Supplementary Material for End-to-End Referring Video Object Segmentation with Multimodal Transformers

Adam Botach, Evgenii Zheltonozhskii, Chaim Baskin
Technion – Israel Institute of Technology
{botach, evgeniizh, chaimbaskin}@campus.technion.ac.il

A. Additional Method Details

A.1. Loss and Cost Functions

In this section we present the full definitions of the Dice [8] and Focal [5] cost and loss functions used in Sections 3.5 and 3.6 of the paper.

Let $m_{\text{GT}} = \{m^t_{\text{GT}}\}_{t=1}^{T_{\mathcal{I}}}$ and $m_{\text{Pred}} = \{m^t_{\text{Pred}}\}_{t=1}^{T_{\mathcal{I}}}$ be ground-truth and prediction mask sequences respectively. Also, denote by N_p the number of pixels a mask has and by $p_i \in \mathbb{R}$ the i^{th} pixel in a given mask.

A.1.1 Dice Cost and Loss

Given two segmentation masks m_A, m_B, the Dice coefficient [8] between the two masks is defined as follows:

$$\text{DICE}(m_A, m_B) = \frac{2 * |m_A \cap m_B|}{|m_A| + |m_B|}, \tag{A.1}$$

where for a mask m, $|m| = \sum_{i=1}^{N_p} p_i$ and $|m_A \cap m_B| = \sum_{i=1}^{N_p} p_i^A \cdot p_i^B$. Note that in practice we also add a smoothing constant $s = 1$ to both the numerator and denominator of the above expression to avoid possible division by 0.

Given the above, the Dice cost $\mathcal{C}_{\text{Dice}}$ between the mask sequences m_{GT} and m_{Pred} is defined as

$$\mathcal{C}_{\text{Dice}}(m_{\text{GT}}, m_{\text{Pred}}) = -\tfrac{1}{T_{\mathcal{I}}} \textstyle\sum_{t=1}^{T_{\mathcal{I}}} \text{DICE}(m^t_{\text{GT}}, m^t_{\text{Pred}}). \tag{A.2}$$

Similarly, the Dice loss $\mathcal{L}_{\text{Dice}}$ between the two sequences is defined as

$$\mathcal{L}_{\text{Dice}}(m_{\text{GT}}, m_{\text{Pred}}) = \textstyle\sum_{t=1}^{T_{\mathcal{I}}} 1 - \text{DICE}(m^t_{\text{GT}}, m^t_{\text{Pred}}). \tag{A.3}$$

A.1.2 Focal Loss

The Focal loss [5] between two corresponding segmentation masks m^t_{GT} and m^t_{Pred} for time step t is defined as

$$\text{FL}(m^t_{\text{GT}}, m^t_{Pred}) = \tfrac{1}{N_p} \textstyle\sum_{i=1}^{N_p} -\alpha_i^T (1 - p_i^T)^\gamma \log(p_i^T), \tag{A.4}$$

where p_i^T is the probability predicted for the ground-truth class of the i^{th} pixel:

$$p_i^{\text{T}} = \begin{cases} p_i^{\text{Pred}} & p_i^{\text{GT}} = 1 \\ 1 - p_i^{\text{Pred}} & \text{otherwise,} \end{cases} \tag{A.5}$$

and $\alpha_i^T \in [0, 1]$ is a class balancing factor defined as

$$\alpha_i^T = \begin{cases} \alpha & p_i^{\text{GT}} = 1 \\ 1 - \alpha & \text{otherwise.} \end{cases} \tag{A.6}$$

Following [5, 11] we use $\alpha = 0.25, \gamma = 2$. We refer to [5] for more information about these hyperparameters.

Given the above, the Focal loss $\mathcal{L}_{\text{Focal}}$ between the ground-truth and predicted mask sequences m_{GT} and m_{Pred} is defined as

$$\mathcal{L}_{\text{Focal}}(m_{\text{GT}}, m_{\text{Pred}}) = \sum_{t=1}^{T_{\mathcal{I}}} \text{FL}(m^t_{\text{GT}}, m^t_{Pred}). \tag{A.7}$$

B. Additional Dataset Details

B.1. A2D-Sentences & JHMDB-Sentences

A2D-Sentences [2] contains 3,754 videos (3,017 train, 737 test) with 7 actors classes performing 8 action classes. Additionally, the dataset contains 6,655 sentences describing the actors in the videos and their actions. JHMDB-Sentences [2] contains 928 videos along with 928 corresponding sentences describing 21 different action classes.

B.2. Refer-YouTube-VOS

The original release of Refer-YouTube-VOS [9] contains 27,899 text expressions for 7,451 objects in 3,975 videos. The objects belong to 94 common categories. The subset with the *first-frame expressions* contains 10,897 expressions for 3,412 videos in the train split and 1,993 expressions for 507 videos in the validation split. The subset with the *full-video expressions* contains 12,913 expressions for 3,471

videos in the train split and 2,096 expressions for 507 videos in the validation split. Following the introduction of the RVOS competition[1], only the more challenging *full-video expressions* subset is publicly available now, so we use this subset exclusively in our experiments. Additionally, this subset's original validation set was split into two separate competition validation and test sets of 202 and 305 videos respectively. Since ground-truth annotations are available only for the training set and the test server is currently closed, we report results exclusively on the competition validation set by uploading our predictions to the competition's server[2].

C. Additional Implementation Details

C.1. Temporal Encoder Modifications

The original architecture of Video Swin Transformer [6] contains a single temporal down-sampling layer, realized as a 3D convolution with kernel and stride of size $2 \times 4 \times 4$ (the first dimension is temporal). However, since our multimodal Transformer expects per-frame embeddings, we removed this temporal down-sampling step by modifying the kernel and stride of the above convolution to size $1 \times 4 \times 4$. In order to achieve this while maintaining support for the Kinetics-400 [3] pretrained weights of the original Swin configuration, we summed the pretrained kernel weights of the aforementioned convolution on its temporal dim, resulting in a new $1 \times 4 \times 4$ kernel. This solution is equivalent to (but more efficient than) duplicating each frame in the input sequence before inserting it into the temporal encoder.

C.2. Multimodal Transformer

We employ the same Transformer architecture proposed by Carion et al. [1]. The decoder layers are fed with a set of $N_q = 50$ object queries per input frame. For efficiency reasons we only utilize 3 layers in both the encoder and decoder, but note that more layers may lead to additional performance gains, as demonstrated by Carion et al. [1]. Also, similarly to Carion et al. [1], fixed sine spatial positional encodings are added to the features of each frame before inserting them into the Transformer. No positional encodings are used for the text embeddings, as in our experiments using sine embeddings have led to reduced performance and learnable encodings had no effect compared to using no encodings at all.

C.3. Instance Segmentation

The spatial decoder G_{Seg} is an FPN-like [4] module consisting of several 2D convolution, GroupNorm [12] and ReLU layers. Nearest neighbor interpolation is used for the upsampling steps. The segmentation kernels and the feature maps of $\mathcal{F}_{\text{Seg}}$ are of dimension $D_s = 8$ following [10].

C.4. Additional Training Details

We use $D = 256$ as the feature dimension of the multimodal Transformer's inputs and outputs. The hyper-parameters for the loss and matching cost functions are $\lambda_r = 2, \lambda_d = 5, \lambda_f = 2$.

Following Carion et al. [1] we utilize AdamW [7] as the optimizer with weight decay set to 10^{-4} during training. We also apply gradient clipping with a maximal gradient norm of 0.1. A learning rate of 10^{-4} is used for the Transformer and $5 \cdot 10^{-5}$ for the temporal encoder. The text encoder is kept frozen.

Similarly to Carion et al. [1] we found that utilizing auxiliary decoding losses on the outputs of all layers in the Transformer decoder expedites training and improves the overall performance of the model.

During training, to enhance model's position awareness, we randomly flip the input frames horizontally and swap direction-related words in the corresponding text expressions accordingly (e.g., the word 'left' is replaced with 'right').

We train the model for 70 epochs on A2D-Sentences [2]. The learning rate is decreased by a factor of 2.5 after the first 50 epochs. In the default configuration we use window size $w = 8$ and batch size of 6 on 3 RTX 3090 24GB GPUs. Training takes about 31 hours in this configuration. On Refer-YouTube-VOS [9] the model is trained for 30 epochs, and the learning rate is decreased by a factor of 2.5 after the first 20 epochs. In the default configuration we use window size $w = 12$ and batch size of 4 on 4 A6000 48GB GPUs. Training takes about 45 hours in this configuration.

D. Additional Experiments

D.1. Ablations

Number of object queries. To study the effect of the number of object queries on MTTR's performance, we train and evaluate our model on A2D-Sentences using window size $w = 6$ and different values of N_q. As shown in Tab. D.1, the best performance is achieved for $N_q = 50$. Our hypothesis is that when using lower values of N_q the resulting set of object queries may not be diverse enough to cover a large set of possible object detections. On the other hand, using higher values of N_q may require a longer training schedule to obtain good results, as the probability of each query being matched with a ground-truth instance (and thus updated) at each training iteration is lower.

D.2. Analysis of the Effect of TSVS

To further illustrate and analyze the effect of the *temporal segment voting scheme* (TSVS), we refer to the zebras example in the third row of Figure 3 (in the paper) and the orange

[1] https://youtube-vos.org/dataset/rvos/

[2] https://competitions.codalab.org/competitions/29139

N_q	IoU		mAP
	Overall	**Mean**	
10	**70.1**	61.0	42.5
50	69.5	**61.8**	**44.0**
100	69.2	61.3	41.5
200	68.5	60.8	42.8
300	67.4	59.7	42.7

Table D.1. Ablation on number of object queries.

text query. Without TSVS, a prediction would have to be made for each frame in the video *separately*. Hence, as the correct zebra (marked in orange) is not yet visible in the first two frames, one of the other visible zebras in each of these frames may be wrongly selected. With TSVS, however, the predictions of the correct zebra in the final three frames vote *together* as part of a sequence, and due to the high reference scores of these predictions, this sequence is then selected over all other instance sequences. This results in only the correct zebra being segmented throughout the video (i.e., no zebra is segmented in the first two frames), as expected.

References

1. Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision (ECCV)*, pages 213–229. Springer, August 2020. (cited on p. 2)
2. Kirill Gavrilyuk, Amir Ghodrati, Zhenyang Li, and Cees G. M. Snoek. Actor and action video segmentation from a sentence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. (cited on pp. 1 and 2)
3. Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. (cited on p. 2)
4. Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. (cited on p. 2)
5. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. (cited on p. 1)
6. Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. *arXiv preprint arXiv:2106.13230*, 2021. (cited on p. 2)
7. Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. (cited on p. 2)
8. Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-Net: fully convolutional neural networks for volumetric medical image segmentation. In *Fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016. (cited on p. 1)
9. Seonguk Seo, Joon-Young Lee, and Bohyung Han. URVOS: Unified referring video object segmentation network with a large-scale benchmark. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision (ECCV)*, pages 208–223. Springer, August 2020. (cited on pp. 1 and 2)
10. Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision (ECCV)*, pages 213–229. Springer, August 2020. (cited on p. 2)
11. Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8741–8750, June 2021. (cited on p. 1)
12. Yuxin Wu and Kaiming He. Group normalization. In *European Conference on Computer Vision (ECCV)*, pages 3–19, September 2018. (cited on p. 2)