Dynamic Graph Modules for Modeling Object-Object Interactions in Activity Recognition: Supplementary Material

Hao Huang[1]
hhuang40@ur.rochester.edu
Luowei Zhou[2]
luozhou@umich.edu
Wei Zhang[1]
wzhang45@ur.rochester.edu
Jason J. Corso[2]
jjcorso@umich.edu
Chenliang Xu[1]
chenliang.xu@rochester.edu

[1] University of Rochester
Rochester, New York, USA
[2] University of Michigan,
Ann Arbor, Michigan, USA

1 Coordinates updating

At time step $t > 1$, suppose the top-left and bottom-right coordinates of the m-th node in hidden graph are $(m^{t-1}_{x,1}, m^{t-1}_{y,1}, m^{t-1}_{x,2}, m^{t-1}_{y,2})$, and the coordinates of the n-th proposal in the t-th feature map are $(n^t_{x,1}, n^t_{y,1}, n^t_{x,2}, n^t_{y,2})$. The normalized weight (IoU) between the m-th node in the hidden graph and the n-th proposal in the t-th feature map is $\boldsymbol{F}'_{\boldsymbol{l}}(\boldsymbol{b}^t_n, \boldsymbol{x}_m)$. The coordinate of $m^t_{x,1}$ is computed as:

$$\begin{cases} m^t_{x,1} = \frac{1}{2}(m^{t-1}_{x,1} + \sum_{n=1}^{N} \boldsymbol{F}'_{\boldsymbol{l}}(\boldsymbol{b}^t_n, \boldsymbol{x}_m) n^t_{x,1}) \ , \\ m^t_{y,1} = \frac{1}{2}(m^{t-1}_{y,1} + \sum_{n=1}^{N} \boldsymbol{F}'_{\boldsymbol{l}}(\boldsymbol{b}^t_n, \boldsymbol{x}_m) n^t_{y,1}) \ , \\ m^t_{x,2} = \frac{1}{2}(m^{t-1}_{x,2} + \sum_{n=1}^{N} \boldsymbol{F}'_{\boldsymbol{l}}(\boldsymbol{b}^t_n, \boldsymbol{x}_m) n^t_{x,2}) \ , \\ m^t_{y,2} = \frac{1}{2}(m^{t-1}_{y,2} + \sum_{n=1}^{N} \boldsymbol{F}'_{\boldsymbol{l}}(\boldsymbol{b}^t_i, \boldsymbol{x}_m) n^t_{y,2}) \ . \end{cases} \tag{1}$$

2 The Structure of Fusion Layers

The average-pooled feature produced by the 3D ConvNet is denoted as $\boldsymbol{f} \in \mathbb{R}^{C\times 1}$ where $C = 2048$. The graph module feature $\boldsymbol{q}_t \in \mathbb{R}^{C'\times 1}$ where $C' = 1024$. We fuse both graph module feature and 3D ConvNet feature to recognize actions. The fusion layers are illustrated in Fig. 1. We keep the size of the fused feature $\boldsymbol{z}_t$ to $C' \times 1$ and forward this feature into a multi-layer perceptron to get the final recognition results.

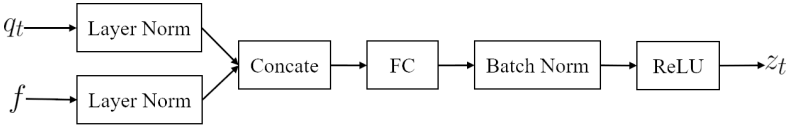

Figure 1: Fusion layers to fuse the graph module feature and 3D ConvNet feature at time step t.

3 Implementation Details

We first train our backbone 3D model [1, 5] on Kinetics dataset and then fine-tune it on the target datasets. For Something-Something dataset, we randomly sample 32 frames from each video. For ActivityNet dataset, as the video length is much longer, we first segment each activity instance into several clips (around 5 seconds) with the overlap rate fixed to 20%. The sampled frames are used to train our backbone 3D model. Following [4], sampled frames are randomly scaled with shorter side resized to a random integer number in [256, 320]. Then we randomly crop out an area of 224×224 and randomly flip frames horizontally before forwarding them to the backbone model. The Dropout [2] before the classification layer in backbone model is set to 0.5. We train our backbone model with a batch size of 24. We set the initial learning rate to 0.00125. We apply stochastic gradient descent (SGD) optimizer and set momentum to 0.9 and weight decay to 0.0001. We adopt cross-entropy loss during our training. We adopt cross-entropy loss during our training.

Next, we describe how we train our streaming dynamic graph module. For each input frame, we propose RoI proposals using RPN [3] with ResNet-50 pre-trained on Microsoft COCO. For Something-Something dataset, we keep the top 20 proposals each frame and set the number of nodes in hidden graph to be 5. For ActivityNet dataset, as video scenes are more complex and contain more objects, we keep the top 40 proposals and increase the number of graph nodes to 10. We fix the backbone 3D ConvNet and only train our graph module, fusion layers and classification layer. We adopt the same learning strategy as the fine-tuning of the backbone.

For the static model, we first train the streaming model following the strategy above for 3 epochs as a warm-up. Then we concatenate the graph module feature with the backbone feature using the fusion layers described in Sec. 2. At the same time, we reduce the learning rate by a factor of 10. The parameters of the backbone remain fixed during training.

References

[1] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 4724–4733. IEEE, 2017.

[2] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[3] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[4] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417, 2018.

[5] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.