TT-SPN: Twin Transformers with Sinusoidal Representation Networks for Video Instance Segmentation

Can Kocagil
Bilkent University
Departmant of EEE
can.kocagil@ug.bilkent.edu.tr

Abstract

Video instance segmentation is the recently introduced computer vision research that aims at joint detection, segmentation, and tracking of instances in the video domain. Recent methods proposed highly sophisticated and multi-stage networks that are practically unusable. Hence, simple yet effective approaches are needed to be used in practice. To fill the gap, we propose an end-to-end transformer based video instance segmentation module with Sinusoidal Representation Networks (SPN), namely TT-SPN, to address this problem. TT-SPN views the VIS task as a direct sequence prediction problem in single-stage that enables us to aggregate temporal information with spatial one. Set of video frame features are extracted by twin transformers that then propagated to the original transformer to produce a set of instance predictions. These produced instance level information is then passed through modified SPNs to get end instance level class ids and bounding boxes and self-attended 3-D convolutions to get segmentation masks. At its core, TT-SPN is a natural paradigm that handles the instance segmentation and tracking via similarity learning that enables system to produce a fast and accurate set of predictions. TT-SPN is trained end-to-end with set-based global loss that forces unique predictions via bipartite matching. Thus, the general complexity of the pipeline is significantly decreased without sacrificing the quality of segmentation masks. For the first time, the VIS problem is addressed without implicit CNN architectures thanks to twin transformers with being one of the fastest approaches. Our method can be easily divided into its sub-components to produce separate instance masks and bounding boxes that will make it unified approach for many vision tasks. We benchmark our results on YouTube-VIS dataset by comparing competitive baselines and show that TT-SPN outperforms the base VIS model by a significant margin. Code is available at `https://github.com/cankocagil/TT-SPN`.

1. Introduction

Instance based segmentation and object detection in images and videos are fundamental problems in the context of computer vision. Different from image instance segmentation, the new problem aims at simultaneous detection, segmentation and tracking of object instances in videos [31]. It is first introduced in the video instance segmentation paper [31] with a novel algorithm called Mask-Track R-CNN. Video instance segmentation is a crucial task for spatio-temporal understanding in video domain with applications to video-editing, autonomous driving, pedestrian tracking, augmented reality, robot vision and lot more. Since it requires both segmentation and tracking, it is a more challenging task compared to image level instance segmentation. Furthermore, it helps us to encode spatio-temporal raw data to meaningful insights along the video as it has richer content compared to visual spatial data. With the addition of temporal dimension to our decoding process, we further get information about the motion, viewpoint variations, illuminations, occlusions, deformations and local ambiguities from the video frames. Hence, video instance segmentation gained popularity as a research area and it attracts the community along the line of research for video understanding recently.

State-of-the-art approaches developed very complicated architectures with multiple networks and mostly based on human-oriented post-processing approaches (e.g.., Non-Maximum Suppression) to produce high quality segmentation masks and bounding boxes. Generally, tracking-by-detection (top-down approaches) [31, 3, 5] or spatio-temporal embedded clustering [6] (bottom-up) based approaches are proposed to tackle the VIS task. In top-down approaches, image-level instance segmentation masks are produced then associated in temporal dimension via complex hand-crafted rules to advance spatial predictions to spatio-temporal predictions that complicated the process of

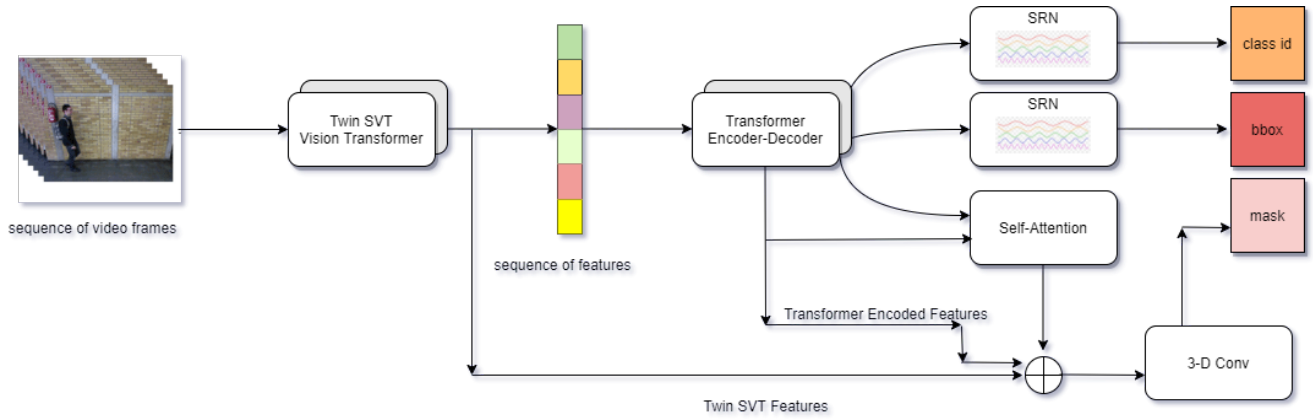

Figure 1: **Overall Pipeline of TT-SPN.** Given the video frames, set of image features are extracted by twin transformer then passed through to classical transformer architecture to get instance level predictions. These predictions then passed to separate SPN branches to produce set of class ids, confidences, bounding boxes and self-attended convolutions to get segmentation masks. Overall predictions follow the video frame order.

decoding and lead to be practically unusable. Whereas in bottom-up methods, instance level pixels in formations are clustered in spatio-temporal embedding space with non-overlapping regions that heavily based on dense prediction quality [6] and requires multiple networks to produce end VIS results. Hence, simple yet effective, single-stage, practically usable and end-to-end trainable approaches are highly desirable.

In this paper, we propose a novel approach, namely TT-SPN, to reduce the overall pipeline complexity without compromising the speed of the predictions and its quality to produce VIS results. Overall pipeline is depicted in figure 1. Given the video frames, a set of image features are extracted by twin transformer then passed through to classical transformer architecture to get instance level predictions. These predictions then passed to separate SPN branches to produce a set of class ids, confidences, bounding boxes and self-attended convolution module to get segmentation masks. In nature, all of the sub-tasks of VIS (classification, detection, segmentation and tracking) are related tasks. Hence, the output of one task can provide significant information to another task that will all facilitate all sub-module mutually. By realizing this, TT-SPN is also facilitated by the paradigm of handling the sub-tasks in one module. Simultaneously, since there are no human-designed rules for all individual tasks of VIS, instance level feature quality is another significant part of TT-SPN that is accomplished by the twin transformer module. Twin transformer is state-of-the-art architecture based on spatially oriented vision transformers and recently proposed in the paper [9]. From the publication of classical transformers in the context of NLP [28], transformer are placed *de facto* method for variety of NLP tasks (e.g., machine translation and seq-to-seq problems). For the first time, vision transformers are proposed in the paper [11] and demonstrate the power of transformer in the context of computer vision. However, the problem was the computational complexity that is quadratic to image size. To suppress, a variety of vision transformers are proposed and demonstrated that carefully designed global and local attention mechanisms may outperform classical CNN architectures in dense prediction tasks [11, 27, 9]. A workaround is the locally-grouped self-attention (or self-attention in non-overlapped windows as in the recent Swin Transformer [20]), where the input is spatially grouped into non-overlapped windows and the standard self-attention is computed only within each sub-window [9]. Even the Swin transformers decrease the overall computational complexity, it cannot make connections between non-overlapping region via attention. To overcome this, twin transformers are proposed in the paper [9] that introduce the spatially separable self-attention (SSSA) to alleviate this challenge. SSSA is composed of locally-grouped self-attention (LSA) and global sub-sampled attention (GSA) [9]. We found that the instance level features produced by twin transformers are highly optimized with respect to its counter parts in conventional CNNs. The overall scheme is depicted in figure 2

Here, we also introduce modified Sinusoidal Representation Networks for the classification and object detection tasks. Sinusoidal Representation Networks are proposed in the paper [25] and demonstrate that implicitly defined, continuous, differentiable signal representations parameterized by neural networks have emerged as a powerful paradigm, offering many possible benefits over conventional represen-

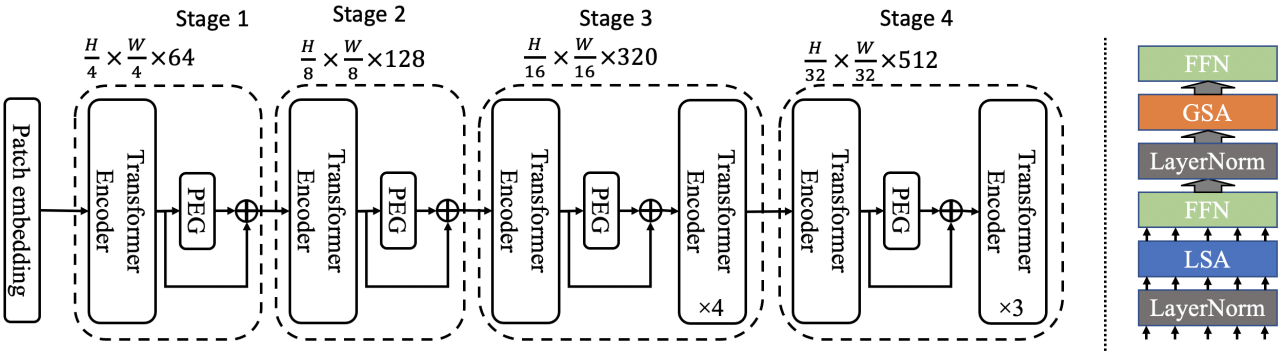

Figure 2: **Architecture of Twin Transformer** Given the video frames, Twins-SVT interleaves locally-grouped attention (LSA) and global sub-sampled attention (GSA) in hierarchical stages. PEG stands for position encoding generator.

tations [25]. They introduce o leverage periodic activation functions for implicit neural representations and demonstrate that these networks, dubbed sinusoidal representation networks or SIRENs, are ideally suited for representing complex natural signals and their derivatives [25]. In this paper, we adapted the SIREN architecture for dense prediction tasks, and modified it to reuse for bounding box and class id predictions. Our modified SPN networks consist of multiple of SIREN layers with dropouts. As a last layer of SPNs, fully connected layer with GELU non-linearity [15] is placed to produce end results. The proposed architecture of SPNs are depicted in figure 3.

Instance segmentation and tracking are other significant aspects of the VIS task. To perform natural, efficient and mutually inclusive segmentation and tracking, we adapted the instance sequence matching and tracking module from the VisTR [29]. The instance sequence matching performs bipartite graph matching between the output instance sequence and the ground-truth instance sequence, and supervises the sequence that uniquely assigns predictions and their annotations [29]. Thus, TT-SPN can maintain the order directly. The instance sequence segmentation accumulates the mask features for each instance across multiple frames through self-attention and segments the mask sequence for each instance through 3D convolutions to get end results [29].

Our main contributions are summarized as follows.

- We propose a highly efficient video instance segmentation module built upon vision and classical transformers with Sinusoidal Representation Networks that views the VIS task as an end-to-end set prediction problem.
- To our best knowledge, it is the first time that video instance segmentation problem is addressed without any implicit CNN architectures, instead vision transformers (e.g., twin transformers) are used as a instance level features generator.
- Without any knitting, TT-SPN is one of the fastest methods as runs in 55.3 FPS and achieves competitive accuracy on YouTube-VIS as 39.3 % mAP by outperforming the VIS baseline model by a significant margin.

The rest of the paper is organized as follows. We begin with the related work done in the context of video instance segmentation and the comparison with the related tasks. Then, we explain our methodology and empirical study along with the research. After that, we introduce our algorithm with the details of the implementations. We show our empirical results based on the VIS evaluation metrics. We end up with further improvements and a conclusion.

2. Related Work

There are various approaches in the literature for solving the task of video instance segmentation, as it is often considered multi-stage problem, i.e., segmentation/detection and tracking components are handled in different stages. However, recent studies come up with single stage, simple, computationally effective methods to deal with the problem, although the performance of single-stage approaches does not outperform the multi-stage approaches. Hence, computer vision research community extends the work done in the paper [31] by proposing either the variant of Mask-Track R-CNN or new approaches for video instance segmentation tasks. STEm-Seg is another novel algorithm for this task and in particular, they model a video clip as a single 3D spatio-temporal volume, and propose a novel approach that segments and tracks instances across space and time in a single stage [1]. Then, Chung-Ching Lin et.al. proposed a

Variotional Auto-Encoder based segmentation tracking algorithm for video instance segmentation task as it builds a shared encoder and three parallel decoders, yielding three disjoint branches for predictions of future frames, object detection boxes, and instance segmentation masks [16]. To facilitate the research along with the problem, Jiale Cao et.al, proposed another single-stage novel algorithm called SipMask that preserves instance-specific spatial information by separating mask prediction of an instance to different sub-regions of a detected bounding-box [6]. Then, VisTR is proposed as single-stage transformer based VIS architecture that views VIS task as a direct end-to-end parallel sequence decoding/prediction problem [29]. Some of our work is adapted from the VisTR module. To be specific, we integrated their instance sequence matching and segmentation module to supervise and segments the instances as a complete. The instance sequence matching performs bipartite graph matching between the output instance sequence and the ground-truth instance sequence, and supervises the TT-SPN so TT-SPN learn the similarity between instances [29]. The instance sequence segmentation module perform self-attended 3-D convolutions to learn pixel-level similarity.

Hence, there are various different approaches for solving temporal domain instance level segmentation problem, here in this work, we propose our approach of solving video instance segmentation problem as we consider it direct set of prediction problem.

Even the concept of video instance segmentation can be classified as a new task, there are various similar problems addressed by the researchers in the literature such as image level instance segmentation, video object detection, video object tracking, and video object segmentation. We will briefly describe the similar problems as follows.

2.1. Image-level Instance Segmentation

Instance segmentation not only group pixels into different semantic classes, but also group them into different object instances [12]. A two-stage paradigm is usually adopted, which first generate object proposals using a Region Proposal Network (RPN), and then predict object bounding boxes and masks using aggregated RoI features [12]. In our case, we not only generate segmentation masks for individuals but also associates them in the video sequences.

2.2. Video Object Detection

Video object detection aims at detecting objects in videos, which is first proposed as part of ImageNet visual challenge [24]. Even the association and providing identity improves the detection quality, this challenge is limited to spatially preserved evaluation metrics for per-frame detection and does not require joint object detection and tracking [31]. However, in our case, we aim joint detection, segmentation and tracking as oppose to video object detection task.

2.3. Video Object Tracking

Video object tracking task is generally considered as detection-based and detection-free tracking approaches. In detection-based tracking algorithms, object are jointly detected and tracked such that tracking part improves the detection quality whereas in detection-free approaches we're given initial bounding box and try to track that object across video frames [26, 31]. As detection-based approaches are similar to our case, video instance segmentation requires temporal segmentation masks.

Hence, as oppose to previous fundamental computer vision tasks, video instance segmentation requires multi-disciplinary and aggregated approaches.

2.4. Video Instance Segmentation

As video instance segmentation task is supervised, it requires human oriented high-quality annotations for bounding boxes and binary segmentation masks with predefined categories. Let C_i be the object categories belong to the dataset D for $i = 1, ..., K$ where K is the number of unique categories including background in the D. Then, let $B_j^{t_i}$ and $S_j^{t_i}$ are the j^{th} bounding box and binary mask for $j^{th} \in C_1, ..., C_K$ object in video frame $t_i \in T$ where T represents the number of frames in the given video sequence. Assuming that when inference phase, VIS algorithm produces $N \in C_1, ..., C_K$ instance hypothesis such that $H_{N_j}^{t_i}$ represents the prediction for N_j^{th} instance and t_i^{th} time produced by VIS. Hence, $H_{N_j}^{t_i}$ includes confidence score $s_j^{t_i} \in [0, 1]$ as the probability of identification of the instance with predefined category, $\hat{B_j^{t_i}}$ and $\hat{S_j^{t_i}}$. Hence, we are trying to minimize between the human-created annotations and produced hypothesis as it requires fast and optimal detection, tracking and segmentation estimations.

3. Proposed Approach: TT-SPN

We propose end-to-end transformer based video instance segmentation module with Sinusoidal Representation Networks (SPN), namely TT-SPN, to address the VIS task. Our method, TT-SPN, views the VIS task as a direct set of prediction problem in single-state that enables us to aggregate temporal information with spatial information. Set of video frame features are extracted by twin transformers that then propagated to the original transformer to produce sequence of instance predictions. These produced instance level information by transformers are then passed through modified Sinusoidal Representation Networks to get end instance level class ids and bounding boxes, and self-attended 3-D

convolutions to get segmentation masks. At its core, TT-SPN is a natural paradigm that handles the tracking and segmentation via similarity learning that enables the system to produce a fast and accurate set of predictions. Instance sequence matching algorithm is adapted from [29] to track instances across video frames. TT-SPN is trained end-to-end with set-based global loss that forces unique predictions via bipartite matching. Thus, general complexity of pipeline is significantly decreased without sacrificing quality of segmentation masks. For the first time, VIS problem is addressed without implicit CNN architectures thanks to twin transformers with being one of the fastest approaches. Our method can be easily divided its sub-components to produce separate instance masks and bounding boxes that will make it unified approach for many vision tasks. In this section, TT-SPN is divided into its sub-modules and details are described.

3.1. Twin Transformers

Recently, twins are proposed in the paper [9], and demonstrate that the spatially oriented vision transformers can outperform the classical CNNs [9]. Here, we integrated Twins-SVT network to our case to produce instance level features. There twin transformer is based on a spatially separable self-attention (SSSA) network that consist of locally-grouped self-attention (LSA) and global sub-sampled attention (GSA) [9]. Thanks to its spatially separable module, the quality of features are increased by a significant margin. In the subsections, we describe the SSSA module in detail.

3.1.1 Locally-grouped self-attention (LSA)

In LSA, 2-D feature maps are divided into sub-windows that enable the self-attention within each sub-window. Features maps are divided into m x n sub-windows, that lead to each window consist of $\frac{HW}{mn}$ elements where H,W represents image dimensions. By dividing the image into m x n region the computational cost is decreased from $O(H^2W^2d)$ to $O(\frac{H^2W^2}{mn}d)$ where d is the self-attention dimension. At that point, we did not make any further relation to non-overlapping regions in the windows. Hence, here GSA module comes into play.

3.1.2 Global sub-sampled attention (GSA)

As we need further localization in the self-attention mechanism, global self-attention is required to make connections in non-overlapping regions. In GSA module, a single representative key in formations from the locally attended windows are used to compute global attention. However, with the computation of global-attention, the computation cost would increase to $O(H^2W^2d)$. To prevent this, locally attended features are sub-sampled via average pooling, depth-wise strided convolutions and regular strided convolutions. The results show that regular strided convolutions perform best [9]. Mathematically, SSSA module performs the following computations.

$$\begin{aligned} a^l_{i,j} &= LSA(LayerNorm(a^{l-1}_{i,j})) + a^{l-1}_{i,j}, \\ a^l_{i,j} &= FFN(LayerNorm(a^l_{i,j})) + a^l_{i,j}, \\ a^{l+1} &= GSA(LayerNorm(a^l)) + a^l, \\ a^{l+1} &= FFN(LayerNorm(a^{l+1})) + a^{l+1} \end{aligned} \tag{1}$$

for $i = 1, ..., m$ and $j = 1, ...n$ where LSA denotes locally-grouped self-attention, GSA denotes global sub-sampled attention, FFN denotes feed-forward network and LayerNorm denotes layer normalization layer [2]. Both attention modules performed in multi-headed way.

3.2. Classical Transformers

Classical transformer architecture with 6 encoder layers, 6 decoders layers, with GELU activation [15] is adapted to perform instance-wise query generation. The output of the classical transformer is proposals of instances plus additional no object queries. The usage of transformer is quite similar to the one in object detection model DETR [7]. During the training, bipartite matching is performed to supervises the model by uniquely assigning predictions with ground truths. Prediction with no match should yield a "no object" class prediction so the number of instance queries should be larger than the number of instances in video frames. At its core, the transformer consists of its encoder and decoder structure that is discussed in the following subsections.

3.2.1 Spatio-Temporal Positional Encoding

Since the transformer architecture is permutation-invariant, spatio-temporal positional encoding is necessary to model the precise location information. Spatio-temporal positional encoding is based on sinusoidal waves and it is 3-D version of the classical positional encoding. Our positional encoding has 3 different dimensions that are temporal, horizontal and vertical. Let d final concatenated channel position encoding dimension, then we independently used $d/3$ sinusoidal functions with different frequencies as follows.

$$PE(pos, i) = \begin{cases} \sin(pos, w_k), & \text{for i = 2k} \\ \cos(pos, w_k), & \text{for i = 2k +1;} \end{cases} \tag{2}$$

where $w_k = 1/10000^{2k/\frac{d}{3}}$, pos is the position in that dimension. As in the case of conventional positional encoding, these 3-D position encodings are added to the input.

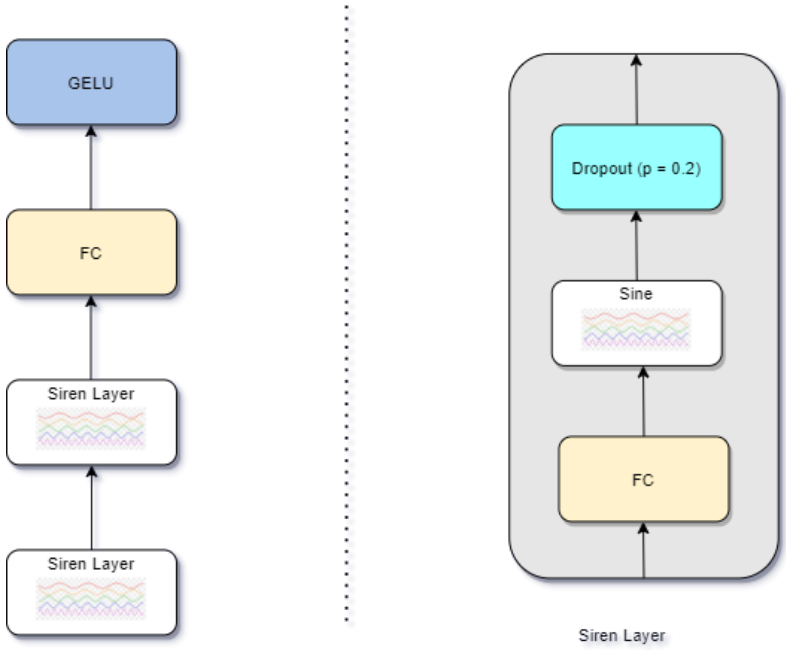

Figure 3: **Architecture of Sinusoidal Representation Networks.** It consist of two SIREN layers with one extra fully connected layer with GELU activation.

3.2.2 Transformer Encoder

The transformer encoder layer with the size of 6 is adapted to learn instance-wise similarities that will be later propagated to the decoding layer to produce end instance level queries. The extracted features from the twin transformer is passed to a single convolutional layer with 256 output latent size. Hence, the input to the transformer encoder is in the shape of $\mathbb{R}^{NxLxHxW}$ where N is batch size, L is latent size, H and W are height and width of output of the single convolutional layer. Note that the temporal order is preserved according to the input order. Each encoder layer performs a multi-headed self-attention mechanism as conventional.

3.2.3 Transformer Decoder

The sequence of encoded features is then passed through the transformer decoder layer to produce sequence of instance query predictions. In this layer, a sequence of learnable instance queries are also passed the decoder layer. The instance queries are a fixed number of input embeddings to represents the total number of instance predictions. The number of instance queries is always bigger than the number of instances in the image to be safe. Bipartite matching performs instance wise assignment uniquely, the exposed predictions are called "no object queries". For example, assuming we produce n_t instance prediction in frame t, then let q be the size of general instance queries so that $q > n_t$ in all frames.

3.3. Sinusoidal Representation Networks

Sinusoidal Representation Networks are a recently proposed method in the paper [25] for representation learning. The sinusoidal layers consist of fully connected layers with its unique initialization introduced in the paper [25] with sinusoidal activation layer. The overall architecture is depicted in the figure 3. In this work, we modified their architecture for our case by adding internal dropout layers between the sinusoidal layers with end GELU nonlinearity [15] to produce instance level end features. These end features are then propagated to classification, bounding box detection and instance segmentation branches. We realized that the periodic activations for implicit neural representations and demonstrate that these networks, dubbed sinusoidal representation networks or SIRENs, are ideally suited for dense prediction tasks. Our ablation studies demonstrate that periodic activation functions for end prediction layers can be suited for dense prediction tasks.

3.4. Instance Sequence Matching

An important aspect of TT-SPN, namely Instance Sequence Matching, is adapted from the paper [29] to uniquely assign the instance predictions with ground truths via bipartite matching to supervises the model. Further, this

module enables us to infer the precise order of predicted instances that later enables the tracking instances across the video. The matching loss takes both class predictions and the similarity of predicted and ground truth into account [7]. Let y denote the ground truth boxes set of objects, and $\tilde{y} = \tilde{y}_{i=1}^{N}$ the set of N predictions. Our loss produces an optimal bipartite matching between predictions and ground truth. To compute bipartite matching between two sets, the following minimization is computed.

$$\tilde{\sigma} = \underset{\sigma}{\operatorname{argmin}} \sum_{i}^{N} L_{match}(y_i, \tilde{y_{\sigma_{(i)}}}) \tag{3}$$

where $L_{match}(y_i, \tilde{y_{\sigma_{(i)}}})$ is instance-wise matching cost between a ground truth and prediction. This assignment problem is computed with Hungarian method that is a combinatorial optimization algorithm that solves the assignment problem in polynomial time [7].

The matching procedure takes into account both class predictions and the similarity of the predicted and ground truth boxes. Let each element i of the annotation is denoted by $y_i = (c_i, b_i)$ where c_i target class and b_i is a vector denotes the ground truth normalized coordinates. These coordinates are organized as center, height and width and they are relative to the image size. Then, for the prediction with index $\sigma(i)$, let $\tilde{p_{\sigma(i)}}(c_i)$ denote class probability and $\tilde{b_{\sigma(i)}}$ as predicted box. Hence, we can define $L_{match}(y_i, \tilde{y_{\sigma_{(i)}}})$ as follows.

$$L_{match}(y_i, \tilde{y_{\sigma_{(i)}}}) = -\tilde{p_{\sigma(i)}}(c_i) + L_{box}(b_i, \tilde{b_{\sigma_{(i)}}}) \tag{4}$$

This procedure supervises a model and plays an important role in the heuristic assignment process. In classical object detection or instance segmentation tasks (e.g. Mask R-CNN [13]), these procedures are counterpart of the matching proposal or anchors to ground truths. Significantly different from the classical approaches, bipartite matching assigns uniquely. At this point, we assigned the predictions with their ground truths so we need to compute loss, in our case, *Hungarian Loss* for all matched pairs. Given the one-to-one assignment, *Hungarian Loss* computes the loss as a linear combination of negative log-likelihood for class prediction, a box and mask loss for the sequence of instances as follows.

$$\begin{aligned} L_{Hungarian}(y, \tilde{y}) = &- \sum_{i=1}^{N} \log \tilde{p_{\sigma(i)}}(c_i) + \sum_{i=1}^{N} L_{box}(b_i, \tilde{b_{\sigma_{(i)}}}) \\ &+ \sum_{i=1}^{N} L_{mask}(m_i, \tilde{m_{\sigma_{(i)}}}) \end{aligned} \tag{5}$$

where $\tilde{\sigma}$ is optimal assignment computed previously. This loss is used to train the model in end-to-end fashion. Next, we need to define L_{box} and L_{mask}. L_{box} is similarly computed as in DETR [7] as follows.

$$\begin{aligned} L_{box}(b_i, \tilde{b_{\sigma_{(i)}}}) = &\frac{1}{T} \sum_{t=1}^{T} \lambda_{IoU} * L_{IoU}(b_{i,t}, \tilde{b_{\sigma_{(i),t}}}) \\ &+ \frac{1}{T} \sum_{t=1}^{T} \lambda_{L1} * \left\| b_{i,t} - \tilde{b_{\sigma_{(i),t}}} \right\| \end{aligned} \tag{6}$$

where λ_{IoU} and λ_{L1} are hyperparameters. Note that losses are normalized with the number of instances inside the frame.

3.5. Instance Sequence Segmentation

Another important aspect of TT-SPN, namely Instance Sequence Segmentation module, is adapted from the paper [29] to produce end segmentation masks. Internally, this module accumulates the instance features of the frames then segmentation is performed on these accumulated features. For each frame, instance predictions gathered by the end decoder layer of the transformer and transformer encoded features gathered by the end encoder layer of the transformer are passed through the self-attention module. These attended features are then fused with the features gathered by the twin transformer and the encoded features generated from the end encoder of the transformer. These procedure is very similar as in the case of VisTR [29] and DETR [7]. The instance level features with different sizes are then fed into deformable convolution layer [10] that augments the spatial sampling locations in the modules with additional offsets and learning the offsets from target tasks, without additional supervision [10]. Then, the fused maps that are in the shape of $\mathbb{R}^{1xCxTxtHxW}$ where C is channel dimension, T is temporal dimension, H and W are spatial feature dimension are fed into 3-D convolution layer with group normalization [30] and GELU non-linearity [15]. At the end layer, single convolutional layer is with 1 output channel dimension placed to get segmentation masks. Finally, we need to define L_{mask} to complete the loss function. L_{mask} is computed by combining dice [22] and focal loss [18] as follows.

$$\begin{aligned} L_{mask}(m_i, \tilde{m_{\sigma_{(i)}}}) = &\frac{1}{T} \sum_{t=1}^{T} \lambda_{mask} * L_{dice}(m_i, \tilde{m_{\sigma_{(i)}}}) \\ &+ \frac{1}{T} \sum_{t=1}^{T} \lambda_{mask} * L_{focal}(m_i, \tilde{m_{\sigma_{(i)}}}) \end{aligned} \tag{7}$$

Figure 4: Example predictions of TT-SPN on arbitrary test video. We intend to point out the occlusions, point view variations, systematic artifact on overlapping objects and scale of instances. From left upper corner to right lower corner, visuals are sequential predictions of our model.

4. Results

In this section, we demonstrate our results on YouTube-VIS dataset [31]. YouTube-VIS is a large and scalable dataset consist 2,883 high-resolution YouTube videos, 2,238 training videos, 302 validation videos and 343 test videos. A category label set consist of 40 common objects such as person, animals and vehicles with a total of 4,883 unique video instances that yield 131k high-quality human-oriented annotations. Since the evaluation of the test set is closed, the evaluation results are based on the validation set.

4.1. Implementation Details

We inherited hyperparameters used in twin transformer [9] for the first stage of the TT-SPN. Hence, the embedding dimension is selected as 64, patch size 4, local patch size 7, and depth is 1. In the same order, the hyperparameters of the second stage of twin transformer is, 128, 2, 7, 1. For the third stage, 256, 2, 7, 5 are selected as hyperparameters of the third stage. In the final stage, embedding size is 512, patch size is 2, local patch size is 7 and depth is 4. Here, depth means the number of transformer blocks described in the twin transformer section. Please refer to figure 2. The hidden size of single convolutional layer is selected as 256. In the classical transformer, there are 6 encoder layers and 6 decoding layers with multi-head size 8. The internal activation of transformer block is GELU [15] in all encoder-decoders. In the SPN stage, dropout probabilities are selected as 0.2. All SPN layers are initialized with their specialized initialization scheme described in the paper [25].

Then, the largest number of the annotated video length in YouTube-VIS is 36 [31], we select this value as input video length. Thus, no post-processing is required for associating different clips from one video. In this way, our model is end-to-end trainable in singe stage. As our model predicts 10 objects per video frame, we set the query number to 360.

TT-SPN is implemented via PyTorch 1.8 [23]. Thanks to its simple building blocks, TT-SPN is generalizable and scalable to other frameworks and vision tasks. We also provide separate instance segmentation and object detection version of TT-SPN in our project page [1].

In the phase of training, we optimized all layers with AdamW [21] starting with the learning rate of 1e-4 and decays 0.1 per 3 epochs. TT-SPN is trained with 18 epochs and batch size selected as 16. The classical transformer weights are initialized from DETR [7] that is pretrained in COCO [19]. All video frames are normalized with ImageNet mean and standard deviation values by per-channel fashion. Then, all video frames are resized to 300 x 540 to fit the GPU. We have used only random horizontal flip with probability 0.5 as video data augmentation. TT-SPN is trained on single Tesla K80 GPU of 8GB RAM with 5 days.

In the phase of inference, there is no change in the TT-

[1] https://github.com/cankocagil/TT-SPN---Object-Detection

Method	Backbone	FPS	mAP	AP50	AP75	AR1	AR10
Mask Track R-CNN [31]	ResNet-50	20.0	30.3	51.1	32.6	31.0	35.5
STEm-Seg [1]	ResNet-50	-	30.6	50.7	33.5	31.6	37.1
STEm-Seg [1]	ResNet-101	2.1	34.6	55.8	37.9	34.4	41.6
MaskProp [3]	ResNet-50	-	40.0	-	42.9	-	-
MaskProp [3]	ResNet-101	-	42.5	-	45.6	-	-
VisTR [29]	ResNet-50	69.9	36.2	59.8	36.9	37.2	42.4
VisTR [29]	ResNet-101	57.7	40.1	64.0	45.0	38.3	44.9
TT-SPN	Twin Transformer	55.3	39.3	61.3	38.2	40.0	41.3

Table 1: **Video Instance Segmentation mAP (%) results on YouTube-VIS [31].** Results are obtained from theirs papers.

SPN architecture. Hence, our model's training and inference shapes are completely same. Additionally, no hand crafted post processing is necessary to associate instances across the video frames. We set the threshold to kept instances whose score is higher than the determined threshold to get end result. We set this threshold as 0.6. There were some instances identified as different classes in video frames. At that time, we use the most frequently predicted category.

4.2. Evaluation Metrics

Evaluations are made with the standard evaluation metrics in image instance segmentation with modification adapted to our new task [31]. Specifically, the metrics are average precision (AP) and average recall (AR) [31] with various conditions. AP is defined as the area under the precision-recall curve [31]. The confidence score is used to plot the curve. AP is averaged over multiple intersection-over-union (IoU) thresholds [31]. Average recall describes the area doubled under the Recall-IoU curve. As a conditional AP and AR, we follow the COCO evaluation procedure as it requires 10 IoU thresholds from 50% to 95% at step 5%. As we are in video domain, we need to include temporal consistencies in our evaluations, e.g., even the model produces successful segmentations, if it fails to track the instances, it points the bad performance. Hence, our IoU computation is different from image instance segmentation because each instance contains a sequence of masks [31] so IoU computation is extended to batch of video frames by taking accumulation of IoUs across video frames. The IoU computation is as follows. Here, m_t^i represents ground truth and $\tilde{m_t^i}$ represents the hypothesis.

$$IoU(i,j) = \frac{\sum_{t=1}^{T} \left| m_t^i \cap \tilde{m_t^i} \right|}{\sum_{t=1}^{T} \left| m_t^i \cup \tilde{m_t^i} \right|} \tag{8}$$

4.3. Main Results

We compare TT-SPN with other state-of-the-art methods in VIS in terms of speed and accuracy on YouTube-VIS. Since our method is single stage and end-to-end trainable, we give priority to compare our method with single stage and end-to-end trainable approaches. We compared TT-SPN with Mask Track R-CNN [31], MaskProp [3], VisTR [29] and STEm-Seg [1]. Summary of results are presented in table 1.

Without any knitting, TT-SPN is one of the fastest methods as runs in 55.3 FPS in single GPU and achieves competitive accuracy on YouTube-VIS as 39.3 % mAP by outperforming VIS baseline model by a significant margin.

In terms of speed, TT-SPN is placed as the second winner among state-of-the-art VIS models. The current winner in terms of speed is VisTR [29] as it runs at 57.7 with ResNet-101 [14] backbone and at 69.9 with ResNet-50 backbone [14]. TT-SPN outperforms the current VIS baseline model Mask Track R-CNN, which runs at 20.0 FPS, by significant margin in terms of speed. This margin is originated from the simple attention based mechanism of TT-SPN that requires least steps to produce VIS predictions. Another competitive method, STEm-seg, runs at 2.1 FPS that is highly unusable in real time purposes. The speed of MaskProp is not mentioned in their paper [3]. Note that data loading and preprocessing step times are not included in mentioned results.

In terms of accuracy, TT-SPN outperforms Mask Track R-CNN by a significant margin, as our model achieves 39.3 % mAP score on validation set of YouTube-VIS wherease Mask Track R-CNN achieves 30.3 % mAP score. This significant margin is originated from the structure of TT-SPN, which consist of state-of-the-art approaches in all components. Further, TT-SPN also outperform STEm-seg by large margin as STEm-seg achieves 34.6 % mAP score with ResNet-101 backbone. As TT-SPN is similar to VisTR, VisTR with ResNet-101 backbone outperforms TT-SPN by 0.8 % mAP score whereas TT-SPN outperforms VisTR with ResNet-50 backbone by 3.1 % mAP score. The current winner, MaskProp achieves 46.6 % mAP score and it outperforms TT-SPN by a large margin. The gap between TT-SPN and MaskProp is originated from the multi-network design of MaskProp, which consist of Spatio-temporal Sampling Network [4], Feature Pyramid Network [17], Hybrid Task Cascade Network [8] and High-Resolution Mask Refine-

ment post processing [3]. With being one of the simplest VIS architecture, TT-SPN achieves one of the fastest and accurate results among all competitors. Further, TT-SPN can easily divided into its sub components to perform individual VIS tasks, i.e., object detection, instance segmentation and classification. These makes our approach simple, unified and real-time without sacrificing the quality of instance masks.

5. Conclusion

In this work, we proposed an end-to-end transformer based video instance segmentation module with Sinusoidal Representation Networks (SPN), namely TT-SPN, to address the video instance segmentation task. TT-SPN, views the VIS task as direct sequence prediction problem in single state that enables us to aggregate temporal information with spatial information. To produce high quality features extracted from the video frames, we utilized the twin transformer. Classical transformer is used to produce a sequence of instance predictions that later passed through the modified Sinusoidal Representation Networks to get end results. TT-SPN is a natural paradigm that handles tracking via similarity learning that enables the system to produce a fast and accurate set of predictions. TT-SPN is trained end-to-end with set-based global loss that forces unique predictions via bipartite matching that yields a decrease in general complexity of the pipeline without sacrificing quality of segmentation masks. For the first time, the VIS problem is addressed without conventional CNN architectures thanks to twin transformers with being one of the fastest approaches. Our method can be easily divided into its subcomponents to produce separate instance masks and bounding boxes that will make it unified approach for many vision tasks. We believe that video instance segmentation is a crucial task in the line of video understanding and will innovate the computer vision research community. Our project page is at `https://github.com/cankocagil/TT-SPN` and seperate detection/segmentation version of TT-SPN is at `https://github.com/cankocagil/TT-SPN---Object-Detection`.

References

[1] A. Athar, S. Mahadevan, A. Ošep, L. Leal-Taixé, and B. Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos, 2020.

[2] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization, 2016.

[3] G. Bertasius and L. Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation, 2020.

[4] G. Bertasius, L. Torresani, and J. Shi. Object detection in video with spatiotemporal sampling networks, 2018.

[5] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. *2016 IEEE International Conference on Image Processing (ICIP)*, Sep 2016.

[6] J. Cao, R. M. Anwer, H. Cholakkal, F. S. Khan, Y. Pang, and L. Shao. Sipmask: Spatial information preservation for fast image and video instance segmentation, 2020.

[7] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers, 2020.

[8] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, C. C. Loy, and D. Lin. Hybrid task cascade for instance segmentation, 2019.

[9] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, and C. Shen. Twins: Revisiting the design of spatial attention in vision transformers, 2021.

[10] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks, 2017.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation, 2014.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn, 2018.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[15] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus), 2020.

[16] C.-C. Lin, Y. Hung, R. Feris, and L. He. Video instance segmentation tracking with a modified vae architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection, 2017.

[18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection, 2018.

[19] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2015.

[20] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

[21] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019.

[22] F. Milletari, N. Navab, and S.-A. Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation, 2016.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

[25] V. Sitzmann, J. N. P. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions, 2020.

[26] J. Son, M. Baek, M. Cho, and B. Han. Multi-object tracking with quadruplet convolutional neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3786–3795, 2017.

[27] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017.

[29] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia. End-to-end video instance segmentation with transformers, 2021.

[30] Y. Wu and K. He. Group normalization, 2018.

[31] L. Yang, Y. Fan, and N. Xu. Video instance segmentation, 2019.