From VIS To OVIS: A Technical Report To Promote The Development Of The Field

Wenbo Li[†], Xuesheng Li , Qiwei Xu , Chen Li
University of Electronic Science and Technology of China
Chengdu DELU Dynamics Ltd
Huapohen AI Lab
liwenbo@huapohen.com, allanlxs@uestc.edu.cn, {xuqiwei, lichen}@deludynamics.com

Abstract

Occluded Video instance segmentation(OVIS) is a new vision task that has emerged in this years and is processed by video deep learning algorithms. It uses continuous video frames as input, generally ranging from a few frames to hundreds of frames. Before OVIS, there has a task called VIS. To tackle the task of OVIS and VIS, we design a new alghorithm called SimVTR, which based on DETR and VisTR. During the experiment, although we acquire the 27.66 mAP on OVIS test, 25.18m AP on OVIS val, and 31.9 mAP on VIS test, we have found a surprising phenomena that the evaluation mechanism is not sensitive to our mothod SimVTR. When we only use one frame to inference, the model can acquire the similar mAP as dozens frames. SimpleVTR trade off and optimizes the computing resources and effects of end-to-end video instance segmentation algorithm. We used one RTX1080Ti (11G) to experiment, and the batch size can change from 1 to 16 frames. We were surprised to find that only one frame can also get a very high score in inference. The VIS and OVIS cocoapi have some unreasonable place in ytvoseval.py. In this technical report, we prudently point out the phenomena that the evaluation mechanism could have some bug. If this is true, we need check our model to promote the process of the video instance segmentation.

1. Introduction of SimVTR

Occluded video instance segmentation (OVIS) [30]and video instance segmentation(VIS) [2][9][10][11] is a new vision task that has emerged in recent years and is processed by deep learning algorithms. VIS and OVIS occupies an important position in industrial production, such as autonomous driving, film and television post-processing, and video editing. The instance segmentation task of video uses continuous frames as input. These continuous frames are extracted from a small segment of video, usually ranging from a few frames to dozens of frames, and output continuous frames with the results of the instance mask segmentation. The difference from the classic image instance segmentation task is that the video task segments specific instances from consecutive frames, and at the same time, the instances between frames can be matched. However, the classic segmentation algorithms such as Mask R-CNN[17] and SOLO[6] do instance segmentation for a single image, and the instances between images within a batch do not need a matching relationship. The recent processing methods of VIS tasks such as MaskTrack-RCNN[9], STEm-Seg[11], etc. require carefully designed modules to process images and then perform tracking and matching, leaving the question of speed and complexity. The QueryTrack[29] is strong, but the we can not get the concrete realization now. The newly proposed method VisTR[2] is based on DETR[1]. It processes continuous frame images end-to-end without complicated post-processing. The VisTR model architecture is simple and the effect is competitive, but there are problems such as large memory usage of the model, not friendly to small objects, and insufficient fineness of the instance mask. Based on VisTR, we design a model call SimVTR to solve the problem of GPU memory consumption, and finally used one GTX1080Ti for training. In order to use Transformer[3] for OVIS tasks, some optimization proposals are pointed out.

We empirically pointed out the following proposals:

- After decoupling the frame and the instance in the query embedding, any frame can be inferred. That is, the training uses 16 frames and the inference can use 64 frames or only one frame.
- DCN[12] is an important component in SimVTR. Because the feature map of the instances is obtained by copying feature map in mask head FPN[7], the training difficulty increases after removing DCN and DCNv2[13] is more adaptable to network adjustments than DCN.
- The mask head can be simplified and some modules such as 3D convolution can be removed from VisTR, but it needs to follow the processing pipeline of DETR's FPN.
- In our experiment, using the AdaBelief[15] optimi-

zer can speed up training, and the learning rate setting should not greater than 1e-4.

- During inference, we can just input one frame the the model, and then acquire a high mAP on OVIS val/test dataset. The phenomena reflect the most urgent question that the youtubevos cocoapi could not adapt the number of frames. We think this is our most contribution for promoting the video instance segmentation despite we can not acquire a high mAP on test dataset in VIS 2021 and OVIS 2021[30].

2. Related Work

DETR. DETR[1] is based on ResNet[16] framework and Transformer[3] framework. As backbone, ResNet is used as the CNN feature extractor to extract the features of the last layer of each block. The feature of the output layer of the last block are used as the memory of the Transformer[3]. DETR uses the seq2seq[18] mechanism of the encoder-decoder of the Transformer framework to perform sequence prediction. The prediction of the network output uses the Hungarian algorithm to construct a unique prediction through bipartite graph matching. The instance segmentation head of DETR is a simplified FPN[7]. DETR is different from the classic single-stage and two-stage target detection algorithms, and there is no elaborate manual design such as NMS post-processing and predefined anchor.

VisTR. On the basis of DETR, VisTR[2] extends the original processing pipeline of single-frame images to the processing pipeline of consecutive frames for VIS. In the feature extraction stage of ResNet, the continuous frame images are placed in the batch channel. In the Transformer stage, the batch channel size is set to 1, and the number of consecutive frames and instances are combined in one channel at the same time, and the size is equal to the number of frames multiplied by the number of instances. VisTR is based on the sequence processing capabilities of the Transformer in the DETR framework, and processes the instance tracking among consecutive frames through the Hungarian algorithm. VisTR uses additional defined query embedding to query the cross-frame instance in the output feature layer of the backbone, and processes consecutive frames at the same time, achieving a balance of speed and efficiency, and constructs an end-to-end processing method to make real-time video instance segmentation possible. However, in the model inference stage, the fixed frame input limits the number of video frames, and for a video with a small number of frames, it will take the same time as a fixed frame in the depth model stage.

QueryTrack. In the youtubevos 2021 track 2: VIS, the QueryTrack acquire 52.3 mAP on test datasets. QueryTrack, based on Mask RCNN, Sparse RCNN and DETR, is a strong VIS model, but we do not use.

3. SimpleVTR

In this paper, SimpleVTR follow the design patterns of DETR and VisTR, and move towards optimizing the VisTR network first to reduce the computing resources and stabilize the index effect, and then explore more network architectures to improve the effect. Our experiments is pretrained on VOS 2021 training dataset and OVIS 2021 training dataset, and is tested on the development server of OVIS 2021. Finally, the version with the smallest changes we called SimpleVTR was submitted to the testing server of OVIS.

3.1. Architecture

The architecture of SimpleVTR is shown in Figure 1. Without bells and whistles, we aim to reduce computing resources with minimal changes, and specially marked the key operation DCN for the instance segmentation task.

3.2. Deformable Convolution

We want to explore what is the indispensable module added to the VisTR architecture on the basis of DETR. We located the 3x3 DCN[12] module. During the experiment, we observed some interesting phenomena. Without the DCN module, it is difficult to reduce the loss during training. In addition, after training a network with a DCN module for a period of time, and then replacing it with an ordinary convolution module, the loss will rise sharply. It can be seen that the DCN module is very important to the network. We point out here: the DCN module us a big trick and an indispensable key operation for optimizing the VisTR-like network. SimpleVTR retains the original DCN to experiment on GTX1080Ti, and we have stacked two layers of 3x3 DCN at the output layer.

3.3. Backbone

The backbone of DETR uses ResNet [16], which is only used for feature extraction. As the number of frames increases, the GPU memory occupied by ResNet will increase accordingly. Both VisTR and DETR have not explored the redundancy of the backbone in Transformer-based tasks, and we want to save the small goals ignored by the VisTR architecture, because classification and bbox only use the last layer of the 4 layers taken from the backbone.

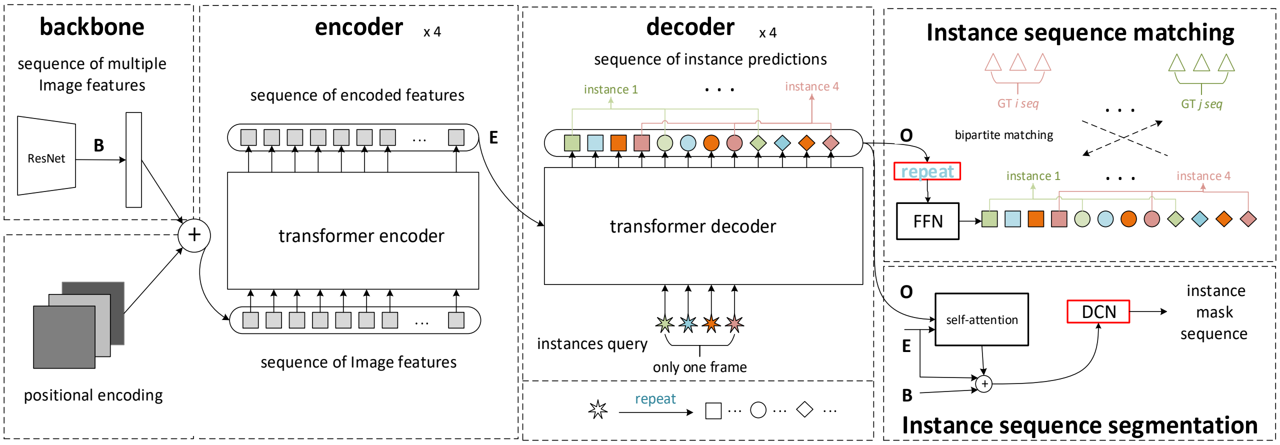

Figure 1: Architecture of SimpleVTR. We decouple *frames* and *instances* to reduce computing resources, and restore *frames* through *repeat* operation. Key operation DCN is marked with a red box. In the inference stage of SimpleVTR, *frames* can be set to any size.

We does not use Transormer[3][27] or improved Tansformer vision architecture[23][24][25][26] as backbone to replace ResNet. In the final SimpleVTR version submitted to the testing server, we still retained the original ResNet as backbone.

3.4. Query Embedding

The query embedding of SimpleVTR is defined as:

$$self.query\_embed = nn.Embedding(self.instances, hidden\_dim) \quad (1)$$

In the above formula, *self.instances* is the predefined maximu-m number of instances of a single frame *10*. There are no frames in the definition of query_embed. We separate frames and instances, keep instances and ignore frames. The decoupling of frames and instances is a key step in the single-card training VIS task. While reducing the computing resources, it also facilitates the input of arbitrary frames for the inference stage. We consider that the instances will almost always exist in consecutive frames, except for a few occlusion situations, which is OVIS[10] task. After *self.query_embed* is processed by Transformer, we will take a simple copy operation to reflect the frames。

3.5. Transformer

Transformer is derived from Attention Is All You Need[3] and has a powerful ability to process sequences. We take the features of the last layer taken out by backbone as memory for Transformer to query, that is, output the feature maps required by class task and bbox task. We reduce the encoder and decoder layers of Transformer from *k=6* to *k=4*. Because query embedding decouples frames and instances, it is faster to operate and consumes less GPU memory. The query embedding output by Transformer needs to be copied in *N=frames* copies, written as follows with einops[21] expression:

$$x = repeat\big(x, \text{"k 1 ins hw} \rightarrow \text{k 1 (frames ins) hw"}, \text{frames} = \text{self.frames}\big) \quad (2)$$

The output feature ***x*** is used as input for *class_head*, *bbox_head*, and *mask_head*. SimpleVTR follows the processing framework of VisTR, and the effect of copying *N=frames* ***x*** on *class_head* and *bbox_head* is acceptable according to the experimental results. The reasons for this analysis may be:

- the category of the instance does not change among consecutive frames
- *bbox_head* does not participate in the mask calculation, nor does it participate in the inference of the video instance segmentation. For the segmentation task, only the loss of bbox is used when calculating the total loss.

3.6. Mask Head

In SimpleVTR, we removed the 3D convolution derived from VisTR, the output layer is stacked with two DCNs, the kernel size of the convolution is 3, and the second DCN directly outputs the instance segmentation mask. We have observed that after the feature map is gradually enlarged in resolution through FPN, the memory usage will gradually increase, especially after FPN performs the last nearest neighbor interpolation. In order to adapt to the GPU memory of the GTX 1080Ti single card, we directly cancelled the 3D convolution, because we considered that after the copy operation, a new *instance* channel is added, and the 3D convolution is based on DCN to further integrate features, and its contribution to AP is limited. In other words, the more important operation is DCN instead of 3D convolution.

4. Experiments

4.1. Model Settings

The SimpleVTR we proposed inherits VisTR's hyperparameter settings based on DETR. The seed is set to 25 during inference, because we observe that the seed setting is different and mAP fluctuates close to 1 point. We first pretrain the SimpleVTR on VIS 2021 training dataset, and train the SimpleVIR and OVIS 2021. We set the learning rate to *1e-5*. For optimizer, we did not use AdamW[14] or Adam[20] but chose AdaBelief[15], in which *weight_decay* is set to *1e-4, weight_decouple* and *rectify* are both set to *True*. The number of encoder and decoder layers of Transformer is set to *4*. The number of input frames during training is change from 1 to*16, such as 1,2,4,8,10,16*. Due to *frames* and *instances* are decoupled by query embedding, we tried different sizes of input frames during inference, and found that mAP was the highest when the number of input frames was set to only one frames. It's shocking. We found an interesting phenomenon. SimpleVTR inherits VisTR's inference pipline and sets different frame numbers when inferring on the development dataset, and the obtained mAP are nearly different.

4.2. Results

SimpleVTR get 25.18 mAP on OVIS 2021 development dataset, and get 27.66 mAP on OVIS 2021 test dataset.

5. Conclusion

SimpleVTR optimizes the network based on the structure of VisTR, which reduces the computing resources required during training, and brings the possibility of training a larger number of video frames and a larger number of instances. Although SimpleVTR did not improve the effect of instance segmentation, it pointed out some problems about youtube cocoapi evaluation mechanism and key points in the original VisTR,.

Reference

[1] NicolasCarion, FranciscoMassa, GabrielSynnaeve, Nicolas Usunier, AlexanderKirillov, and SergeyZagoruyko. End-to-End Object Detection with Transformers. In *ECCV*, 2020.

[2] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, Huaxia Xia. End-to-End Video Instance Segmentation with Transformers. In *CVPR* 2021.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, 2017.

[4] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2018.

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

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

[7] Tsung-Yi Lin, Piotr Doll´ar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017.

[8] Tsung-YiLin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal Loss for Dense Object Detection. In *ICCV*, 2017.

[9] Linjie Yang, Yuchen Fan, and Ning Xu. Video Instance Segmentation. In *ICCV*, 2019.

[10] Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip H.S. Torr, Song Bai. Occluded Video Instance Segmentation. arXiv preprint arXiv:2102.01558, 2021

[11] Ali Athar, Sabarinath Mahadevan, Aljoˇsa Oˇsep, Laura LealTaix´e, and Bastian Leibe. Stem-seg: Spatio-temporal embeddings for Instance Segmentation in Videos. In *ECCV*, 2020.

[12] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable Convolutional Networks. In *ICCV*, 2017.

[13] Xizhou Zhu, Han Hu, Stephen Lin, Jifeng Dai. Deformable ConvNets v2: More Deformable, Better Results. In *CVPR*, 2019.

[14] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2017.

[15] Juntang Zhuang, Tommy Tang, Sekhar Tatikonda, Nicha Dvornek, et al. AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients. In *NeurIPS*, 2020.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[17] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

[18] Ilya Sutskever, Oriol Vinyals, Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *NIPS*, 2014.

[19] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, et al. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. In *CVPR*, 2019.

[20] Diederik P. Kingma, Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2014.

[21] Alex Rogozhnikov, Cristian Garcia, et al. Einops, https://github.com/arogozhnikov/einops, 2018

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

[23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. arXiv preprint arXiv: 2103.14030, 2021.

[24] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, Chunhua Shen. Twins: Revisiting the Design of Spatial Attention in Vision Transformers. arXiv preprint arXiv: 2104.13840, 2021.

[25] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, et al. MLP-Mixer: An all-MLP Architecture for Vision. arXiv preprint arXiv:2105.01601, 2021.

[26] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, et al. ResMLP: Feedforward networks for image classification with data-efficient training. arXiv preprint arXiv:2105.03404, 2021.

[27] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021.

[28] Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV, 2016.

[29] Tracking Instances as Queries. Shusheng Yang, Yuxin Fang, Xinggang Wang, Yu Li, Ying Shan, Bin Feng, Wenyu Liu. arXiv preprint arXiv: 2106. 11963, 2021.

[30] Occluded Video Instance Segmentation . Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip H.S. Torr, Song Bai . arXiv preprint arXiv: 2102.01558, 2021.