Few-Shot Video Object Detection

Qi Fan[1], Chi-Keung Tang[1], and Yu-Wing Tai[1,2]

[1] The Hong Kong University of Science and Technology, [2] Kuaishou Technology
fanqics@gmail.com, cktang@cs.ust.hk, yuwing@gmail.com

1 More Implementation Details

1.1 Matching Network Architecture

We adopt the multi-relation head from FSOD [3] as our matching network (MN), which consists of three relation heads for learning to match support and query features $\{f_s, f_q\} \in \mathbb{R}^{1\times C\times 7\times 7}$ in multiple levels.

Global-relation head. Designed to learn a global matching embedding, this head first concatenates f_s and f_q along the channel dimension to feature $f_c^{'} \in \mathbb{R}^{1\times 2C\times 7\times 7}$, which is then average pooled to $f_c \in \mathbb{R}^{1\times 2C\times 1\times 1}$. Finally, a MLP $\mathcal{M}$ containing three fully connected layers with ReLU (except the last one) is applied to f_c to predict the matching score $s_g = \mathcal{M}(f_c)$.

Patch-relation head. Designed to learn a non-linear metric to capture the complex relation between patches, this head is derived from the RelationNet [5] where the concatenated feature $f_c^{'}$ is fed to a small convolution network, which consists of two 3×3 average pooling operators at the first and last layers separately, two 1×1 convolutional layers for reducing and then restoring dimensions, and one 3×3 convolutional layer (all convolutional layers are equipped with ReLU). Note that all these operations and layers use one stride and zero padding to generate the final feature vector $f_q \in \mathbb{R}^{1\times C\times 1\times 1}$. Finally, a fully connected layer is employed to generate the matching score s_p, and a sibling fc layer to generate the box prediction for better supervision from multi-task learning.

Local-relation head. Designed to capture the pixel-level relation between support and query features, f_s and f_q are first processed using a weight-shared convolution layer with ReLU. Then their pixel-wise relation is calculated by depth-wise correlation [4] with the resulting feature vector $f_d \in \mathbb{R}^{1\times C\times 1\times 1}$ fed to a fully connected layer to generate the matching score s_l.

These three relation heads cooperate together to capture the relation between support and query features in different levels. The final matching score is obtained by summing all the aforementioned matching scores: $s = s_g + s_p + s_l$.

1.2 Deformable RoIAlign

Deformable RoIAlign dynamically changes its sample locations according to the input features. In our implementation, two frame features are concatenated and sent to the deformable RoIAlign so that it is aware of the object positions in both frames and therefore dynamically adapt the sample locations to enlarge the search region to capture objects in both frames.

1.3 Training Details

The stride of the Res5 block is reduced to 1 to increase feature map resolution. We replace its regular convolutional layer with the dilated convolutional layer to keep the effective receptive field. Following common practices, the low-level layers (Res1 and Res2) are fixed and only the high-level layers are trained. As for the inputs, the query image is resized to (600, 1000) where the shorter and longer sizes are respectively no longer than 600 and 1000 pixels. We also adopt the multi-scale training for query images during training. As for the support and aligning query images, they are cropped and resized to 320×320 size with extended 16-pixels around the target object and the cropped images are saved to the disk for efficient training to avoid repeating the crop for the same image.

1.4 Evaluation Details

During inference, the final score of each box is obtained by multiplying the matching score predicted by TMN and the corresponding objectness score generated by TPN to suppress the scores of boxes containing hard background[1].

Instead of setting a fixed support set which is only used for support images, we exploit a support set which can fully utilize the val and test set in a dynamic manner for more comprehensive evaluation on all videos.

The following description applies to the val set which is similar to the test set. Our *dynamic* support set contains "offline" and "online" support sets. The support images in the offline support set are derived from the randomly selected val set videos V_{offline}. The support features f_{offline} are pre-computed and saved to the hard drive for efficient evaluation[2]. Then we can load the pre-computed support features to the model to perform detection on query videos. When performing evaluation on V_{offline}, we build the online support sets by randomly selecting images from the remaining videos, and the support features are online generated for the evaluation[3]. In this way, we avoid the "cheap matching" between same objects which is degraded to the single object tracking task. The dynamic support set can dynamically decide the support sets for different videos, and therefore efficiently utilizes the entire val set to perform evaluation without leaving a fraction of videos as the specialized support set. Note that the video-level annotation is much more expensive and time-consuming than the image-level annotation. With our dynamic support set we can avoid wastage of valuable video data.

Note that the finetuning-based methods cannot be directly compared with matching-based methods because of the former's high requirement for support sets, which requires training on novel classes in a reserved support set. This

[1] The background may have high matching score because of the similar appearance with supports. The low objectness score predicted by TPN can down-weigh the overall score to alleviate this influence.

[2] One class has one corresponding support feature in the $C \times 1 \times 1$ size and C is the channel number.

[3] We only use them for V_{offline} with a small number of videos without reusing again. It is also feasible to first save them to the hard drive.

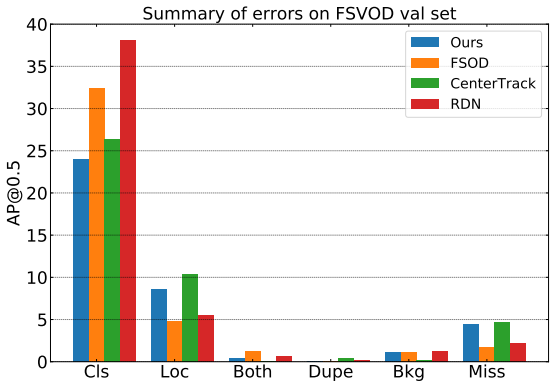

Fig. 1. Error type analysis of different methods on FSVOD-500 val set. Lower is better.

limits the application of the finetuning-based methods, because it is impossible to exhaustively annotate all videos[4] for each novel class. This issue can be solved by annotating a special support set (it is very time and money consuming) to finetune these models.

2 Error Type Analysis

To conduct an in-depth investigation of different models on the FSVOD task, we analyze the error types on four representative models, namely, FSOD [3], RDN [2], CenterTrack [6] and our FSVOD using a general toolbox TIDE [1] which segments object detection errors into six types and measures the contribution of each error by isolating its effect on overall performance (refer to [1] for more details).

Figure 1 indicates that all methods suffer from classification errors on our FSVOD-500 dataset, revealing that the core problem of FSVOD lies on few-shot learning in distinguishing and classifying novel classes, which cannot be replaced by other video understanding tasks.

For individual performances: FSOD has the lowest localization and missing error thanks to the high-quality proposals generated by its attention RPN; the VID-based model RDN mainly suffers from classification error because it generates too many background proposals which exacerbate the following matching procedure; the MOT-based model CenterTrack has lower classification error benefiting from the robust tube-based feature, but it suffers higher localization and missing errors caused by its lower recall. Our approach has the lowest classification error benefiting from our strategically designed TMN+ which leverages the representative tube-based features generated by TPN.

[4] In our case, for a novel class, we need to annotate at least one video and one support image containing a different object belonging to the same class to avoid "cheap matching".

Way	Shot	AP	AP_{50}	AP_{75}
1	1	44.0	68.6	45.7
1	5	46.5	71.9	48.3
2	1	39.6	61.0	41.3
2	5	45.2	69.7	47.0
5	1	31.9	49.1	33.1
5	5	42.8	65.7	44.7

Table 1. Experimental results on FSVOD-500 val set of our model under different few-shot evaluation settings.

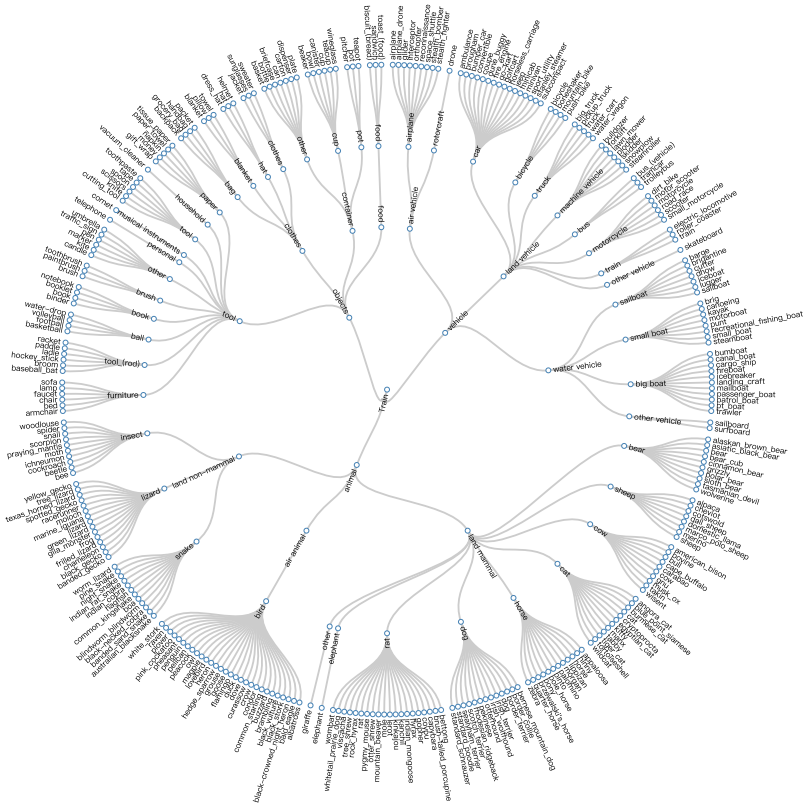

Fig. 2. Class hierarchy of FSVOD train set.

From the above error analysis, we conclude that solving the few-shot matching problem is the most essential future direction for FSVOD. We show more experimental results under different few-shot evaluation settings in Table 1.

3 Full Dataset Hierarchy

The full dataset hierarchy of FSVOD is shown in Figure 2 (train set) and Figure 3 (val and test sets).

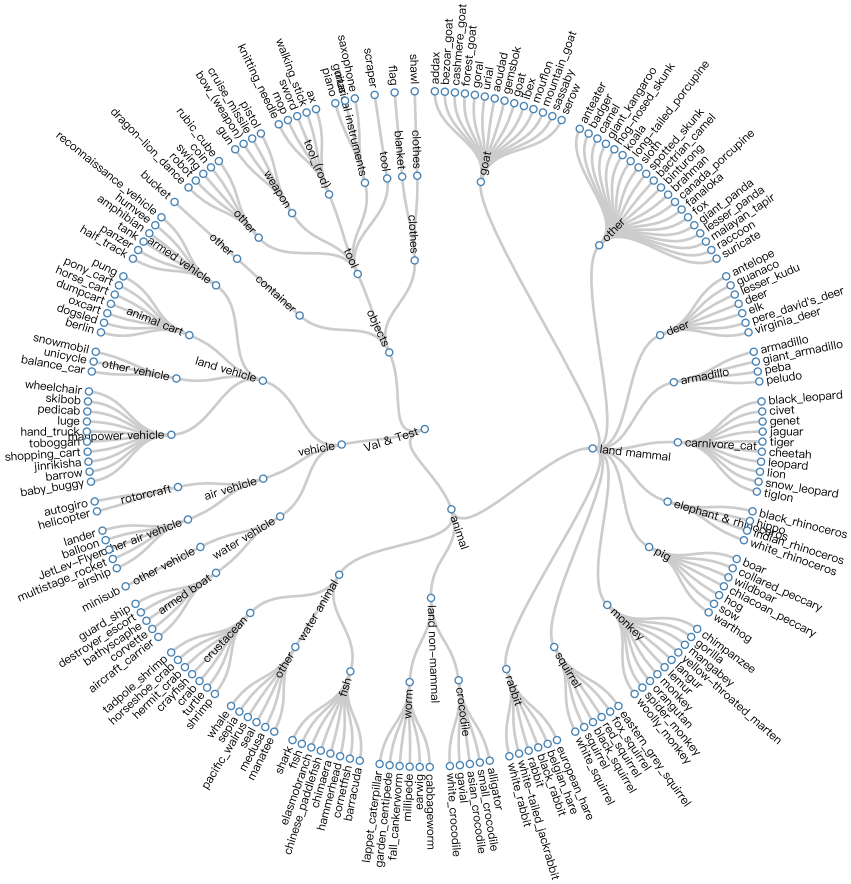

Fig. 3. Class hierarchy of FSVOD val and test sets.

4 Object Localization in Massive Videos

In the main paper, we propose a common realistic problem: *Given a bunch of videos, how can we index and localize all novel objects of interest as video clips?.*

The practical solution is to detect objects in these videos and index/localize frames based on the detection results. Specifically, if there is a detection prediction for the target class, we index/localize this video frame.

The fully-supervised methods (*e.g.*, object detection and multiple object tracking) can not solve this problem, because the interested objects can belong to arbitrary classes.

The single object tracking tasks can not solve this problem, because the video is massive and arbitrary, while the single object tracking requires the per-video annotated template for the first frame. Furthermore, the interested class may occur in discrete video clips and there are possibly multiple objects for the target class. Thus the single object tracking methods cannot handle these realistic cases.

The image-level few-shot learning tasks (*e.g.*, few-shot image/video classification) can not solve this problem, because the interested objects is probably very small, while the image classification cannot properly represent small objects.

The few-shot image object detection cannot properly solve this problem, because its methods are specifically designed for still images without the consideration for the temporal information.

The video object detection based methods are better than image detection based method because of their better detection results. The multi-object tracking methods significantly improve the precision/recall/F1 performance thanks to the tube-based tracking. Note these methods are all adapted for few-shot learning. Our method has the best performance thanks to our tube proposal network and temporal matching strategy.

References

1. Bolya, D., Foley, S., Hays, J., Hoffman, J.: Tide: A general toolbox for identifying object detection errors. In: ECCV (2020) 3
2. Deng, J., Pan, Y., Yao, T., Zhou, W., Li, H., Mei, T.: Relation distillation networks for video object detection. In: ICCV (2019) 3
3. Fan, Q., Zhuo, W., Tang, C.K., Tai, Y.W.: Few-shot object detection with attention-rpn and multi-relation detector. In: CVPR (2020) 1, 3
4. Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., Yan, J.: Siamrpn++: Evolution of siamese visual tracking with very deep networks. In: CVPR (2019) 1
5. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: CVPR (2018) 1
6. Zhou, X., Koltun, V., Krähenbühl, P.: Tracking objects as points. In: ECCV (2020) 3