Learning Video Representations from Correspondence Proposals (CVPR `19) 논문 리뷰

CPNet : Correspondence Proposals Network

2020. 05. 김성욱

Learning Video Representations from Correspondence Proposals

Xingyu Liu*
Stanford University

Joon-Young Lee
Adobe Research

Hailin Jin
Adobe Research

Abstract

Correspondences between frames encode rich information about dynamic content in videos. However, it is challenging to effectively capture and learn those due to their irregular structure and complex dynamics. In this paper, we propose a novel neural network that learns video representations by aggregating information from potential correspondences. This network, named CPNet, can learn evolving 2D fields with temporal consistency. In particular, it can effectively learn representations for videos by mixing appearance and long-range motion with an RGB-only input. We provide extensive ablation experiments to validate our model. CPNet shows stronger performance than existing methods on Kinetics and achieves the state-of-the-art performance on Something-Something and Jester. We provide analysis towards the behavior of our model and show its robustness to errors in proposals.

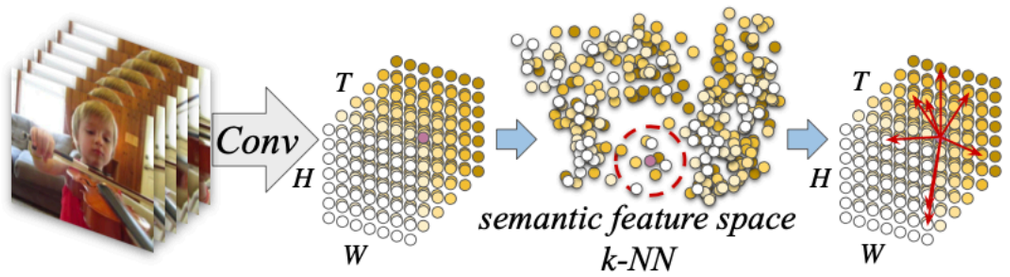


Figure 1: We view video representation tensor as a point cloud of features with $T \times H \times W$ points. For each point (e.g. the purple point), its k potentially corresponding points are the k-NN in C-dimensional semantic space from other frames. Our CP module will learn and aggregate all these potential correspondences.

erations within a local neighborhood (e.g. convolution) or global feature re-weighting (e.g. non-local means) for inter-frame relation reasoning thus cannot effectively capture correspondence: stacking local operations for wider coverage is inefficient or insufficient for long-range correspondences while global feature re-weighting fails to include positional

Introduction

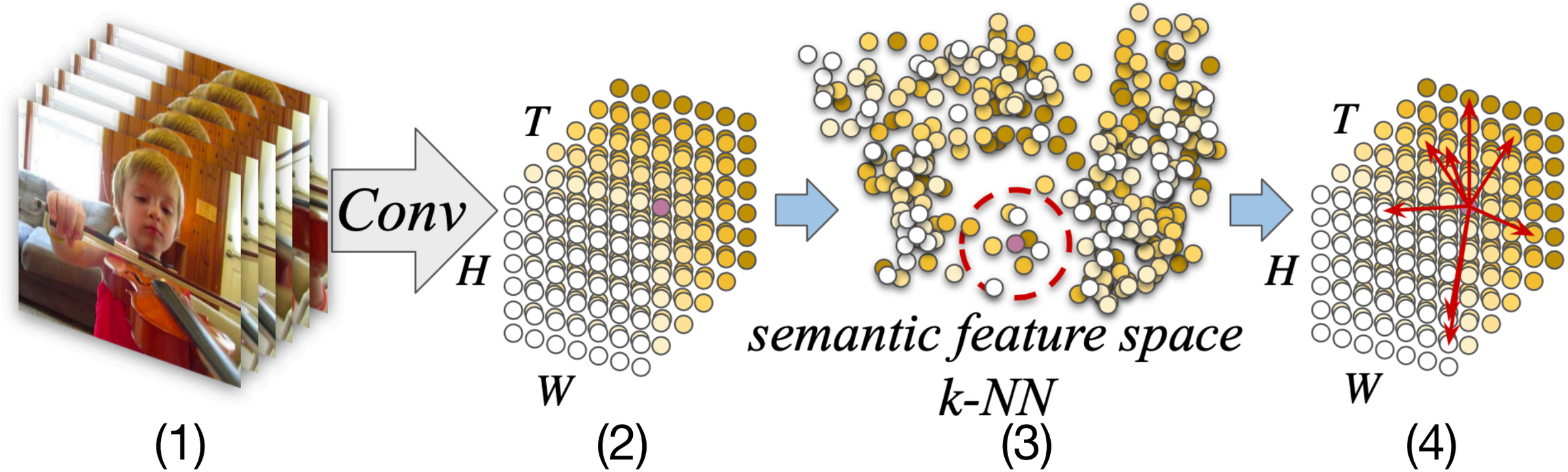
- Video Representations를 잘 표현하는 모델은 **static한 형태 (appearance)**와 **dynamic change**를 잘 학습할 수 있어야 한다.

- 비디오의 동적 특성은 Temporal하게 나타나며, 한 프레임에서 객체는 다른 프레임에서 대응하는 곳을 갖을 수 있고, 이때 Semantic 특징도 같은 방법으로 전달될 수 있다.

- 주어진 객체를 대응하는 다른 프레임에서의 객체는 일반적으로 제한된 범위의 셋을 갖는다. (내용 수정.. 설명 애매)

- CP Module : CNN을 결합하여 static appearance feature와 dynamic motion feature가 동시에 고려되어 학습되는, 시간적 공간 정보가 포함된 모듈

CP Module



(1) (2) (3) (4)

- (1) 입력 sequence : H` x W` x 3 x T
- (2) CNN을 통해 추출된 Feature Map : H x W x T x C(feature_dim)
- (3) feature_dim dimension 공간에서 각 (2)의 각 feature vector들의 분포 및 (2)에서 선택된 feature vector와 인접한 k개의 **다른 시점 feature vector(대응, Semantic)**들을 선택
- (4) 선택된 k개의 feature vector를 화살표로 표현 : 이를 Correspondence한 것으로 간주

CP Module Architecture

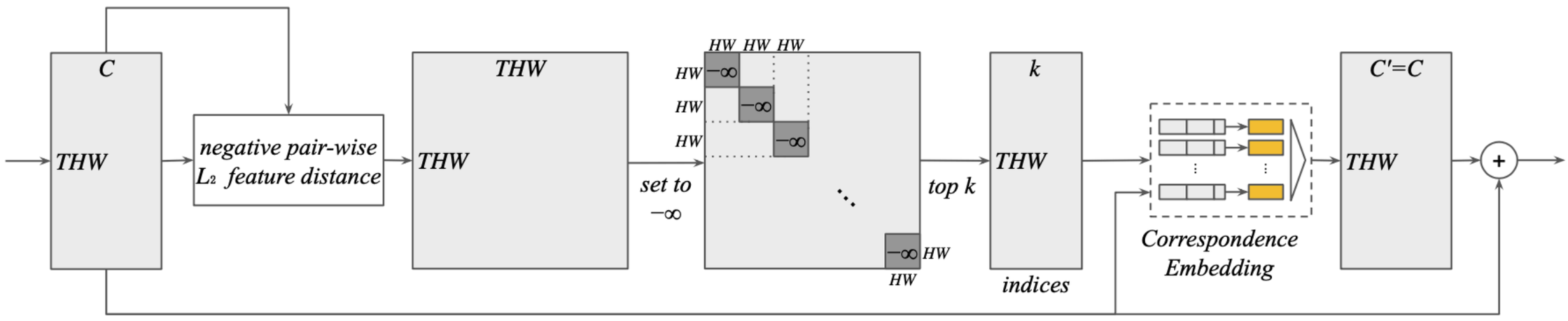


Figure 2: CP module architecture. Gray boxes denote tensors, white boxes denote operators and orange boxes denote neural networks with trainable weights. The dashed box represents the Correspondence Embedding layer, whose architecture is illustrated in detail in Figure 3.

- CP Module의 입출력은 THW x C 형태의 Matrix
- THW 개의 feature들을 point cloud 형태로 표현, 이를 기반으로
  1) k-nn grouping (corresponding), **2) Correspondence Embedding** (Representation)
  수행
- Spatio / Temporal 속성을 모두 고려한 End-to-End 아키텍쳐

T: Temporal == # of frames
H, W : Spatio dimension
C : channel dimension

CP Module : k-nn grouping

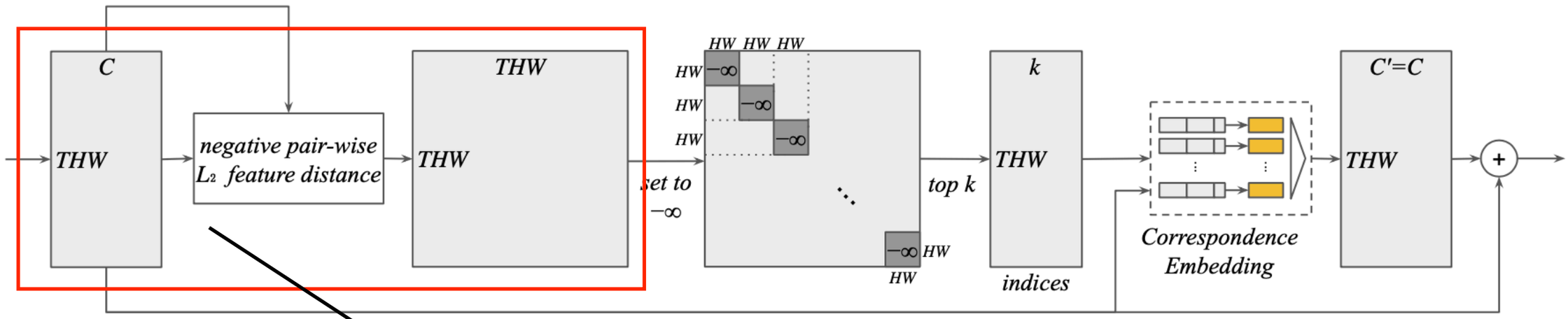
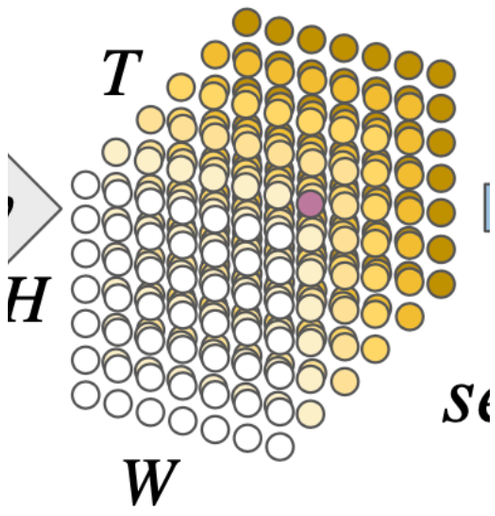
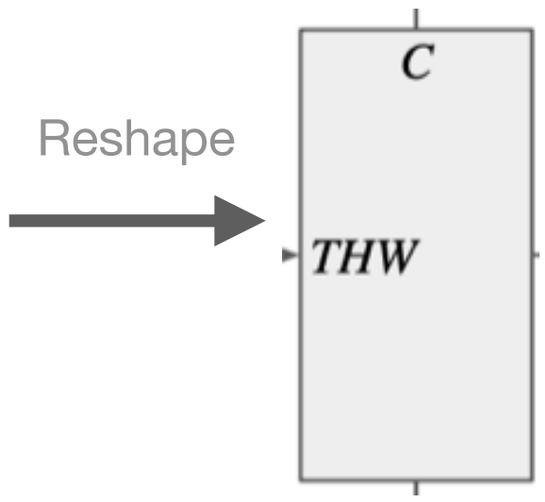
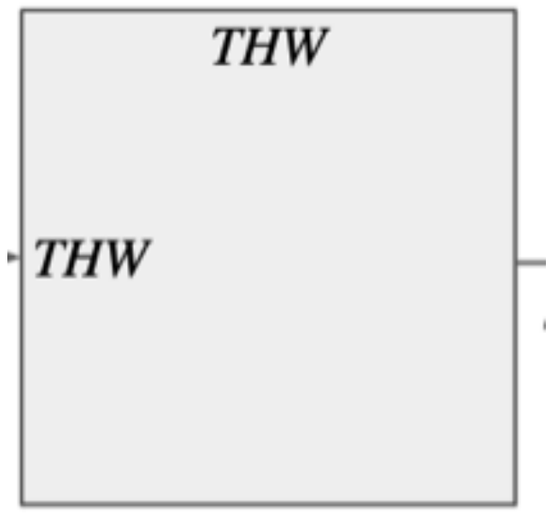


Figure 2: CP module architecture. Gray boxes denote tensors, white boxes denote operators and orange boxes denote neural networks with trainable weights. The dashed box represents the Correspondence Embedding layer, whose architecture is illustrated in detail in Figure 3.







T x H x W x C

(T x H x W) x C

== CP Module Input shape
== CP Module Output shape

[THW x C] * [THW x C]^T

거리 == 벡터 내적 == 전치 후 행렬곱

CP Module : k-nn grouping

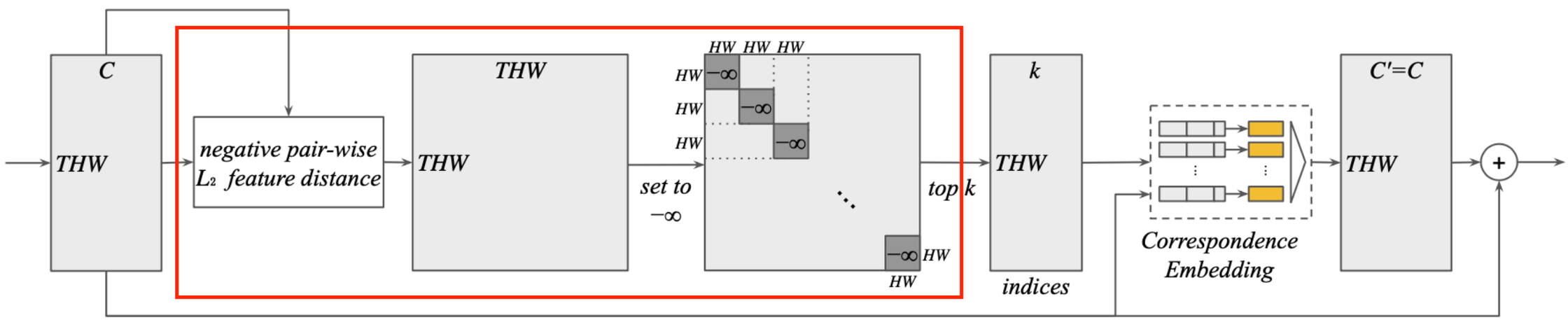
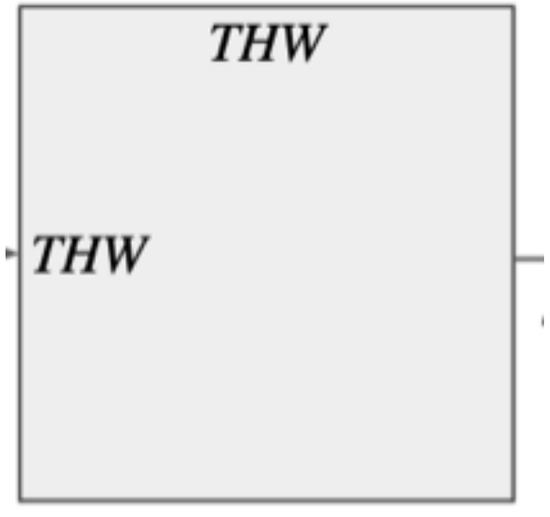
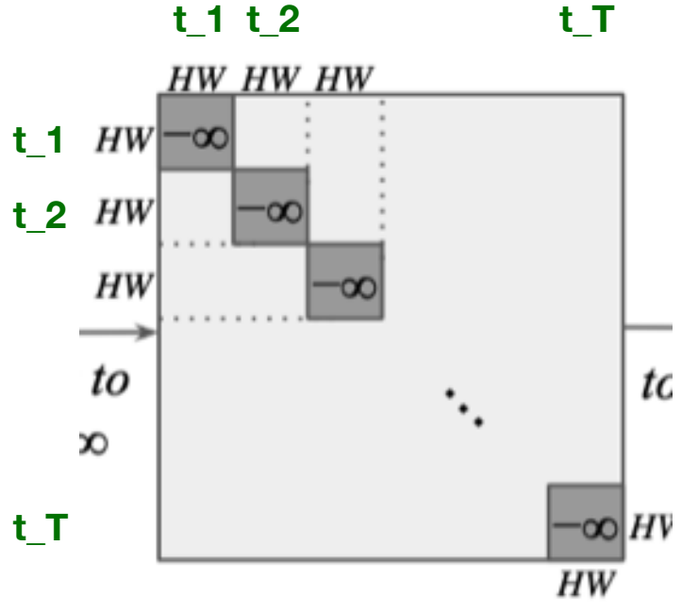


Figure 2: CP module architecture. Gray boxes denote tensors, white boxes denote operators and orange boxes denote neural networks with trainable weights. The dashed box represents the Correspondence Embedding layer, whose architecture is illustrated in detail in Figure 3.



모든 시간, 위치별로 다른 모든 경우와의 음의 거리(L2) 계산



동일한 시점에는 음의 무한대 설정

- 특정 시점 t_i의 w_j, h_k 위치의 Feature와 다른 모든 feature간의 거리 계산. (같은 시점에서는 선택되지 않게 음의 무한대 설정)

Negative L2 Distance
== Similarity

CP Module : k-nn grouping

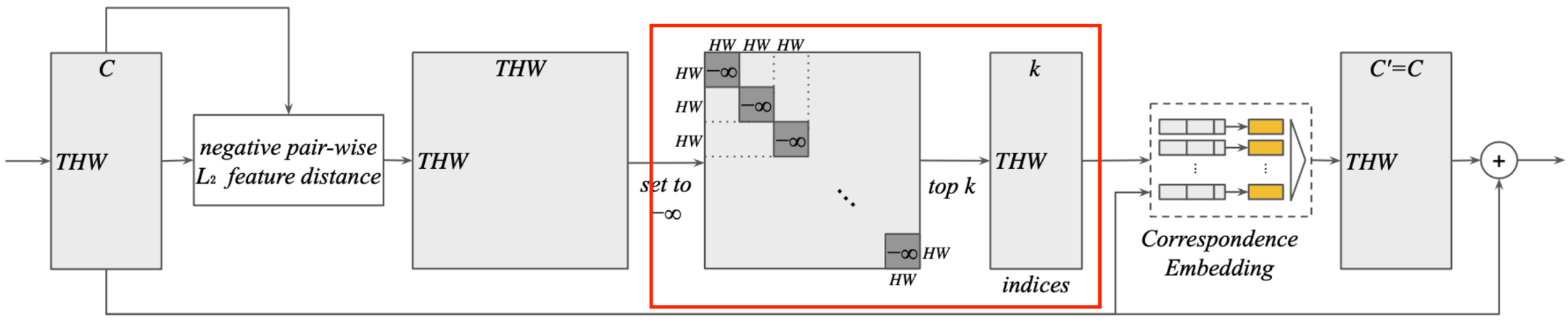
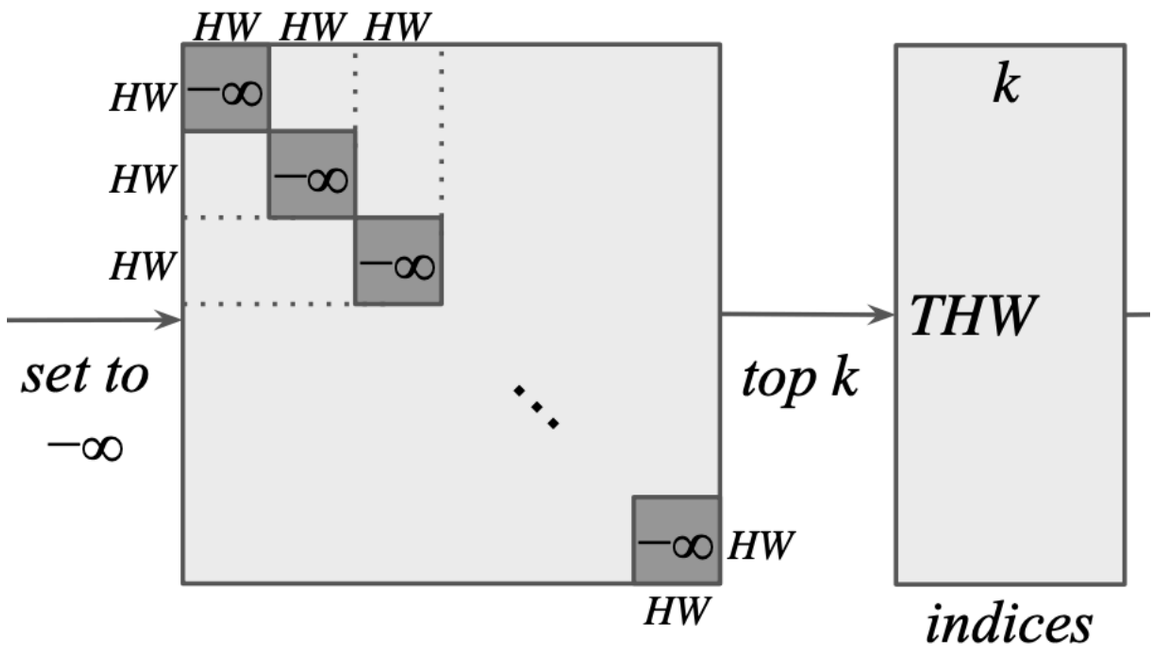


Figure 2: CP module architecture. Gray boxes denote tensors, white boxes denote operators and orange boxes denote neural networks with trainable weights. The dashed box represents the Correspondence Embedding layer, whose architecture is illustrated in detail in Figure 3.

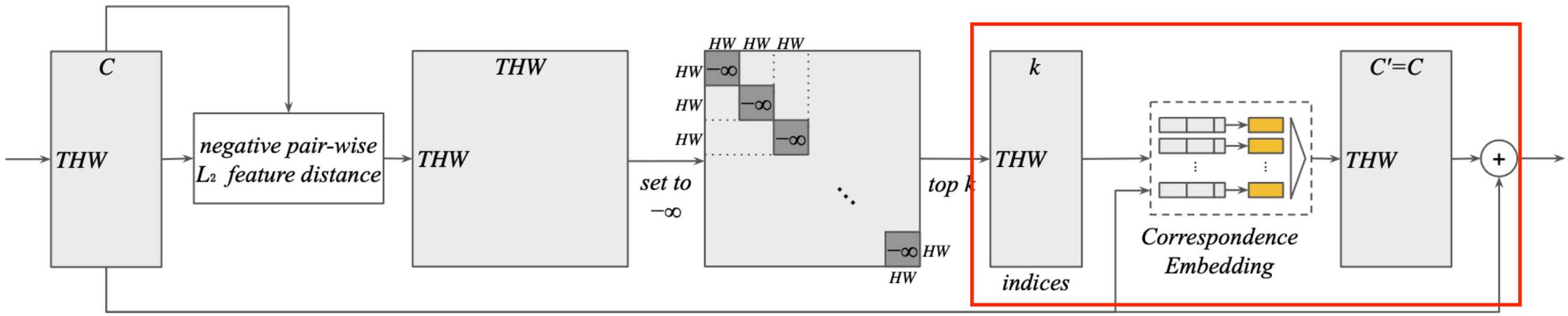


Select top k-nearest features

- 계산된 negative L2 Dist Matrix에서 각 행마다 값이 큰 top k개의 feature를 선정

- 선택된 각 feature는 현재 선택된 feature에 대응 후보로 여겨질 수 있음.

CP Module : Correspondence Embedding



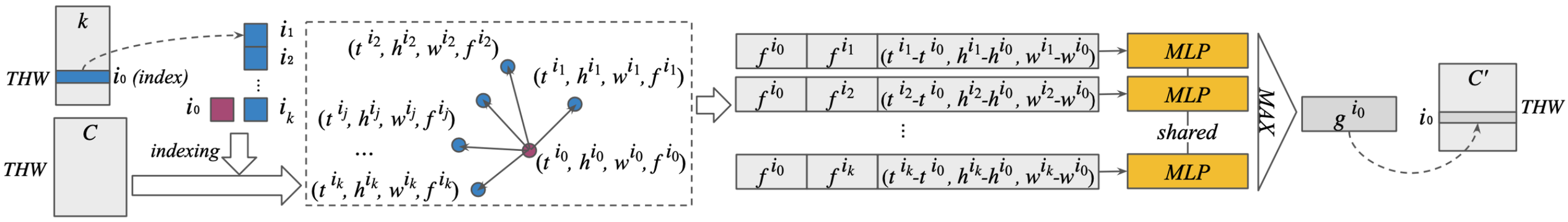
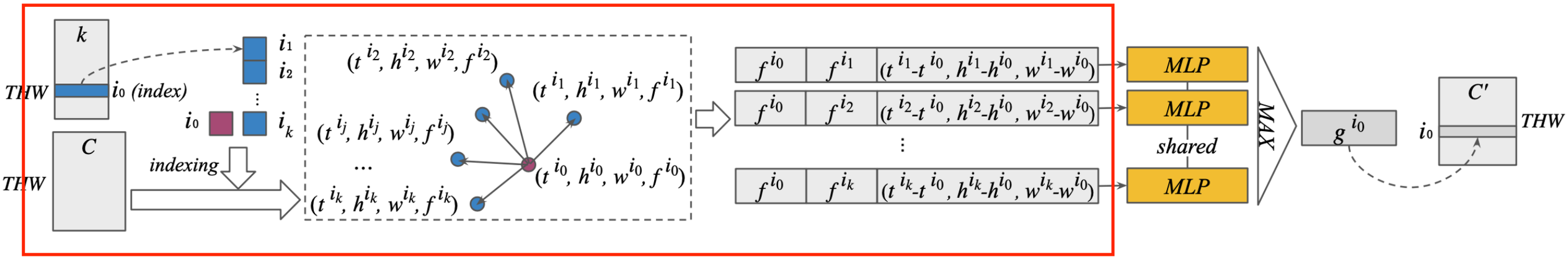


Figure 2: CP module architecture. Gray boxes denote tensors, white boxes denote operators and orange boxes denote neural networks with trainable weights. The dashed box represents the Correspondence Embedding layer, whose architecture is illustrated in detail in Figure 3.

- Goal : 선택된 Correspondence Proposals의 feature들에 대해서 Representations을 학습시키기 위한 Layer
- 다른 frame 상의 대응하는 위치로의 feature의 모션을 학습할수 있게 됨.

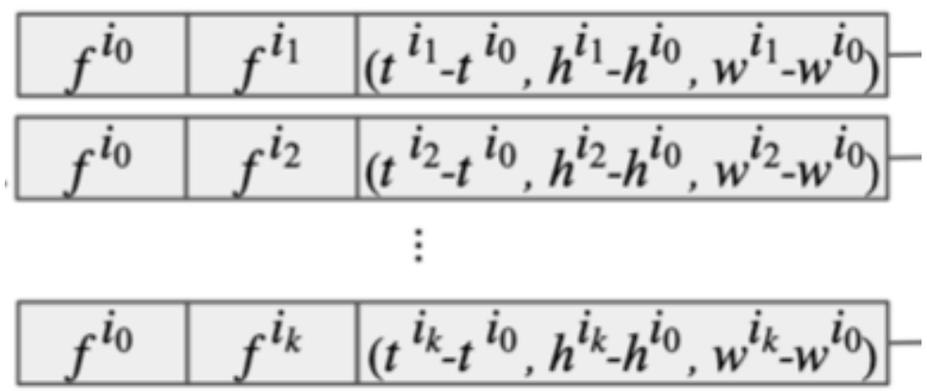
CP Module : Correspondence Embedding



- Top k-1 의 경우 한 시점의 대응 관계만을 찾기 때문에 비디오 전체 frame 에 대한 representation이 고려되지 못할 수 있음. 따라서 논문에서는 큰 수의 k를 사용함

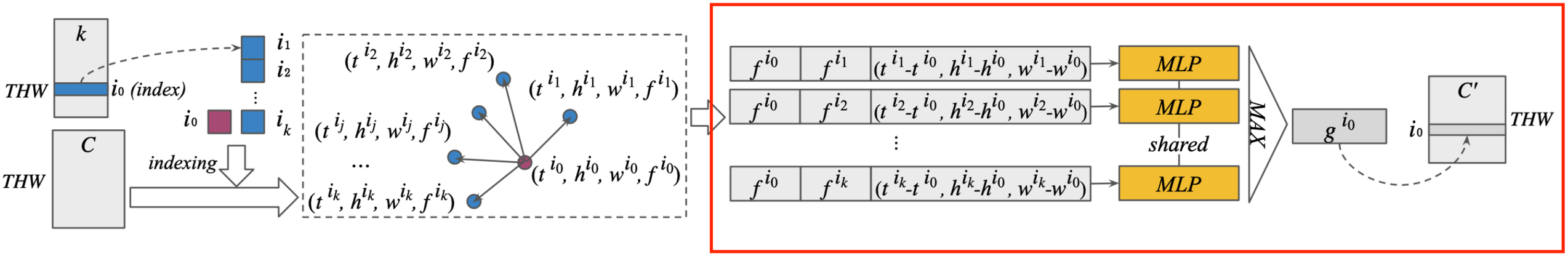
- Input feature i_0 : $(t^{i0}, h^{i0}, w^{i0}, f^{i0})$
  j-th k-NN. : $(t^{ij}, h^{ij}, w^{ij}, f^{ij})$ where $j \in \{1,2,\ldots,k\}$

- -> spatiotemporal displacements
  $[t^{ij} - t^{i0}, h^{ij} - h^{i0}, w^{ij} - w^{i0}] \in R^3$



t, h, w는 각각 [0 - 1]로 normalize

CP Module : Correspondence Embedding



- k개의 차분된 feature들은 가중치를 공유하는 MLP 네트워크에 입력된다.

- k개의 MLP 출력은 각 element-wise 로 Max 값을 찾아 최종 feature g 를 얻는다.

$$g^{i_0} = \underset{j \in \{1,2,\ldots,k\}}{MAX} \{\zeta(f^{i_0}, f^{i_j}, t^{i_j} - t^{i_0}, h^{i_j} - h^{i_0}, w^{i_j} - w^{i_0})\}$$

- CP Module은 Convolutional 네트워크의 중간 레이어로 삽입될 수 있으며, 논문에서는 ResNet의 Residual Block의 RELU 전에 사용함.
  == CPNet

A Failing of Several Previous Methods

Figure 4: An “up” example in our toy dataset.

- Long-range Temporal한 feature의 모션 학습이 가능한지에 대한 검증을 위해 Toy Dataset으로 기존의 방법들과 비교 수행

- Toy Dataset
  - 32x32 size 이미지 x 4 frame
  - 2x2 size의 흰 점 : 위치 변화에 따라 left, right, up, down 클래스
    : 이동 거리는 frame마다 임의의 7~9픽셀
  - Training set : 1000 / Validation set : 200

A Failing of Several Previous Methods

Table 1: Architectures for toy experiment

layer	I3D NL Net [33]	ARTNet [32]	TRN [39]	C2D CPNet (**ours**)
$conv_1$	$3 \times 3 \times 1$,16	$3 \times 3 \times 3$,16	$3 \times 3 \times 1$,16	$3 \times 3 \times 1$,16
	NL block	-	-	CP module
$conv_2$	$1 \times 1 \times 3$,16 $3 \times 3 \times 1$,16	SMART- $3 \times 3 \times 3$,16	$3 \times 3 \times 1$,16	$3 \times 3 \times 1$,16
	pooling, fc	pooling, fc	pooling, temporal relation, fc	pooling, fc
train	27.8	26.8	27.1	**97.9**
val	26.4	25.9	26.9	**97.4**

- Receptive Fields : 모두 5x5가 되게끔 설정 (동일한 조건에 각 방법들의 특징 모듈 사용)
- CPNet은 overfitting(학습 가능) 되었으나, 다른 방법들은 거의 26~27의 정확도로 random에 가까운 성능(1 / 4)으로 학습 실패
- ARTNet, TRN : 작은 Receptive Fields 로 인해 실패
- I3D NL Net은 NL block이라는 global receptive field를 갖지만, **NL block이 positional information을 갖는데 실패**하였기 때문에 long-range motion에 대한 학습에 실패함
- 논문에서는 NL block에 메모리나 연산량을 크게 증가시키지 않고 pairwise positional information을 반영하는 것이 쉽지 않다는 점에서, CPNet의 이점을 강조함.
  : CPNet과 NL Net 모두 THW x THW 크기의 distance matrix를 구하는 것은 갖지만, 그 다음 출력 Matrix를 만들기 위해 CP 모듈은 추가적인 행렬곱 없이 k개의 feature를 추출하고 이로부터 output matrix를 만듦.

Experiment Results

- Ablation studies

(a) number of CP modules

model	top-1	top-5
C2D	56.9	79.5
1 CP	60.3	82.4
2 CPs	60.4	82.4
4 CPs	61.0	**83.1**
6 CPs	**61.1**	**83.1**

(c) CP module positions

model	top-1	top-5
C2D	56.9	79.5
res_3	60.4	82.4
res_4	**60.8**	**82.8**
res_5	59.2	81.6

(b) Ablation on CP module's k values used in training and testing time.

top-1/top-5 accuracy		test					
		$k=1$	$k=2$	$k=4$	$k=8$	$k=16$	$k=32$
train	$k=1$	**59.9/82.3**	59.2/81.6	56.6/79.4	52.5/76.1	49.0/72.6	44.6/58.5
	$k=2$	59.1/81.8	**60.2/82.5**	59.6/81.8	56.9/80.1	53.0/77.1	48.9/73.5
	$k=4$	59.0/81.2	60.2/82.4	**60.5/82.6**	59.0/81.7	55.3/79.2	49.2/73.5
	$k=8$	53.4/76.3	56.8/79.5	59.6/81.9	**60.7/82.8**	59.7/82.1	57.0/80.3
	$k=16$	51.3/75.1	53.8/77.3	56.8/79.7	59.8/82.1	**60.6/82.8**	59.2/81.8
	$k=32$	52.6/76.6	53.8/77.7	55.5/79.1	58.2/80.8	60.0/82.2	**60.4/82.4**

Experiment Results

- Comparison with Other Architectures

(d) Kinetics validation accuracy of architectures in Table 2. Clip length is 8 frames.

frame rate	1/12 of original frame rate				1/4 of original frame rate			
val configuration	1-clip, 1 crop		25-clip, 10 crops		1-clip, 1 crop		25-clip, 10 crops	
accuracy	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5
C2D	56.9	79.5	61.3	83.6	54.1	77.4	60.8	83.3
C3D [28]	58.3	80.7	64.4	85.8	55.0	78.5	63.3	85.2
NL C2D Net [33]	58.6	81.3	63.3	85.1	55.3	78.6	62.1	84.2
ARTNet [32]	59.1	81.1	65.1	86.1	56.1	78.7	64.2	85.6
CPNet (Ours)	**61.1**	**83.1**	**66.3**	**87.1**	**57.2**	**80.8**	**64.9**	**86.5**

- Large Models on Kinetics

Table 4: Large RGB-only models on Kinetics validation accuracy. Clip length for NL Net and our CPNet is 32 frames.

model	params (M)	top-1	top-5
I3D Inception [3]	25.0	72.1	90.3
Inception-ResNet-v2 [2]	50.9	73.0	90.9
NL C2D ResNet-101 [33]	48.2	75.1	91.7
CPNet C2D ResNet-101 **(ours)**	**42.1**	**75.3**	**92.4**

Experiment Results

- Results on Something-Something

(a) Something-Something v2 Results

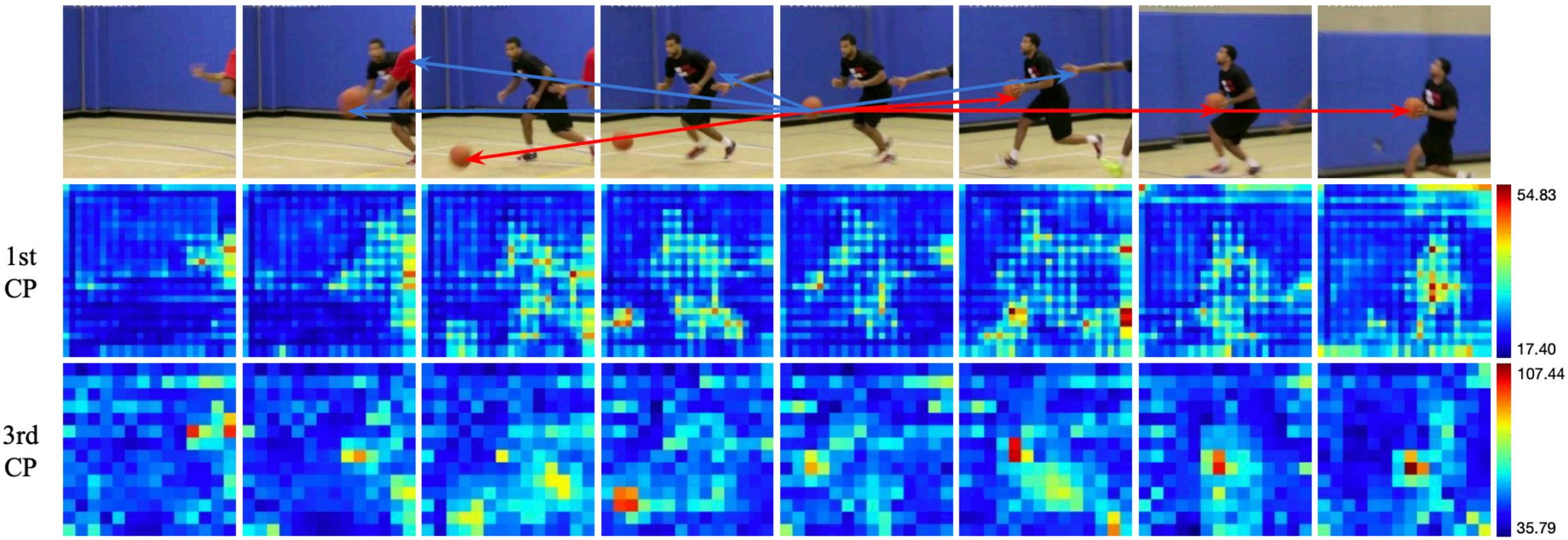
model	params (M)	val		test	
		top-1	top-5	top-1	top-5
Goyal et al. [12]	22.2	51.33	80.46	50.76	80.77
MultiScale TRN [39]	22.8	48.80	77.64	50.85	79.33
Two-stream TRN [39]	46.4	55.52	83.06	56.24	83.15
C2D Res18 baseline	10.7	35.24	64.49	-	-
C2D Res34 baseline	20.3	39.64	69.61	-	-
CPNet Res18, 5 CP **(ours)**	**11.3**	54.08	82.10	53.31	81.00
CPNet Res34, 5 CP **(ours)**	**21.0**	**57.65**	**83.95**	**57.57**	**84.26**

- Results on Jester

(b) Jester v1 Results

model	params (M)	val	test
BesNet [11]	37.8	-	94.23
MultiScale TRN [39]	22.8	95.31	94.78
TPRN [35]	22.0	95.40	95.34
MFNet [20]	41.1	96.68	96.22
MFF [19]	43.4	96.33	96.28
C2D Res34 baseline	20.3	84.73	-
CPNet Res34, 5 CP **(ours)**	**21.0**	**96.70**	**96.56**

Experiment Results

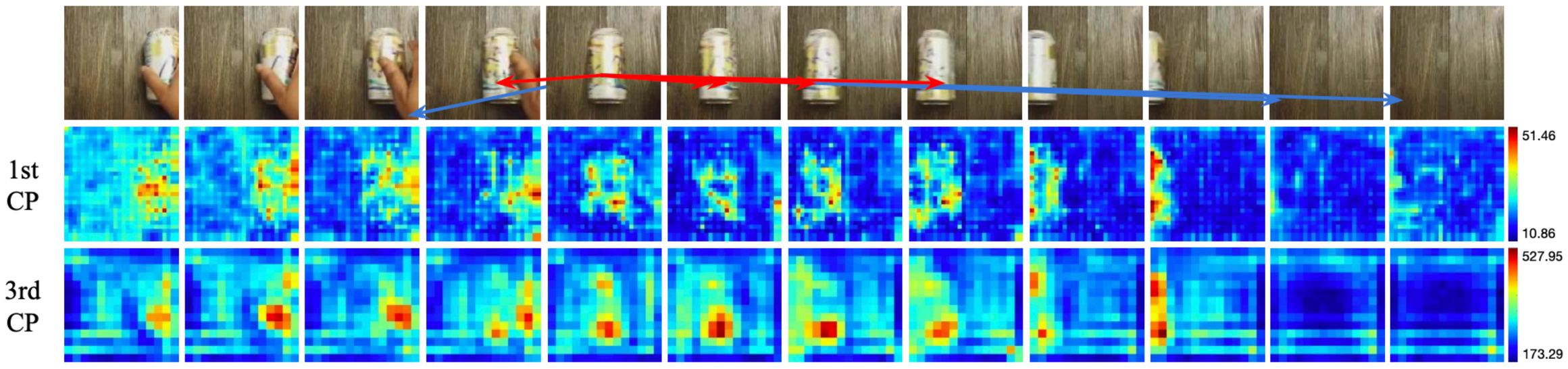


(a) A video clip with label “playing basketball” from Kinetics validation set.

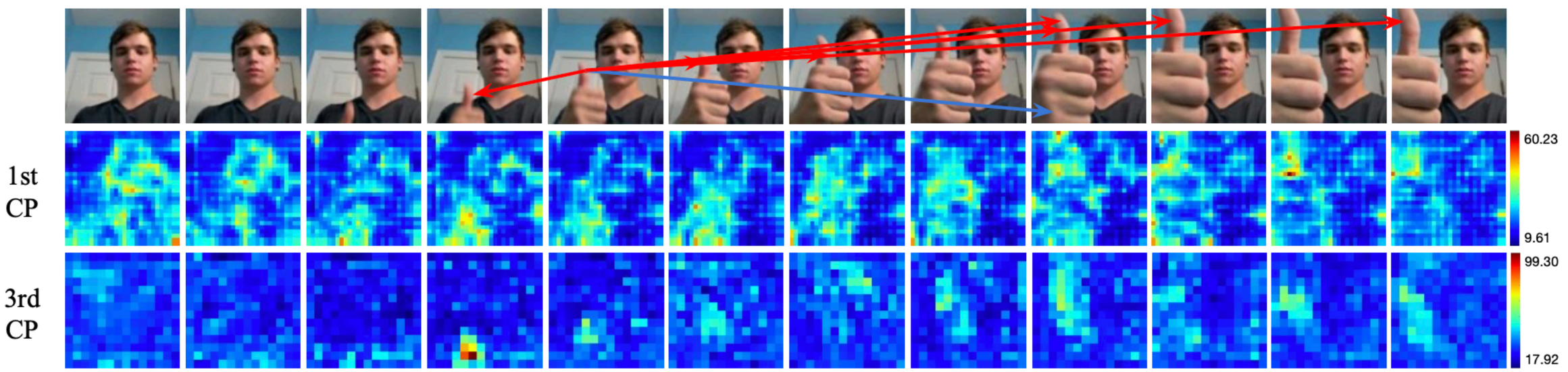
빨간 화살표 : 시작점 i_0에 대응하는 k개의 위치를 연결

파란 화살표 : 시작점 i_0에 대응하는 k개에 선택되지 않은 위치를 연결

Experiment Results



(b) A video clip with label “Rolling something on a flat surface” from Something-Something v2 validation set.



(c) A video clip with label “Thumb Up” from Jester v1 validation set.

빨간 화살표 : 시작점 i_0에 대응하는 k개의 위치를 연결

파란 화살표 : 시작점 i_0에 대응하는 k개에 선택되지 않은 위치를 연결