its node features. We use the controller head to obtain the instance-sensitive convolution filters of the reference frame. The total loss function of our model can be written as:

$$L_{\text{total}} = \lambda_1 L_{\text{det}} + \lambda_2 L_{\text{mask}} + \lambda_3 L_{\text{edge}} \tag{6}$$

The loss function is applied on the target frame while the segmentation loss L_{mask} is also applied on the reference frame.

Testing: Given a video sequence, our method follows the online strategy to processes a pair of consecutive frames I_{t-1} (reference frame) and I_t (target frame) iteratively until the end of the video sequence. For the first frame, since there is no reference frame, we only use the plain detection and segmentation branches (*i.e.* no spatial-temporal features nor kernel warping module) to get its instance predictions. The full tracking results can be easily obtained by connecting the tracking prediction of each pair of consecutive frames. However, it is possible that an instance in the reference frame I_{t-1} is not matched to any instances in the current frame I_t, indicating that the instance is occluded in I_t. In this case, our method puts the unmatched instance identity together with its cropped node feature h_i^v as well as its instance-sensitive convolution filters $\theta_{t-1}(x, y)$ in a memory. When proceeding to the next pair of frames I_t and I_{t+1}, we will add the unmatched node feature as an existing instance node at frame t and replace the convolution filters of the newly estimated reference frame $\theta_t(x, y)$ with $\theta_{t-1}(x, y)$ since an instance displacement within a short time interval Δt is usually small. To make the assumption hold, we only keep the unmatched instance in the memory within the time interval Δt, which is set to 7 in our experiments.

4. Experiments

4.1. Datasets and Metrics

In this section, we conduct our experiments on the YouTubeVIS [32] dataset, which contains 2238 training, 302 validation and 343 test video clips. The dataset is annotated with 40 categories, object bounding boxes, segmentation masks and instance identity labels.

Video instance segmentation is evaluated by the metrics of AP (Average Precision) and AR (Average Recall). To achieve a good performance, the method must accurately segment instances and associate instances across frames correctly at the same time.

Iterations	AP	AP_{50}	AP_{75}	AR_1	AR_{10}
1	32.1	53.9	34.1	31.7	36.5
2	35.3	56.4	37.8	33.9	38.9
3	36.5	58.6	39.0	35.5	40.8
4	36.0	58.1	38.8	35.0	40.3

Table 1. Performance on YoutubeVIS dataset when the number of update iterations in GNN increases from 1 to 4.

4.2. Implementation Details

Network: We use ResNet-50 [12] as our feature extraction backbone and build our detection branches as same as CenterNet [36]. We also use FPN module in our framework. Ground truth instances are assigned to different FPN levels according to their sizes. Relation modeling and feature fusion are processed on each output feature map from FPN. Following the settings of segmentation module in CondInst [22], we use three convolution layers as our dynamic mask FCN head, thus, the dynamic controller head have to output 169-channel parameter maps including $H \times W$ instance-sensitive filter parameter vectors for each potential instance. The kernel size of deformable convolution [7] used in propagation branch is 3×3.

Training: We use backbone model pretrained on COCO dataset and train the whole network on YouTubeVIS [32] dataset on 2 Titan X GPUs for 24 epochs. The initial learning rate is 1e-2 and batch size is 16. The learning rate is reduced by a factor of 10 at epoch 16 and 22. We also use multi-scale data augmentation during training.

4.3. Ablation Experiments

In this section we conduct some ablation experiments to study the improvement from each module and find the best hyper-parameters for them.

Iterations of GNN Update: To study the impact from the iterations of the GNN feature update, we set the update iterations from 1 to 4 and evaluate the model on Youtube-VIS [32] validation dataset. From Table 1, we can observe that AP on validation dataset increases when update iterations increase from 1 to 3, because each pixel in feature map can obtain more relevant information from its neighboring instances from last frame. However, when iterations is larger than 3, AP drops. The reason is that when layer number of graph neural network increases, the node and edge features will converge to the average feature, making the network lose its expressive power [20]. So the classification and segmentation heads are not able to predict accurate results from these features that contain less information.

Feature Fusion Strategy: We argue that feature fusion based on relation modeling is important for improving the accuracy of video instance segmentation. To prove it, we use different feature fusion strategies in our ablation experiments. From Table 2, we find that models using fusion strategy based on relation modeling get much higher AP than models without fusion. The reason is that the former outputs more accurate masks and tracking relations based on received temporal information. The fusion strategy based on relation modeling also outperforms the strategy that fuses feature maps of two frames directly by using the concatenation of them as input. This is because, compared to the scene-level information, the instance-level relations and instance corresponding information are more helpful while they are ignored by methods that directly use CNN to fuse features.

Feature Fusion	Association Strategy	Propagation	AP
No fusion	Embedding	✗	31.6
Concat	Embedding	✗	32.1
Relation	Offset	✗	32.6
Relation	Embedding	✗	33.4
Relation	Edge Feature	✗	34.8
No fusion	Embedding	✓	32.0
Relation	Embedding	✓	33.9
Relation	Edge Feature	✓	35.2

Table 2. Ablation experiment results on feature fusion, tracking strategy and mask information propagation module. Multi-scale training is not used. “Relation” and “Edge features” denote our feature fusion and tracking strategy based on gnn, while “Embedding” and “Offset” denote tracking strategy following other VIS or MOT method.

Joint Detection, Segmentation and Tracking: To prove that jointly learning detection, segmentation and tracking based on instance-level relations is beneficial for improving the performance on video instance segmentation task, we conduct the experiments using different association strategies. For “Embedding” association strategy in Table 2, we follow the strategy in MaskTrack R-CNN [32] by adding an extra head to generate an embedding feature and using this embedding to associate the most similar candidates with corresponding instances. The embedding head is trained with classification loss as same as that in MaskTrack R-CNN. As for “Offset” association strategy, following CenterTrack [35], we predict center point offsets across frames for each instance and use these offsets and IoU to match instances with candidates. We compare the performance of these two association strategies with our edge features based association strategy. We find that association strategy based on edge features outperforms both of them. The association based on offsets performs worse because YoutubeVIS [32] dataset is annotated 1 frame per 5 frame from videos, while offset tracking relies on high fps videos. Meanwhile, different from MOT situation where the categories only include person and vehicle, which means the motion pattern of objects is relatively easy to learn, in YoutubeVIS, there are various classes of objects and it is

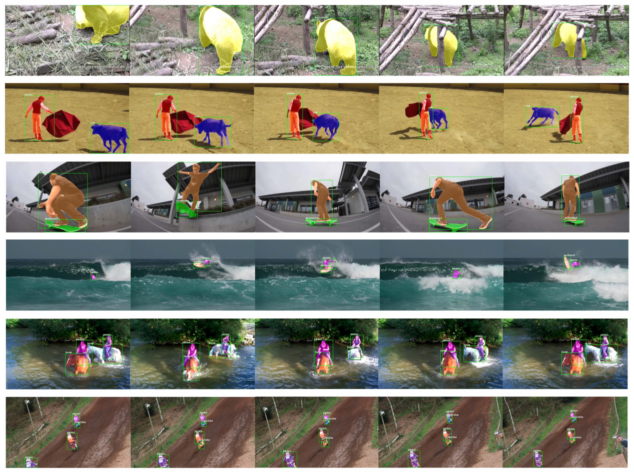


Figure 5. Visualization results of our method on YoutubeVIS dataset. Best viewed in color.

hard to model their motion patterns. As for association based on embedding, though the embedding head is trained together with segmentation head, it treats the association between two frames as a N-classes classification problem, where N is the instance in the previous frame. However, N varies in each frame and it is hard to learn the classifier. Moreover, each of the candidate in current frame is classified separately, without using information from other candidates. Relation modeling based on GNN utilizes the features from all instances and candidates and a binary classifier for edges is also more robust, so it performs much better than embedding-based association.

Mask Information Propagation Branch: Most of the VIS methods utilize temporal information by feeding fused feature maps into prediction heads, but ignores the concrete mask-level information provided by historical instance masks. From Table 2, we find that adding a mask information propagation branch is helpful to improve AP value. The reason is that by merging the warped filter information from last frame, the mask head not only segments instances based on fused feature map $\hat{F}_t$, but also utilizes the instance-level mask information, motion and alignment information from the previous frame. In addition to fusing temporal information via GNN, using the propagation branch will offer more detailed shape information for the mask prediction head.

4.4. Main Results

We compare our method with existing VIS methods in Table 3. We use metrics AP, AP_{75} and AP_{50} to evaluate these methods. To eliminate the effect from backbones, all the methods in table use ResNet-50 [12] as the backbone. We can observe that our method achieves higher AP value than most of the existing VIS methods. Especially, the AP value of our method is 6.2% higher than baseline method

Method	AP	AP_{50}	AP_{75}	AR_1	AR_{10}
DeepSORT[29]	26.1	42.9	26.1	27.8	31.3
FEELVOS[24]	26.9	42.0	29.7	29.9	33.4
OSMN[33]	27.5	45.1	29.1	28.6	33.1
MaskTrack R-CNN[32]	30.3	51.1	32.6	31.0	35.5
MaskProp[2]	40.0	-	42.9	-	-
STEm-Seg[1]	30.6	50.7	33.5	31.6	37.1
VisTR[27]	34.4	55.7	36.5	33.5	38.9
SipMask[5]	33.7	54.1	35.8	-	-
VisSTG(ours) w/o ms	35.2	55.7	38.0	33.6	38.5
VisSTG(ours)	36.5	58.6	39.0	35.5	40.8

Table 3. The results of video instance segmentation on the YouTube-VIS validation dataset. We compare our method to other existing methods by metrics of AP, AP_{50} and AP_{75}. All of the methods use ResNet-50 as backbone. Our method outperforms most of the existing methods.

MaskTrack R-CNN [32] and 2.1% higher than the recent method VisTR which is based on transformer. Note that AP of our method is lower than that of MaskProp [2]. We believe it is caused by the complicated post-processing steps and a stronger network, while our method deals with the segmentation, detection and tracking in a single framework, and thus has much faster FPS than [2] (22 *vs.* 2). Some visualization results of our method on the YouTube-VIS validation dataset are shown in Figure 5. Images in each row are sampled from the same video and the instance masks with the same color belong to the same object. We observe that our method can track and segment well in various situations, even some instances are occluded or overlapped.

5. Conclusion

We proposed an end-to-end video instance segmentation approach that simultaneously learns classification, detection, segmentation and tracking. To better utilize spatial-temporal information from previous frames and related instances, we use GNN to model the instance-level relations and guide the spatial-temporal information fusion. Detection and segmentation results are predicted from fused node features, while tracking results can be obtained from edge features via a binary edge classifier simultaneously. To further improve the segmentation accuracy, we add a propagation branch to obtain history mask-level information and use the fused instance-sensitive filters to predict masks by a dynamic mask head for each instance. Our method utilizes instance-level spatial-temporal information for all VIS subproblems, achieving good performance on YoutubeVIS validation dataset.

References

[1] Ali Athar, Sabarinath Mahadevan, Aljosa Osep, Laura Leal-Taixé, and Bastian Leibe. Stem-seg: Spatio-temporal em-

beddings for instance segmentation in videos. In *European Conference on Computer Vision*, pages 158–177. Springer, 2020.

[2] Gedas Bertasius and Lorenzo Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9739–9748, 2020.

[3] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9157–9166, 2019.

[4] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6247–6257, 2020.

[5] Jiale Cao, Rao Muhammad Anwer, Hisham Cholakkal, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. Sipmask: Spatial information preservation for fast image and video instance segmentation. *arXiv preprint arXiv:2007.14772*, 2020.

[6] Shengjia Chen, Zhixin Li, and Zhenjun Tang. Relation r-cnn: A graph based relation-aware network for object detection. *IEEE Signal Processing Letters*, 27:1680–1684, 2020.

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[8] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272. PMLR, 2017.

[9] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[10] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312. Springer, 2014.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6409–6418, 2019.

[14] Xiaolong Jiang, Peizhao Li, Yanjing Li, and Xiantong Zhen. Graph neural based end-to-end data association framework for online multiple-object tracking. *arXiv preprint arXiv:1907.05315*, 2019.

[15] Jiahe Li, Xu Gao, and Tingting Jiang. Graph networks for multiple object tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 719–728, 2020.

[16] Yuxi Li, Ning Xu, Jinlong Peng, John See, and Weiyao Lin. Delving into the cyclic mechanism in semi-supervised video object segmentation. *Advances in Neural Information Processing Systems*, 2020.

[17] Weiyao Lin, Huabin Liu, Shizhan Liu, Yuxi Li, Rui Qian, Tao Wang, Ning Xu, Hongkai Xiong, Guo-Jun Qi, and Nicu Sebe. Human in events: A large-scale benchmark for human-centric video analysis in complex events. *arXiv preprint arXiv:2005.04490*, 2020.

[18] Jonathon Luiten, Philip Torr, and Bastian Leibe. Video instance segmentation 2019: A winning approach for combined detection, segmentation, classification and tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[19] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016.

[20] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

[21] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719, 2020.

[22] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. *arXiv preprint arXiv:2003.05664*, 2020.

[23] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9627–9636, 2019.

[24] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9481–9490, 2019.

[25] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9197–9206, 2019.

[26] Yongxin Wang, Xinshuo Weng, and Kris Kitani. Joint detection and multi-object tracking with graph neural networks. *arXiv preprint arXiv:2006.13164*, 2020.

[27] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv preprint arXiv:2011.14503*, 2020.

[28] Xinshuo Weng, Yongxin Wang, Yunze Man, and Kris M Kitani. Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6499–6508, 2020.

[29] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[30] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12193–12202, 2020.

[31] Minghao Xu, Hang Wang, Bingbing Ni, Qi Tian, and Wenjun Zhang. Cross-domain detection via graph-induced prototype alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12355–12364, 2020.

[32] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5188–5197, 2019.

[33] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6499–6507, 2018.

[34] Yufeng Zhang, Lianghui Ding, Yuxi Li, Weiyao Lin, Mingbi Zhao, Xiaoyuan Yu, and Yunlong Zhan. A regional distance regression network for monocular object distance estimation. *Journal of Visual Communication and Image Representation*, page 103224, 2021.

[35] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020.

[36] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.