Supplementary Material: Long-Term Feature Banks for Detailed Video Understanding

Chao-Yuan Wu[1,2] Christoph Feichtenhofer[2] Haoqi Fan[2]
Kaiming He[2] Philipp Krähenbühl[1] Ross Girshick[2]

[1]The University of Texas at Austin [2]Facebook AI Research (FAIR)

1. Backbone Architecture

We use ResNet-50 I3D [5, 6] with non-local blocks [11] as the 'backbone' of our model. Following Wang *et al.* [12], the network only downsamples the temporal dimension by a factor of two. Table 1 presents the exact specification.

Stage	Specification	Output size
conv_1	$5\times7\times7$, 64, stride 1, 2, 2	$32\times112\times112$
pool_1	$1\times3\times3$ max, stride 1, 2, 2	$32\times56\times56$
res_2	$\begin{bmatrix} 3\times1\times1, 64 \\ 1\times3\times3, 64 \\ 1\times1\times1, 256 \end{bmatrix}\times3$	$32\times56\times56$
pool_2	$2\times1\times1$ max, stride 2, 1, 1	$16\times56\times56$
res_3	$\begin{bmatrix} 3(1)\times1\times1, 128 \\ 1\times3\times3, 128 \\ 1\times1\times1, 512 \end{bmatrix}\times4$, NL: 1, 3	$16\times28\times28$
res_4	$\begin{bmatrix} 3(1)\times1\times1, 256 \\ 1\times3\times3, 256 \\ 1\times1\times1, 1024 \end{bmatrix}\times6$, NL: 1, 3, 5	$16\times14\times14$
res_5	$\begin{bmatrix} 3(1)\times1\times1, 512 \\ 1\times3\times3, 512 \\ 1\times1\times1, 2048 \end{bmatrix}\times3$	$16\times14\times14$

Table 1. ResNet-50 NL-I3D [5, 6, 11] backbone used in this paper. Here we assume input size $32\times224\times224$ (frames$\times$width$\times$height). 'NL: $i_0, i_1, \ldots$' in stage 'res_j' denotes additional non-local blocks [11] after block i_0, i_1, $\ldots$ of res_j. $3(1)\times1\times1$ denotes that we either use a $3\times1\times1$ or a $1\times1\times1$ convolution. Specifically, we use $3\times1\times1$ for block 0, 2 of res_3, block 0, 2, 4 of res_4, and block 1 of res_5, and use $1\times1\times1$ for the rest.

2. AVA Person Detector

We use Faster R-CNN [9] with a ResNeXt-101-FPN [8, 13] backbone for person detection. The model is pre-trained on ImageNet for classification, and on COCO for keypoint detection. The model obtains 56.9 box AP and 67.0 keypoint AP on COCO keypoints. Model parameters are available in Detectron Model Zoo [4]. We fine-tune the model on AVA bounding boxes from the training videos for 130k iterations with an initial learning rate of 0.005, which is decreased by a factor of 10 at iteration 100k and 120k. To improve generalization, we train with random scale jittering (from 512 to 800 pixels). The final model obtains 93.9 AP@50 on the AVA validation set.

3. LFB *vs*. Improving Backbones

A large body of recent research focuses on improving 3D CNN architectures, *i.e.*, improving modeling of short-term patterns. This paper, on the other hand, aims at improving the modeling of long-term patterns. How do these two directions impact video understanding differently?

We plot the per-class impact of LFB in Fig. 1, the per-class impact of improving backbone in Fig. 2, and compare them in Fig. 3. The error bars are plus/minus one standard error around the mean, computed from 5 runs. We see that they lead to improvement in different action classes. Using LFB leads to improvement in many interactive actions, such as 'play musical instrument' or 'sing to', while improving backbone leads to improvement in more standalone actions, such as 'hand shake' or 'climb'. This suggests that improving long-term modeling (through LFB) and short-term modeling (through improving backbone) are complementary; we believe both are important for future video understanding research.

4. AVA STO Regularization

To address the overfitting issue of STO on AVA, we experimented with a number of regularization techniques, as summarized in Table 2. We found that dropout [10] was insufficient to regularize an STO, but injecting 'distractors', *i.e.*, randomly sampled features, into the features being attended during training was very effective. We report STO results with 'distractor' training for AVA unless otherwise stated. For STO on other datasets, we did not observe obvious overfitting.

3D CNN	STO	STO + 0.5 dropout	STO + 0.8 dropout	STO + distractors
22.1	20.2	20.1	20.9	**23.2**

Table 2. STO **with different regularization techniques on AVA** (mAP in %).

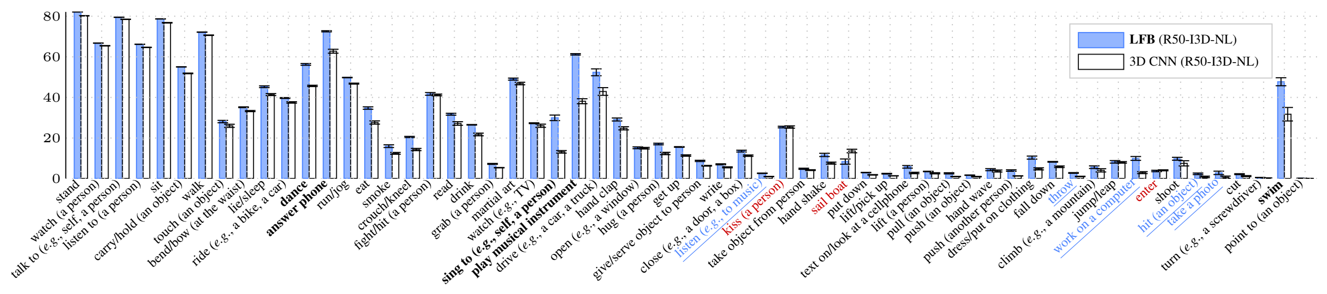

Figure 1. **Impact of LFB.** We compare per-class AP of 3D CNN (22.1 mAP) and LFB model (25.5 mAP) on AVA. LFB leads to larger improvement on *interactive* actions, *e.g*., 'sing to', 'play musical instrument', or 'work on a computer'. (**Bold:** 5 classes with the largest absolute gain. Blue: 5 classes with the largest relative gain. Red: classes with decreased performance. Classes are sorted by frequency.)

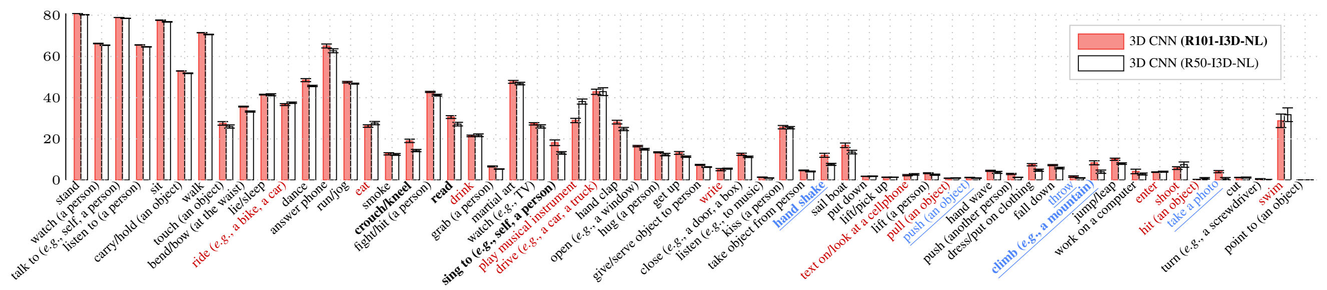

Figure 2. **Impact of improving backbone.** We compare per-class AP of 3D CNN with the default backbone (R50-I3D-NL; 22.1 mAP) and a stronger backbone (R101-I3D-NL; 23.0 mAP) on AVA. Improving backbone leads to larger improvement in standalone actions, such as 'crouch/kneel', 'read', or 'hand shake'. (**Bold:** 5 classes with the largest absolute gain. Blue: 5 classes with the largest relative gain. Red: classes with decreased performance.)

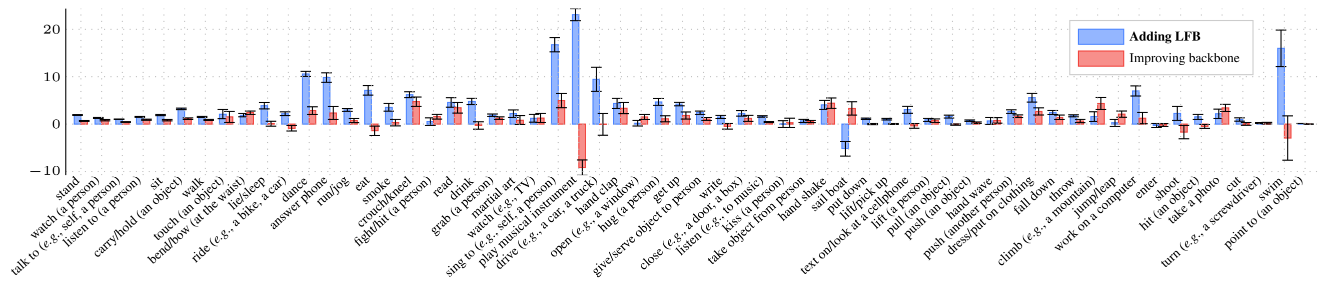

Figure 3. **Adding LFB *vs*. improving backbone.** We compare the absolute improvement (in AP) brought by LFB and the improvement brought by improving backbone. We see that they lead to improvement in different action classes. This suggests that improving long-term modeling (through LFB) and short-term modeling (through improving backbone) are complementary; we believe both are important for future video understanding research.

5. Training Schedule for EPIC-Kitchens

We train the verb models for 36k iterations with 10^{-5} weight decay and a learning rate of 0.0003, which is then decreased by 10 times at iteration 28k and 32k. For the noun models, we train for 50k iterations with weight decay 10^{-6} and a learning rate of 0.001, which is decreased by 10 times at iteration 40k and 45k.

6. EPIC-Kitchens Inference

We sample training clips such that the center of the clip falls within a training segment. For testing, we sample one center clip per segment, resize such that the short side is 256 pixels, and use a single center crop of 256×256. We compute the probability of an action as the product of the softmax scores, weighted by a prior μ, *i.e.*

$$P(\text{action} = (v, n)) \propto \mu(v, n)P(\text{verb} = v)P(\text{noun} = n),$$

where μ is a prior estimated as the count of (v, n) pair divided by count of n in training annotations.

7. Object Detector for LFB of EPIC-Kitchens Noun Model

We use Faster R-CNN [9] with ResNeXt-101-FPN [8, 13] backbone for the object detector. The detector is pre-trained on Visual Genome [7] with 1600 class labels defined in Anderson *et al*. [1]. We fine-tune this model on the 'new' training split (defined in Baradel *et al*. [2]) of EPIC-Kitchens for 90k iterations, with random scale jittering (from 512 to 800 pixels). We use an initial learning rate of 0.005, which is decreased by a factor of 10 at iteration 60k and 80k. The final model achieves 2.4 AP on the 'new' validation split. The AP is low because with the new train/val split, most of the classes are unseen during training. In addition, most of the classes have zero instance in the new, smaller validation set, and we calculate the average precision of those classes as 0. This is also not comparable to the performance reported in [3], where the model is trained on the full training set, and evaluated on the unreleased test sets.

8. Charades Training Schedule

We train the models to predict the 'clip-level' labels, *i.e*., the union of the frame labels that fall into the clip's temporal range. We train the baseline 3D CNNs with the default 24k schedule with learning rate 0.02, which is decreased by a factor of 10 at iteration 20k. To train LFB models, we use a 2-stage approach following STRG [12]. We first train the model without the FBO using the 24k schedule, and then add FBO, freeze backbone, and train for half of the schedule (12k iterations, so 36k in total). This schedule helps prevent overfitting that was observed when training directly with FBO in one stage. For training STO, we observed worse performance with this 2-stage approach, so we report the STO performance using the default 24k schedule.

9. Charades NL Block Details

	R50-I3D-NL		R101-I3D-NL	
	pre-act	**post-act**	pre-act	**post-act**
STO	39.0	39.6	40.5	41.0
LFB NL	39.5	**40.3**	41.5	**42.5**
3D CNN	38.3		40.3	

Table 3. **Pre-activation *vs*. post-activation** NL' **on Charades.**

We experimented with two variants of NL' design: a *pre-activation* (the default variant described in paper), and a *post-activation* variant, where we move ReLU after the skip connection, and move the layer normalization after the output linear layer. For both variants, LFB consistently outperforms STO and baseline 3D CNN (Table 3). We choose post-activation as default for Charades due to the stronger performance. For AVA and EPIC-Kitchens, we did not observe any noticeable difference between the two variants.

References

[1] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, 2018. 3

[2] F. Baradel, N. Neverova, C. Wolf, J. Mille, and G. Mori. Object level visual reasoning in videos. In *ECCV*, 2018. 3

[3] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling egocentric vision: The EPIC-kitchens dataset. In *ECCV*, 2018. 3

[4] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron, 2018. 1

[5] C. Gu, C. Sun, D. A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, et al. AVA: A video dataset of spatio-temporally localized atomic visual actions. In *CVPR*, 2018. 1

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[7] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017. 3

[8] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1, 3

[9] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 3

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014. 1

[11] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018. 1

[12] X. Wang and A. Gupta. Videos as space-time region graphs. In *ECCV*, 2018. 1, 3

[13] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 1, 3