Activity Recognition

Final project for Vision and Perception by Prof. Fiora Pirri

Son Tung Nguyen - Wei Xu - Zijian Xue

September 28, 2018

1 Introduction

Activity recognition has been a fundamental problem in computer vision [1]. Huge progress has been achieved with the success of convolutional neural network (CNN) on image recognition or object detection tasks. In this work, we study different solutions to recognize and organize human activities into multiple categories. To be precise, our goal is to classify accurately a given video into one of ten categories of housework activities.

2 Related works

Human activity recognition is a well studied problem. Some of the past works on this topic include training support vector machine classifiers on bag-of-words features [2, 3]. A more recent solution is to train a CNN to capture spatial information and to use a long short term memory (LSTM) network to exploit temporal features of videos. [1] was one of the first examples on this idea. Their works showed that the proposed system performed very well on three different problems: activity recognition, image captioning and video description.

3 Proposed method

3.1 Model architecture

Our model architecture follows the work of [1]. Figure 1 illustrates the standard architecture of our system. For each frame, we have a CNN model to output spatial features. This CNN model was pretrained on Imagenet and kept untrainable during our training. The output of the CNN layer is forwarded to the LSTM layer to capture temporal relationship of frames. After this, we calculate the softmax probabilities of each frame and then output the average as our final prediction.

3.2 Improvements

We carried on multiple experiments on different ideas to enhance the ability of the system.

VGG-16 or Inception pretrained weights We had two options for the pretrained CNN layer: VGG-16 [4] or Inception V3 [5]. So, we decided to try both. The result was as expected when using Inception V3 weights resulted in better performance (Table 1).

Longer sequence Different video sequence was also tried. Experiments showed that using longer sequence made the system better (Table 1).

More LSTM layers Stacking more LSTM layers typically enhanced the performance of the system. We tried to stack our system with 1, 2 and 3 layers and found that stacking resulted in higher accuracy.

Linear interpolating We tried to linear interpolate the loss function as suggested by [6]. To be more exact, we weighted our loss function so that the model focuses more on later time steps in the sequence. This is an alternative to just sum up the softmax loss of every time steps.

Exploit spatial information from object detection model We attempted to inject spatial information from a Mask-RCNN [7] model on every frame into our system. That is, for every frame, we ran inference by Mask-RCNN model and returned three vectors: *mask*, *bbox* and *prob*. Each vector is T-length depicted a pre-defined set of T objects. The *mask* and *bbox* vectors described the mask and bounding box areas of T objects. The *prob* vector described the probabilities of detecting these T objects. Figure 2 illustrates an example of this architecture.

4 Experiments

Datasets We used a subset of ActivityNet [8] as our data source for training and testing. For each video, we extracted video segment that contains interested activities. Each video segment is at most 10 second long. In the end, we had 856 video segments and split them into train set of 680 segments and test set of 176 segments.

Training settings To optimize our network, we used Adam optimization [9] with learning rate of 0.1. Each experiment was optimized for 100 epochs. Figure 3 shows the training profile of our network and suggests that the model suffered from over-fitting.

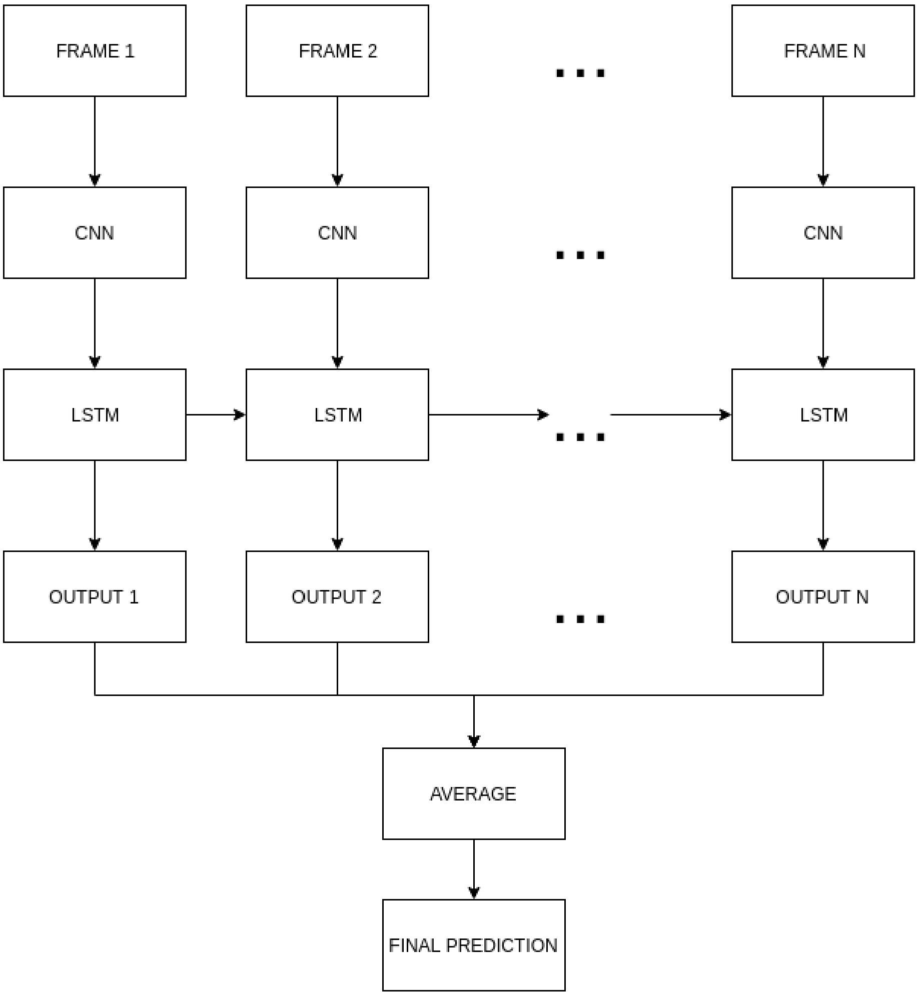

Figure 1: Standard model architecture.

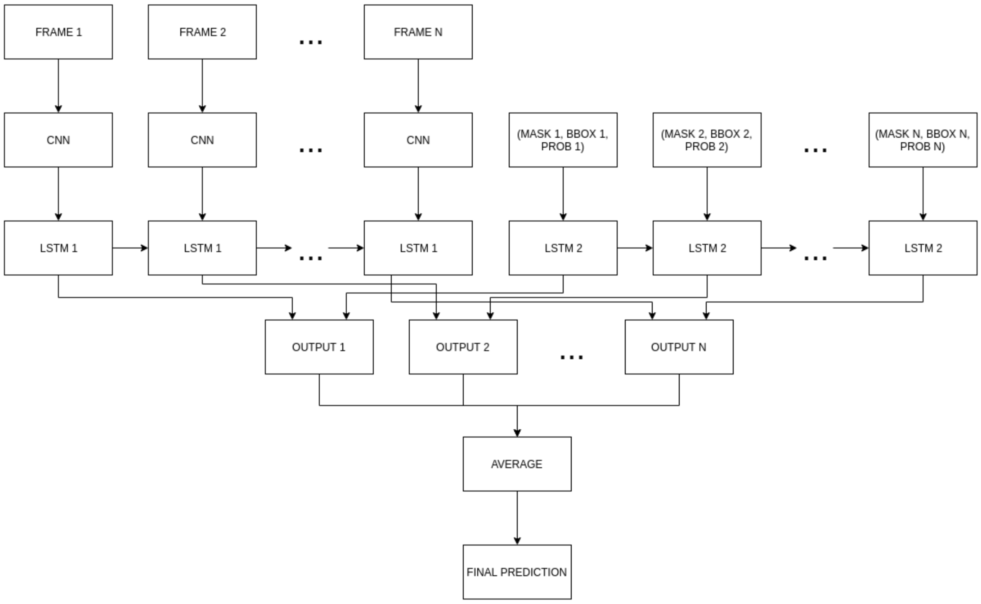

Figure 2: Model architecture that exploits spatial information from object detector.

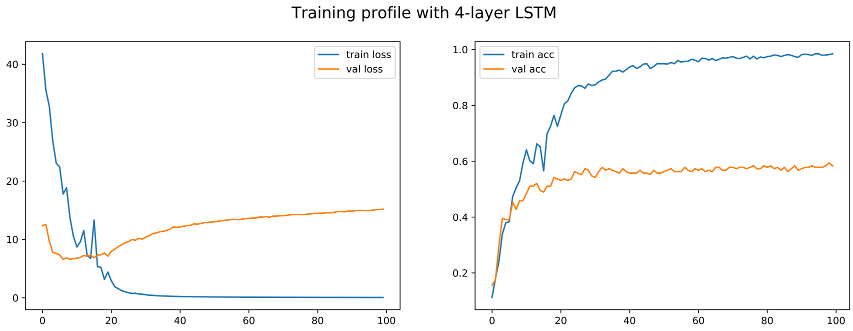

Figure 3: Training profile with 4 LSTM layers.

Table 1: Pretrained weights and longer sequence.

Pretrained weights	**25**	**50**	**100**
VGG-16	0.447	0.494	0.484
Inception V3	0.546	0.557	0.593

Table 2: Stacking different numbers of LSTM layers.

	1	**2**	**3**	**4**	**5**
Accuracy	0.536	0.531	0.557	0.562	0.552

Table 3: Using additional spatial information.

Using prob	**Using mask**	**Using bbox**	**Accuracy**
1	0	0	0.557
0	0	1	0.567
0	1	0	0.562
1	1	1	0.531

References

[1] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 677–691, April 2017.

[2] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2008.

[3] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *2013 IEEE International Conference on Computer Vision*, pp. 3551–3558, Dec 2013.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2818–2826, 2016.

[6] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4694–4702, June 2015.

[7] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017.

[8] B. G. Fabian Caba Heilbron, Victor Escorcia and J. C. Niebles, "Activitynet: A large-scale video benchmark for human activity understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 961–970, 2015.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.