Region-based Non-local Operation for Video Classification

Guoxi Huang and Adrian G. Bors

Department of Computer Science, University of York, York YO10 5GH, UK
E-mail: {gh825, adrian.bors}@york.ac.uk

***Abstract*—Convolutional Neural Networks (CNNs) model long-range dependencies by deeply stacking convolution operations with small window sizes, which makes the optimizations difficult. This paper presents region-based non-local (RNL) operations as a family of self-attention mechanisms, which can directly capture long-range dependencies without using a deep stack of local operations. Given an intermediate feature map, our method recalibrates the feature at a position by aggregating the information from the neighboring regions of all positions. By combining a channel attention module with the proposed RNL, we design an attention chain, which can be integrated into the off-the-shelf CNNs for end-to-end training. We evaluate our method on two video classification benchmarks. The experimental results of our method outperform other attention mechanisms, and we achieve state-of-the-art performance on the Something-Something V1 dataset.**

I. Introduction

With the rapid development of the Internet, videos have become the main multimedia resource of information, and the analysis of video information is in high demand. Video classification attracts increasing research interest, given the numerous applications for this area. As Convolutional Neural Networks (CNNs) demonstrated high capability for learning visual representations in the image domain, it is natural to attempt to apply CNNs to the video area. An effective way to extend CNN from image to video domain is by changing the convolution kernels from 2D to 3D, aka 3D CNN [1], [2] or by adding recurrent operations to CNNs [3], [4].

The models based on convolutional or recurrent operations capture long-range dependencies by deeply stacking local operations with small window sizes. However, the deep stack of local operations limits the efficiency of message delivery to distant positions, and makes the optimization difficult [5], [6]. To mitigate the optimization difficulties, Wang *et al.* proposed the non-local (NL) operation [7] that works as a self-attention mechanism [8] to capture long-range dependencies directly by exploiting the inner-interactions between positions regardless of their positional distance, which we revisit in Section III-A. However, in the non-local operation, the calculation of the relation between two positions only relies on the information from these two positions while not fully utilizing the information around them. As a result, its calculation of positional relationships is not robust to noise or unrelated features, especially in high resolution, which has been emphasized in [9].

In this paper, we investigate the non-local operation [7] and propose a region-based non-local (RNL) operation based on the non-local mean concept [9], which enhances the calculation of positional relationships by fully utilizing the information from neighboring regions. The proposed RNL operation endows CNNs with a global view of input features without needing a deep stack of local operations to ease the optimization difficulties. In Figure 1, we illustrate an example to demonstrate that the proposed RNL operation can better capture positional relationships than NL operation. There are two advantages of the proposed RNL compared with the original NL: first of all, RNL is more robust to noise or unrelated features; secondly, the RNL is more computationally efficient. Meanwhile, we present various instantiations of the RNL operation to meet different application requirements. By adding RNL operation into the off-the-shelf CNNs, we obtain a new video classification architecture named region-based non-local network. In order to evaluate the effectiveness of our method, we conduct video classification experiments on two large-scale video benchmarks, Kinetics-400 [2] and Something-Something V1 [10]. Our models outperform the baseline and other popular attention mechanisms, and achieve state-of-the-art performance on Something-Something V1.

II. Related Work

Spatio-temporal Networks. With the tremendous success of CNNs on image classification tasks [5], [11]–[18]. Some research studies have attempted to extend the applications of CNNs to video-based classification tasks [2], [3], [19]–[21]. Among them, the two-stream model [19] and its variant [22] learn temporal evolution by using jointly the optical flow stream and the RGB stream for video classification. The recent video models [2]–[4], [20] leverage long short-term memory (LSTM) to fuse frame-level CNN representations for modeling long-term temporal relationships. However, 2D CNN+LSTM [2] empirically shows lower performance than two-stream architectures. CNNs employing 3D convolution processing [1], [2], [23] represent a promising research direction for spatio-temporal representation learning, but the training of 3D CNNs has huge computational demands. Some research studies have devoted to simplifying 3D CNNs, such as P3D [24], TSM [25], S3D [26], CSN [27], X3D [28]. Nevertheless, the inefficiency of message delivery caused by the deep stacking of local operations in 3D CNNs remains

(a) RNL attention maps (b) NL attention maps

Fig. 1. Examples of visualizing the attention maps of RNL and NL operations in the res4 stage of ResNet on a video clip from Kinetics-400. Given a reference position, an ideal non-local operation should only highlight the regions related to the reference position. In the same video clip, the NL operation has almost the same attention maps at different reference positions while the proposed RNL operation presents query-specific attention maps, which demonstrate that the proposed RNL operation can better compute the relationships between positions.

serious, and there is not much research on this problem, which is the main theme of this paper.

Attention Mechanisms. Attention mechanisms have been initally used for machine translation [29]. Recent works [7], [30]–[32] would embed task-specific attention mechanisms to CNNs to boost up performance and robustness in visual tasks. In computer vision, attention mechanisms can be decomposed into two components, channel attention - focusing on 'what' is meaningful, and spatial (or spatio-temporal) attention - focusing on 'where' is informative [32]. For example, The Squeeze-and-Excitation (SE) module is a representative channel attention mechanism, which utilizes global average-pooled features to exploit the inter-channel relationships. Inspired by the classic non-local mean algorithm [9] for image denoising, Wang, *et al.* [7] introduced the self-attention concept [8] from machine translation to large-scale visual classification tasks, and proposed non-local (NL) operation for video classification. The NL operation was initially designed to learn spatio-temporal attention. However, Cao *et al.* [33] observe that NL can only capture the global context of channels, aka channel attention. Moreover, they demonstrate that the intrinsic natures of the NL operation and SE module [30] are the same while the implementation of the SE module is rather economical.

In this paper, we redesign the non-local operation and propose the region-based non-local operation which increases the effectiveness and efficiency in capturing the spatio-temporal attention. Yue *et al.* [34] also aimed to improve the NL operation, proposing a compact generalized version of the NL operation by integrating channel attention and spatio-temporal attention into a compact module. However, their work do not improve the effectiveness of NL operation. Instead of simplifying the NL, we focus on improving the effectiveness of NL for better capturing the spatio-temporal attention.

III. Non-local Methods for Video Classification

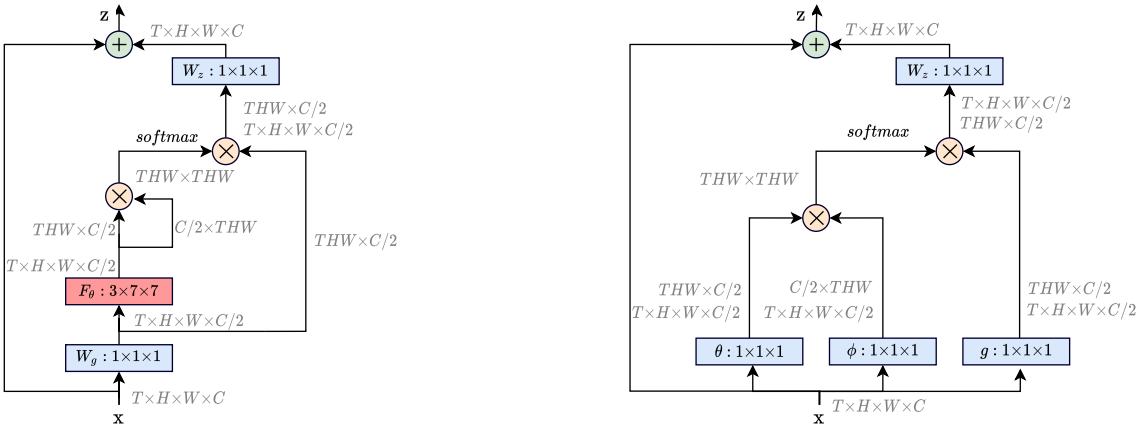
A. Revisiting the Non-local (NL) Operation

Intuitively, the non-local operation [7], illustrated in Figure 2 (b), strengthens the feature in a certain position via aggregating the information from other positions. The estimated value for a position, is computed as a weighted sum of the feature values of all other positions. Formally, we denote $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{THW \times C}$ as the input and output of an NL operation, flattened along the space-time directions, where T, H, W and C are temporal length (depth), height, width and the number of channels, respectively. Then, the NL operation can be described as:

$$\begin{aligned} \mathbf{y}_i &= \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} w_{i,j} \mathbf{W}_g \mathbf{x}_j, \\ w_{i,j} &= f(\mathbf{x}_i, \mathbf{x}_j), \end{aligned} \tag{1}$$

where $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^C$ are the i-th and j-th element of $\mathbf{x}$, i is the index of a reference position, and j enumerates all possible positions. $\mathbf{W}_g$ is a learnable weight matrix that computes a representation of $\mathbf{x}_j$, and $\mathcal{C}(\mathbf{x})$ is the normalization factor. Meanwhile, $w_{i,j}$ is a weight, representing the relationship between positions i and j, which is calculated by pairwise similarity function $f(\cdot, \cdot)$. Regarding the form for $f(\cdot, \cdot)$, Wang *et al.* [7] propose four instantiations for the non-local operation, of which the embedded Gaussian form is described as $f(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta(\mathbf{x}_i)^{\mathrm{T}} \phi(\mathbf{x}_j)}$, $\mathcal{C}(\mathbf{x}) = \Sigma_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j)$, where θ and ϕ represent linear transformations, implemented with $1 \times 1 \times 1$ convolutions.

Attention Maps of the Non-local Operation. In the NL operation, each output element $\mathbf{y}_i$ is a weighted average of the input features over all positions $\mathbf{x}_j$, and therefore each $\mathbf{y}_i$ has a corresponding attention weight map calculated by $f(\cdot, \cdot)$, highlighting the areas related to position i. In Figure 1 (b), we randomly pick one video from Kinetics-400 and visualize the attention maps of NL at two different reference positions, one



(a) RNL block

(b) NL block [7]

Fig. 2. Diagrams of implementing the NL and RNL operations in (b) and (a), respectively, indicating the shaping and the reshaping operations of a tensor together with the connections. $\otimes$ denotes matrix multiplication while $\oplus$ denotes element-wise addition. The blue boxes denote $1 \times 1 \times 1$ convolutions, and the red box F_θ denotes a $3 \times 7 \times 7$ channel-wise separable convolution or an average/max pooling layer.

of which is located in the background area while the other is located in the region of the moving object. In the original NL operation, its attention maps with different reference positions are almost the same, which indicates that this fails to capture the positional relations. The NL operation realistically learns channel-wise attention rather than spatio-temporal attention.

We redesign the non-local operation as a spatio-temporal attention mechanism, namely the region-based non-local operation (RNL). Figure 1 (a) shows that our RNL operation only highlights the regions related to the reference position, which indicates that the proposed RNL operation can effectively learn spatio-temporal attention.

B. Region-based non-local (RNL) Operation

The initial idea for the RNL operation is that the relation between two positions in a video representation should not rely on just their own features but also on those features from their neighborhoods. Therefore, for each position i of input sample $\mathbf{x}$, we define a cuboid region $\mathcal{N}_i$ of fixed size centered at position i. The calculation of the relationship $w_{i,j}$ between positions i and j is redefined as:

$$w_{i,j} = f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j)), \tag{2}$$

where, $\theta(\cdot)$ denotes an information aggregation function that separately summarizes the features in a region for each channel. Function $\theta(\cdot)$ is given by

$$\theta(\mathcal{N}_i) = \sum_{k \in \mathcal{N}_i} \mathbf{u}_k \odot \mathbf{x}_k, \tag{3}$$

where $\odot$ denotes element-wise multiplication and $\mathbf{u}_k$ denotes a vector shared by all cuboid regions $\mathcal{N}_i$. As there is no channel interaction in $\theta(\cdot)$, it can be implemented as channel-wise [1] separable convolutions [35], or as average/max pooling. By replacing the expression of $w_{i,j}$ from equation (1) with the expression from (2), the RNL operation can be written as:

$$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j)) \mathbf{x}_j. \tag{4}$$

From equation (4), we can see that by employing the RNL operation, the new feature of each position is a weighted sum of the old features from all positions, where the weights are calculated by the similarity function $f(\cdot, \cdot)$ according to the similarity between the target region, and all the other regions. The proposed RNL operation enhances the calculation of positional relations by fully utilizing the information from the neighboring regions, which increases the robustness to noise or unrelated features, Hence, the RNL operation can learn more meaningful representations in comparison with NL.

For the form of function $f(\cdot, \cdot)$, in addition to adopting the Gaussian version and the Dot product version as in [7], we also propose a new form, called the Cosine version. Specifically, the **Gaussian** form of $f(\cdot, \cdot)$ is given by

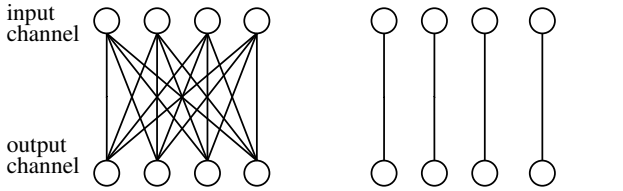
$$f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j)) = e^{\theta(\mathcal{N}_i)^\mathsf{T} \theta(\mathcal{N}_j)}. \tag{5}$$

The **Dot product** form of $f(\cdot, \cdot)$ measures the relation between two regions by using the dot-product similarity:

$$f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j)) = \theta(\mathcal{N}_i)^\mathsf{T} \theta(\mathcal{N}_j). \tag{6}$$

However, the dot-product similarity takes into account both the vector angle and the magnitude, as $\theta(\mathcal{N}_i)^\mathsf{T} \theta(\mathcal{N}_j) = \|\theta(\mathcal{N}_i)\| \|\theta(\mathcal{N}_j)\| \cos \psi_{i,j}$, where $\psi_{i,j}$ is the angle between vectors $\theta(\mathcal{N}_i)$ and $\theta(\mathcal{N}_j)$. It is preferable to replace dot-product similarity with the cosine similarity, ignoring the

[1] Also referred to as "depth-wise". We use the term "channel-wise" to avoid confusions with the network depth.



(a) conventional convolution (b) channel-wise separable convolution

Fig. 3. Illustrations of the conventional convolution (a) and the channel-wise separable convolution (b). The total number of connections of the channel-wise separable convolution [35] is reduced to $\frac{1}{C}$ of that of the conventional convolution.

vector magnitude and resulting in a value within the range $[-1, 1]$. The **Cosine** form of $f(\cdot, \cdot)$ is expressed as:

$$f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j)) = \mathrm{ReLU}(\frac{\theta(\mathcal{N}_i)^{\mathsf{T}}\theta(\mathcal{N}_j)}{\|\theta(\mathcal{N}_i)\|\|\theta(\mathcal{N}_j)\|}) = \mathrm{ReLU}(\cos\psi_{i,j}). \quad (7)$$

When $f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j)) < 0$, it indicates that the features in positions i and j are not related. As the new feature in a cetrain position should only be determined by those related features, we use the ReLU function to restrict the output of $f(\cdot, \cdot)$ to be non-negative. The normalization factor is set as $\mathcal{C}(\mathbf{x}) = \sum_{\forall j} f(\theta(\mathcal{N}_i), \theta(\mathcal{N}_j))$ for the Gaussian version from (5), and set as $\mathcal{C}(\mathbf{x}) = THW$ for the Dot-product and Cosine versions from equations (6) and (7), respectively.

C. Region-based non-local Block

In order to embed the RNL operation into the off-the-shelf CNNs without influencing the results provided by the pre-trained kernels, we embed the RNL operation into a residual style block [5], named the RNL block. The Gaussian RNL block, defined by (5), is written as a matrix form as:

$$\mathbf{z} = \mathbf{y}\mathbf{W}_z + \mathbf{x}, \quad (8)$$

$$\mathbf{y} = softmax(F_\theta(\mathbf{x}\mathbf{W}_g)(F_\theta(\mathbf{x}\mathbf{W}_g))^{\mathsf{T}})\mathbf{x}\mathbf{W}_g, \quad (9)$$

where $\mathbf{z}$ is the output that represents the feature after recalibration, $\mathbf{W}_z \in \mathbb{R}^{\frac{C}{2}\times C}$ and $\mathbf{W}_g \in \mathbb{R}^{C\times\frac{C}{2}}$ are learnable weight matrices, which are implemented as $1 \times 1 \times 1$ convolutions, and '$+\mathbf{x}$' denotes a residual term. F_θ denotes the operation that corresponds to the matrix form of function $\theta(\cdot)$ from equation (3). We present the architectures of the Gaussian RNL block and the Gaussian embedding version of the original NL block in Figure 2. We can observe that the original NL block illustrated in Figure 2 (b) uses four $1 \times 1 \times 1$ convolutions, while the proposed RNL block shown in Figure 2 (a) uses two $1 \times 1 \times 1$ convolutions and one channel-wise separable convolution, which reduces the computational complexity significantly.

Next, we explain two main implementations of the region information aggregation function F_θ in RNL operation.

1) Channel-wise Separable Convolutions. It is worthwhile to note that, in principle, the candidates for implementing F_θ should not fuse together information across channels. Otherwise, the new feature embedding might fail to represent its original information, which is why we cannot adopt conventional convolutions. In contrast, channel-wise separable convolution [35], exemplified in Figure 3, is a perfect candidate for the implementation of F_θ, as there is no interaction between the channels. An additional benefit that the channel-wise separable convolution brings is that it reduces the computation and the parameters by a factor of C, compared with the conventional convolution. The kernel size of the channel-wise separable convolution has a significant impact on performance, as it corresponds to how large a region $\mathcal{N}_i$ is considered for information aggregation. We will explore the effectiveness of various kernel sizes, in Section IV-A.

2) Average/Max Pooling. The other implementation options for F_θ are the average pooling and max pooling, which have been widely adopted for information aggregation. Although it shows a relatively weaker capability than the implementation of channel-wise separable convolution, average/max pooling adds no extra parameters to the models.

TABLE I
THE ARCHITECTURE OF THE RNL NETWORK. THE KERNEL SIZE AND THE OUTPUT SIZE ARE SHOWN IN THE SECOND AND THIRD COLUMNS, RESPECTIVELY. THE RNL BLOCKS ARE INSERTED AFTER THE RESIDUAL BLOCKS SHOWN IN BRACKETS, WHERE THE TEMPORAL SHIFT MODULES [25] ARE EMBEDDED INTO THE CONVOLUTIONAL LAYERS.

Layer	Operation	Output size
conv1	$1 \times 7 \times 7$, 64, stride 1,2,2	$8 \times 112 \times 112$
pool1	$1 \times 3 \times 3$, 64, stride 1,2,2	$8 \times 56 \times 56$
res2	$\begin{bmatrix} 1\times1\times1, 64 \\ 1\times3\times3, 64 \\ 1\times1\times1, 256 \end{bmatrix} \times 3$	$8 \times 56 \times 56$
res3	$\begin{bmatrix} 1\times1\times1, 128 \\ 1\times3\times3, 128 \\ 1\times1\times1, 512 \end{bmatrix}_{\mathrm{RNL}} \times 4$	$8 \times 28 \times 28$
res4	$\begin{bmatrix} 1\times1\times1, 256 \\ 1\times3\times3, 256 \\ 1\times1\times1, 1024 \end{bmatrix}_{\mathrm{RNL}} \times 6$	$8 \times 14 \times 14$
res5	$\begin{bmatrix} 1\times1\times1, 512 \\ 1\times3\times3, 512 \\ 1\times1\times1, 2048 \end{bmatrix} \times 3$	$8 \times 7 \times 7$

D. Attention Chain

When the proposed RNL block can learn the long-range dependencies for each position in the spatio-temporal dimension, the squeeze-excitation (SE) block [30] can learn the long-range dependencies in the channel dimension. In order to capture both spatio-temporal attention and channel-wise attention in a single module, we embed the SE block [30] together with the RNL block to form an attention chain module (SE+RNL). Firstly, we modify the SE block [30], where the squeeze operation F_{sq} is expressed as:

$$\mathbf{s}' = F_{sq}(\mathbf{x}) = \frac{1}{THW}\sum_{i=1}^{THW}\mathbf{x}_i, \quad (10)$$

and the excitation operation F_{ex} is expressed as:

$$\mathbf{s} = F_{ex}(\mathbf{s}') = \mathbf{W}_2\mathrm{ReLU}(\mathrm{BN}(\mathbf{W}_1\mathbf{s}')), \quad (11)$$

where $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{2}\times C}$ and $\mathbf{W}_2 \in \mathbb{R}^{C\times\frac{C}{2}}$ are learnable weights, which can be implemented with fully-connected (FC) layers.

In the excitation operation F_{ex}, we add a batch normalization (BN) layer [36] right after the FC layer $\mathbf{W}_1$ to reduce the internal covariate shift. Subsequently, we reshape $\mathbf{s} \in \mathbb{R}^C$ into $\mathbb{R}^{1\times C}$. The output of the SE block is given by :

$$\mathbf{v} = \mathbf{x} \oplus \mathbf{s}, \tag{12}$$

where $\oplus$ refers to the element-wise addition broadcasting in unmatched dimensions (replicate $\mathbf{x}$ to match the dimension of $\mathbf{s}$). After that, we place the RNL block after the SE block to form an attention chain.

E. The Network Architecture

The RNL block is designed to be compatible with most existing CNNs. It can be plugged into a CNN at any processing stage, resulting in an RNL network. For the implementation, we use ResNet-50 [5] with the temporal shift modules (TSM) [25] as the backbone network to build our model (RNL TSM), and its structures is provided in Table I. The TSM is a lightweight module enabling 2D CNNs to achieve temporal modeling by shifting part of the channels along the temporal dimension, which facilitates the information exchange among neighboring frames. In this architecture, we keep the temporal size constant, which means all the layers in the network only reduce the spatial size of the input features. The backbone network is also the baseline for our experiments.

IV. Experiments

We perform video classification experiments on two standard video benchmarks, Kinetics-400 [2] and Something-Something V1 [10]. Kinetics-400 is a large-scale video classification benchmark that consists of ∼300K video clips, classified into 400 categories. Something-Something V1 consists of ∼108K videos from 174 categories. We report Top-1, Top-5 accuracy on the validation sets and the computational cost (in GFLOPs) of a single, spatially center-cropped clip to comprehensively evaluate the effectiveness and efficiency. Figure 1 and Figure 5 visualize some examples of the attention maps of RNL operation, which shows RNL operation can correctly learn the relations between positions.

Training and Inference. Our models are pretrained on ImageNet [37]. For the training, we follow the setting from [7] and use a spatial size of 224 × 224, which is randomly cropped from a resized video frame. The temporal size is set as 8 frames unless otherwise specified. In order to prevent overfitting, we add a dropout layer after the global pooling layer. We optimize our models using the Stochastic Gradient Descent, and train the models for 50 epochs with a cosine decay learning rate schedule. The batch size is set at 64 across multiple GPUs. For Kinetics, the initial learning rate, weight decay and dropout rate are set to 0.01, 1e-4 and 0.5 respectively; for Something-Something, these hyper-parameters are set to 0.02, 8e-4, and 0.8 respectively. In the inference, we follow the common setting in [7], [25]. Unless stated otherwise, we uniformly sample 10/2 clips for Kinetics-400/Something-Something V1, and perform spatially fully convolutional inference (three crops of size 256 × 256 to cover the spatial dimensions) for all clips, and the video-level prediction is obtained by averaging all the clip prediction scores of a video.

A. Ablation Studies

We explore the most efficient and effective form of RNL operation on Kinetics-400. By default, the function $f(\cdot,\cdot)$ of RNL operation is implemented by using the equation (5), and F_θ is implemented by a channel-wise separable convolution with a kernel size of $3 \times 7 \times 7$, unless otherwise specified. Following the results from [7], we add RNL blocks to the res3 and res4 stages in the architecture shown in Table I. Our exploration is organized in three parts. First, we search for the effective kernel size of F_θ in RNL blocks. Next, we evaluate the performance of various instantiations of RNL and find out the efficient and effective one. Finally, we combine the selected version of RNL with an SE block to form an attention chain module.

Kernel Size. The kernel size of F_θ (determining the size of region $\mathcal{N}_i$) in the RNL block has a significant impact on the performance as it affects what the RNL operation would learn. Large kernels are supposed to be robust to noise, while small kernels would consider the details and fine structures from video sequences. By considering that the features learned by the kernel from the temporal and spatial dimensions are different, we separately explore the temporal and spatial sizes of the kernel by fixing one while varying the other. The results are shown in Table II (a). We observe that in the temporal dimension, the size of 3 surpasses other options regardless of the spatial size of the kernel, while in the spatial dimension, the size of 7 is the best option. Therefore, we expect the kernel of $3\times7\times7$ is the best option in space and time, and it has been verified through our grid search. Concurrently, we evaluate the influence of the kernel size of F_θ to the model performance by visualizing the attention maps of the RNL operation, shown in Figure 4, where the RNL operation considers the highlighted areas to have strong relations with the reference position, indicated by a red point. Figure 4 shows that a kernel of a small size spatially, such as 1×1, tends to incorrectly interpret the relations between some background areas and the foreground areas. In contrast, a kernel with larger spatial size can learn more precise relations between such positions. For example, the kernel of size 7×7 precisely highlights the moving object in in Figure 4 when the reference position is located at the moving object. However, too large kernels could also lead to performance degradation. For example, the kernel of size $3\times9\times9$ has a lower accuracy than the kernel of size $3\times7\times7$ (73.51% vs. 73.66%), and the kernel of $7 \times 7 \times 7$ shows a lower performance than the kernel of size $3 \times 7 \times 7$ (73.11% vs. 73.66%). The kernel of size $1\times1\times1$ has a lower accuracy than the others except for $7\times1\times1$ and $7\times7\times7$, which verifies our assumption that the relation between two positions should not rely on just their own features but also on features from their neighborhoods.

Instantiations. There are various solutions for $f(\cdot,\cdot)$ from equation (4) and for F_θ from equation (9), as discussed in

Fig. 4. Visualization the attention maps of the RNL block when considering different kernel sizes in the res3 stage by giving the reference position (red point). When the reference point is located at the moving object, the RNL operation with proper kernel size should just highlight the related moving regions.

TABLE II
EXPLORATION OF THE EFFECTIVENESS AND EFFICIENCY OF VARIOUS RNL MODULES ON KINETICS-400. FOR THE MODELS IN (A) AND (C), WE INSERT ONE GAUSSIAN RNL BLOCK INTO THE RES3 STAGE OF RESNET-50.

Kernel size	Top-1 (%)	Kernel size	Top-1 (%)
$1 \times 1 \times 1$	73.28	$3 \times 3 \times 3$	73.53
$3 \times 1 \times 1$	73.41	$3 \times 5 \times 5$	73.27
$7 \times 1 \times 1$	73.12	$\mathbf{3 \times 7 \times 7}$	**73.66**
$1 \times 3 \times 3$	73.32	$3 \times 9 \times 9$	73.51
$1 \times 7 \times 7$	73.43	$7 \times 7 \times 7$	73.11
$1 \times 9 \times 9$	73.32	$7 \times 9 \times 9$	73.30

(a) RNL blocks with different kernel sizes of F_θ.

# RNL	Method($f(\cdot,\cdot)$)	Top-1 (%)
1	Dot-product	73.22
	Gaussian	**73.66**
	Cosine	73.46
5	dot-product	74.16
	Gaussian	**74.68**
	Cosine	74.40

(b) Instantiations of the RNL with different form of $f(\cdot,\cdot)$.

Method (F_θ)	Top-1 (%)	GFLOPs	Params
channel-wise conv	73.66	1.65	2.67M
average pooling	73.22	1.65	0.26M
max pooling	73.47	1.65	0.26M

(c) Instantiations of RNL with different implementations of F_θ.

Section III-B and Section III-C, respectively. In the following, we conduct ablation studies on the instantiations by fixing a specific choice for either $f(\cdot,\cdot)$ or F_θ while changing the other. The operation F_θ can be implemented as a channel-wise separable convolution or as the average/max pooling, the stride of which is set as 1, and the padding of which is half of the kernel size. From the results shown in Table II (c), we can see that the channel-wise separable convolution implementation achieves a higher accuracy with +0.44% and +0.19% than the average and max pooling, respectively. However, the implementation of average/max pooling is more efficient and adds fewer parameters (-2.4M) to the model compared to the channel-wise separable convolution. We instantiate three versions of the RNL operation, such as Gaussian, Dot-product and Cosine, provided in equations (5), (6) and (7) respectively. The results are shown in Table II (b). By adding a single RNL block into the backbone network, the result of the Gaussian RNL outperforms the Dot-product and Cosine versions. Moreover, the performance of all installations of the RNL operation can be further improved by stacking more RNL blocks. The model with 5 Gaussian RNL blocks (3 in the res4 stage and 2 in the res3 stage) gains an additional 1.02% accuracy increase in comparison with adding a single RNL block.

B. Evaluation

In order to evaluate the efficiency and effectiveness of our method in comparison with other attention mechanisms, we reimplement the original NL network [7], GCNet [33] (a simplified NL network), SE network [30] and CBAM network [32]. Table III presents the results on Kinetics and Something-Something. We can see that the proposed RNL block achieves higher performance than other attention mechanisms. Notably, the network with 5 RNL blocks outperforms the network

TABLE III
COMPARISONS BETWEEN VARIOUS VISUAL ATTENTION MECHANISMS ON KINETICS-400 AND SOMETHING-SOMETHING V1.

Dataset	Model	Top-1 (%)	FLOPs (G)	# Param (M)
Kinetics-400	baseline	72.80	32.89	24.33
	+ 5 SE	73.70	32.89	24.79
	+ 5 CBAM	73.99	32.90	24.80
	+ 5 GC	73.76	32.90	24.79
	+ 5 NL	74.41	49.38	31.69
	+ 5 RNL	**74.68**	**41.15**	**35.48**
	+ 5 [SE+RNL]	**74.97**	**41.16**	**35.95**
Something-Something V1	baseline	46.63	32.89	24.33
	+ 5 NL	48.25	49.38	31.69
	+ 5 RNL	**49.24**	**41.15**	**35.48**
	+ 5 [SE+RNL]	**49.47**	**41.16**	**35.95**

with 5 NL blocks with +0.27% on Kinetics and +1% on Something-Something, while the computational complexity required in FLOPs of the RNL network is 8.23G less than that of the NL network. Furthermore, by adding 5 blocks of the attention chain (SE + RNL), as described in Section III-D, to the backbone network, the performance is further improved (74.97% on Kinetics and 49.47% on Something-Something). In the visualization examples of the RNL and NL blocks, shown in Figure 1, we observe that the attention maps of the RNL block would only highlight those regions related to the reference positions. However, the attention maps of the original NL block always highlight the same regions for different reference positions. The observation demonstrates that the RNL block can capture the spatio-temporal attention while the NL block only captures the channel attention.

C. Comparisons with the State-of-the-Art

We compare the proposed method with the state-of-the-art methods on Kinetics-400 and Something-Something V1. In

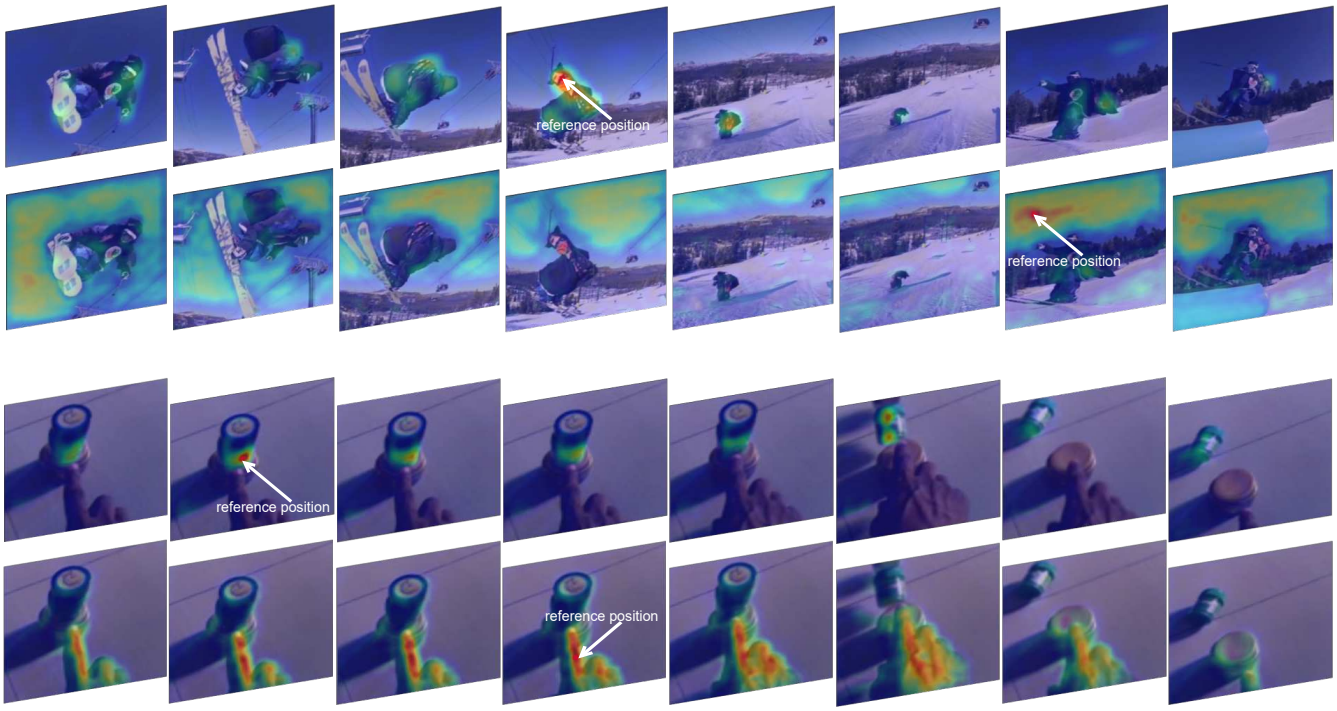


Fig. 5. Visualization of attention maps of the RNL in the res3 stage, with different reference positions on frames from Kinetics (1st row) and Something-Something (2nd row). Given a video clip, the RNL operation only highlights those regions related to the reference position.

TABLE IV
RESULTS ON KINETICS-400.

Model	Backbone	Training Frames	Top-1	Top-5
I3D RGB [2]	Inception	64	72.1	90.3
S3D-G RGB [26]	Inception	64	74.7	93.4
TSM [25]	ResNet-50	8	74.1	91.2
TSM [25]	ResNet-50	16	74.7	-
NL I3D [7]	ResNet-50	32	74.9	91.6
Slow [38]	ResNet-50	8	74.9	91.5
SlowFast [38]	ResNet-50	4+32	75.6	92.1
RNL TSM (ours)	**ResNet-50**	**8**	**75.6**	**92.3**
RNL TSM (ours)	**ResNet-50**	**16**	**77.2**	**93.1**
RNL TSM$_{En}$ (ours)	**ResNet-50**	**8+16**	**77.4**	**93.2**
NL I3D [7]	ResNet-50	128	76.5	92.6
NL I3D [7]	ResNet-101	128	77.7	93.3
SlowFast [38]	ResNet-101	16+64	78.9	93.5
LGD-3D RGB [39]	ResNet-101	128	79.4	94.4

TABLE V
RESULTS ON SOMETHING-SOMETHING V1.

Model	Backbone	Frames×Crop×Clip	Top-1	Top-5
I3D [40]	ResNet-50	64=32×1×2	41.6	72.2
NL I3D [40]	ResNet-50	64=32×1×2	44.4	76.0
NL I3D + gcn [40]	ResNet-50	64=32×1×2	46.1	76.8
TSM [25]	ResNet-50	8=8×1×1	45.6	74.2
TSM [25]	ResNet-50	16=16×1×1	47.2	77.1
TSM$_{En}$ [25]	ResNet-50	24=(8+16)×1×1	49.7	78.5
RNL TSM (ours)	**ResNet-50**	**8=8×1×1**	**47.3**	**-**
RNL TSM (ours)	**ResNet-50**	**16=16×1×1**	**49.4**	**-**
RNL TSM$_{En}$ (ours)	**ResNet-50**	**24=(8+16)×1×1**	**51.3**	**80.6**
SmallBig [41]	ResNet-50	48=8×2×3	48.3	78.1
SmallBig [41]	ResNet-50	96=16×2×3	50.0	79.8
SmallBig$_{En}$ [41]	ResNet-50	144=(8+16)×2×3	51.4	80.7
RNL TSM (ours)	**ResNet-50**	**48=8×2×3**	**49.5**	**78.4**
RNL TSM (ours)	**ResNet-50**	**96=16×2×3**	**51.0**	**80.3**
RNL TSM$_{En}$ (ours)	**ResNet-50**	**144=(8+16)×2×3**	**52.7**	**81.5**
RNL TSM (ours)	**ResNet-101**	**48=8×2×3**	**50.8**	**79.8**
RNL TSM$_{En}$ (ours)	**R101 + R50**	**144=(8+16)×2×3**	**54.1**	**82.2**

order to achieve the best performance on Kinetics-400, we increase the number of training epochs from 50 to 100. The performance comparisons are summarized in Tables IV and V, where RNL TSM refers to the model with 5 attention chain blocks. Note that using the same approach, the models with deeper backbone networks or longer clips as training inputs would consistently result in better performance in comparison with shallower backbone networks. on Kinetics, we use a shallower network, such as ResNet-50, as the backbone, and the length of our input video clips is at least 8 times shorter than other methods, yet our results are highly competitive with those of the other approaches.

On Something-Something V1, when using ResNet-50 as the backbone, the ensemble version of our model, the RNL TSM$_{En}$, using $\{8, 16\}$ frames as inputs, achieves a higher accuracy than other approaches, w.r.t., single-clip & center-crop (Top-1: 51.3%) and multi-clip & multi-crop (Top-1: 52.7%). When adopting ResNet-101 as the backbone, we gain extra performance boost (Top-1: 50.8% vs. 49.5%). Moreover, the ensemble of the deep model of 8 frame inputs and the shallow model of 16 frame inputs achieves the best accuracy (Top-1: 54.1%). All these results further demonstrate the effectiveness and efficiency of the proposed method.

V. Conclusion

In this work, we presented the region-based non-local operation (RNL), a novel self-attention mechanism that effectively captures long-range dependencies by exploiting pairwise region relationships. The RNL blocks can be easily embedded into the off-the-shelf CNNs architectures for end-to-end training. We have performed ablation studies to investigate the effectiveness of the proposed RNL operation in various settings. To verify the efficiency and effectiveness of the proposed methodology, we conducted experiments on two video benchmarks, Kinetics-400 and Something-Something V1. The results of the proposed method are shown to outperform the baseline and other recently proposed attention methods. Furthermore, we achieve state-of-the-art performance on Something-Something V1, which has demonstrated the powerful representation learning ability of our models.

References

[1] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 4489–4497.

[2] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 4724–4733.

[3] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2015, pp. 4694–4702.

[4] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2015, pp. 2625–2634.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016, pp. 770–778.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 7794–7803.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.

[9] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, vol. 2. IEEE, 2005, pp. 60–65.

[10] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag *et al.*, "The" something something" video database for learning and evaluating visual common sense." in *ICCV*, vol. 1, no. 4, 2017, p. 5.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Int. Conf. Learn. Rep. (ICLR)*, 2015.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2015, pp. 1–9.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for comp. vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016, pp. 2818–2826.

[15] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *AAAI Conference on Artificial Intelligence*, 2016.

[16] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Comp. Vision and Pattern Recog. (CVPR)*, 2017, pp. 5987–5995.

[17] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 4700–4708.

[18] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019.

[19] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 568–576.

[20] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. Snoek, "Videolstm convolves, attends and flows for action recognition," *Computer Vision and Image Understanding*, vol. 166, pp. 41–50, 2018.

[21] G. Huang and A. G. Bors, "Learning spatio-temporal representations with temporal squeeze pooling," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP)*, 2020.

[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 *Proc. Eur. Conf. Comput. Vis. (ECCV), vol LNCS 9912*, 2016, pp. 20–36.

[23] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2010, pp. 140–153.

[24] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3d residual networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 5533–5541.

[25] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 7083–7093.

[26] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," in *Proc. Eur. Conf. Comput. Vis. (ECCV), vol. LNCS 11219*, 2018, pp. 305–321.

[27] D. Tran, H. Wang, L. Torresani, and M. Feiszli, "Video classification with channel-separated convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 5552–5561.

[28] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020, pp. 203–213.

[29] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Int. Conf. Learn. Representations (ICLR)*, 2015.

[30] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 7132–7141.

[31] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 3156–3164.

[32] S. Woo, J. Park, J. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.

[33] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCV-w)*, 2019.

[34] K. Yue, M. Sun, Y. Yuan, F. Zhou, E. Ding, and F. Xu, "Compact generalized non-local network," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 6510–6519.

[35] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 4510–4520.

[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn. (ICML) - Volume 37*, 2015, p. 448–456.

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2009, pp. 248–255.

[38] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019, pp. 6202–6211.

[39] Z. Qiu, T. Yao, C.-W. Ngo, X. Tian, and T. Mei, "Learning spatio-temporal representation with local and global diffusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019, pp. 12 056–12 065.

[40] X. Wang and A. Gupta, "Videos as space-time region graphs," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 399–417.

[41] X. Li, Y. Wang, Z. Zhou, and Y. Qiao, "Smallbignet: Integrating core and contextual views for video classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020, pp. 1092–1101.