Supplementary Material: TraMNet - Transition Matrix Network for Efficient Action Tube Proposals

Anonymous ACCV 2018 submission

Paper ID 153

In this supplementary material, we first present implementation details in Section 1. Next, we show how optical flow and appearance based feature contribute to the final performance in Section 2. Finally, we show how the transition matrix helps in dynamic classes over AMTNet-L in Section 3.

1 Implementation details

All models are trained with a batch size of 16 on two 1080Ti GPUs (11GB VRAM each) for AMTNet-L and TraMNet. Whereas, ACT-L is trained on 4 GPUs because it requires fitting feature maps of 6 frame into GPU memory. We used Pytorch library to implement all the models (ACT-L, AMTnet-L, and TraMNet) by following the implementation of Singh *et al.*[1] closely from their GitHub repository.

UCF1101-24. We trained all our models using the data-loader function provided by Singh *et al.* [1] where 5000 training images are picked for each class to train. The RGB stream is trained with an initial learning rate of 0.001 and it is dropped by a factor of 10 after $110K$ iterations, and the model is trained for $180K$ iterations for all the different detection frameworks, i.e., ACT-L, AMTMet-L, and TraMNet. The flow stream is trained with an initial learning rate of 0.001 and it is dropped by a factor of 10 after $150K$ iterations, and the model is trained for $220K$ iterations for all the different detection frameworks, i.e., ACT-L, AMTMet-L, and TraMNet.

DALY. We use all the action instances of actions available even if the number of frames annotated in a given instance is less than 2. AMTNet-L and TraMNet requires at least two consecutive frames to train, but in case of DALY dataset, there are action instances where only one frame is annotated, we use annotated frame twice to make a sequence of two frames, as mentioned in the paper this happens for 12% of action instances. Remaining instances contribute according to the number of frames annotated per instance.

The RGB stream is trained with an initial learning rate of 0.001 and it is dropped by a factor of 10 after $12K$ iterations, and the model is trained for $30K$ iterations for all the different detection frameworks, i.e., AMTNet-L and TraMNet. The Flow stream is trained with an initial learning rate of 0.001 and it is dropped by a factor of 10 after $15K$ iterations, and the model is trained for $50K$ iterations for all the different detection frameworks, i.e., AMTNet-L and TraMNet.

Input Data Preparations. The number of input video frames to CNNs is more than 1 in our case, e.g., 6 for ACT-L action detection framework. The optical flow stream uses a stack of 5 frames as input for each input training example in the sequence. In our case, sequence length (s) is more than (in case AMTNet-L and TraMNet equal to 2

and ACT-L equal to 6), so 2 or 6 ($s = 2$) stacks of 5 frames are used as input for flow stream in case of TraMNet for flow stream.

The number of input frames for the RGB stream is equal to the sequence length, e.g., 2 for TraMNet. Each optical flow image is a 3 channel image, where first two channels are flow in x and y direction respectively and the third channel is the magnitude (square-root of the sum of squares) of flow in both directions. We computed dense optical flow between each pair of successive video frames using the algorithms of [2].

The SSD relies on **data augmentation** to boost the performance, we extended the data augmentation [3] function provided by Singh *et al.*[1] in their GitHub repository to pass as input a sequence of frames in place of a single frame. The same data augmentation was applied to all the frames in a training batch.

VGG weight initialisation. The weights of VGG network (base network) are initialised with weights from a pre-trained ImageNet model [4][1] for appearance- and flow-based SSD networks for both UCF101-24 and DALY dataset.

2 RGB and Flow Contribution

In Figure 1, we show how each stream individually performs and contributes the performance of fused model using *mean-fusion* [5]. We can observe that the RGB stream works better than the flow stream and fusion always gives improvements over the RGB stream. Interestingly, the RGB stream is most dominant and performs comparable to the fusion model, hence one can only use RGB stream to improve the run-time speed of the whole system at the cost of a small performance drop.

3 Ablation Study: Performance on Classes with Movements

There are many action classes with a lot of spatial movement in the UCF101-24 dataset, unlike DALY dataset where only "CleaningFloor" had spatial movements. In Table 2, we show class-wise performance at detection threshold $\delta = 0.2$. We consider a class to have a considerable spatial movements if the class contributes to the off-diagonal elements in the transition matrix at $\Delta = 5$. The "BasketballDunk", "Skiing", "VolleyballSpiking" are the classes which contribute most to the off-diagonal entries in the transition matrix. It is clear from the Table 2 that the transition matrix helps to improve the detection performance considerably for those classes (highlighted in red) which have signifiacnt spatial movements. It is important to note that we build a transition matrix for each scale separately, our method is limited in case of when an actor moves towards or away from camera, or camera moves in that direction. One such case is biking observed from the front view of the biker. Solution to cover such movements is to build one transition matrix for all the pyramids. However, our base network is SSD, which has inconsistent feature dimension across the scales of the pyramid, hence, concatenated features from different scales would result in inconsistent dimensionality.

The classes with multiple actors (e.g. "IceDancing", "SalsaSpin", "Fencing") suffer in the TraMNet framework. We think it is due to the fact that the anchor micro-tubes

[1] https://gist.github.com/weiliu89/2ed6e13bfd5b57cf81d6

Table 1. Action localisation results on untrimmed videos from UCF101-24 split1. The table is divided into 4 parts. The first part lists approaches which have single frames as input; the second part approaches which take multiple frames as input; the third part contemplates the reimplemented versions of approaches in the second group; lastly, we report TraMNet's performance.

Methods	Train Δ	Test Δ	$\delta = 0.2$	$\delta = 0.5$	$\delta = 0.75$	$\delta = .5{:}.95$	Acc %
T-CNN [6]	NA	NA	47.1	–	–	–	–
MR-TS [7]	NA	NA	73.5	32.1	02.7	07.3	–
Saha *et al.* [8]	NA	NA	66.6	36.4	07.9	14.4	–
SSD [1]	NA	NA	73.2	46.3	15.0	20.4	–
AMTnet [9] rgb-only	1,2,3	1	63.0	33.1	00.5	10.7	–
ACT [5]	1	1	76.2	49.2	19.7	23.4	–
Gu *et al.* [10] ([7] + [11])	NA	NA	–	**59.9**	–	–	–
SSD-L with-trimming	NA	NA	76.2	45.5	16.4	20.6	92.0
SSD-L	NA	NA	76.8	48.2	17.0	21.7	92.1
ACT-L RGB	1	1	74.2	47.9	18.6	22.4	88.0
ACT-L Flow	1	1	72.8	43.9	12.3	18.6	86.2
ACT-L RGB+Flow	1	1	77.9	50.8	19.8	**23.9**	91.4
AMTnet-L RGB	1	1	76.2	46.9	18.0	21.9	89.1
AMTnet-L Flow	1	1	73.7	43.6	11.3	17.5	88.2
AMTnet-L RGB+Flow	1	1	**79.4**	**51.2**	19.0	23.4	**92.9**
TraMNet (ours) RGB	1	1	76.8	46.9	18.9	22.2	88.5
TraMNet (ours) Flow	1	1	74.9	43.1	11.2	17.6	87.1
TraMNet (ours) RGB+FLow	1	1	79.0	50.9	**20.1**	**23.9**	92.4

are class invariant and movement of some classes might be creating confusion in these classes. Adding class-specific anchor micro-tubes and respectively making regression layers class specific could help alleviate this problem. In some static classes (e.g. "Glof-Swing" or "TennisSwing") we observe that TraMNet is better than both ACT-L and AMTNet-L, where as in other static classes (e.g. "CricketBowling" and "RopeClimbing"), ACT-L is better. We think it because ACT uses features from 6 frames and able to classify the classes better.

Table 2. Spatio-temporal detection results (video APs in %) on UCF101-24 at $\delta = 0.2$. Classes with movement in the dataset are highlighted in red colour.

Actions	Basketball	BasketballDunk	Biking	CliffDiving	CricketBowling	Diving	Fencing	FloorGymnastics
ACT-L	50.3	78.0	78.3	79.2	70.5	100.0	89.9	98.9
AMTNet-L	53.6	79.4	82.2	81.7	65.7	100.0	90.7	99.5
TraMNet	51.9	81.8	80.0	81.6	60.6	100.0	87.0	98.8

Actions	GolfSwing	HorseRiding	IceDancing	LongJump	PoleVault	RopeClimbing	SalsaSpin	SkateBoarding
ACT-L	69.8	96.1	87.9	81.9	88.9	99.4	43.6	89.8
AMTNet-L	77.6	96.1	86.5	88.6	92.9	98.1	46.9	88.0
TraMNet	76.5	96.1	83.5	86.6	88.8	97.7	41.7	89.2

Actions	Skiing	Skijet	SoccerJuggling	Surfing	TennisSwing	TrampolineJumping	VolleyballSpiking	WalkingWithDog
ACT-L	81.1	87.8	95.0	67.0	35.0	67.6	44.0	90.4
AMTNet-L	82.4	93.5	94.9	68.9	40.5	65.7	42.0	90.4
TraMNet	85.2	93.6	94.4	65.6	42.8	66.8	53.5	91.2

References

1. Singh, G., Saha, S., Sapienza, M., Torr, P., Cuzzolin, F.: Online real-time multiple spatiotemporal action localisation and prediction. In: IEEE Int. Conf. on Computer Vision. (2017)
2. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. (2004)
3. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. arXiv preprint arXiv:1512.02325 (2015)
4. Liu, W., Rabinovich, A., Berg, A.C.: Parsenet: Looking wider to see better. arXiv preprint arXiv:1506.04579 (2015)
5. Kalogeiton, V., Weinzaepfel, P., Ferrari, V., Schmid, C.: Action tubelet detector for spatio-temporal action localization. In: IEEE Int. Conf. on Computer Vision. (2017)
6. Hou, R., Chen, C., Shah, M.: Tube convolutional neural network (t-cnn) for action detection in videos. In: IEEE Int. Conf. on Computer Vision. (2017)
7. Peng, X., Schmid, C.: Multi-region two-stream R-CNN for action detection. In: ECCV 2016 - European Conference on Computer Vision, Amsterdam, Netherlands (2016)
8. Saha, S., Singh, G., Sapienza, M., Torr, P.H.S., Cuzzolin, F.: Deep learning for detecting multiple space-time action tubes in videos. In: British Machine Vision Conference. (2016)
9. Saha, S., Singh, G., Cuzzolin, F.: Amtnet: Action-micro-tube regression by end-to-end trainable deep architecture. In: IEEE Int. Conf. on Computer Vision. (2017)
10. Gu, C., Sun, C., Vijayanarasimhan, S., Pantofaru, C., Ross, D.A., Toderici, G., Li, Y., Ricco, S., Sukthankar, R., Schmid, C., et al.: Ava: A video dataset of spatio-temporally localized atomic visual actions. arXiv preprint arXiv:1705.08421 (2017)
11. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2017) 4724–4733