Encouraging LSTMs to Anticipate Actions Very Early
Supplementary Material

Mohammad Sadegh Aliakbarian[1,3], Fatemeh Sadat Saleh[1,3], Mathieu Salzmann[2], Basura Fernando[1], Lars Petersson[1,3], Lars Andersson[3]
[1]Australian National University, [2]CVLab, EPFL, Switzerland, [3]Smart Vision Systems, CSIRO
firstname.lastname@data61.csiro.au, mathieu.salzmann@epfl.ch, basura.fernando@anu.edu.au

In this supplementary material, we analyze different aspects of our approach via several additional experiments. While the main paper discusses action anticipation, here, we focus on evaluating our approach on the task of action recognition. Therefore, we first provide a comparison to the state-of-the-art action recognition methods on three standard benchmarks, and evaluate the effect of exploiting additional optical flow features for both action recognition and anticipation. We then analyze the effect of our different feature types in several loss functions, the influence of the number of hidden units and of our average pooling in LSTMs, and, finally, the effect of our multi-stage LSTM architecture.

1. Comparison to State-of-the-Art Action Recognition Methods

We first compare the results of our approach to state-of-the-art methods on UCF-101, JHMDB-21 and UT-Interaction in terms of average accuracy over the standard training and testing partitions. In Table 1, we provide the results on the UCF-101 dataset. Here, for the comparison to be fair, we only report the results of the baselines that do not use any other information than the RGB image and the activity label (we refer the readers to the baselines' papers and the survey [6] for more detail). In other words, while it has been shown that additional, handcrafted features, such as dense trajectories and optical flow, can help improve accuracy [19, 20, 9, 13, 1], our goal here is to truly evaluate the benefits of our method, not of these features. Note, however, that, as discussed in the next section of this supplementary material, our approach can still benefit from such features. As can be seen from the table, our approach outperforms all these RGB-based baselines. In Tables 2 and 3, we provide the results for JHMDB-21 and UT-Interaction. Again, we outperform all the baselines, even though, in this case, some of them rely on additional information such as optical flow [5, 21, 14, 15, 10] or IDT Fisher vector features [14]. We believe that these experiments show the effectiveness of our approach at tackling the action recognition problem.

Table 1. Comparison with state-of-the-art methods on UCF-101 (average accuracy over all training/testing splits). For the comparison to be fair, we focus on the baselines that, as us, only use the RGB frames as input.

Method	Accuracy
Dynamic Image Network [1]	70.0%
Dynamic Image Network + Static RGB [1]	76.9%
Rank Pooling [4]	72.2%
DHR [4]	78.8%
Zhang et al. [26]	74.4%
LSTM [16]	74.5%
LRCN [2]	68.8%
C3D [17]	82.3%
Spatial Stream Net [13]	73.0%
Deep Network [7]	65.4%
ConvPool (Single frame) [25]	73.3%
ConvPool (30 frames) [25]	80.8%
ConvPool (120 frames) [25]	82.6%
Ours	**83.3%**
Diff. to State-of-the-Art	+0.7%

2. Exploiting Optical Flow

Note that our approach can also be extended into a two-stream architecture to benefit from optical flow information, as state-of-the-art action recognition methods do. In particular, to extract optical flow features, we made use of the pre-trained temporal network of [13]. We then computed the CNN features from a stack of 20 optical flow frames (10 frames in the x-direction and 10 frames in the y-direction), from $t - 10$ to t at each time t. As these features are potentially loosely related to the action (by focusing on motion), we merge them with the input to the second stage of our multi-stage LSTM. In Table 4, we compare the results of our modified approach with state-of-the-art methods that also exploit optical flow. Note that our two-stream approach

Table 2. Comparison with state-of-the-art methods on JHMDB-21 (average accuracy over all training/testing splits). Note that while the methods of [5, 21, 14, 15] use motion/optical flow information and [14] uses IDT Fisher vector features, our method yields better performance.

Method	Accuracy
Where and What [15]	43.8%
DP-SVM [14]	44.2%
S-SVM [14]	47.3%
Spatial-CNN [5]	37.9%
Motion-CNN [5]	45.7%
Full Method [5]	53.3%
Actionness-Spatial [21]	42.6%
Actionness-Temporal [21]	54.8%
Actionness-Full Method [21]	56.4%
Ours	**58.3%**
Diff. to State-of-the-Art	+1.9%

Table 3. Comparison with state-of-the-art methods on UT-Interaction (average accuracy over all training/testing splits). Note that while the methods of [14] uses motion/optical flow information and IDT Fisher vector features, our method yields better performance.

Method	Accuracy
D-BoW [12]	85.0%
I-BoW [12]	81.7%
Cuboid SVM [11]	85.0%
BP-SVM [8]	83.3%
Cuboid/Bayesian [12]	71.7%
DP-SVM [14]	14.6%
Yu et al. [23]	83.3%
Yuan et al. [24]	78.2%
Waltisberg et al. [18]	88.0%
Ours	**90.0%**
Diff. to State-of-the-Art	+2.0%

yields accuracy comparable to the state-of-the-art.

We also conducted an experiment to evaluate the effectiveness of incorporating optical flow in our framework for action anticipation. To handle the case where less than 10 frames are used, we padded the frame stack with gray images (with values 127.5). Our flow-based approach achieved 86.8% for earliest and 91.8% for latest prediction on UCF-101, thus showing that, if runtime is not a concern, optical flow can indeed help increase the accuracy of our approach.

We further compare our approach with the two-stream network [13], designed for action recognition, applied to the task of action anticipation. On UCF-101, this model achieved 83.2% for earliest and 88.6% for latest prediction, which our approach with optical flow clearly outperforms.

Table 4. Comparison with the state-of-the-art approaches that use optical flow. For the comparison to be fair, we focus on the baselines that, as us, use RGB frames+optical flow as input.

Method	Accuracy
Spatio-temporal ConvNet [7]	65.4%
LRCN + Optical Flow [2]	82.9%
LSTM + Optical Flow [16]	84.3%
Two-Stream Fusion [3]	92.5%
CNN features + Optical Flow [13]	73.9%
ConvPool (30 frames) + OpticalFlow [25]	87.6%
ConvPool (120 frames) + OpticalFlow [25]	88.2%
VLAD3 + Optical Flow [9]	84.1%
Two-Stream ConvNet [13]	88.0%
Two-Stream Conv.Pooling [25]	88.2%
Two-Stream TSN [22]	91.5%
Ours + Optical Flow	91.8%

Table 5. Importance of the different feature types using different losses. Note that combining both types of features consistently outperforms using a single one. Note also that, for a given model, our new loss yields higher accuracies than the other ones.

Feature	Sequence Learning	Accuracy
Context-Aware	LSTM (CE)	72.38%
Action-Aware	LSTM (CE)	74.24%
Context+Action	MS-LSTM (CE)	78.93%
Context-Aware	LSTM (ECE)	72.41%
Action-Aware	LSTM (ECE)	77.20%
Context+Action	MS-LSTM (ECE)	80.38%
Context-Aware	LSTM (LGL)	72.58%
Action-Aware	LSTM (LGL)	77.63%
Context+Action	MS-LSTM (LGL)	81.27%
Context-Aware	LSTM (Ours)	72.71%
Action-Aware	LSTM (Ours)	77.86%
Context+Action	MS-LSTM (Ours)	83.37%

3. Effect of Different Feature Types

Here, we evaluate the importance of the different feature types, context-aware and action-aware, on recognition accuracy. To this end, we compare models trained using each feature type individually with our model that uses them jointly. For all models, we made use of LSTMs with 2048 units. Recall that our approach relies on a multi-stage LSTM, which we denote by *MS-LSTM*. The results of this experiment for different losses are reported in Table 5. These results clearly evidence the importance of using both feature types, which consistently outperforms using individual ones in all settings.

Table 6. Influence of the number of hidden LSTM units and of our average pooling strategy in our multi-stage LSTM model. These experiments were conducted on the first splits of UCF-101 and JHMDB-21.

Setup	Average Pooling	Hidden Units	UCF-101	JHMDB-21
Ours (CE)	wo/	1024	77.26%	52.80%
Ours (CE)	wo/	2048	78.09%	53.43%
Ours (CE)	w/	2048	78.93%	54.30%
Ours (ECE)	wo/	1024	79.10%	55.33%
Ours (ECE)	wo/	2048	79.41%	56.12%
Ours (ECE)	w/	2048	80.38%	57.05%
Ours (LGL)	wo/	1024	79.76%	55.70%
Ours (LGL)	wo/	2048	80.10%	56.83%
Ours (LGL)	w/	2048	81.27%	57.70%
Ours	wo/	1024	81.94%	56.24%
Ours	wo/	2048	82.16%	57.92%
Ours	w/	2048	83.37%	58.41%

4. Robustness to the Number of Hidden Units

Based on our experiments, we found that for large datasets such as UCF-101, the 512 hidden units that some baselines use (e.g. [2, 16]) do not suffice to capture the complexity of the data. Therefore, to study the influence of the number of units in the LSTM, we evaluated different versions of our model with 1024 and 2048 hidden units (since 512 yields poor results and higher numbers, e.g., 4096, would require too much memory) and trained the model with 80% training data and validated on the remaining 20%. For a single LSTM, we found that using 2048 hidden units performs best. For our multi-stage LSTM, using 2048 hidden units also yields the best results. We also evaluated the importance of relying on average pooling in the LSTM. The results of these different versions of our MS-LSTM framework are provided in Table 6. This shows that, typically, more hidden units and average pooling can improve accuracy slightly.

5. Effect of the LSTM Architecture

Finally, we study the effectiveness of our multi-stage LSTM architecture at merging our two feature types. To this end, we compare the results of our MS-LSTM with the following baselines: A single-stage LSTM that takes as input the concatenation of our context-aware and action-aware features (Concatenation); The use of two parallel LSTMs whose outputs are merged by concatenation and then fed to a fully-connected layer (Parallel). A multi-stage LSTM where the two different feature-types are processed in the reverse order (Swapped), that is, the model processes the action-aware features first and, in a second stage, combines them with the context-aware ones; The results of this comparison are provided in Table 7. Note that both multi-stage LSTMs outperform the single-stage one and the two parallel LSTMs, thus indicating the importance of treating the two types of features sequentially. Interestingly, processing context-aware features first, as we propose, yields higher accuracy than considering the action-aware ones at the beginning. This matches our intuition that context-aware features carry global information about the image and will thus yield noisy results, which can then be refined by exploiting the action-aware features.

Table 7. Comparison of our multi-stage LSTM model with diverse fusion strategies. We report the results of simple concatenation of the context-aware and action-aware features, their use in two parallel LSTMs with late fusion, and swapping their order in our multi-stage LSTM, i.e., action-aware first, followed by context-aware. Note that multi-stage architectures yield better results, with the best ones achieved by using context first, followed by action, as proposed in this paper.

Feature Order	Sequence Learning	Accuracy
Concatenation	LSTM	77.16%
Parallel	2 Parallel LSTMs	78.63%
Swapped	MS-LSTM (Ours)	78.80%
Ours	MS-LSTM (Ours)	83.37%

Furthermore, we evaluate a CNN-only version of our approach, where we removed the LSTM, but kept our average pooling strategy to show the effect of our MS-LSTM architecture on top of the CNN. On UCF-101, this achieved 69.53% for earliest and 73.80% for latest prediction. This shows that, while this CNN-only framework yields reasonable predictions, our complete approach with our multistage LSTM benefits from explicitly being trained on multiple frames, thus achieving significantly higher accuracy (80.5% and 83.4%, respectively). While the LSTM could in principle learn to perform average pooling, we believe that the lack of data prevents this from happening.

Acknowledgement. Authors thank Oscar Friberg for his assistance in conducting additional experiments of the supplementary material.

References

[1] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould. Dynamic image networks for action recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition CVPR*, 2016.

[2] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[3] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[4] B. Fernando, P. Anderson, M. Hutter, and S. Gould. Discriminative hierarchical rank pooling for activity recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[5] G. Gkioxari and J. Malik. Finding action tubes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 759–768, 2015.

[6] S. Herath, M. Harandi, and F. Porikli. Going deeper into action recognition: A survey. *Image and Vision Computing*, 60:4 – 21, 2017. Regularization Techniques for High-Dimensional Data Analysis.

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[8] K. Laviers, G. Sukthankar, D. W. Aha, M. Molineaux, C. Darken, et al. Improving offensive performance through opponent modeling. In *Proceedings of AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 58–63.

[9] Y. Li, W. Li, V. Mahadevan, and N. Vasconcelos. Vlad3: Encoding dynamics of deep features for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1951–1960, 2016.

[10] S. Ma, L. Sigal, and S. Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1942–1950, 2016.

[11] M. Ryoo, C.-C. Chen, J. Aggarwal, and A. Roy-Chowdhury. An overview of contest on semantic description of human activities (sdha) 2010. In *Recognizing Patterns in Signals, Speech, Images and Videos*, pages 270–285. Springer, 2010.

[12] M. S. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1036–1043. IEEE, 2011.

[13] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[14] K. Soomro, H. Idrees, and M. Shah. Online localization and prediction of actions and interactions. *arXiv preprint arXiv:1612.01194*, 2016.

[15] K. Soomro, H. Idrees, and M. Shah. Predicting the where and what of actors and actions through online action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2648–2657, 2016.

[16] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using lstms. *CoRR, abs/1502.04681*, 2, 2015.

[17] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497. IEEE, 2015.

[18] D. Waltisberg, A. Yao, J. Gall, and L. Van Gool. Variations of a hough-voting action recognition system. In *Recognizing patterns in signals, speech, images and videos*, pages 306–312. Springer, 2010.

[19] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3551–3558, 2013.

[20] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4305–4314, 2015.

[21] L. Wang, Y. Qiao, X. Tang, and L. Van Gool. Actionness estimation using hybrid fully convolutional networks. *arXiv preprint arXiv:1604.07279*, 2016.

[22] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.

[23] T.-H. Yu, T.-K. Kim, and R. Cipolla. Real-time action recognition by spatiotemporal semantic and structural forests. In *BMVC*, volume 2, page 6, 2010.

[24] F. Yuan, V. Prinet, and J. Yuan. Middle-level representation for human activities recognition: the role of spatio-temporal relationships. In *ECCV*, pages 168–180. Springer, 2010.

[25] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702, 2015.

[26] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang. Real-time action recognition with enhanced motion vector cnns. *arXiv preprint arXiv:1604.07669*, 2016.