Spatio-temporal Prompting Network for Robust Video Feature Extraction Supplementary Material

Guanxiong Sun[1, 2], Chi Wang [1], Zhaoyu Zhang [1], Jiankang Deng [2, 3], Stefanos Zafeiriou [3], Yang Hua[1]

[1]Queen's University Belfast [2]Huawei UKRD [3]Imperial College London
{gsun02, cwang38, zzhang55,Y.Hua}@qub.ac.uk, {j.deng16,s.zafeiriou}@imperial.ac.uk

We introduce detailed implementations of STPN. We use Python 3.7 and PyTorch 1.8.1 [8], and conduct experiments on NVIDIA Tesla V100-32GB GPUs.

1. STPN on CVT

CVT [10] is the default transformer encoder used in MixFormer [1] for visual object tracking. The architecture of CVT is slightly different from other transformer encoders [6, 4] mainly because CVT uses convolutional layers to generate down-scaled feature maps whereas other encoders are down-scaled by reshaping and concatenating. As a result, the predicted DVP by STPN should also be compatible with the convolution-based downscale layers in CVT.

The convolution-based downscale layer in CVT is implemented by Conv2d layers with kernel size 3×3 and stride 2. The predicted dynamic video prompts (DVPs) consist of N_P embeddings. To enable Conv2d on DVPs, we increase N_P from the default value of 5 to 9, so that we can reshape the DVPs to a 3×3 feature map. The reshaped DVPs are then passed into the convolution-based downscale layer to generate down-scaled embeddings. Additionally, we add zero padding of size 2 on the reshaped DVPs before passing it to the Conv2d layer to ensure the output DVPs have the same size as the input. Therefore, the input size and the output size of DVPs are the same which is 3x3. Finally, the output DVPs are reshaped back to 9 embeddings and then prepended with down-scaled feature maps for the following transformer layer.

2. Training and Inference

Details of hyper-parameters we used for video object detection (VOD), video instance segmentation (VIS), and visual object tracking (VOT) are listed in the Table 1. For VOD and VIS, following the training protocols in FasterRCNN [9] and MinVIS [2], the whole model is trained end-to-end in a single stage. In contrast, following the training protocol in MixFormer [1], the training process is divided into two stages. The parameters in the score prediction head [1] are trained in the second stage. All other parameters including the DVP predictor and CVT backbone are trained in the first stage.

3. Grad-CAM Details

We use the EigenCAM [7] for visualising the class activation maps (CAM) for STPN on the task of video object detection. During the visualisation progress, two extra modifications are needed compared with the normal grad-cam visualisation for the task of image classification. Firstly, we need to formulate a customised "reshape" transformation that integrates the stored activations in the FasterRCNN [9] output features (from the feature pyramid network (FPN) [5]). Specifically, we re-scale all feature levels of FPN to the same scale, the scale of 64× in our implementation. Secondly, we need to construct a "target" function that generates CAMs optimised for specific bounding boxes, such as their score or their intersection over union with the original bounding boxes. More implementation details can be found in the GitHub page [1].

4. t-SNE Details

Following the protocol in FGFA [11], we categorise the ImageNet VID Val set into three groups: fast-speed, medium-speed, and slow-speed subsets. The definition is based on the Motion Intersection over Union (mIoU) metric which measures the IoU of the same object in the nearby frames (±10 frames). The specific thresholds are mIoU > 0.9 (slow), mIoU $\in [0.7, 0.9]$ (medium), and mIoU < 0.7 (fast). Slow-speed subset usually has higher quality than the medium-speed and the fast-speed subsets. Therefore, STPN improves the FasterRCNN detector more significantly in the medium-speed and fast-speed subsets.

[1] https://github.com/jacobgil/pytorch-grad-cam

Task	Optimizer	Batch Size	Base LR	LR Drop Rate	LR Schedule	Total	Weight Decay
VOD	ADAMW [3]	8	0.000025	0.1	6E	9E	0.05
VIS		64	0.0001		4000I	6000I	0.05
VOT_{stage1}		32	0.001		400E	500E	0.0001
VOT_{stage2}		32	0.0001		30E	40E	0.0001

Table 1. Details of hyper-parameters for training STPN on three different video tasks, i.e., video object detection (VOD), video instance segmentation (VIS), and visual object tracking (VOT). E and I denote epoch and iteration, respectively.

t-SNE requires a classification label for each sample feature. So we need to convert the feature maps within bounding boxes into sample features. Given a bounding box, we use the RoIPooling [9] operation to generate a sample feature (proposal). Specifically, we use the ground-truth bounding boxes of the ImageNet VID Val set to generate sample features. The labels of each sample feature are set as the label of the corresponding ground-truth bounding box. In this way, we can compare the quality of feature maps obtained by FasterRCNN with and without STPN.

References

[1] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *CVPR*, 2022. 1

[2] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. In *NeurIPS*, 2022. 1

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[4] Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1

[5] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1

[7] Mohammed Bany Muhammad and Mohammed Yeasin. Eigen-cam: Class activation map using principal components. In *IJCNN*, 2020. 1

[8] 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 *NIPS*, 2019. 1

[9] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2

[10] Haiping Wu, Bin Xiao, Noel C. F. Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *ICCV*, 2021. 1

[11] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*, 2017. 1