Two-Stream convolutional nets for action recognition in untrimmed video

Kenneth Jung
Stanford University
kjung@stanford.edu

Song Han
Stanford University
songhan@stanford.edu

Abstract

We extend the two-stream convolutional net architecture developed by Simonyan for action recognition in untrimmed video clips. The main challenges of this project are first replicating the results of Simonyan et al, and then extending the pipeline to apply it to much longer video clips in which no actions of interest are taking place most of the time. We explore aspects of the performance of the two-stream model on UCF101 to elucidate the current barriers to better performance in both UCF101 and untrimmed video. We also explore whether or not training on the background videos from the Thumos Challenge dataset (in which no actions of interest occur in the video) improves action recognition.

1. Introduction

The two-stream architecture presented in Simonyan et al [8] for action classification in video achieved state of the art performance on the UCF-101 dataset despite being quite conceptually simple. Like a radically simplified primate visual cortex, it splits the task of visual processing into two parallel streams - *spatial* encoding (analogous to the ventral, or *what* pathway), and *motion* encoding (analogous to the *where*, or dorsal pathway). Much of its success stemmed from the decision to bypass the difficult problem of directly training a convolutional net to extract motion information from video, as was attempted done by Karpathy et al [6]. Instead, they opted to use pre-computed dense optical flow fields, stacked over consecutive frames, as inputs to the temporal network.

However, the model was only evaluated on the UCF-101 dataset, which consists of short video clips in which the entire video is classified as belonging to one of 101 action classes [9], and each video is trimmed such that the action is occurring for the entire extent of the video. It is unclear whether and how it will scale to the more general problem of recognizing actions in unconstrained video in which actions may begin and end at any time, and will likely occupy only a very brief fraction of the total duration of the video (see Figure 1 for a comparison of the number of frames per video in each dataset). We note that this problem is distinct from recent work of Gkioxari et al [3], which focuses on action localization *in space*. Here, we apply the two-stream architecture on long video clips in which, for most of the time, no action of interest is taking place, and evaluate the model's ability to recognize actions. We take advantage of the Thumos Challenge 2014 dataset [5], which contains much longer video clips annotated with the UCF101 action classes. We first replicate the key results from Simonyan et al. Then we apply the trained two-stream model to untrimmed videos from the Thumos validation dataset.

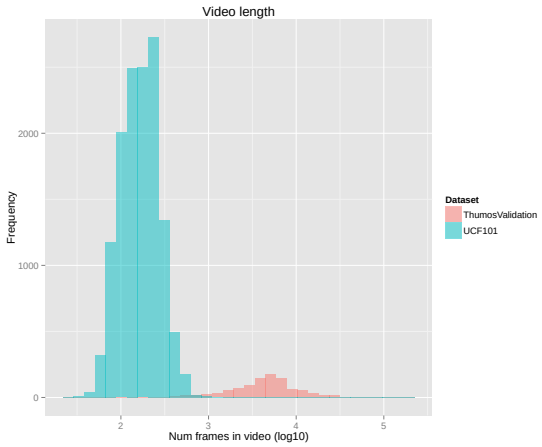

Figure 1. Trimmed versus untrimmed video: Number of frames in the UCF101 versus Thumos Validation Set video collections.

2. Methods

2.1. Replication of key findings from Simonyan et al

We first set out to replicate key findings from Simonyan et al using Caffe [4]. Specifically, we aimed to fit spatial and temporal models that achieved similar performance on UCF101 as reported previously. All base models were trained and tested using Caffe; we have included the prototxt files used for all parts of this work for reference. We made extensive use of custom Caffe data layers written in

C++ for this project; all source code not in the standard Caffe distribution in included in a zip file with this submission.

2.1.1 Pre-computing optical flow fields

As described previously, we used the OpenCV implementation of Brox optical flow field estimation to calculate x and y flow fields for each consecutive pair of frames for all videos used in this study. x and y flow fields for each frame pair were stored in individual files as compressed jpeg files after subtracting the mean flow per flow field, and scaling to the range 0-255; a jpeg quality parameter of 25 was used for all files. The throughput of this step was roughly 0.06 seconds per frame pair for the UCF101 video files. Throughput on the Thumos Background and Validation videos was reduced because those videos had to be resized to 320x240 from their original size of 240x180.

2.1.2 Training the temporal nets

Due to differences in our computational resources relative to Simonyan et al, our replication effort differed in some respects from the training procedures and model architectures described therein. Most of the computational heavy lifting was performed on an Amazon EC2 g2.2xlarge instance with an nVidia K520 GPU with 4GB memory. This is a rather less GPU memory than was available to Simonyan et al, and using a temporal window size of $L = 10$ consecutive optical flow fields per input, along with their net architecture, would have required us to use a rather small batch size of 16. Based on advice from Serena Yeung, we therefore opted to use an Alexnet architecture, along with $L = 5$. This allowed us to use a batch size of 128 for training the temporal net. We implemented a custom Caffe data layer in C++ to construct stacks of flow fields as described in Simonyan. Specifically, the data layer randomly picked a frame, f, from each video in the mini-batch and then stacked the x and y flow fields for frame pairs $(f, f+1), \ldots, (f+5, f+6)$ into a single input volume with $2L = 10$ channels. This data layer was also responsible for rescaling the flow fields to their original scales. A random 227x227 crop of the original frame was performed, along with random horizontal flipping; these operations were of course applied to the entire flow stack. Similar to the training schedule used by Simonyan, we trained this model for 120,000 iterations starting with an initial learning rate of 0.01 decreased by a factor of 10 every 40,000 iterations. As reported previously, we used a dropout rate of 0.9 in the fully connected layers to control overfitting. Training was performed on the training data from the first UCF101 train-test split.

We were also interested in whether or not including Thumos background videos as an additional class, i.e., videos in which it is guaranteed that none of the UCF101 actions are taking place, would help performance in untrimmed video. We therefore fine-tuned the last fully connected layer of the temporal net using the trained temporal net on UCF101 videos plus a random sample of flow stacks from the first 300 background videos. For this finetuning, we trained with a batch size of 128 for 45,000 iterations, with an initial learning rate of 0.001. The learning rate was decreased by a factor of 0.1 every 15,000 iterations. We used a dropout of 0.9 during fine-tuning.

2.1.3 Training the spatial nets

The spatial net model is similar to that reported in Simonyan et al. Because they reported results for simply fine-tuning the final layer of a pretrained net on the UCF101 data that were realistically within the margin of error for a fully trained net, we opted to fine-tune a net pre-trained on ImageNet to this new task. For this net, we started with the VGG_CNN_M_2048 model from the Caffe model zoo and fine-tuned the last layer for 80,000 iterations using an initial learning rate of 0.001, reduced by a factor of 10 every 20,000 iterations, and a batch size of 64. As for the temporal net, we had to implement a custom Caffe data layer that would sample a random frame from the each video file in each mini-batch; these frames were also randomly cropped to 224x224 and randomly horizontally flipped. Finally, similar to the temporal nets, we fine-tuned our spatial net using background videos for 45,000 iterations with an initial learning rate of 0.001, reduced by a factor of 10 every 15,000 iterations. As for the temporal net, training was performed on the training data form the first UCF101 train-test split.

2.1.4 Evaluating the temporal and spatial nets on UCF101 Test videos

In order to evaluate our spatial and temporal nets on UCF101 videos, we had to implement yet more custom Caffe data layers to perform the test time procedure described by Simonyan et al. Specifically, the temporal net test time code chose 25 frames from the entire extent of each test video. From each of these frames, 10 optical flow field stacks were constructed using the four corners plus centers of each stack, plus their horizontal flips. Thus, the temporal net was run on 250 inputs for each input video clip. An analogous process was run for the spatial net, again using a custom data layer. We used a modified version of the `extract_features.cpp` program included with Caffe to extract the softmax probabilities from running the spatial and temporal nets on the UCF101 training and test videos. The softmax probabilities were average-pooled over all 250 of the crops/flips for each video clip, yielding a single vector of class probabilities for each video clip in UCF101. Note

that these last steps were performed in `R` - we apologize for using a computing environment the TAs are unlikely to be familiar with, but it was simply the most familiar environment for us! R scripts are included in the submission.

2.2. Applying the two-stream model to unclipped video

Applying the two-stream model to untrimmed video is uncharted territory, and it was not clear how to extend the test time procedure used by Simonyan et al to untrimmed video. Clearly, we don't want to simply take 25 equally spaced frames from a 10,000 frame video! We opted for a simple sampling strategy - we would simply use every tenth frame from the Thumos validation set video files. Thumos validation videos are all 30 frames per second, so this corresponds to sampling the videos every third of a second. Given this sampling density, there are still two issues we wanted to evaluate. First, the sparsity of actions of interest in untrimmed video suggested that we might benefit from explicitly learning a null, or `BACKGROUND` class. We addressed this question by testing spatial and temporal models fine-tuned with Thumos background video data in addition to the UCF101 files. Second, we wished to know whether or not pooling input over a larger time interval, t_{pool}, would help performance. This was motivated by the observation that the performance achieved in UCF101 was heavily dependent on pooling the net outputs over all 250 inputs from each video clip. We addressed this question by varying the pooling parameter t_{pool} from 1 second to 2 and 4 seconds. Finally, as minor point, we evaluated the use of different methods for combining the spatial and temporal model outputs - specifically, we evaluated two non-linear models - neural nets with ReLU activations and dropout and gradient boosted trees - in addition to model averaging. As before, custom data layers and modified versions of `extract_features.cpp` were used in these experiments, along with the `gbm` package in `R` for fitting the gradient boosted model and pylearn2 for fitting the neural net models.

Finally, we note that due to time constraints, we were not able to run our models on the full Thumos validation set of 1010 videos. Instead, we used only the first 339 videos, comprised of 173,380 frames of video. The primary bottleneck was pre-computing the optical flow for these files - running the trained models took relatively little time in comparison. One consequence of this is that many action classes are not represented at all in the validation set that was evaluated.

Model	Train acc per frame	Test acc per frame	Train acc per video	Test acc per video
Simonyan spatial	NA	NA	NA	72.7
Simonyan temporal	NA	NA	NA	81.0
Simonyan mean	NA	NA	NA	85.9
Spatial	83.8	54.4	98.9	68.8
Temporal	87.5	54.9	98.2	75.2
Mean	95.8	67.5	99.7	82.2

Table 1. Training and test set video level accuracy on UCF101 Test set 1 per frame and per video.

3. Results

3.1. Two-stream performance in UCF101

Our first goal was to approximate the two-stream model results on UCF101. The performance of our spatial and temporal models, along with model averaging on the UCF101 test set, is shown in table 1. First, we note that our performance is within 2-5% of the state of the art results previously reported by Simonyan. Once we achieved this level of performance, we did not make more effort to improve performance as we deemed this sufficient to provide insight into the performance of the two-stream architecture on untrimmed video. Second, note that we provide accuracies at the per-frame level in addition to the per-video accuracies reported by Simonyan. We also include training set accuracies. This is to point out two observations about the two-stream architecture applied to UCF101. First, even with very aggressive dropout and reduced model capacity, our temporal stream is still showing substantial overfitting, especially at the per-video level. This suggests that substantial performance gains are to be had from simply getting more data, more data augmentation, and more constrained/better regularized models. Second, the spatial and temporal nets have roughly equal accuracy at the per-frame level, but the test time procedure benefits the temporal net substantially more than the spatial net (75.2% from 54.9% versus 68.8% from 54.5%). This suggests that some difference in the distribution of activations of the temporal versus spatial nets over the time course of these trimmed videos that might be exploited in future work.

We also observed that the spatial and temporal nets are good at recognizing quite different action classes (Figure 2). Table 2 shows the top ten classes sorted by degree of difference in the test set accuracy of the temporal versus spatial nets at the per-video level. Note that among these classes, the differences in accuracies are all in favor of the temporal net, with some of the differences being greater than 50%!

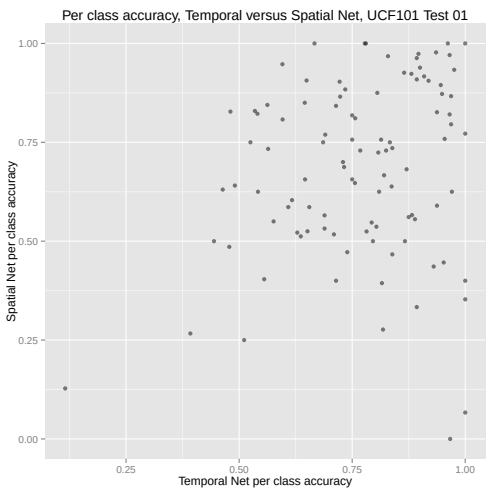

Figure 2. Per-class accuracies in UCF101 Test for spatial and temporal nets (R = 0.206)

We hypothesize that this reflects the fact that the spatial net must recognize actions by essentially recognizing objects in the frames, while the temporal net recognizes actions based on motion. These results suggest that the classes in Table 2 do not have distinguishing objects for the spatial net to learn to recognize, at least at the scale of the UCF101 dataset. Alternatively, it may be that the distinguishing image components for these actions classes are simply not represented well by the pre-trained net. This problem could then perhaps be alleviated by a larger training set such that the spatial net can be trained successfully from scratch. We also examined the confusion matrix for the spatial and temporal nets on the UCF101 test set. Heatmaps of the confusion matrices are provided as supplementary figures. We note that the temporal net is confused by class pairs such as "Shaving beard" and "Brushing teeth". The temporal net is also confused by the motion fields for "Basketball", "Band Marching", and "Ice Dancing". The spatial net, in contrast, appears to show confusion about classes such as "Brushing Teeth", "Apply Lipstick", and "Apply Eye Makeup", i.e., sets of classes in which the frames contain a common main subject (i.e., a face) and the distinguishing objects are often small and without much in the way of texture. This is consistent with the idea that the temporal stream is learning to recognize distinctive motions as opposed to simply responding to objects blurred by motion in the flow field. We must note, however, that the small size of the test set (¡ 3300 videos and 101 classes) suggests that these results are subject to high variance.

3.2. Performance in untrimmed video

We evaluated the performance of the two-stream model, plus non-linear models trained on per-frame softmax probabilities from the two-stream model, on the first 339 videos

Class	Temporal acc - Spatial acc
Surfing	96.7
Jumping Jack	93.3
Hammer Throw	64.7
Skiing	60.0
Sky Diving	55.9
Archery	54.2
Pole Vault	50.6
Diving	49.4
Drumming	37.2

Table 2. Per class accuracy in UCF101 Test set 1 - differences in performance between the spatial and temporal nets.

of the Thumos Validation set. In the following discussion, we evaluated the performance of the models as follows. First, softmax probabilities were extracted for every tenth frame in the videos, or every 1/3 of a second. Second, these probabilities were average pooled over non-overlapping time intervals of length t_{pool} of either 1 second, 2 seconds, or 4 seconds. These probabilities were then evaluated for accuracy using the AUC at different thresholds of the class probabilities (using a single threshold for all action classes), and for precision and recall. Note that Thumos task 1 essentially asks, whether or not each of the UCF101 actions occur at any time during the video. Thus, our predictions over this dataset was distilled down to a matrix whose rows are class probabilities for each video. The AUCs of the various models is shown in Table 3. In this table, GBM refers to a gradient boosted tree model trained on per-frame softmax probabilities from both the temporal and spatial nets. The GBM model was fit using initially for 1000 iterations using a shrinkage parameter of 0.0025 on 90% of the training data, and an interaction depth of 6. After determining the optical number of iterations using the multinomial loss on the held out training data the model was fit for that many iterations on the full training set. NN refers to a 3 layer neural network model using ReLU activations and droput = 0.5, with 100 hidden units per layer. Figure 3 shows the partial ROC plots and precision versus recall curves the the models using t_{pool} = 4 seconds.

We note the following trends in the results. First, training to recognize an explicit background class using data from background videos appears to help performance as measured by AUC in the spatial and temporal models. However, this benefit is diminished by simply using a larger t_{pool} and provides little benefit for the non-linear models. In fact, we observe the puzzling phenomenon where the NN model does better without having seen background examples, which runs counter to the trend in the other models including the GBM model. It is not clear why that is, though we note that the training procedure for these networks was not uniform. Basically, we used a held out por-

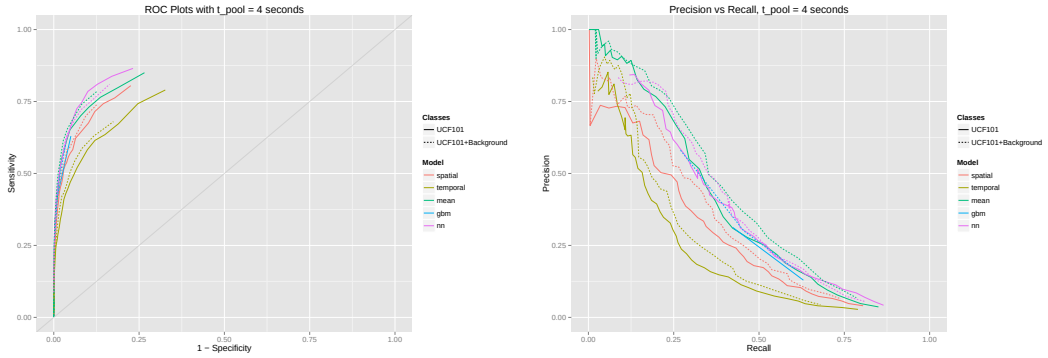

Figure 3. ROC and Precision vs Recall of NN on untrimmed video, t_{pool} = 4 seconds

tion of the training data as a validation set. The models were initially trained until the objective on the validation set had not improved for 3 epochs. The models were then further trained using *all* of the training data until the objective on the validation set was equal to the objective on the training set during the first run, or until our patience and/or time ran out. Unfortunately, this happened rather sooner for the NN model with background examples, so it may be that this model was simply not trained adequately. Second, all things being equal, it appears to help to pool over larger t_{pool}. However, this effect is relatively weak, though quite consistent across the models. Third, the benefit from non-linear models over simple model averaging of the spatial and temporal nets is marginal at best. Fourth, it is odd that the temporal net substantially underperforms the spatial net in this dataset given the edge it has in UCF101. We hypothesize that this is due to the specific action classes that appear in the first 339 videos of the validation set. Clearly, we should have at least randomly chosen videos from the validation set instead of processing them in order! Finally, we note that the AUC is not a very good measure of accuracy in a task such as this because the ground truth matrix is very sparse - there are only 400 actions out of 34,239 possible action annotations in 339 videos and 101 action classes. Specifically, we note that a recall of 50%, the precision gap between the best and worst models is about 15%.

4. Discussion

We have approximately replicated the results of Simonyan's two-stream architecture on UCF101 and applied the resulting models to untrimmed video from the Thumos Validation dataset. We found that training explicitly to recognize background, or "no action of interest" videos, helps performance. We also found that increasing the time frame over which we pool class probabilities has little impact on performance, and that in this dataset, model averaging will likely perform as well as more complex non-linear models.

Model	$t_{pool} = 1$	$t_{pool} = 2$	$t_{pool} = 4$
Spatial	0.874	0.877	0.878
Temporal	0.815	0.817	0.819
Mean	0.884	0.887	0.887
GBM	0.885	0.891	0.893
NN	**0.901**	**0.902**	**0.902**
Spatial + Background	0.882	0.884	0.885
Temporal + Background	0.830	0.832	0.835
Mean + Bacground	0.899	**0.902**	**0.902**
GBM + Background	0.890	0.894	0.894
NN + Background	0.892	0.893	0.894

Table 3. Thumos Task 1. Action recognition in untrimmed video. Due to time constraints, we were only able to run the system on 339 out of 1010 video clips from the validation set.

We hypothesize that this simpler model ensemble is more resistant to differences in between the UCF101 and Thumos videos. Our project met its primary endpoints of replicating Simonyan's work and learning about the issues that pertain to action recognition in untrimmed video. However, it is also quite clear that this project has significant limitations that would be fun to address in future work. We note that this approach completely ignores the sequential nature of the input aside from pooling information in small temporal windows. Additionally, the covariance structure of the action classes is completely ignored. However, we view these limitations as minor due to the limited size of the dataset. Rather we believe that the first orders of business if one were to make a serious attempt to scale the two-stream approach to untrimmed video would be get a lot more data and to deal with the overfitting problem.

More broadly, the approach of pre-processing frames in order to get optical flow fields is computationally intensive and very much against the spirit of most recent work using deep convolutional nets for computer vision in that we cannot back-propagate errors through the motion estima-

tion procedure. At least three plausible avenues of attack are apparent for dealing with this rather distasteful inelegance. First, recent work on using convolutional nets for image segmentation and depth map prediction [7, 2] suggests that it may be possible to train a convolutional net using flow fields as targets directly. This network could then be used as a pretrained network for using motion for action recognition, with the fine-tuning of the entire stack. However, this approach strikes us as practical but still somewhat unsatisfying. It presupposes that dense motion fields are a good representation for action recognition. It is clear that visual systems as embodied by, say, our visual cortex, does not maintain a dense optical flow map as such. An alternative approach is to learn a representation of video frames that is useful for action recognition directly from consecutive video frames, as was originally envisioned in Karpathy et al [6], and embodied in one form by the C3D features desribed in [11]. However, our opinion is that the really interesting solutions to this problem will use recurrent architectures to explicitly model sequences of video frames at a high level [1, 10]. While our opinion is rather uninformed, we note that is a lot better informed than it was when we started this class! :-)

References

[1] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, abs/1411.4389, 2014.

[2] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. *CoRR*, abs/1406.2283, 2014.

[3] G. Gkioxari and J. Malik. Finding action tubes. *CoRR*, abs/1411.6031, 2014.

[4] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093, 2014.

[5] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`, 2014.

[6] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 1725–1732, 2014.

[7] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014.

[8] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 568–576, 2014.

[9] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.

[10] N. Srivastava, E. Mansimov, and R. Salakhtudinov. Unsupervised learning of video representation using lstms.

[11] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. C3D: generic features for video analysis. *CoRR*, abs/1412.0767, 2014.