SmallBigNet: Integrating Core and Contextual Views for Video Classification

Xianhang Li[12*], Yali Wang[1*], Zhipeng Zhou[1*], and Yu Qiao[12†]

[1]ShenZhen Key Lab of Computer Vision and Pattern Recognition, SIAT-SenseTime Joint Lab, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
[2]SIAT Branch, Shenzhen Institute of Artificial Intelligence and Robotics for Society

Abstract

Temporal convolution has been widely used for video classification. However, it is performed on spatio-temporal contexts in a limited view, which often weakens its capacity of learning video representation. To alleviate this problem, we propose a concise and novel SmallBig network, with the cooperation of small and big views. For the current time step, the small view branch is used to learn the core semantics, while the big view branch is used to capture the contextual semantics. Unlike traditional temporal convolution, the big view branch can provide the small view branch with the most activated video features from a broader 3D receptive field. Via aggregating such big-view contexts, the small view branch can learn more robust and discriminative spatio-temporal representations for video classification. Furthermore, we propose to share convolution in the small and big view branch, which improves model compactness as well as alleviates overfitting. As a result, our SmallBigNet achieves a comparable model size like 2D CNNs, while boosting accuracy like 3D CNNs. We conduct extensive experiments on the large-scale video benchmarks, e.g., Kinetics400, Something-Something V1 and V2. Our SmallBig network outperforms a number of recent state-of-the-art approaches, in terms of accuracy and/or efficiency. The codes and models will be available on https://github.com/xhl-video/SmallBigNet.

1. Introduction

3D convolution has been widely used for deep video classification [1, 23]. In particular, spatio-temporal factorization is preferable to reduce computation cost as well as overfitting [23, 32]. However, temporal convolution in this form is usually operated on a limited view, which often contains unrelated video contexts. As shown in Fig.1(a), temporal convolution (e.g., $3 \times 1 \times 1$) is performed over the yellow tube of a *High Jump* video. Apparently, for the blue box at t, the yellow boxes at $t-1$ and $t+1$ provides almost useless even harmful contexts. For example, the yellow box at $t-1$ contains the arm of the athlete. However, the arm movement is not quite critical to recognize *High Jump*. Hence, the context at $t-1$ tends to be redundant. Furthermore, the yellow box at $t+1$ contains the upper body of another sitting person, without any clues about the moving athlete. Hence, the context at $t+1$ tends to be noisy. By aggregating these contexts with the blue box at t, temporal convolution often leads to a weak and unstable spatio-temporal representation that is not discriminative to recognize *High Jump*.

To tackle the problem above, we creatively introduce a novel and concise SmallBig unit in Fig.1(b), where the big view branch can flexibly provide the small view branch with discriminative contexts from a larger spatio-temporal receptive field. Via aggregating such contextual clues, the small view branch tends to learn key spatio-temporal representations for video classification. Note that, our SmallBig design is different from the SlowFast design [7], in terms of both motivation and mechanism. In particular, SlowFast is motivated by mimicking two-stream fusion. Hence, it feeds input frames of two temporal rates to build up two 3D CNNs (i.e., slow and fast pathways), and applies lateral connections to integrate them into a unified framework. Alternatively, our SmallBig is motivated by releasing the contextual receptive fields of 3D CNN itself. Hence, we introduce two distinct views (i.e., small and big branches), and discover the most activated context of big view to enhance the core representations of small view.

More specifically, we propose two distinct operations in our SmallBig unit. First, we perform 3D max pooling in the big view branch, which can discover the most activated contexts from a broader 3D tube to enhance the process of the small view branch. For example, we treat the blue box at t as center, and find its corresponding $3 \times 3 \times 3$ yellow tube

*Equally-contributed first authors (xianhang710@gmail.com, (yl.wang, zp.zhou)@siat.ac.cn)
†Corresponding author (yu.qiao@siat.ac.cn)

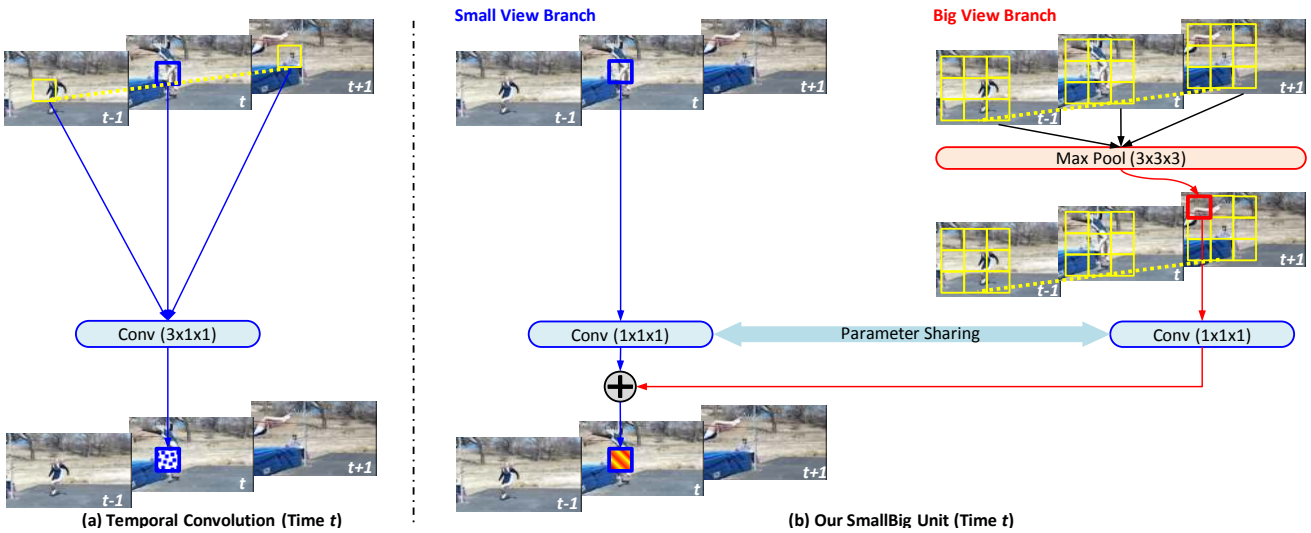

Figure 1. Motivation. As shown in Subplot (a), temporal convolution is operated on a limited view (yellow tube), which often contains useless video contexts, e.g., the yellow box at $t+1$ contains the upper body of another sitting person, without any clues about the moving athlete. Aggregating such contexts would be harmful to recognize *High Jump*. To alleviate it, we propose a novel and concise SmallBig unit with two views, where the big view branch can provide the small view branch with the most activated contexts in a broader 3D receptive field. Such cooperation allows our SmallBig unit to learn more discriminative spatio-temporal representations for video classification.

from $t-1$ to $t+1$ in Fig.1(b). Subsequently, we apply max pooling over this tube to identify its most activated features, i.e., the red box at $t+1$. As we can see, this box contains the take-off pose of the athlete. Clearly, it provides more discriminative cues to recognize *High Jump*, comparing to the yellow boxes applied in the limited view of temporal convolution (Fig. 1(a)). Second, we propose to share convolution parameters between the small and big view branches. This operation improves the compactness of our SmallBig unit, while boosting accuracy.

Finally, we build up our SmallBig network (SmallBigNet) in a ResNet style. By progressively applying a number of SmallBig units in a residual block, we enlarge the cooperative power of two views with richer contexts from a broader 3D receptive field. As a result, our SmallBig network can gradually learn a key spatio-temporal representation for video classification, when the layer is going deeper. To evaluate it, we perform extensive experiments on the widely-used video benchmarks, e.g., Kinetics400 [13], Something-Something V1 and V2 [10]. Under the same setting, our SmallBig network outperforms the recent state-of-the-art methods, in terms of accuracy and/or efficiency.

2. Related Works

2D CNNs for Video Classification. Over the past years, video classification has been mainly driven by deep learning frameworks [1, 7, 21, 27, 28, 31]. One of the widely-used 2D frameworks is two-stream CNNs [21], which can learn video representations respectively from RGB and optical flows. To further boost performance, a number of extensions have been proposed by deep descriptors [25], spatio-temporal fusions [8, 9], key volume mining [34], attention [26], sequential modeling with RNNs [6, 18, 20], temporal segment networks [27], temporal relational networks [33], temporal shift module [15], etc. In particular, temporal shift module (TSM) [15] moves feature along the temporal dimension, which achieves the performance of 3D CNN but maintains the complexity of 2D CNN. However, it may lack the comprehensive capacity of understanding spatio-temporal dynamics in videos. Alternatively, our SmallBig design can effectively exploit the most-activated contexts from a broader 3D view, and learn key spatio-temporal representations with cooperation of two different views.

3D CNNs for Video Classification. 3D CNNs have become popular for spatio-temporal learning, by treating time as the third dimension of convolutions [11, 12, 22]. However, such operation introduces many more parameters, which makes 3D convolution harder to train. To alleviate such problem, I3D [1] has been proposed by inflating 2D convolution into 3D. But still, the heavy computation burden limits the power of these full 3D CNNs. Recent studies have shown that factorizing 3D convolution is preferable to reduce complexity as well as boost accuracy, e.g., P3D [19], R(2+1)D [23], S3D-G [32], etc. However, temporal convolution in these approaches is performed on a limited view, where the unrelated contexts often reduce its capacity of learning video representations.

Learning Long-Term Video Dependency. Alternatively, learning long-term dependency has been highlighted for video classification [4, 5, 16, 28, 29, 30]. One of the most popular models is the nonlocal network [28]. However, this approach mainly aggregates global relations to as-

sist video classification, which may not fully exploit fine contexts in the local tube. On the contrary, our approach gradually enlarges contextual receptive fields in a SmallBig block. Hence, it allows us to learn key video representation progressively from local view to global view. Finally, a SlowFast network has been proposed in [7]. The key difference is that, it uses input frames of two temporal rates to mimic two-stream fusion of 3D CNNs, while our SmallBig uses two spatio-temporal views on a 3D CNN itself to exploit contexts for enhancing core video features.

3. SmallBig Unit

In this section, we first analyze temporal convolution, and then explain how to design our SmallBig unit.

Temporal Convolution. Without loss of generality, we use a widely-used $3 \times 1 \times 1$ temporal convolution filter as illustration. Specifically, we denote $\mathbf{x}_t^{(h,w)}$ as a feature vector at spatial location of (h, w) and temporal frame of t. Additionally, we denote $\Theta = [\Theta_\alpha,\ \Theta_\beta,\ \Theta_\gamma]$ as parameters in this $3 \times 1 \times 1$ convolution filter. As shown in Fig.1(a), temporal convolution applies Θ to encode video dynamics from $t-1$ to $t+1$, w.r.t., each spatial location (h, w),

$$\begin{aligned}\mathbf{y}_t^{(h,w)} &= \textbf{\textit{TemConv}}(\Theta,\ \mathbf{x}_t^{(h,w)},\ \{\mathbf{x}_{t-1}^{(h,w)},\ \mathbf{x}_{t+1}^{(h,w)}\}),\\ &= \Theta_\beta \mathbf{x}_t^{(h,w)} + [\Theta_\alpha \mathbf{x}_{t-1}^{(h,w)} + \Theta_\gamma \mathbf{x}_{t+1}^{(h,w)}],\end{aligned} \tag{1}$$

where $\mathbf{y}_t^{(h,w)}$ is the output vector of spatio-temporal representation. As mentioned in the introduction, temporal convolution is performed with spatio-temporal contexts of $\mathbf{x}_{t-1}^{(h,w)}$ and $\mathbf{x}_{t+1}^{(h,w)}$. Such limited view often weakens the discriminative power of $\mathbf{y}_t^{(h,w)}$.

SmallBig Unit. To address the problem above, we propose to discover the contexts of $\mathbf{x}_t^{(h,w)}$ from a broader spatio-temporal receptive field. This leads to a novel and concise SmallBig unit with parameters $\Psi = [\Psi_\rho,\ \Psi_\nu]$,

$$\begin{aligned}\mathbf{y}_t^{(h,w)} &= \textbf{\textit{SmallBig}}(\Psi,\ \mathbf{x}_t^{(h,w)},\ \{\mathbf{x}_k^{(i,j)}\}),\\ &= \underbrace{\Psi_\rho \mathbf{x}_t^{(h,w)}}_{small\ view} + \underbrace{\Psi_\nu \text{MaxPool}(\{\mathbf{x}_k^{(i,j)}\})}_{big\ view}.\end{aligned} \tag{2}$$

Next, we mainly explain two key operations in this unit.

1) 3D Max Pooling Over Big View. To further release the spatio-temporal location constraints of contexts, we propose to work on a broader $T_\epsilon \times H_\epsilon \times W_\epsilon$ tube (e.g., $3 \times 3 \times 3$) centered at (t, h, w). In particular, we perform max pooling over the feature vectors $\{\mathbf{x}_k^{(i,j)}\}$ in this 3D tube. As a result, we can identify the most activated contextual feature from a bigger view. Compared to $\mathbf{x}_{t-1}^{(h,w)}$ and $\mathbf{x}_{t+1}^{(h,w)}$ in the temporal convolution, this max-pooled feature is often more discriminative to capture key video dynamics, e.g., take-off pose of athlete in the red box of Fig.1(b). By aggregating such contexts for $\mathbf{x}_t^{(h,w)}$, our SmallBig unit can reduce redundancy and promote robustness of spatio-temporal learning.

2) Parameter Sharing Between Small & Big Views. After obtaining the most activated contextual feature, we apply $1 \times 1 \times 1$ pointwise convolution filters Ψ_ρ and Ψ_ν to further encode the representation in the small and big view branches. Specifically, we propose to share the parameters between filters of two views, i.e., $\Psi_\rho = \Psi_\nu$, in order to increase model compactness. Via this operation, the size of our SmallBig unit is reduced as that of 2D convolution. In this case, our SmallBig unit can efficiently enhance $\mathbf{y}_t^{(h,w)}$ with cooperation of two views.

4. Exemplar: SmallBig-ResNet

After introducing the SmallBig unit, we illustrate how to adapt it into a residual style block, and then build up a SmallBig network from ResNet23 (or ResNet50).

SmallBig Blocks. As shown in Fig.2, we first introduce two widely-used residual blocks for comparison, i.e., 2D convolution in Subplot(a) and 3D convolution in Subplot(b), where 2D convolution consists of three layers, i.e., two $1 \times 1 \times 1$ and one $1 \times 3 \times 3$. For 3D convolution, we follow [28] to apply inflation [1] on the first $1 \times 1 \times 1$, which leads to a $3 \times 1 \times 1$ temporal convolution.

For our SmallBig blocks, we adapt the layers of 2D convolution gradually into the SmallBig unit, as shown in Subplot(c)-(e). Note that, for a typical SmallBig block in Subplot(e), we set the pooling size as $T \times 3 \times 3$ in the big view branch of last layer, where T is the total number of sampled video frames. The main reason is that, after $3 \times 3 \times 3$ max pooling in the big view branch of middle layer, $1 \times 3 \times 3$ convolution further enlarges spatial receptive field. To balance spatio-temporal view, we further enlarge temporal receptive field in the big view branch of last layer.

Finally, we introduce an extra SmallBig unit on top of a typical SmallBig block, which leads to a full SmallBig block in Subplot(f). In the big view branch of the extra SmallBig unit, we operate max pooling over the global spatio-temporal tube of $T \times H \times W$. In this way, the full SmallBig block can progressively integrate the most activated contexts from local view to global view. Furthermore, the pooling operation in this extra unit actually produces a global feature vector (after conv), which is irrelevant to spatio-temporal location. Hence, we naturally adapt this vector as attention (with sigmoid), and apply it for channel-wise product aggregation. Lastly, besides of parameter sharing between two views, we propose to use the bottleneck-like design in this extra SmallBig unit, e.g., input : output channels is 4:1 for its 1st convolution, while input : output channels is 1:4 for its 2nd convolution. This is used to reduce computation cost of our full SmallBig block.

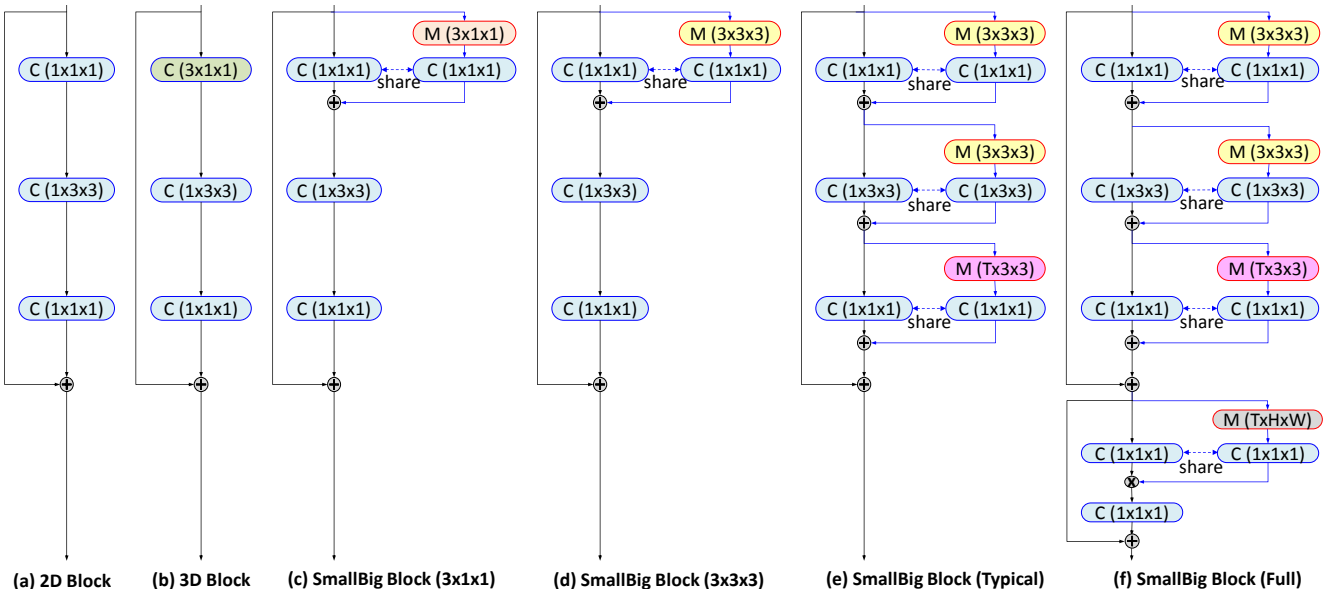

Figure 2. SmallBig Blocks. C: Convolution. M: Max pooling. We design a number of SmallBig blocks by adapting all the 2D convolution layers progressively as the SmallBig unit. More explanations can be found in Section 4.

Layer		Output Size
conv1	$1 \times 7 \times 7$, 64, stride $1 \times 2 \times 2$	$8 \times 112 \times 112$
pool1	$1 \times 3 \times 3$ max, stride $1 \times 2 \times 2$	$8 \times 56 \times 56$
res2	$\begin{pmatrix} 1 \times 1 \times 1, 64 \\ 1 \times 3 \times 3, 64 \\ 1 \times 1 \times 1, 256 \end{pmatrix} \times 1$ (or 3)	$8 \times 56 \times 56$
res3	$\begin{pmatrix} 1 \times 1 \times 1, 256 \\ 1 \times 3 \times 3, 256 \\ 1 \times 1 \times 1, 512 \end{pmatrix} \times 2$ (or 4)	$8 \times 28 \times 28$
res4	$\begin{pmatrix} 1 \times 1 \times 1, 512 \\ 1 \times 3 \times 3, 512 \\ 1 \times 1 \times 1, 1024 \end{pmatrix} \times 3$ (or 6)	$8 \times 14 \times 14$
res5	$\begin{pmatrix} 1 \times 1 \times 1, 1024 \\ 1 \times 3 \times 3, 1024 \\ 1 \times 1 \times 1, 2048 \end{pmatrix} \times 1$ (or 3)	$8 \times 7 \times 7$
global average pool, fc		$1 \times 1 \times 1$

Table 1. 2D backbone of our SmallBig network: ResNet23 (or ResNet50). We construct SmallBig-ResNet by adapting each 2D residual block as a SmallBig block such as Fig.2(c)-(f). The input is $8 \times 224 \times 224$, which is sampled from a 64-frame clip with temporal stride of 8.

SmallBig-ResNet. After building up the SmallBig blocks, we construct SmallBig network from ResNet23 (or ResNet50) in Table 1. The size of input clip is $8 \times 224 \times 224$. For each 2D residual block, we replace it with any of our SmallBig blocks in Fig.2(c)-(f). Note that, the number of parameters in SmallBig-ResNet is comparable to that of 2D ResNet, due to parameter sharing. Moreover, the parameters of SmallBig-ResNet can be directly initialized from those of 2D ResNet, which has been well pretrained on ImageNet. This simplifies the initialization issue and boosts our SmallBig-ResNet in practice.

5. Further Discussion: SmallBig vs. Nonlocal

As mentioned before, our SmallBig design is related to the well-known nonlocal operation [28], which also leverages the spatio-temporal contexts in a broader view. Hence, we further discuss differences between our design and this SOTA architecture. For convenience, we denote $\{\mathbf{x}_k^{(i,j)}\}_{all}$ as all the feature vectors in the global tube of $T \times H \times W$. For $\mathbf{x}_t^{(h,w)}$ at (t, h, w), the nonlocal operation actually finds its highly similar contexts from $\{\mathbf{x}_k^{(i,j)}\}_{all}$. Specifically, we rewrite this operation as a formulation of two views with parameter $\mathbf{V} = [\mathbf{V}_\theta,\ \mathbf{V}_\phi,\ \mathbf{V}_g,\ \mathbf{V}_o]$,

$$\begin{aligned} \mathbf{y}_t^{(h,w)} &= \textit{NonLocal}(\mathbf{V},\ \mathbf{x}_t^{(h,w)},\ \{\mathbf{x}_k^{(i,j)}\}_{all}) \\ &= \mathbf{x}_t^{(h,w)} + \mathbf{V}_o \sum\nolimits_{all} f(\mathbf{x}_t^{(h,w)}, \mathbf{x}_k^{(i,j)}) g(\mathbf{x}_k^{(i,j)}) \\ &= \mathbf{x}_t^{(h,w)} + \mathbf{V}_o \sum\nolimits_{all} s_k^{(i,j)} \mathbf{V}_g \mathbf{x}_k^{(i,j)} \\ &= \underbrace{\mathbf{x}_t^{(h,w)}}_{small\ view} + \underbrace{\mathbf{V}_o \mathbf{V}_g \sum\nolimits_{all} s_k^{(i,j)} \mathbf{x}_k^{(i,j)}}_{big\ view}. \end{aligned} \tag{3}$$

$s_k^{(i,j)}$ is the similarity score between $\mathbf{x}_t^{(h,w)}$ and $\mathbf{x}_k^{(i,j)}$. It is computed from a kernel function, e.g., embedded Gaussian $f(\mathbf{x}_t^{(h,w)}, \mathbf{x}_k^{(i,j)}) = \exp[(\mathbf{V}_\theta \mathbf{x}_t^{(h,w)})^\top (\mathbf{V}_\phi \mathbf{x}_k^{(i,j)})]/C_x$ with a normalization term C_x. Additionally, $g(\mathbf{x}_k^{(i,j)}) = \mathbf{V}_g \mathbf{x}_k^{(i,j)}$ is a linear transformation of $\mathbf{x}_k^{(i,j)}$. Hence, we move $\mathbf{V}_g$ out of summation $\sum$.

Via comparing the big view branch in Eq.(2) and (3), we find that both mechanisms exhibit the spirit of visual attention. However, our SmallBig design contains the distinctive characteristics as follows. **First**, the goals of visual

attention are different. The nonlocal operation uses similarity comparison as soft attention, which aims at finding the similar contexts for $\mathbf{x}_t^{(h,w)}$. Such contexts implicitly assist video classification by modeling spatio-temporal dependency. Alternatively, our SmallBig unit uses max pooling as hard attention, which aims at finding the key contexts around $\mathbf{x}_t^{(h,w)}$. Such contexts are more explicit and discriminative to boost classification accuracy, since they are highly activated to recognize different video classes. **Second**, the receptive fields of visual attention are different. The nonlocal operation directly works on the global spatio-temporal tube to learn long-term relations, which may ignore key video details for classification. Alternatively, our SmallBig unit works on the local spatio-temporal tube to capture fine video clues. More importantly, we gradually enlarge the receptive field of big view branch in the SmallBig block, allowing us to learn video representation progressively from local view to global view. Our experiments also show that the SmallBig network steadily outperforms the nonlocal network.

6. Experiment

Data Sets. We perform the experiments on the large-scale video benchmarks, i.e., Kinetics400 [13], Something-Something V1 and V2 [10]. Kinetics400 consists of around 300k videos from 400 categories. Something-Something V1/V2 consists of around 108k/220k videos from 174 categories. We mainly evaluate all the models on the validation set, where we report Top1 & Top5 accuracy (%) and GFlops to comprehensively evaluate accuracy and efficiency.

Training. For all the data sets, we follow [28] to use the spatial size of 224×224, which is randomly cropped from a scaled video whose shorter side is randomly sampled in $[256, 320]$ pixels. For Kinetics400, the input clip consists of 8 frames, which are sampled from 64 consecutive frames with temporal stride 8. We train our models with 110 epochs, where we set the weight decay as 1e-4, and utilize the cosine schedule of learning rate decay. For SmallBig-ResNet23, we set the initial learning rate as 0.02 and the batch size as 128. For SmallBig-ResNet50/101, we set the initial leaning rate as 0.00625 and the batch size as 128. For Something-Something V1 and V2, we divide a video into 8 segments and then randomly choose one frame in each segment. We train our models with 50 epochs. The initial learning rate is 0.01, and it decays at 30, 40, 45 epochs respectively. Finally, we apply batch normalization individually for each view (right after convolution). All the models are pretrained on ImageNet, including BN in the small view branch of each layer. For BN in the big view branch, we initialize its scale parameter as zero. This design makes the initial state of our SmallBig network as the original ResNet.

Inference. Following [7, 28], we rescale the video frames with the shorter side 256 and take three crops (left, middle, right) of size 256×256 to cover the spatial dimensions. Unless stated otherwise, we uniformly sample 10/2 clips for Kinetics400 / Something-Something V1 and V2. We average their softmax scores for video-level prediction.

6.1. Evaluation on Kinetics400

In the following, we perform extensive ablation studies to investigate various distinct characteristics in our SmallBig network. Then, we further evaluate accuracy and efficiency of our SmallBig networks by a comprehensive comparison with the recent state-of-the-art approaches.

Effectiveness of SmallBig. We apply ResNet23 (R23) of Table 1 as backbone, and adapt the SmallBig block ($3 \times 1 \times 1$) of Fig.2(c) into all the residual stages. For comparison, we also adapt the 3D block ($3 \times 1 \times 1$) of Fig.2(b) in the same way. As shown in Table 2, our SmallBig-R23 outperforms its 2D and 3D counterparts. Note that, even though we do not further enlarge the spatio-temporal receptive field in the big view branch, our SmallBig block ($3 \times 1 \times 1$) still achieves a better result than the 3D block ($3 \times 1 \times 1$). It illustrates that, the most activated contexts found by max pooling is a preferable guidance to learn key video representation, compared to temporal convolution.

Stage of using SmallBig blocks. We use the above SmallBig-R23 ($3 \times 1 \times 1$) to evaluate which stage may be important for SmallBig design. In Table 3, we gradually recover our SmallBig blocks as the original 2D blocks, from bottom to top. As expected, the middle blocks (e.g., res4) are often more important than the bottom and top blocks. The main reason is that, the receptive field is too small (or big) in the bottom (or top) blocks, enlarging 3D view tends to find useless (or similar) contexts. On the contrary, the middle blocks contain the middle-level semantics with a reasonable spatio-temporal receptive field. Hence, the contexts in these blocks would be more discriminative. In our following experiments, we use SmallBig blocks in all the residual stages to achieve the best accuracy.

Broader receptive field in the big view branch. For SmallBig-R23 ($3 \times 1 \times 1$), we further extend 3D receptive field in its big view branch. As shown in Table 4, the accuracy first increases and then decreases. It may be because the diversity of contexts is reduced, when we directly perform max pooling on too big view. As a result, SmallBig-R23 achieves the best performance with $3 \times 3 \times 3$ in Fig.2(d).

More SmallBig layers. The above experiment in Table 4 indicates that, it is not reasonable to enlarge the 3D receptive field directly to a very big view. Hence, we adapt more layers in each residual block to be our SmallBig unit, allowing to extend the 3D receptive field gradually from local to global view. As shown in Table 5, the accuracy is consistently getting better, when more convolution layers are progressively changed as SmallBig. As expected, the setting of

R23	Top1	Top5
2D	64.1	85.4
3D: 3×1×1	68.3	88.2
SmallBig: 3×1×1	**69.0**	**88.6**

Table 2. Effectiveness of SmallBig. We apply ResNet23 (R23) as 2D backbone, and adapt 3D block ($3\times1\times1$) of Fig.2(b) and SmallBig block ($3\times1\times1$) of Fig.2(c) in all the residual stages.

SmallBig-R23	Top1	Top5
R23	64.1	85.4
SmallBig: res2+3+4+5	**69.0**	**88.6**
SmallBig: res3+4+5	68.8	88.5
SmallBig: res4+5	68.6	88.5
SmallBig: res5	64.5	85.6

Table 3. Stage of using SmallBig blocks. As expected, the middle blocks (e.g., res4) are often more important than others.

SmallBig-R23	Top1	Top5
R23	64.1	85.4
SmallBig: 3×1×1	69.0	88.6
SmallBig: 3×3×3	**69.5**	**89.0**
SmallBig: 3×5×5	69.1	88.5
SmallBig: 3×7×7	68.6	88.3
SmallBig: T×3×3	69.3	88.5

Table 4. Broader receptive field in the big view branch.

SmallBig-R23	Top1	Top5
R23	64.1	85.4
SmallBig: 3×3×3(1st layer)	69.5	89.0
SmallBig: 3×3×3(1st layer)+T×3×3(3rd layer)	70.8	89.3
SmallBig: Typical	71.4	90.0
SmallBig: Full	**72.6**	**90.3**

Table 5. More SmallBig layers. The accuracy is consistently better, when more layers are progressively changed as our SmallBig design. For SmallBig: Typical or Full, all blocks refer to Fig.2(e) or (f).

SmallBig-R23	Params	GFlops	Top1	Top5
R23	11.3M	17	64.1	85.4
Avg Pool	13.4M	31	72.2	90.0
Max Pool	13.4M	31	**72.6**	**90.3**
Without Share	22.1M	31	71.6	89.6
With Share	13.4M	31	**72.6**	**90.3**
Single BN	13.3M	17	64.6	85.8
Individual BN	13.4M	31	**72.6**	**90.3**

Table 6. Detailed designs of SmallBig.

Model	Params	GFlops	Top1	Top5
R23	11.3M	17	64.1	85.4
NonLocal-R23	18.7M	34	70.2	89.1
SmallBig-R23	13.4M	31	**72.6**	**90.3**

Table 7. SmallBig vs. NonLocal. Our SmallBig-R23 outperforms NonLocal-R23, showing the superiority of our SmallBig design when finding contexts.

Backbone	Top1	Top5
R23	64.1	85.4
SmallBig-R23	72.6	90.3
R50	70.4	89.1
SmallBig-R50	**76.3**	**92.5**

Table 8. Backbone. Our SmallBig-R23 even outperforms R50.

SmallBig-R50	Top1	Top5
Extra Unit: Simple	75.8	92.1
Extra Unit: Default	**76.3**	**92.5**

Table 9. Extra unit in SmallBig (Full). For comparison, we replace the extra unit in Fig.2(f) by a simplified version.

SmallBig-R23 (Full) achieves the best performance, where all the SmallBig blocks refer to Fig.2(f). In the following, we use the full setting in our experiments.

Detailed designs of SmallBig. We use SmallBig-R23 (Full) to further investigate the detailed designs of SmallBig in Table 6. **Avg Pool vs Max Pool**. We apply different pooling operations in the big view branch. The performance of max pooling is better. Hence, we choose it in our experiments. **Without Sharing vs With Sharing**. We apply different parameter sharing strategies for convolution in our SmallBig unit. As expected, parameter sharing can reduce the model size of our SmallBig-R23 as the original R23. Its accuracy is even slightly better than the non-sharing case. Hence, we share parameters between small and big view branches. **Single BN vs Individual BN**. As mentioned in the implementation details, we apply BN individually for each view, i.e., BN(conv(small))+BN(conv(big)). This would introduce extra flops. To further reduce complexity, we apply a single BN directly on the output representation, i.e., BN(conv(small)+conv(big)). Due to the linearity of convolution and sum operations, this operation is equivalent to BN(conv(small+big)), which only requires a single convolution and decreases flops as 2D CNN. As shown in Table 6, single BN has higher efficiency but much lower accuracy, while individual BN has higher accuracy but lower efficiency. For consistency, we choose Individual BN to achieve a better accuracy.

SmallBig vs. NonLocal. We compare our SmallBig design with the related NonLocal operation. Specifically, we use a preferable setting of NonLocal as suggestion in [28], where NL is added on all the residual blocks of res3 and res4. As shown in Table 7, our SmallBig-R23 outperforms NonLocal-R23. It illustrates that, to boost performance, it is preferable to find the most activated contexts progressively from local to global view, instead of finding dependent contexts directly on the global view.

Deeper backbone. We further investigate the performance of our SmallBig network, with a deeper backbone, e.g., ResNet50 (R50). As shown in Table 8, our SmallBig-R23 even outperforms R50. It illustrates the power of our SmallBig design. Furthermore, SmallBig-R50 outperforms SmallBig-R23, showing the effectiveness of SmallBig in deeper backbones.

Extra unit in SmallBig (Full). As shown in Fig.2(f), we add an extra SmallBig unit with global pooling. Note that, we apply channel-wise product aggregation in the extra unit. Hence, we design a Squeeze-and-Excitation version for comparison. Specifically, we first perform global spatio-temporal pooling, and then add two extra $1\times1\times1$ convolutions. The resulting vector (after sigmoid) is used as channel-wise attention for residual aggregation. In Table 9, our default design outperforms this simplified design in the extra unit. It illustrates that, our default extra unit is a preferable choice of global spatio-temporal aggregation,

Method	Backbone	Frame, Size	Top1	Top5	GFlops×crops
STC[5]	ResNeXt101	32, 112	68.7	88.5	N/A×N/A
ARTNet[24]	R18	16, 112	69.2	88.3	5,875=23.5×250
MFNet[3]	R34	16, 224	72.8	90.4	11×N/A
R(2+1)D[23]	R34	8, 112	74.3	91.4	152×N/A
I3D[1]	Inception	64, 224	71.1	89.3	108×N/A
S3D-G[32]	Inception	64, 224	74.7	93.4	71.4×N/A
A^2-Net[2]	R50	8, 224	74.6	91.5	41×N/A
SlowOnly[7]	R50	8, 224	74.9	91.5	1,257=41.9×30
GloRe[4]	R50	8, 224	75.1	N/A	867=28.9×30
TSM[15]	R50	8, 224	74.1	91.2	990=33×30
TEI[17]	R50	8, 224	74.9	91.8	990=33×30
TSM[15]	R50	16, 224	74.7	N/A	1,950=65×30
TEI[17]	R50	16, 224	76.2	92.5	1,980=66×30
SlowFast[7]	R50+R50	36=4+32, 224	75.6	92.1	1,083=36.1×30
NL I3D[28]	R50	32, 224	74.9	91.6	N/A×N/A
NL I3D[28]	R50	128, 224	76.5	92.6	8,460=282×30
Our SmallBig	R50	8, 224	**76.3**	**92.5**	1,710=57×30
CoST[14]	R101	8, 224	75.5	92.0	N/A×N/A
GloRe[4]	R101	8, 224	76.1	N/A	1,635=54.5×30
CPNet[16]	R101	32, 224	75.3	92.4	N/A×N/A
NL I3D[28]	R101	32, 224	76.0	92.1	N/A×N/A
NL I3D[28]	R101	128, 224	77.7	93.3	10,770=359×30
SlowFast[7]	R101+R101	40=8+32, 224	77.9	93.2	3,180=106×30
SlowFast[7]	R101+R101	80=16+64, 224	78.9	93.5	6,390=213×30
Our SmallBig	R101	32, 224	77.4	93.3	5,016=418×12
Our SmallBig$_{En}$	R50+R101	40=8+32, 224	**78.7**	**93.7**	5,700=475×12

Table 10. Comparisons with SOTA on Kinetics400 validation set (RGB input). Our 8-frame SmallBig-R50 outperforms 32-frame Nonlocal-R50 with a higher accuracy, and uses $4.9\times$ less GFlops than 128-frame Nonlocal-R50 but with a competitive accuracy. Its accuracy is even slightly better than 32-frame Nonlocal-R101. Moreover, with the comparable GFlops, our 8-frame SmallBig-R50 outperforms 36-frame SlowFast-R50. All these results show that, our SmallBig network is an accurate and efficient model for video classification.

with cooperation of two views.

Comparison with the SOTA approaches. We make a comprehensive comparison in Table 10, where our SmallBig network outperforms the recent SOTA approaches. **First**, our 8-frame SmallBig-R50 outperforms 32-frame Nonlocal-R50 [28] (Top1 acc: 76.3 vs. 74.9), and it uses $4.9\times$ less GFlops than 128-frame Nonlocal-R50 but achieves a competitive accuracy (Top1 acc: 76.3 vs. 76.5). Moreover, it is even slightly better than 32-frame Nonlocal-R101 (Top1 acc: 76.3 vs. 76.0). All these results clearly illustrate that, our SmallBig network is a more accurate and efficient approach than the nonlocal network, for modeling contexts in video classification. **Second**, with the comparable GFlops, our 8-frame SmallBig-R50 outperforms 36-frame SlowFast-R50 [7] (Top1 acc: 76.3 vs. 75.6). It indicates the importance of SmallBig in context exploitation of 3D CNN itself. Additionally, we perform score fusion over 8-frame SmallBig-R50 and 32-frame SmallBig-R101, which mimics two-steam fusion with two temporal rates. When testing, we use 4 clips and 3 crops per clip to maintain computation. Our SmallBig$_{En}$ achieves a better accuracy than SlowFast, using the same number of frames. **Finally**, our 8-frame SmallBig-R50 outperforms 8-frame TSM-R50 [15] (Top1 acc: 76.3 vs. 74.1). It shows that, spatio-temporal learning of SmallBig is more effective than temporal shift of TSM.

Method	Backbone	Frame,1Clip,1Crop	V1		V2		GFlops
			Top1	Top5	Top1	Top5	
TSN [33]	Inception	8	19.5	-	-	-	16
TRN$_{multiscale}$ [33]	Inception	8	34.4	-	-	-	16
ECO [35]	Incep+R18	8	39.6	-	-	-	32
ECO [35]	Incep+R18	16	41.4	-	-	-	64
ECO$_{En}$Lite [35]	Incep+R18	92	46.4	-	-	-	267
TSM [15]	R50	8	45.6	74.2	-	-	33
TSM [15]	R50	16	47.2	77.1	-	-	65
TSM$_{En}$ [15]	R50	24=8+16	49.7	78.5	-	-	98
Our SmallBig	R50	8	47.0	77.1	59.7	86.7	52
Our SmallBig	R50	16	49.3	79.5	62.3	88.5	105
Our SmallBig$_{En}$	R50	24=8+16	**50.4**	**80.5**	**63.3**	**88.8**	157
Method	Backbone	Frame×Clip×Crop	V1		V2		GFlops
			Top1	Top5	Top1	Top5	
I3D [29]	R50	64=32×2	41.6	72.2	-	-	
NL I3D [29]	R50	64=32×2	44.4	76.0	-	-	
NL I3D + gcn [29]	R50	64=32×2	46.1	76.8	-	-	
CPNet[16]	R34	2,304 =24×16×6	-	-	57.65	83.95	N/A
TSM [15]	R50	48=8×2×3	-	-	59.1	85.6	
TSM [15]	R50	96=16×2×3	-	-	63.4	88.5	
Our SmallBig	R50	48=8×2×3	48.3	78.1	61.6	87.7	
Our SmallBig	R50	96=16×2×3	50.0	79.8	63.8	88.9	N/A
Our SmallBig$_{En}$	R50	144=24×2×3	**51.4**	**80.7**	**64.5**	**89.1**	

Table 11. Comparisons with SOTA on Something-Something V1 and V2 validation set (RGB input). For both V1 and V2, our SmallBig-R50 achieves the best accuracy, w.r.t., single-clip & center-crop and multi-clip & multi-crop. Moreover, our 8-frame SmallBig-R50 even outperforms 48-frame TSM-R50 for V2. For multi-clip & multi-crop, the goal is to report the best accuracy. Hence, GFlops is not taken into account, as suggested by TSM.

6.2. Evaluation on Something-Something V1 & V2

Due to lower resolution and shorter video length in Something-Something V1 and V2, we adopt the slow-only baseline [7] for our SmallBig Net, where we add the SmallBig-Extra unit in Fig.2(f) respectively on top of res3, res4, and res5 stages of this baseline. Following [15], we group the results according to the number of sampled frames in the testing phase, i.e., the single-clip & center-crop case and the multi-clip & multi-crop case. For the multi-clip & multi-crop case, the goal is to report the best performance. Hence, GFlops is not taken into account, as suggested in [15]. The results are shown in Table 11. For both V1 and V2, our SmallBig-R50 achieves the best accuracy, w.r.t., single-clip & center-crop and multi-clip & multi-crop. Moreover, our 8-frame SmallBig-R50 even outperforms 48-frame TSM-R50 [15] for V2 (Top1 acc: 59.7 vs. 59.1). All these results further indicate that, our SmallBig network can effectively boost video classification accuracy.

6.3. Visualization

We visualize and analyze the convolution features learned by SmallBigNet. For comparison, we use the nonlocal network [28] as a strong baseline. Specifically, we feed $8 \times 224 \times 224$ clips respectively into SmallBig-R23 and Nonlocal-R23, and then extract $8 \times 28 \times 28$ convolution feature from res3.2 (after SmallBig and Nonlocal operations). Finally, we average the feature maps along the channel dimension, and show them on the original image.

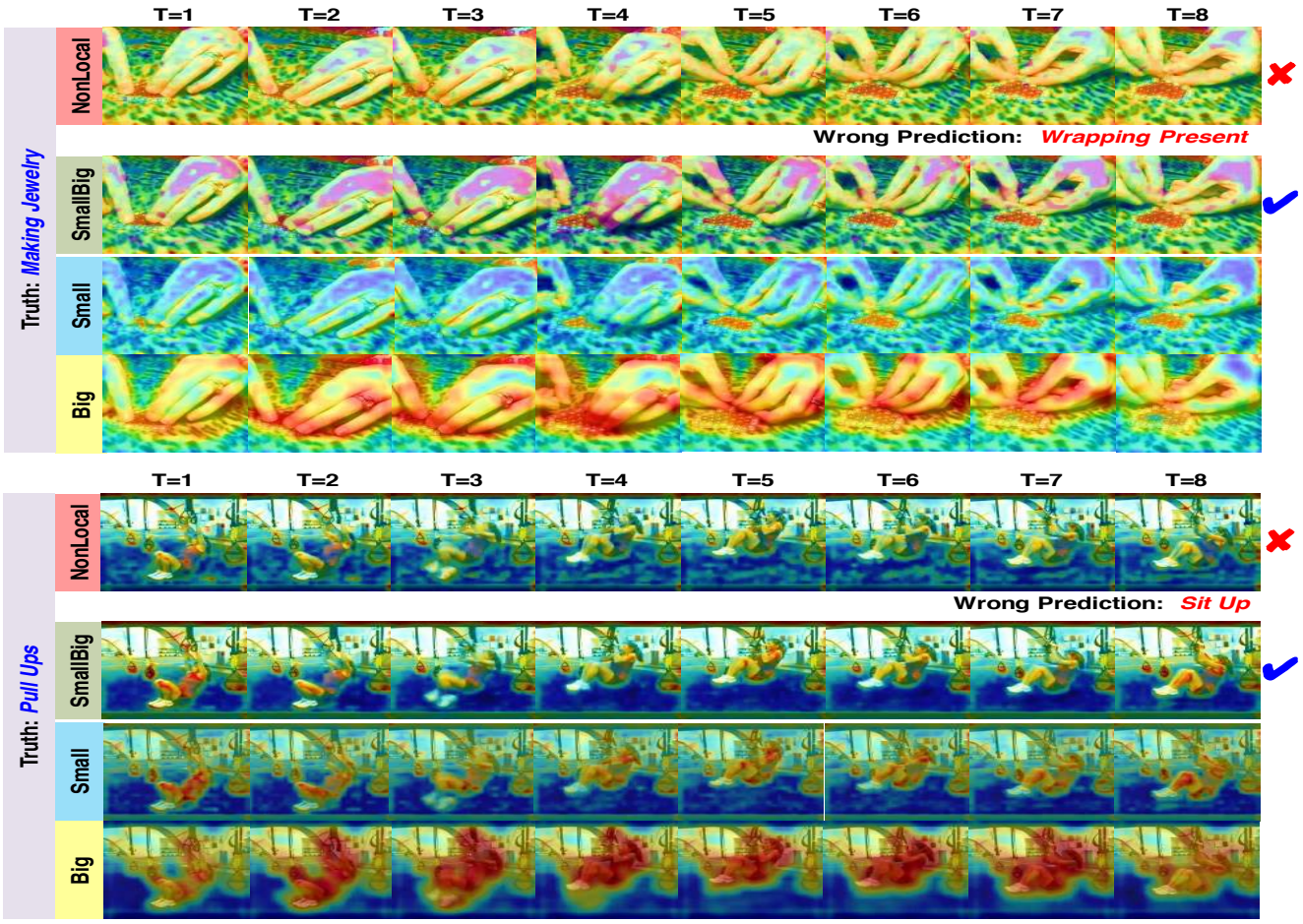

Figure 3. Visualization. Compared to the nonlocal network, our SmallBigNet can discover key video details (e.g., *Making Jewelry*) as well as reduce noisy backgrounds (e.g., *Pull Ups*) for correct prediction. More explanations can be found in Section 6.3.

Fig.3 clearly demonstrates that, our SmallBig network can discover the key video details as well as reduce the noisy backgrounds, compared to the nonlocal network. From this visualization, we also discover that the highly-activated points in the feature map distribute very sparsely in nonlocal, but ours can gather together. This further validates our discussions in Section 5, where our SmallBig network is preferable to learn highly-activated contexts for video classification.

Furthermore, we visualize small and big views from the first layer. As expected, small view tends to capture discriminative core semantics, while big view tends to discover important contextual semantics. For *Making Jewelry*, small view captures hand contour and jewelry object, while big view highlights the regions that contain key hand actions. For *Pull Ups*, small view captures key human parts and objects, while big view highlights the most activated action regions. By aggregating big contextual view to enhance small core view, our SmallBig network is preferable to aggregate the core and contextual views for video classification and can effectively learn spatio-temporal representations.

7. Conclusion

In this work, we propose a concise and novel SmallBig network with cooperation of small and big views. In particular, we enlarge the spatio-temporal receptive field in the big view branch, in order to find the most activated context to enhance core representations in the small view branch. Moreover, we propose a parameter sharing scheme in our design, which allows us to make the SmallBig network compact. Finally, all the experiments show that, our SmallBig network is an accurate and efficient model for large-scale video classification.

Acknowledge This work is partially supported by Science and Technology Service Network Initiative of Chinese Academy of Sciences (KFJ-STS-QYZX-092), Guangdong Special Support Program (2016TX03X276), and National Natural Science Foundation of China (61876176, U1713208), Shenzhen Basic Research Program (JCYJ20170818164704758, CXB201104220032A), the Joint Lab of CAS-HK, Shenzhen Institute of Artificial Intelligence and Robotics for Society.

References

[1] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1, 2, 3, 7

[2] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A^2-nets: Double attention networks. In *NeuIPS*, 2018. 7

[3] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Multi-fiber networks for video recognition. In *ECCV*, 2018. 7

[4] Yunpeng Chen, Marcus Rohrbach, Zhicheng Yan, Shuicheng Yan, Jiashi Feng, and Yannis Kalantidis. Graph-based global reasoning networks. In *CVPR*, 2019. 2, 7

[5] Ali Diba, Mohsen Fayyaz, Vivek Sharma, M.Mahdi Arzani, Rahman Yousefzadeh, Juergen Gall, and Luc Van Gool. Spatio-temporal channel correlation networksfor action classification. In *ECCV*, 2018. 2, 7

[6] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, 2014. 2

[7] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 1, 2, 3, 5, 7

[8] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, 2016. 2

[9] C Feichtenhofer, A Pinz, and A Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 2

[10] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *ICCV*, 2017. 2, 5

[11] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, 2018. 2

[12] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D Convolutional Neural Networks for Human Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013. 2

[13] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. In *arXiv:1705.06950*, 2017. 2, 5

[14] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Collaborative spatiotemporal feature learning for video action recognition. In *CVPR*, 2019. 7

[15] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019. 2, 7

[16] Xingyu Liu, Joon-Young Lee, and Hailin Jin. Learning video representations from correspondence proposals. In *CVPR*, 2019. 2, 7

[17] Zhaoyang Liu, Donghao Luo, Yabiao Wang, Li-Jae Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Tong Lu. Teinet: Towards an efficient architecture for video recognition. *ArXiv*, abs/1911.09435, 2019. 7

[18] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. *CoRR*, abs/1503.08909, 2015. 2

[19] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *ICCV*, 2017. 2

[20] Shikhar Sharma, Ryan Kiros, and Ruslan Salakhutdinov. Action recognition using visual attention. *CoRR*, 2015. 2

[21] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *CoRR*, 2014. 2

[22] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015. 2

[23] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018. 1, 2, 7

[24] Limin Wang, Wei Li, Wen Li, and Luc Van Gool. Appearance-and-relation networks for video classification. In *CVPR*, 2018. 7

[25] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015. 2

[26] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, 2017. 2

[27] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 2

[28] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2, 3, 4, 5, 6, 7

[29] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, 2018. 2, 7

[30] Wenhao Wu, Dongliang He, Xiao Tan, Shifeng Chen, and Shilei Wen. Multi-agent reinforcement learning based frame sampling for effective untrimmed video recognition. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2

[31] Wenhao Wu, Dongliang He, Xiao Tan, Shifeng Chen, Yanming Yang, and Shilei Wen. Dynamic inference: A new approach toward efficient video action recognition. *ArXiv*, abs/2002.03342, 2020. 2

[32] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv:1712.04851*, 2017. 1, 2, 7

[33] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *ECCV*, 2018. 2, 7

[34] Wangjiang Zhu, Jie Hu, Gang Sun, Xudong Cao, and Yu Qiao. A key volume mining deep framework for action recognition. In *CVPR*, 2016. 2

[35] Mohammadreza Zolfaghari, Kamaljeet Singh, and Thomas Brox. Eco: Efficient convolutional network foronline video understanding. In *ECCV*, 2018. 7