Learning Video Representations from Correspondence Proposals Supplementary Material

Xingyu Liu*
Stanford University

Joon-Young Lee
Adobe Research

Hailin Jin
Adobe Research

A. Overview

In this document, we provide more details to the main paper and show extra results on per-class accuracy and visualizations.

In section B, we provide more details on the Kinetics/ResNet-18 ablation experiments (main paper section 5.1). In section C, we provide more details on the baseline architectures in Kinetics/ResNet-18 comparison experiments (main paper section 5.2). In section D, we provide details on the CPNet architecture used in Kinetics/ResNet-101 experiment (main paper section 5.3). In section E, we provide details on the architecture used in Something-Something and Jester experiments (main paper section 5.4 and 5.5). In section F we report the per-class accuracy of C2D model and our CPNet model on Something-Something and Jester datasets. Lastly in section G we provide time complexity of our model and in section H we provide more visualization results on all three datasets.

B. CPNet Architecture in Kinetics/ResNet-18 Experiments

Our CPNet is instantiated by adding a CP module after the last convolution layer of a residual group but before ReLU, as illustrated in Figure 1. For Kinetics/ResNet-18 experiments in main paper section 5.1 and 5.2, each CP module has MLP with two hidden layers. Suppose the number of channels of the input tensor of CP module is C. The number of channels of the hidden layers in the MLPs is then $[C/4, C/2]$. The number of nearest neighbors k is set to 8 for the results in Table 3(a)(c)(d) of the main paper. k varies for the results in Table 3(b). The location of CP module is deduced from the last column of Table 1 for different experiments in section 5.1 of the main paper.

C. Baseline Architectures in Kinetics/ResNet-18 Comparison Experiment

In Table 1, we listed all the architectures used in Kinetics/ResNet-18 comparison experiments, as a supplementary to Table 2 of the main paper. C2D/C3D are vanilla 2D or 3D CNN. ARTNet is pulled directly from [6]. It was designed to have the same number of parameters as its C3D counterpart. NL Net model is adapted from [7], by adding an NL block at the end of each residual group of C2D ResNet-18. CPNet is instantiated in the same way as illustrated in Figure 1. Combined with results in Table 3(d) of the main paper, our CPNet outperforms NL Net and ARTNet in terms of validation accuracy with fewer parameters, showing its superiority.

*Majority of the work done as an intern at Adobe Research.

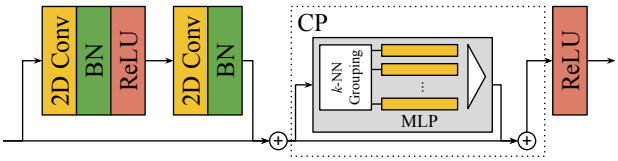


Figure 1: CP module inserted into a residual group of ResNet-18 backbone.

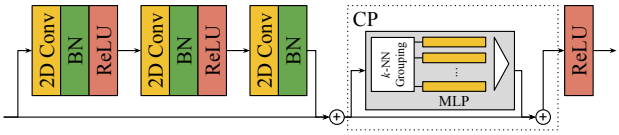


Figure 2: CP module inserted into a residual group of ResNet-101 backbone.

D. CPNet Architecture in Kinetics/ResNet-101 Experiment

We listed CPNet architecture used in Kinetics/ResNet-101 experiment in Table 2. Each residual group in ResNet-101 has three convolution layers. Our CPNet is instantiated by adding a CP module after the last convolution layer of a residual group but before ReLU, as illustrated in Figure 2. Suppose the number of channels of the input tensor of CP module is C. The number of channels of the hidden layers in the MLPs is then $[C/16, C/8]$. The number of nearest neighbors k is set to 4.

We used five CP modules in the architecture. Two CP modules are in res_3 groups with spatial resolution of 28×28 and the rest three are in res_4 groups with spatial resolution

Table 1: Complete Architectures used in Kinetics dataset comparison experiments.

layer	output size	C2D (C3D)	ARTNet [6]	NL C2D Net 6 NL blocks [7]	**CPNet (Ours)** 6 CP modules
conv1	$56 \times 56 \times 8$	$7 \times 7(\times 3), 64$, stride 2, 2(, 1)	SMART $7 \times 7 \times 3, 64$, stride 2, 2, 1	$7 \times 7, 64$, stride 2, 2	$7 \times 7, 64$, stride 2, 2
res_2	$56 \times 56 \times 8$	$\begin{bmatrix} 3 \times 3(\times 3), 64 \\ 3 \times 3(\times 3), 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 64 \\ \text{SMART } 3 \times 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
res_3	$28 \times 28 \times 8$	$\begin{bmatrix} 3 \times 3(\times 3), 128 \\ 3 \times 3(\times 3), 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 128 \\ \text{SMART } 3 \times 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \\ \text{NL block} \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \\ \text{CP module} \end{bmatrix} \times 2$
res_4	$14 \times 14 \times 8$	$\begin{bmatrix} 3 \times 3(\times 3), 256 \\ 3 \times 3(\times 3), 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ \text{SMART } 3 \times 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \\ \text{NL block} \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \\ \text{CP module} \end{bmatrix} \times 2$
res_5	$7 \times 7 \times 8$	$\begin{bmatrix} 3 \times 3(\times 3), 512 \\ 3 \times 3(\times 3), 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \\ \text{NL block} \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \\ \text{CP module} \end{bmatrix} \times 2$
	$1 \times 1 \times 1$	global average pooling, fc 400			
params (M)		10.84 (31.81)	31.81	10.88	10.86

Table 2: CPNet Architectures used in Kinetics large model experiments.

layer	output size	CPNet, 5 CP modules
conv1	$56 \times 56 \times 8$	$7 \times 7, 64$, stride 2, 2
res_2	$56 \times 56 \times 8$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
res_3	$28 \times 28 \times 8$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 2$ $\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \\ \text{CP module} \end{bmatrix} \times 2$
res_4	$14 \times 14 \times 8$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 20$ $\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \\ \text{CP module} \end{bmatrix} \times 3$
res_5	$7 \times 7 \times 8$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	$1 \times 1 \times 1$	global average pooling, fc 400

14×14. Such mixed usage of CP modules at residual groups of different spatial resolutions enables correspondence and motion in different semantic level to be learned jointly. We only listed the case of using 8 frames as input. For 32-frame input, all "8" in the second column of Table 2 should be replaced by 32.

E. Architecture used in Something-Something and Jester Experiments

We listed the CPNet architectures used in Something-Something [3] and Jester [5] experiments in Table 3. CPNet is instantiated in the same way as illustrated in Figure 1. Suppose the number of channels of the input tensor of CP module is C. The number of channels of the hidden layers in the MLPs is then $[C/4, C/2]$. The number of nearest neighbors k is set to 12.

We used five CP modules in the architecture. Two CP modules are in res_3 groups with spatial resolution of 28×28 and the rest three are in res_4 groups with spatial resolution 14×14. We only listed the case of using 12 frames as input. For 24- or 48-frame input, all "12" in the second column of Table 2 should be replaced by 24 or 48.

F. Per-class accuracy of Something-Something and Jester models

To understand the effect of CP module to the final performance, we provide the CPNet's per-class top-1 accuracy gain compared with the respective C2D baseline on Jester in Figure 3 and Something-Something in Figure 5.

We can see that categories that strongly rely on motion (especially in long-range) in videos typically have large accuracy improvement after adding CP module. On the other hand, categories that doesn't require reasoning motion to classify have little or negative gain in accuracy. The results coincide with our intuition that CP module effectively captures dynamic content of videos.

On Jester dataset [5], the largest accuracy improvements are achieved in categories that involve long-range spatial motion such as "Sliding Two Fingers Up", or long-range temporal relation such as "Stop Sign". At the same time,

Table 3: CPNet Architectures used in Something-Something and Jester dataset experiments.

layer	output size	CPNet, 5 CP modules
conv1	$56 \times 56 \times 12$	$7 \times 7, 64$, stride 2, 2
res_2	$56 \times 56 \times 12$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$
res_3	$28 \times 28 \times 12$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$ $\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \\ \text{CP module} \end{bmatrix} \times 2$
res_4	$14 \times 14 \times 12$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 3$ $\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \\ \text{CP module} \end{bmatrix} \times 3$
res_5	$7 \times 7 \times 12$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$
	$1 \times 1 \times 1$	global average pooling, fc 174 or fc 27

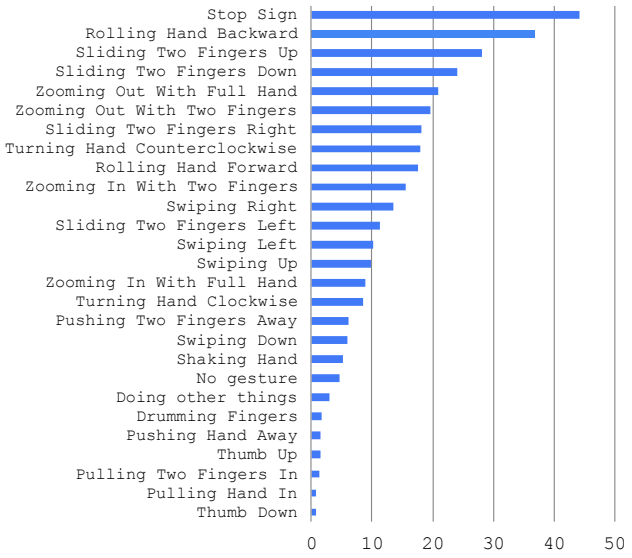


Figure 3: Per-class top-1 accuracy gain in percentage on Jester v1 dataset due to CP module.

categories that don't even need multiple frames to classify, such as "Thump Up" or "Thumb Down", have the smallest accuracy gain.

On Something-Something dataset [3], the largest accuracy improvements are achieved in categories that involve long-range spatial motion such as "Moving away from something with your camera", or long-range temporal relation such as "Lifting up one end of something without letting it drop down". At the same time, categories that don't even need multiple frames to classify, such as "Showing a photo of something to the camera", have the smallest

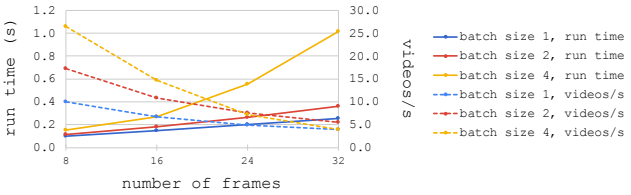


Figure 4: Model run time (solid line) and number of video sequences per second (dashed line) of CPNet with ResNet-34 backbone and spatial size 112×112.

or negative accuracy gain .

G. Model Run Time

In this section, we provide time complexity results of our model. Our CP module can be very efficient in term of computation and memory, for both training and inference.

During training, NL Net [7] computes a $THW \times THW$ matrix followed by a row-wise softmax. The whole process is **differentiable** and all the intermediate values have to be stored for computing gradients during back propagation, which causes huge overhead in memory and computation. Unlike NL Net, our CP module's computation of a $THW \times THW$ matrix results in k integers used for indexing, which is **non-differentiable**. Thus CPNet doesn't compute gradients or store the intermediate values of the $THW \times THW$ matrix, a huge saving compared to NL Net and all other works involving global attention.

During inference, our CPNet is also efficient. We evaluate the inference time complexity of the CPNet model used in Jester v1 experiment. The spatial size is 112×112. The model backbone is ResNet-34. The computing platform is an NVIDIA GTX 1080 Ti GPU with Tensorflow and cuDNN. The model performances with various batch sizes and frame lengths are illustrated in Figure 4. With batch size of 1, CPNet can reach processsing speed of 10.1 videos/s for frame length of 8 and 3.9 videos/s for frame length of 32. The number of videos that can be processed in a given time also increases as batch size increases.

We point out that there exist other more efficient implementations of CP module. In the main paper, we only presented the approach of the finding per-point k-NN in a point cloud via computing a pairwise feature distance matrix of size $THW \times THW$ followed by a row-wise $\arg\text{top}\ k$, which has time complexity of $\mathcal{O}((THW)^2 \cdot (C+k))$. This is the most convenient way to implement in deep learning frameworks such as Tensorflow. However, when deployed on inference platforms, per-point k-NN can be computed by much more efficient approaches with geometric data structures such as k-d tree [1] or Bounding Volume Hierarchy (BVH) [2] in C dimensional space. The time complexity will then be $\mathcal{O}((THW)\log(THW) \cdot (C+k))$, which includes both the construction and traversal of such tree data

structure. Accelerating k-d tree or BVH on various platforms is an ongoing research problem in the computer systems & architectures community and is not the focus of our work.

H. More Visualizations

In this section, we provide more visualizations on examples from Kinetics [4] in Figure 6, Something-Something [3] in Figure 7 and Jester [5] in Figure 8. They further show CP module's ability to propose reasonable correspondences and robustness to errors in correspondence proposal.

Despite what has been shown in the main paper, we also notice some negative examples. For example, in Figure 6(a), when proposing correspondences of the boy's left ice skate, CP module incorrectly proposed the a girl's left ice skate due to the two ice skates' visual features being too similar. CP module also didn't completely overwhelm this wrong proposal after max pooling. However, we notice that this wrong proposal is weak in the output signal: it only activates 3 out of 64 channels during max pooling which is acceptable. We point out that such "error" could also be fixed in later stages of the network or even be beneficial for applications that require reasoning relations between similar but different objects.

References

[1] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 1975. 3

[2] J. H. Clark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 1976. 3

[3] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fründ, P. Yianilos, M. Mueller-Freitag, F. Hoppe, C. Thurau, I. Bax, and R. Memisevic. The "something something" video database for learning and evaluating visual common sense. *CoRR*, abs/1706.04261, 2017. 2, 3, 4

[4] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017. 4

[5] TwentyBN. The 20BN-jester Dataset V1. https://20bn.com/datasets/jester. 2, 4

[6] L. Wang, W. Li, W. Li, and L. V. Gool. Appearance-and-relation networks for video classification. In *CVPR*, 2018. 1, 2

[7] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018. 1, 2, 3

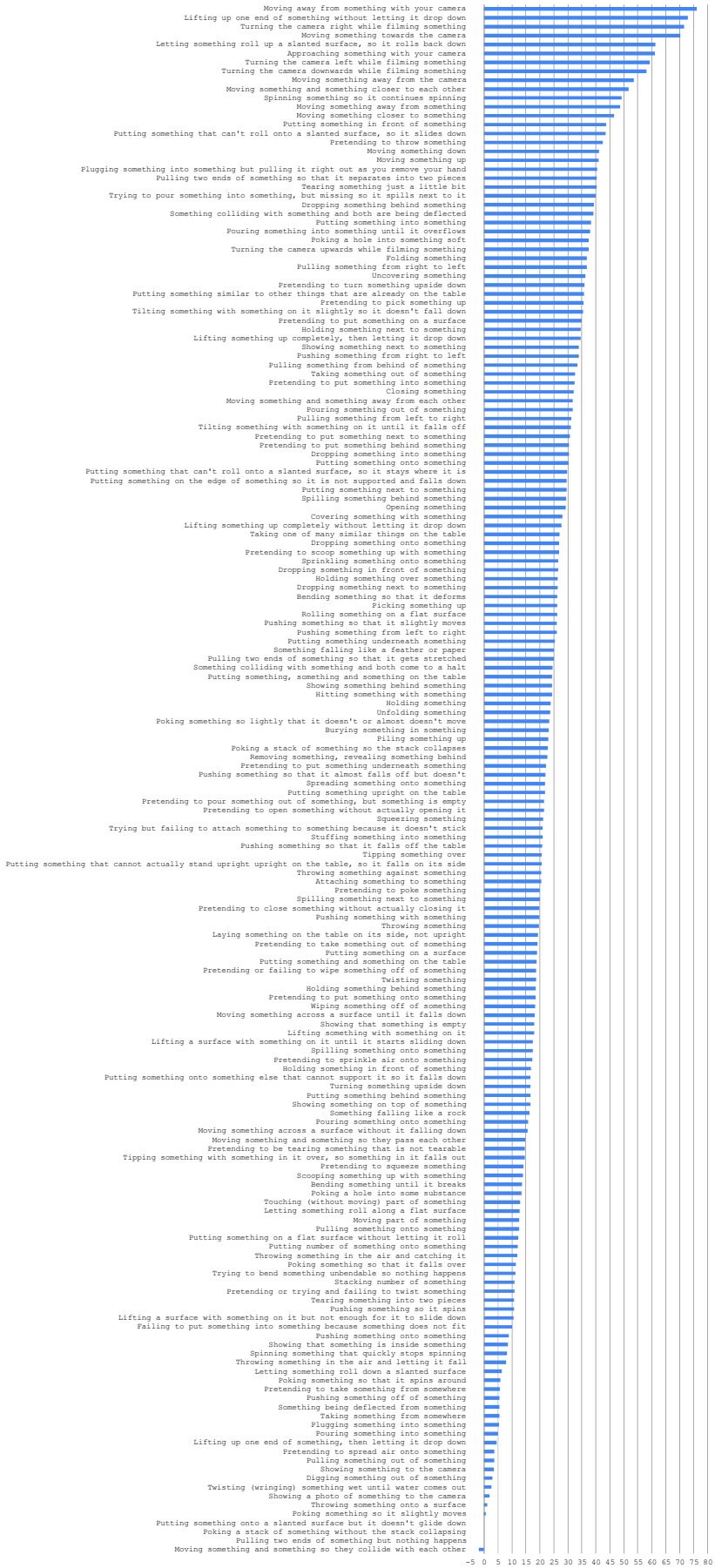
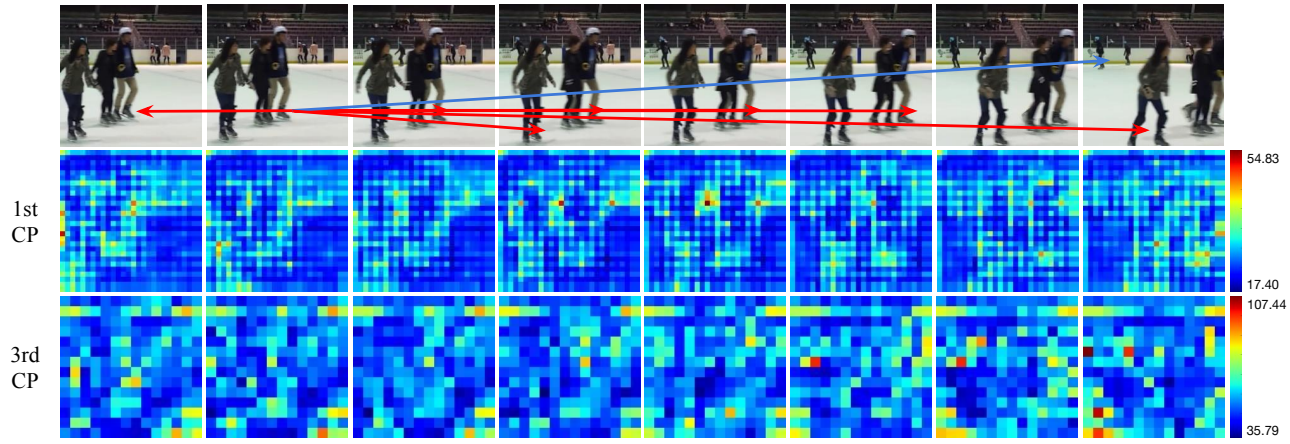
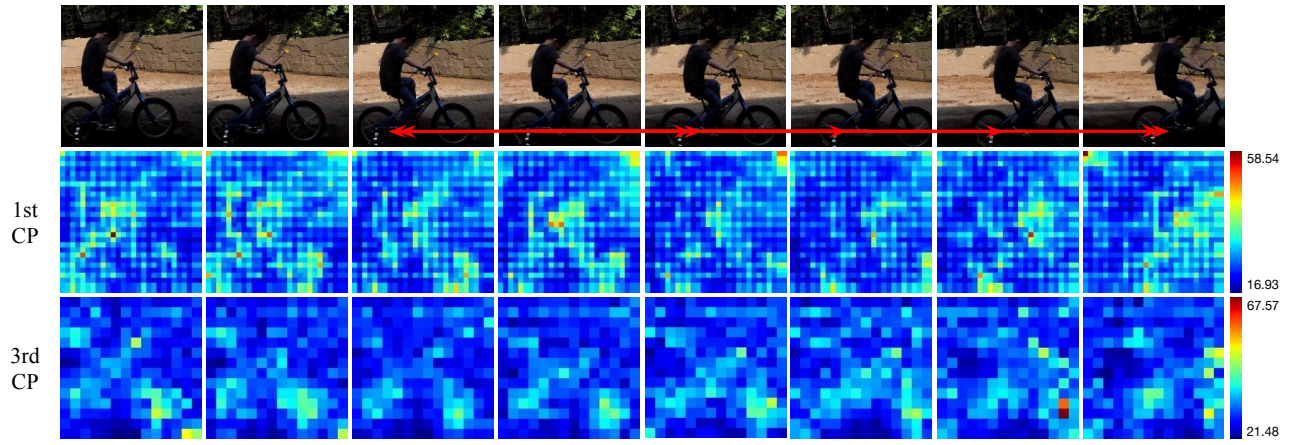


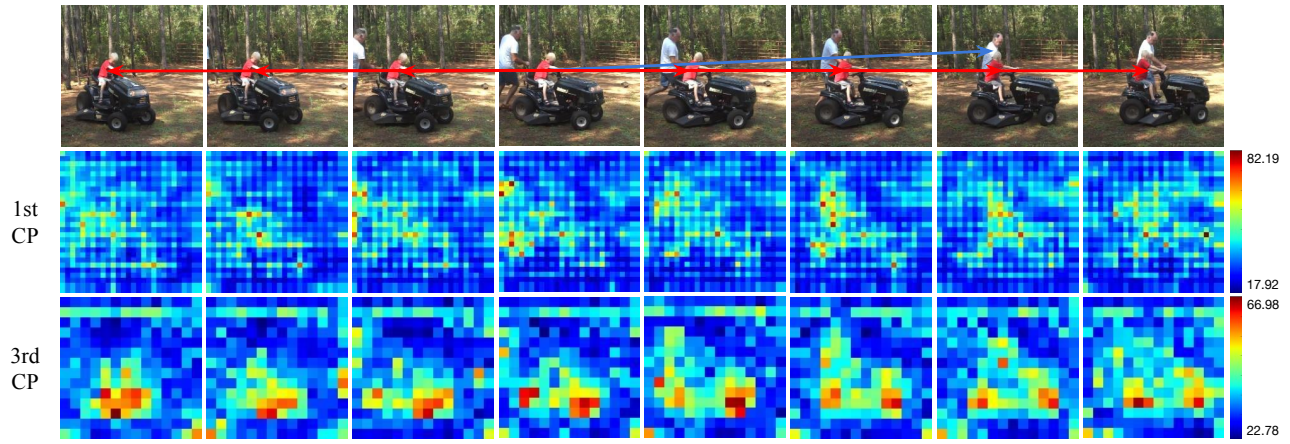
Figure 5: Per-class top-1 accuracy gain in percentage on Something-Something v2 dataset due to CP module.



(a) A video clip with label “ice skating” from Kinetics validation set.

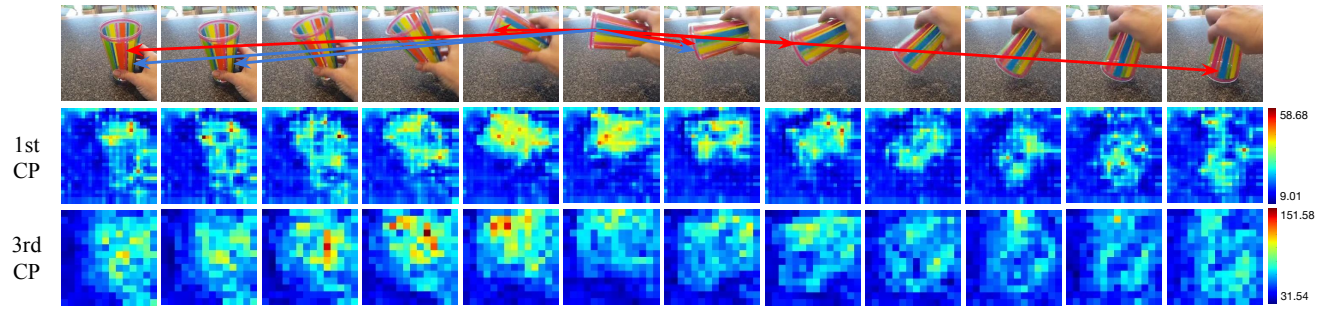


(b) A video clip with label “riding a bike” from Kinetics validation set.

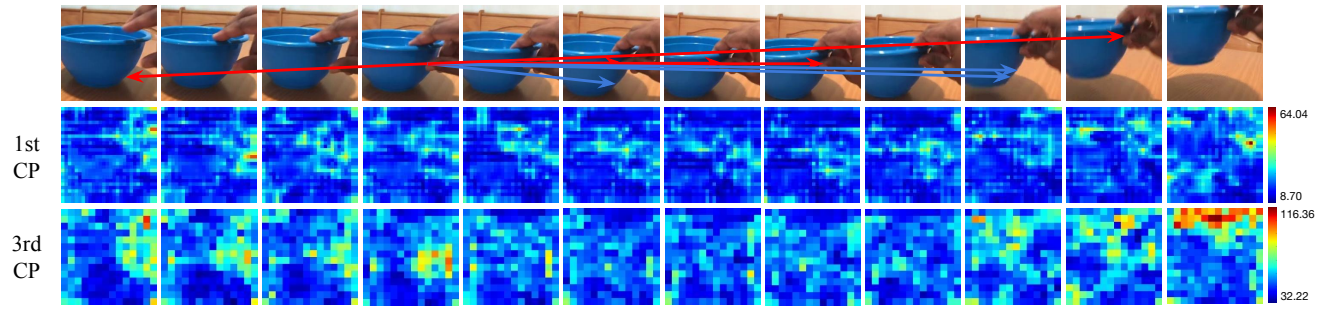


(c) A video clip with label “driving tractor” from Kinetics validation set.

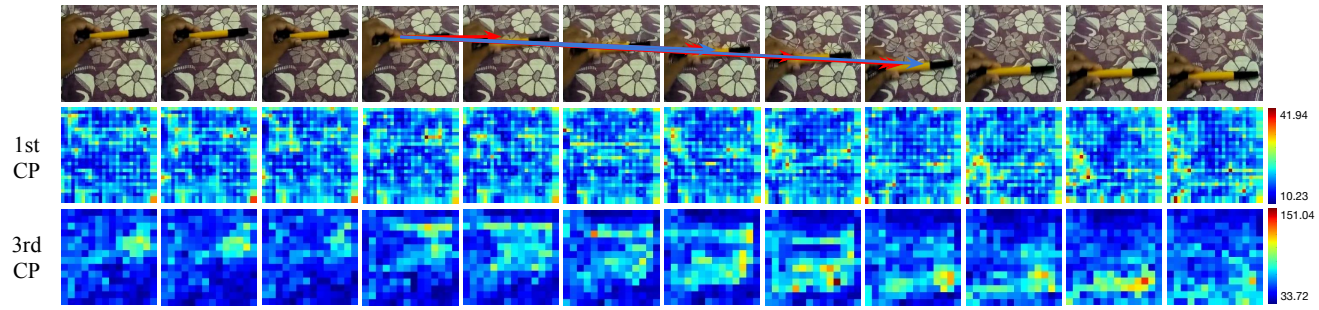
Figure 6: Additional Visualization on our final models on Kinetics dataset. Approach is the same as the main paper.



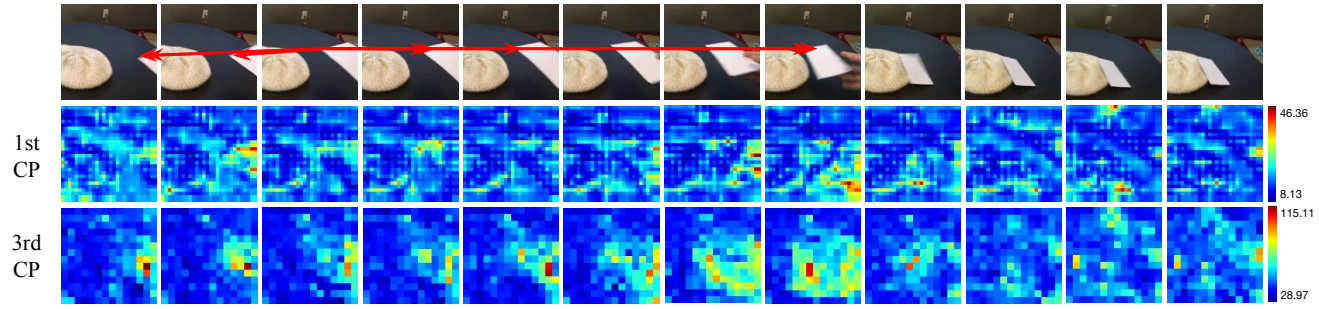
(a) A video clip with label "Turning something upside down" from Something-Something v2 validation set.



(b) A video clip with label "Picking something up" from Something-Something v2 validation set.

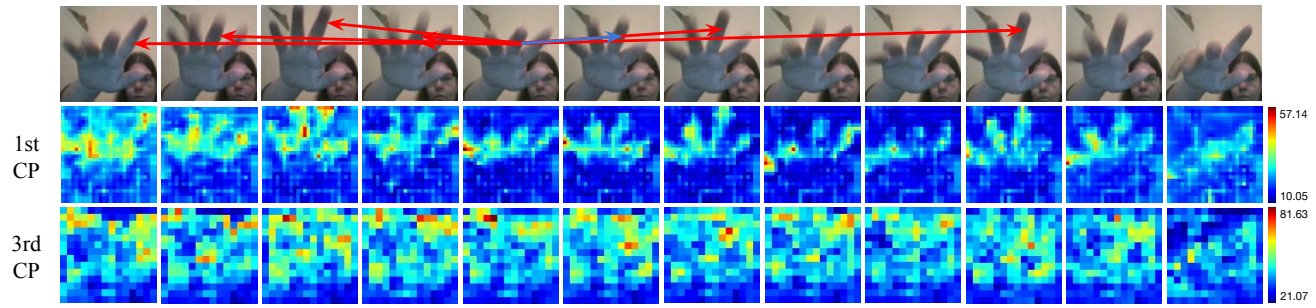


(c) A video clip with label "Moving something down" from Something-Something v2 validation set.

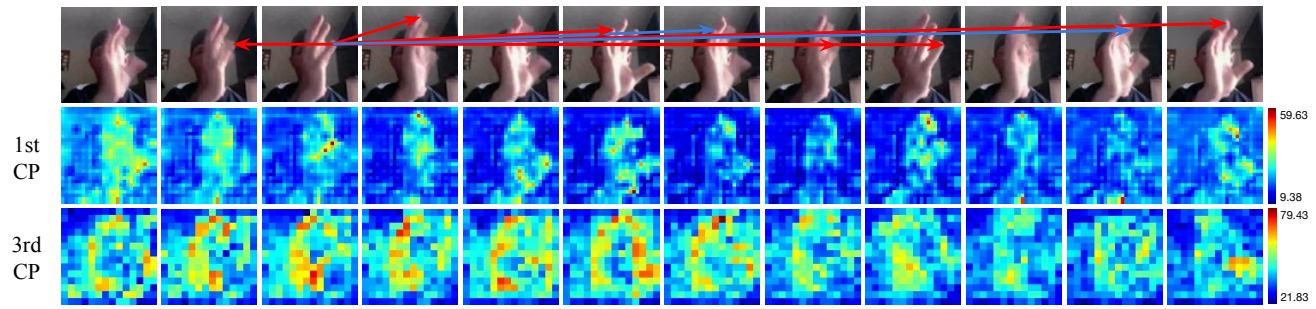


(d) A video clip with label "Dropping something next to something" from Something-Something v2 validation set.

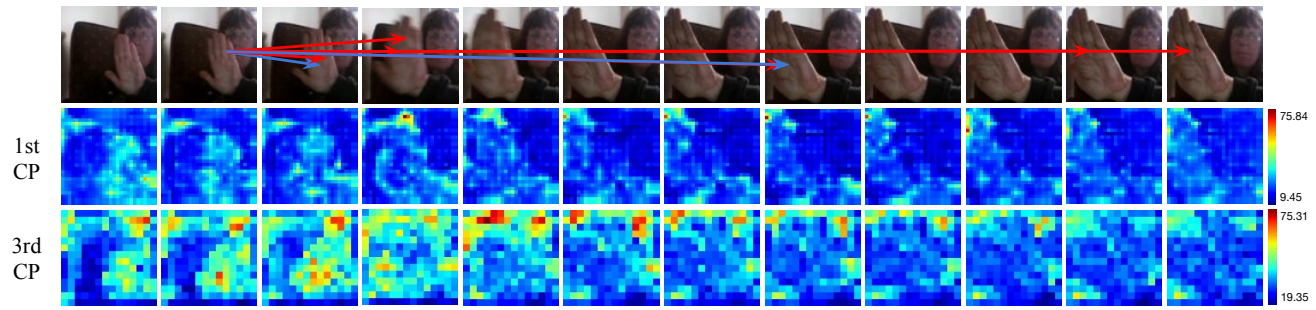
Figure 7: Additional Visualization on our final models on Something-Something v2 dataset. Approach is the same as the main paper.



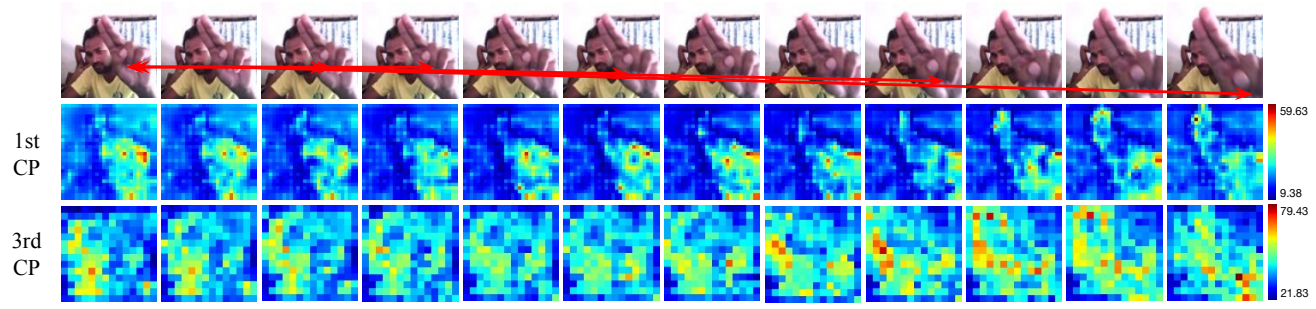
(a) A video clip with label "Drumming Fingers" from Jester v1 validation set.



(b) A video clip with label "Shaking Hand" from Jester v1 validation set.



(c) A video clip with label "Stop Sign" from Jester v1 validation set.



(d) A video clip with label "Pushing Two Fingers Away" from Jester v1 validation set.

Figure 8: Additional Visualization on our final models on Jester v1 dataset. Approach is the same as the main paper.