IEICE TRANS. FUNDAMENTALS, VOL.E104–A, NO.6 JUNE 2021

INVITED PAPER *Special Section on Image Media Quality*

Recent Advances in Video Action Recognition with 3D Convolutions

Kensho HARA[†a)], ***Member***

SUMMARY The performance of video action recognition has improved significantly in recent decades. Current recognition approaches mainly utilize convolutional neural networks to acquire video feature representations. In addition to the spatial information of video frames, temporal information such as motions and changes is important for recognizing videos. Therefore, the use of convolutions in a spatiotemporal three-dimensional (3D) space for representing spatiotemporal features has garnered significant attention. Herein, we introduce recent advances in 3D convolutions for video action recognition.

key words: *video recognition, action recognition, 3D convolutions, survey*

1. Introduction

The use of deep convolutional neural networks (CNNs) in the field of computer vision has expanded significantly in recent years. ImageNet [1], which includes more than a million images, and other large-scale image datasets have contributed substantially to the creation of successful vision-based algorithms because the use of large-scale datasets is extremely important when using deep CNNs, which involve a large number of parameters. In addition to such large-scale datasets, a large number of algorithms, such as those for batch normalization [2] and residual learning [3], have been used to improve image classification performance. Feature representations acquired using very deep CNNs trained on large-scale datasets offer high generalization performance. Using such a feature representation, can significantly improve the performance of several other tasks, including object detection [4], segmentation [5], and image captioning [6].

The performance of video action recognition, which is a task to recognize human actions in videos, has improved significantly by the development of deep CNNs for videos. Similar to image recognition, CNNs with two-dimensional (2D) convolutions pretrained on ImageNet were initially used. A two-stream architecture, which is a popular approach for CNN-based action recognition [7], uses RGB and stacked optical flow frames as appearance and motion information, respectively; it demonstrates that combining two streams improves action recognition accuracy. Numerous methods based on two-stream CNNs have been proposed to achieve further improvements by introducing the advantage of handcrafted features [8], different combining methods of two streams [9]–[11], and modeling of long-range temporal structures [12]. Wang et al. attempted to achieve good practice for the training of very deep two-stream 2D CNNs [13].

Recently, CNNs with spatio–temporal three-dimensional (3D) convolutional CNNs (3D CNNs) have been shown to be more effective than CNNs with 2D kernels in action recognition [14]. Although 3D CNNs have been investigated for action recognition several years ago [15], they do not exhibit the advantages of two-stream 2D-based CNNs [16]. This is primarily due to the relatively small data scale of video datasets that are available for optimizing the immense number of parameters in 3D CNNs, which are much larger than those of 2D CNNs. In addition, 3D CNNs can only be trained on video datasets, whereas 2D CNNs can be pretrained on ImageNet. Recently, however, Carreira and Zisserman achieved a significant breakthrough using the Kinetics dataset, which includes large-scale videos. They also introduced an inflation of 2D kernels pretrained on ImageNet into 3D ones [14]. Following these studies, 3D convolutions are mainly adopted in action recognition methods.

Herein, we introduce recent advances in 3D convolutions for video action recognition. Action recognition approaches by handcrafted methods and early methods of deep learning have been summarized by Aggarwal and Ryoo [17] and Herath et al. [18]. In contrast to these surveys, we focus on 3D convolutions herein. Section 2 explains the basics of 2D and 3D convolutions for action recognition, whereas Sect. 3 describes various network architectures with 3D convolutions. We explain various state-of-the-art advances in Sect. 4 and introduce video datasets for training and evaluation in action recognition in Sect. 5. Finally, Sect. 6 concludes this paper.

2. Convolutions for Video Action Recognition

We explain the convolutions used for video action recognition. As described in the previous section, many studies utilized 2D convolutions for videos a few years ago, similar to image recognition. Recently, 3D convolutions have been mainly used in action recognition tasks because of the release of large-scale video datasets. Furthermore, (2+1)D convolutions, which are extended versions of 3D convolutions, contributed to high recognition performances. Figure 1 shows an overview of the convolutions. In the following paragraphs, we introduce each convolution.

Manuscript received June 30, 2020.
Manuscript revised October 18, 2020.
Manuscript publicized December 7, 2020.

[†]The author is with the National Instutite of Advanced Industrial Science and Technology, Tsukuba-shi, 305-8560 Japan.
a) E-mail: kensho.hara@aist.go.jp

DOI: 10.1587/transfun.2020IMP0012

Copyright © 2021 The Institute of Electronics, Information and Communication Engineers

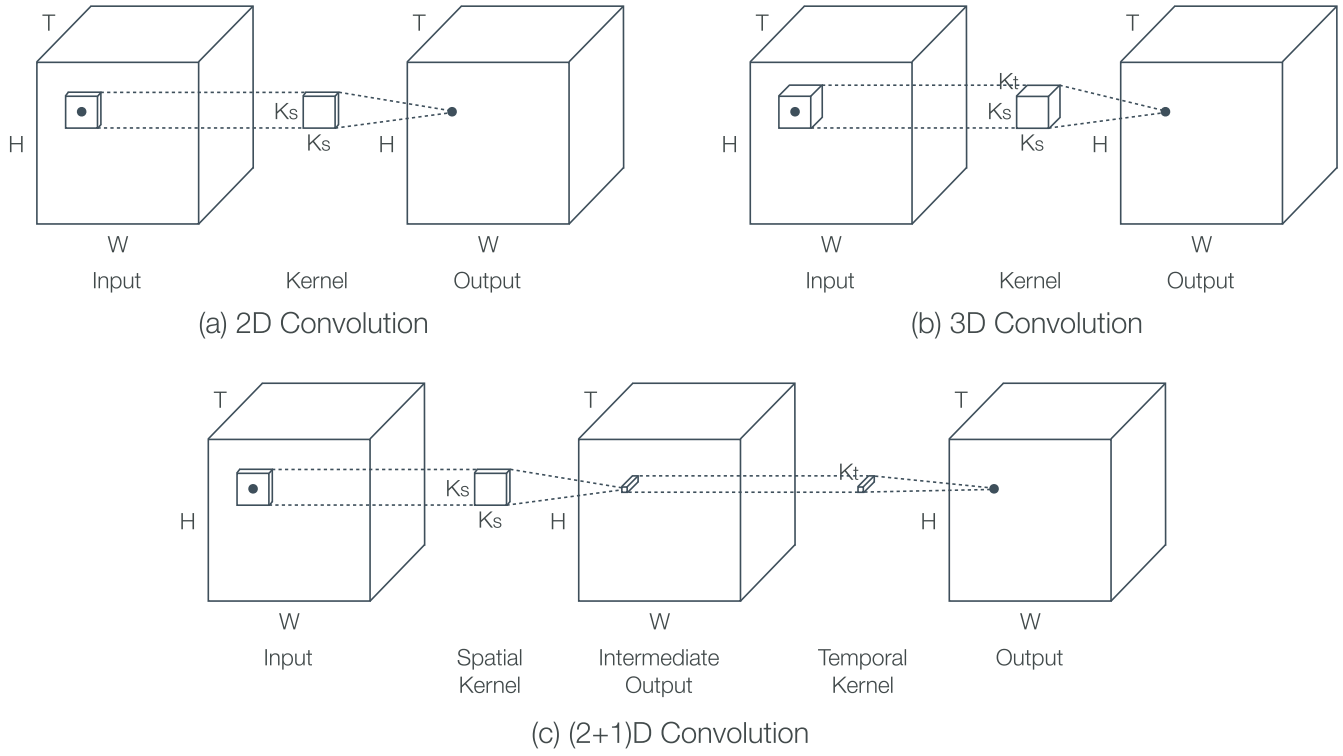

Fig. 1 Convolutions for video action recognition. W, H, and T are width, height, and number of frames of the feature maps, respectively. 2D and 3D convolutions apply $1 \times K_s \times K_s$ and $K_t \times K_s \times K_s$ convolutional kernels, respectively. (2+1)D convolutions apply a $1 \times K_s \times K_s$ kernel at first, and then apply a $K_t \times 1 \times 1$ kernel separately. Note that this figure does not show the number of feature maps.

When applying a 2D $K_s \times K_s$ convolutional kernel into a video, i.e., a spatiotemporal 3D feature map, we applied it for each frame separately. This process is equivalent to directly applying a $1 \times K_s \times K_s$ kernel to a 3D feature map, as shown in Fig. 1(a). When applying $1 \times K_s \times K_s$ convolutional kernels with N channels into a $T \times H \times W$ input feature map with M channels, the computational cost and number of parameters are $THWMK_s^2N$ and MK_s^2N, respectively. Some studies utilized a video as a feature map with 3 (RGB) $\times T$ (the number of frames) channels [7], [8], [13].

The size of a 3D convolutional kernel in the time dimension differs from that of a 2D one, as shown in Fig. 1(b). When applying $K_t \times K_s \times K_s$ convolutional kernels with N channels into a $T \times H \times W$ input feature map with M channels, the computational cost and number of parameters are $THWMK_s^2K_tN$ and $MK_s^2K_tN$, respectively. Compared with 2D convolutions, the computational cost and number of parameters are increased by K_t.

A (2+1)D convolution is a pseudo 3D convolution and comprises a spatial 2D convolution and a temporal 1D convolution, as shown in Fig. 1(c). The separated convolutions reduce the computational cost and number of parameters while facilitating the training of deep neural networks. When applying $1 \times K_s \times K_s$ and $K_t \times 1 \times 1$ convolutional kernels with N_s and N_t channels, respectively, into a $T \times H \times W$ input feature map with M channels, the computational cost and number of parameters are $THWMK_s^2N_s + THWN_sK_tN_t = THW(MK_s^2N_s + N_sK_tN_t)$ and $MK_s^2N_s + N_sK_tN_t$, respectively. When $N_t = N$ and $N_s = (MK_s^2K_tN)/(MK_s^2 + NK_t)$, the computational cost and number of parameters are the same as those of 3D convolutions. Tran et al. demonstrated that (2+1)D convolutions outperformed 3D convolutions in action recognition when the number of parameters of (2+1)D convolutions was the same as that of 3D convolutions [19].

3. Network Architectures with 3D Convolutions

In this section, we introduce 3D CNN architectures for action recognition. In the tables presented, the input sizes of the network architectures listed are based on original publications.

3.1 C3D

C3D proposed by Tran et al. [20] is a popular 3D CNN architecture. Table 1 shows the network architecture of C3D. The architecture of C3D is based on VGG-11 [21] proposed for image classification; 2D convolutions with 3×3 kernels in VGG-11 are replaced with 3D convolutions with $3 \times 3 \times 3$ kernels in C3D. All activation functions of C3D are ReLU [22]. The best kernel temporal depth demonstrated by Tran et al. was three. This value has been used in subsequent architectures.

Table 1 Network architecture of C3D. *C* is the number of classes. *conv*, *maxpool*, and *fc* are the convolutional, max pooling, and fully connected layers, respectively.

Layer	Output Size	Kernel size	Stride
input	$3 \times 16 \times 112 \times 112$		
conv	$64 \times 16 \times 112 \times 112$	$3 \times 3 \times 3$	1, 1, 1
maxpool	$64 \times 16 \times 56 \times 56$	$1 \times 2 \times 2$	1, 2, 2
conv	$128 \times 16 \times 56 \times 56$	$3 \times 3 \times 3$	1, 1, 1
maxpool	$128 \times 8 \times 28 \times 28$	$2 \times 2 \times 2$	2, 2, 2
conv ×2	$256 \times 8 \times 28 \times 28$	$3 \times 3 \times 3$	1, 1, 1
maxpool	$256 \times 4 \times 14 \times 14$	$2 \times 2 \times 2$	2, 2, 2
conv ×2	$512 \times 4 \times 14 \times 14$	$3 \times 3 \times 3$	1, 1, 1
maxpool	$512 \times 2 \times 7 \times 7$	$2 \times 2 \times 2$	2, 2, 2
conv ×2	$512 \times 2 \times 7 \times 7$	$3 \times 3 \times 3$	1, 1, 1
maxpool	$512 \times 1 \times 4 \times 4$	$2 \times 2 \times 2$	2, 2, 2
fc	4096		
fc	4096		
fc	*C*		

C3D is expected to yield spatio–temporal feature representations because it utilizes spatio–temporal 3D convolutions. However, similar to 2D CNNs, the two-stream ensembling of RGB frames and stacked optical flows improve the recognition performance of C3D [16]. Such results have been reported in other 3D architectures [14], [19], [23] as well.

3.2 I3D

An inflated 3D CNN (I3D) [14] is a 3D convolutional network architecture based on GoogLeNet (inception-v1) [24]. Table 2 shows the architecture of an I3D, and Fig. 2 shows the inception module, which is proposed to efficiently compute wider and deeper networks in GoogLeNet. The output of an I3D based on a video clip includes the class probabilities of multiple frames. The average probability of the frames should be calculated to acquire the class probabilities of the video clip.

An important point in an I3D is inflation, i.e., the conversion of successful 2D classification models into 3D ones. The inflation operation repeats the weights of 2D kernels N times along the time dimension and rescales them by dividing by N. Carreira and Zisserman demonstrated that the 3D inflated models of 2D GoogLeNet pretrained on ImageNet achieved better performances compared with non-inflated models [14].

One of the techniques to extend 2D GoogLeNet to 3D I3D is by configuring the kernel temporal depths and strides of the max pooling layers. The depths and strides of the first and second max pooling layers are not three, which is the value of the spatial dimensions, but one. Carreira and Zisserman described that they configured the depths and strides to grow the receptive field gradually in time and to avoid conflating edges from different objects, which can hinder early feature detection [14].

Table 2 Network Architecture of I3D. *C* is the number of classes; *conv*, *maxpool*, and *avgpool* denote the convolutional, max pooling, and average pooling layers, respectively. *Inception* indicates the inception module, as shown in Fig. 2.

Layer	Output Size	Kernel size	Stride
input	$3 \times 64 \times 224 \times 224$		
conv	$64 \times 32 \times 112 \times 112$	$7 \times 7 \times 7$	2, 2, 2
maxpool	$64 \times 32 \times 56 \times 56$	$1 \times 3 \times 3$	1, 2, 2
conv	$64 \times 32 \times 56 \times 56$	$1 \times 1 \times 1$	1, 1, 1
conv	$192 \times 32 \times 56 \times 56$	$3 \times 3 \times 3$	1, 1, 1
maxpool	$192 \times 32 \times 28 \times 28$	$1 \times 3 \times 3$	1, 2, 2
inception ×2	$480 \times 32 \times 28 \times 28$		
maxpool	$480 \times 16 \times 14 \times 14$	$3 \times 3 \times 3$	2, 2, 2
inception ×5	$832 \times 16 \times 14 \times 14$		
maxpool	$832 \times 8 \times 7 \times 7$	$3 \times 3 \times 3$	2, 2, 2
inception ×2	$1024 \times 8 \times 7 \times 7$		
avgpool	$1024 \times 8 \times 1 \times 1$	$2 \times 7 \times 7$	1, 1, 1
conv	$C \times 8$	1, 1, 1	1, 1, 1

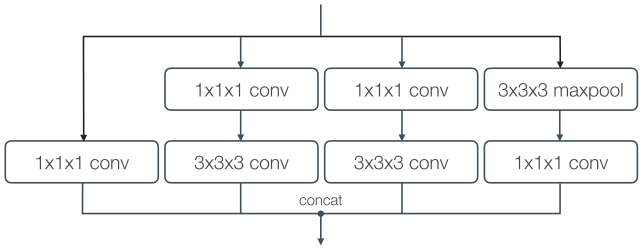

Fig. 2 3D inception module. This 3D inception module replaces the 5×5 convolutional layer of the 2D inception module [24] with not a $5 \times 5 \times 5$ layer but a $3 \times 3 \times 3$ layer.

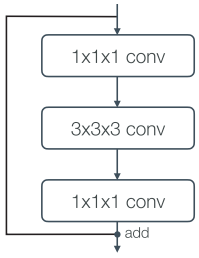

Fig. 3 3D bottleneck residual module.

3.3 3D ResNet

3D ResNet [25]–[27], also known as R3D [19], is a 3D version of the 2D residual network (ResNet) [3]. ResNet provides shortcut connections that allow a signal to bypass one layer and move to the next layer in a sequence, as shown in Fig. 3. Because these connections pass through the network's gradient flows from the latter layers to the early layers, they can facilitate the training of very deep networks. Table 3 shows the architecture of 3D ResNet-50.

The recognition performances of deeper 3D ResNets trained on a large-scale video dataset, such as Kinetics [28],

Table 3 Network Architecture of 3D ResNet-50. C is the number of classes; *conv*, *maxpool*, *avgpool*, and *fc* denote the convolutional, max pooling, average pooling, and fully connected layers, respectively. *Residual* indicates the residual module shown in Fig. 3.

Layer	Output Size	Kernel size	Stride
input	$3 \times 16 \times 112 \times 112$		
conv	$64 \times 16 \times 56 \times 56$	$7 \times 7 \times 7$	1, 2, 2
maxpool	$64 \times 8 \times 28 \times 28$	$3 \times 3 \times 3$	2, 2, 2
residual $\times 3$	$256 \times 8 \times 28 \times 28$	$\begin{bmatrix} 1 \times 1 \times 1, 64 \\ 3 \times 3 \times 3, 64 \\ 1 \times 1 \times 1, 256 \end{bmatrix}$	1, 1, 1
residual $\times 4$	$512 \times 4 \times 14 \times 14$	$\begin{bmatrix} 1 \times 1 \times 1, 128 \\ 3 \times 3 \times 3, 128 \\ 1 \times 1 \times 1, 512 \end{bmatrix}$	2, 2, 2
residual $\times 6$	$1024 \times 2 \times 7 \times 7$	$\begin{bmatrix} 1 \times 1 \times 1, 256 \\ 3 \times 3 \times 3, 256 \\ 1 \times 1 \times 1, 1024 \end{bmatrix}$	2, 2, 2
residual $\times 3$	$2048 \times 1 \times 4 \times 4$	$\begin{bmatrix} 1 \times 1 \times 1, 512 \\ 3 \times 3 \times 3, 512 \\ 1 \times 1 \times 1, 2048 \end{bmatrix}$	2, 2, 2
avgpool	$2048 \times 1 \times 1 \times 1$	$1 \times 7 \times 7$	1, 1, 1
fc	C		

are better than those of shallower models, similar to the image recognition performance of 2D ResNets [3]. Hara et al. demonstrated that improvements using deeper models continued until a depth of 152 was reached [26]. Extended versions of ResNets, such as pre-activation ResNet [29], wide ResNet [30], and ResNeXt [31], have also been investigated, in which ResNeXt-101 achieved the best performance [26].

3.4 P3D, R(2+1)D, and S3D

P3D [32], R(2+1)D [19], and S3D [23] utilize pseudo 3D convolutions; currently, they are often known as (2+1)D convolutions, as described in Sect. 2. (2+1)D convolutions apply spatial 2D convolutions, followed by temporal 1D convolutions. The same idea has been introduced in factorized spatio–temporal convolutional networks [33]. The network architectures of P3D and R(2+1)D are based on the 3D ResNet, and S3D is an I3D-based architecture. These models replace 3D convolutions with (2+1)D convolutions.

P3D introduces different designs of the pseudo 3D convolutions. P3D-A adopts the same design as (2+1)D convolutions.

$$\mathbf{y} = f_t(f_s(\mathbf{x})), \tag{1}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the input and output of the convolution; f_s and f_t are functions that apply 2D spatial and 1D temporal convolutions, respectively. P3D-B applies both spatial 2D and temporal 1D kernels in different pathways in a parallel manner and then accumulates both pathways into the final output.

$$\mathbf{y} = f_s(\mathbf{x}) + f_t(\mathbf{x}). \tag{2}$$

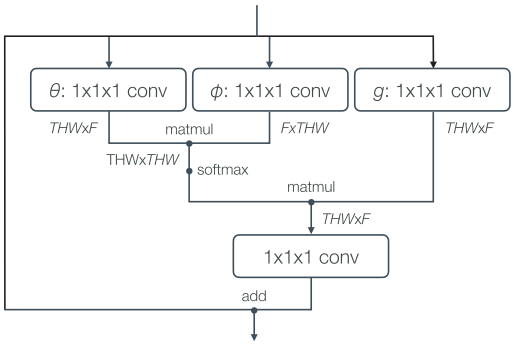

Fig. 4 Non-local module.

P3D-C utilizes a design combining those of P3D-A and P3D-B.

$$\mathbf{y} = f_t(f_s(\mathbf{x})) + f_s(\mathbf{x}). \tag{3}$$

All designs are used with skip connections, similar to the residual module in P3D. Qiu et al. demonstrated that combining the three designs in the network yielded the best performance [32].

Xie et al. investigated top-heavy and bottom-heavy S3Ds, which utilize (2+1)D convolutions in some layers and 2D convolutions in the remaining ones [23]. They demonstrated that a model that maintained the top two layers as (2+1)D convolutions, and the remaining as 2D convolutions yielded a good speed-accuracy tradeoff, and that using (2+1)D convolutions in all layers yielded the best accuracy. Furthermore, they also proposed S3D-G, which introduces a self-attention mechanism after each temporal convolutions in S3D. Its attention mechanism improved the recognition performance.

3.5 Non-Local Neural Networks

Non-local neural networks utilize non-local operations to capture long-term dependencies of actions [34]. Non-local operations compute the response at a position as a weighted sum of the features at all positions in the input feature maps, as shown in Fig. 4.

$$\mathbf{y}_i = \frac{1}{\sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j)} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j), \tag{4}$$

$$f(\mathbf{x}_i, \mathbf{x}_j) = e^{\theta(\mathbf{x}_i)\phi(\mathbf{x}_j)}, \tag{5}$$

where i is the index of an output spatio-temporal position; j is the index that enumerates all possible positions; g, θ, and ϕ are embedded representations. Because the operation considers all spatio-temporal positions, the operation can capture long-term dependencies, which are difficult to capture by convolutions.

Wang et al. used 3D ResNet-50 based non-local network architectures [34], where non-local blocks were inserted into the third and fourth residual blocks of the 3D ResNet-50. They discovered that the non-local 3D ResNet-50 achieved better performances than 3D ResNet-101 with-

Table 4 Network Architecture of SlowFast 3D ResNet-50. C is the number of classes. *conv*, *maxpool*, *avgpool*, and *fc* mean convolutional, max pooling, average pooling, and fully-connected layers, respectively. *concat* concatenates the outputs of slow and fast pathways.

Layer	Slow pathway			Fast pathway		
	Output Size	Kernel size	Stride	Output Size	Kernel size	Stride
input	$3 \times 4 \times 224 \times 224$		$3 \times 32 \times 224 \times 224$			
conv	$64 \times 4 \times 112 \times 112$	$1 \times 7 \times 7$	1, 2, 2	$8 \times 32 \times 112 \times 112$	$5 \times 7 \times 7$	1, 2, 2
maxpool	$64 \times 4 \times 56 \times 56$	$1 \times 3 \times 3$	1, 2, 2	$8 \times 32 \times 56 \times 56$	$1 \times 3 \times 3$	1, 2, 2
residual $\times 3$	$256 \times 4 \times 56 \times 56$	$\begin{bmatrix} 1 \times 1 \times 1, 64 \\ 1 \times 3 \times 3, 64 \\ 1 \times 1 \times 1, 256 \end{bmatrix}$	1, 1, 1	$32 \times 32 \times 56 \times 56$	$\begin{bmatrix} 3 \times 1 \times 1, 8 \\ 1 \times 3 \times 3, 8 \\ 1 \times 1 \times 1, 32 \end{bmatrix}$	1, 1, 1
residual $\times 4$	$512 \times 4 \times 28 \times 28$	$\begin{bmatrix} 1 \times 1 \times 1, 128 \\ 1 \times 3 \times 3, 128 \\ 1 \times 1 \times 1, 512 \end{bmatrix}$	1, 2, 2	$64 \times 32 \times 28 \times 28$	$\begin{bmatrix} 3 \times 1 \times 1, 16 \\ 1 \times 3 \times 3, 16 \\ 1 \times 1 \times 1, 64 \end{bmatrix}$	1, 2, 2
residual $\times 6$	$1024 \times 4 \times 14 \times 14$	$\begin{bmatrix} 3 \times 1 \times 1, 256 \\ 1 \times 3 \times 3, 256 \\ 1 \times 1 \times 1, 1024 \end{bmatrix}$	1, 2, 2	$128 \times 32 \times 14 \times 14$	$\begin{bmatrix} 3 \times 1 \times 1, 32 \\ 1 \times 3 \times 3, 32 \\ 1 \times 1 \times 1, 128 \end{bmatrix}$	1, 2, 2
residual $\times 3$	$2048 \times 4 \times 7 \times 7$	$\begin{bmatrix} 3 \times 1 \times 1, 512 \\ 1 \times 3 \times 3, 512 \\ 1 \times 1 \times 1, 2048 \end{bmatrix}$	1, 2, 2	$256 \times 32 \times 7 \times 7$	$\begin{bmatrix} 3 \times 1 \times 1, 64 \\ 1 \times 3 \times 3, 64 \\ 1 \times 1 \times 1, 256 \end{bmatrix}$	1, 2, 2
avgpool	$2048 \times 1 \times 1 \times 1$	$4 \times 7 \times 7$	1, 1, 1	$256 \times 1 \times 1 \times 1$	$32 \times 7 \times 7$	1, 1, 1
concat	$(2048 + 256) \times 1 \times 1 \times 1$					
fc	C		C			

out non-local blocks, even though the numbers of parameters and layers of the non-local 3D ResNet-50 were smaller.

3.6 SlowFast Networks

SlowFast networks [35] comprises two pathways that are designed based on biological studies. One pathway, known as the slow pathway, captures spatial semantics at a low frame rate, whereas the other one, known as the fast pathway, captures motion at a fine temporal resolution. Table 4 shows the architecture of the SlowFast network. The temporal sizes of the inputs of both pathways are based on the frame rate of the input video instead of its duration. The network only utilizes pure 3D convolutions in the first convolution of the fast pathway. The other convolutions are 2D spatial or 1D temporal convolutions. The slow pathway uses temporal convolutions only in the third and fourth residual blocks because using temporal convolutions in the early layers degrades the accuracy [35]. The fast pathway has a smaller number of feature maps compared with the slow pathway for an efficient computation. Such a thinner architecture of the fast pathway degrades the performance only slightly. The frame rates of the slow and fast pathways used are $1/16$ and $1/2$ of the original videos, respectively. The SlowFast network includes lateral connections between the two pathways to fuse information, similar to two-stream architectures [9]–[11].

3.7 X3D

X3D is an efficient network architecture for video action recognition [36]. The architecture is based on a 2D ResNet structure and the Fast pathway design of SlowFast networks. Feichtenhofer discovered the most efficient architecture by progressively expanding a small base 2D architecture across the following axes: temporal duration, frame rate, spatial resolution, network width, bottleneck width, and depth. One of the architectures, known as X3D-M, is shown in Table 5. X3D-M is expanded across the bottleneck width, temporal resolution, spatial resolution, depth, temporal duration, temporal resolution, temporal duration, and spatial resolution, in that order. These progressive expansions indicate that networks with thin channel dimensions and high spatiotemporal resolution can be effective for video action recognition. Larger X3D-XL achieved performances similar to those of a method combining SlowFast and non-local networks whereas the combined method requires 4.8 times larger multiply–add computations compared with the X3D-XL. The performance of X3D-M is slightly worse than that of X3D-XL although its multiply–add computations are 7.8 times fewer.

3.8 Summary

As above-mentioned, various 3D CNN architectures have been proposed. ResNets, described in Sect. 3.3, were often

Table 5 Network Architecture of X3D-M. *C* is the number of classes; *conv*, *avgpool*, and *fc* denote the convolutional, average pooling, and fully connected layers, respectively. *Subsample* subsamples the input videos in the time dimension.

Layer	Output Size	Kernel size	Stride
input	$3 \times 80 \times 224 \times 224$		
subsample	$3 \times 16 \times 224 \times 224$		5, 1, 1
conv	$24 \times 16 \times 112 \times 112$	$1 \times 3 \times 3, 3 \times 1 \times 1$	1, 2, 2
residual $\times 3$	$24 \times 16 \times 56 \times 56$	$\begin{bmatrix} 1 \times 1 \times 1, 54 \\ 3 \times 3 \times 3, 54 \\ 1 \times 1 \times 1, 24 \end{bmatrix}$	1, 2, 2
residual $\times 5$	$48 \times 16 \times 28 \times 28$	$\begin{bmatrix} 1 \times 1 \times 1, 108 \\ 3 \times 3 \times 3, 108 \\ 1 \times 1 \times 1, 48 \end{bmatrix}$	1, 2, 2
residual $\times 11$	$96 \times 16 \times 14 \times 14$	$\begin{bmatrix} 1 \times 1 \times 1, 216 \\ 3 \times 3 \times 3, 216 \\ 1 \times 1 \times 1, 96 \end{bmatrix}$	1, 2, 2
residual $\times 7$	$192 \times 16 \times 7 \times 7$	$\begin{bmatrix} 1 \times 1 \times 1, 432 \\ 3 \times 3 \times 3, 432 \\ 1 \times 1 \times 1, 192 \end{bmatrix}$	1, 2, 2
conv	$432 \times 16 \times 7 \times 7$	$1 \times 1 \times 1$	1, 1, 1
avgpool	$432 \times 1 \times 1 \times 1$	$16 \times 7 \times 7$	1, 1, 1
fc	$2048 \times 1 \times 1 \times 1$		
fc	C		

used in recent studies as a backbone architecture because they contains various depth configurations, such as ResNet-18, -34, -50, -101, -152, and -200. We can select the depth configuration based on speed-accuracy tradeoff. Instead of 3D convolutions, (2+1)D convolutions (i.e. R(2+1)D), described in Sect. 3.4, were also often used and improved the recognition accuracy whereas using (2+1)D convolutions sometimes increases memory requirements in the training steps as the number of layers increases by two times (2D and 1D convolutions). Adding non-local operations, SlowFast pathways, and optical flow stream helps further improvements of the recognition accuracy.

4. Analyses and Improvements of 3D CNNs

4.1 Analyses of 3D CNNs

3D CNNs are analyzed from various aspects. We classify some analyses and explain them in this section.

4.1.1 Input Sizes

Varol et al. analyzed the temporal duration of the input to 3D CNNs [16]. They indicated that the temporal duration of C3D (= 16 frames) was insufficient to capture spatio–temporal structures of the actions. They investigated the effect of the temporal duration on the recognition accuracy by varying the durations. Their results indicated that longer durations resulted in higher accuracies. Similar results have also been reported in [26], [34], [36].

In addition to the temporal duration, spatial and temporal resolutions affect the recognition performance. Higher spatio–temporal resolutions yielded higher recognition accuracies, and the contribution was significant, as described in Sect. 3.7. However, a tradeoff exists between recognition accuracy and computational costs. A study has been conducted to improve the tradeoff in training, which will be described later in Sect. 4.3.

4.1.2 Motion Information

Ensembling with RGB frames (appearance) and optical flows (motion) improves recognition performance even when using 3D CNNs [14], [16], [19], which can theoretically acquire spatio–temporal representations. This indicates that 3D CNNs cannot utilize motion information when classifying videos. In this regard, Huang et al. tested a model trained with full-length videos on a single subsampled frame (i.e., without motion information) [37]. However, they observed that subsampling resulted in a distribution shift in the temporal dimension, and that subsampling occasionally removed important frames for recognizing the action. To avoid artifacts, they proposed two methods that compensate for the distribution gap and selected the appropriate frames without including any motion information. Their experimental results indicated that when using their proposed methods, the recognition accuracy decreased by only 5% from the 47% accuracy on the Kinetics dataset when evaluating tighter upper bounds without motion information. It is considered likely that their result also indicates that 3D CNNs did not focus on motion information in classification of the videos of Kinetics. Motivated by these results, some studies have been conducted to improve motion representation of 3D CNNs, as described in Sect. 4.2.

4.1.3 Training Data

The scale of training data is important for training 3D CNNs. Kay et al. demonstrated that C3D achieved slightly better performances compared with 2D ResNet-50 when the model was trained on a large-scale Kinetics dataset, although the accuracy of C3D trained on the relatively small UCF-101 dataset was low [28]. Kataoka et al. compared the performance of models pretrained on various datasets [38], such as Kinetics-700 [39] and Moments in Time [40]. Their results indicated that Kinetics-700 pretrained on 3D ResNet-50 achieved the best performance compared with the model pretrained on the other datasets. In addition, to increase the amount of training data using public datasets, they concatenated the Kinetics-700 and Moments in Time datasets from 650K videos/700 and 1M videos/339 categories, respectively, into 1.65M videos/1,039 categories and pretrained 3D ResNet-50 on the concatenated dataset. Their experiment demonstrated that pretraining on the concatenated dataset further improved the accuracy. Ghadiyaram et al. trained 3D CNNs on 65 million videos obtained from a social media website [41]. The video labels were annotated based on the

hash tags of videos, which often include noise. They demonstrated that pretraining on such large-scale videos with noisy labels significantly improved the recognition performance, and that the performance of R(2+1)D-34 improved log-linearly with training data size.

Data augmentation methods for 3D CNNs have been discussed as well. Hara et al. compared various strategies for cropping video clips [42]. Their results indicated that spatio–temporal crop positions should be selected randomly, and that the spatial scale of the cropping should be random but the temporal scale should be fixed.

4.2 Improvements in Motion Representation

Recently, studies have been conducted to improve motion representation. As described in Sect. 4.1.2, some problems are encountered in the motion representation of 3D CNNs. One of the solutions is to use optical flows, but the computational cost of calculating optical flows is high. Inspired by the TV-L1 algorithm [43], which is an optical flow estimation method, Piergiovanni et al. proposed a CNN layer to compute feature flows [44]. Their proposed representation flow layer unrolls the iteration computations of TV-L1 and implements similar computations as a differentiable layer with learnable parameters. Therefore, the representation flow layer can be optimized for action recognition. They applied their layer on lower-resolution feature maps for an efficient computation. Their experimental results showed that 3D and (2+1)D ResNet-18 with representation layers achieved higher accuracies than those without the layers. Inspired by the correlation layer of FlowNet [45], Wang et al. proposed a correlation operator [46], which is similar to the representation flow. The correlation operator computes the correlations between every pair of adjacent frames in the feature maps. Motion information is acquired based on frame-wise correlations (i.e., similarities). The correlation operator is differentiable and implemented with learnable parameters. They inserted their correlation operators into the first, second, and third residual blocks of R(2+1)D-26. Their network involving correlation operators and not requiring optical flows, known as CorrNet, achieved performances similar to those of two-stream backbone models. Zhu et al. used optimized optical flows for action recognition by introducing a hidden two-stream architecture that included an optical flow estimation module [47].

Stroud et al. attempted another approach involving distillation [48]. They trained a network that recognizes actions using optical flows as a teacher network and distills knowledge from a teacher to a student network that uses RGB frames. Because optical flows are only used by the teacher network, which is only used in training, the inference step is efficiently executed without computing flows. Their proposed D3D yielded performances that were comparable with the two-stream approach, without requiring an optical flow in inference. Similarly, ActionFlowNet involves multitask learning that optimizes action recognition and optical flow estimation simultaneously [49], instead of distillation.

Several studies have been conducted to acquire motion information using 2D-based architectures [50]–[54].

4.3 Efficient Training/Inference

It is important to ensure efficient training and inference with 3D CNNs, which can achieve high performances while maintaining a low computational cost. Wu et al. proposed an efficient training method using variable mini-batch shapes with different spatio–temporal resolutions [67]. As described in Sect. 4.1.1, spatio–temporal resolutions (input sizes) relate to the tradeoff between computational costs and recognition accuracies. Training with larger resolutions improves recognition accuracies as well as increases the training time. Their proposed method varies the resolutions from coarse to fine during training. Training SlowFast ResNet-50 using their method slightly improved the accuracy and decreased the training time by 4.5 times. SCSampler, proposed by Korbar et al., reduced the computational cost of action recognition [68]. SCSampler samples a reduced set of salient clips from a video. Recognizing actions using only salient clips is extremely efficient. SCSampler efficiently samples salient clips by directly using compressed videos without MPEG-4 and H.264 decodings, similar to compressed action recognition [69]. Their experimental results demonstrated that using SCSampler not only improved the accuracy, but also significantly reduced the computational cost.

Another approach has been proposed that uses group convolutions [31] and depthwise convolutions [70], which are more efficient than standard convolutions. Luo and Yuille proposed an architecture using group convolutions [71]. They decomposed a 3D $3\times3\times3, F$ convolution into 2D $1\times3\times3, (1-\alpha)F$ and 3D $3\times3\times3, \alpha F$ based on group convolutions. In addition to the efficient computation based on group convolutions, decomposition encourages the channels in each group to concentrate on static semantic features and dynamic motion features separately, thereby affording an easier training. Their proposed method enabled an efficient action recognition without decreasing the recognition accuracy. Tran et al. introduced depthwise convolutions into a 3D ResNet [72]. They replaced the $3 \times 3 \times 3$ convolution of a bottleneck block with a $1 \times 1 \times 1$ traditional convolution and a $3 \times 3 \times 3$ depthwise convolution. They demonstrated that using deeper architectures, such as those with 101 layers and combined with such depthwise convolutions, a slightly better performance was achieved compared with that of the traditional 3D ResNet, with three times fewer FLOPs.

5. Video Datasets

In this section, we introduce popular video datasets for action recognition. We summarize the statistics of the video datasets in Table 6. Although many video datasets exist for other video recognition tasks, such as spatio–temporal action detection [73], action segmentation [74], video caption-

Table 6 Statistics of video datasets for action recognition. *Year* shows not released years of the cited publications but the actual years of releasing datasets. *YouTube** in the row of Moments in Time means YouTube and other internet video sources, such as Flickr, Vine, and Vimeo.

Dataset	Year	Source	Trimmed/ Temporal annotations	Number of classes	Number of videos/instances
HMDB-51 [55]	2011	Movies/YouTube	Yes	51	6,766
UCF-101 [56]	2012	YouTube	Yes	101	13,320
Sports-1M [57]	2014	YouTube	No	487	1,133,158
ActivityNet [58]	2015	YouTube	Yes	200	28,108
Charades [59]	2016	Crowdsorcing	Yes	157	66,500
YouTube-8M [60]	2016	YouTube	No	4,800	8,264,650
Kinetics-400 [28]	2017	YouTube	Yes	400	306,245
Something-Something [61]	2017	Crowdsorcing	Yes	174	108,499
Moments in Time [40]	2018	YouTube*	Yes	339	1,000,000
STAIR-Actions [62]	2018	YouTube/Crowdsorcing	Yes	100	109,478
Kinetics-600 [63]	2018	YouTube	Yes	600	495,547
Something-Something-v2 [64]	2018	Crowdsorcing	Yes	174	220,847
Kinetics-700 [39]	2019	YouTube	Yes	700	650,317
HACS Clips [65]	2019	YouTube	Yes	200	1,500,000
HACS Segments [65]	2019	YouTube	Yes	200	139,000
FineGym [66]	2020	YouTube	Yes	530	32,697

ing [75], learning text-video embeddings [76], and understanding human–object relationships [77], we only discuss datasets used for action recognition.

5.1 Trimmed Video Datasets

Most video datasets contain videos that are temporally trimmed to remove non-action frames. The duration of each video ranges from several seconds to approximately 10 s. These datasets are used for action recognition, in which a video is used as an input and an action label is output.

UCF-101 [56] and HMDB-51 [55] are popular datasets for action recognition. Most videos are collected from YouTube except the videos from movies in HMDB-51. These datasets were released as large-scale video datasets in 2011 and 2012, respectively, and are used as popular benchmark datasets even in 2020. Nevertheless, studies that do not include the performances of the two datasets will increase gradually because the accuracies on the datasets were almost saturated.

Kinetics datasets are the most frequently used video datasets in 2020. Kinetics-400, which contains videos classified into 400 action classes, was released in 2017 [28]; subsequently, the dataset was extended to Kinetics-600 and -700 [39], [63], which have a higher number of classes. The videos in these datasets were collected from YouTube and annotated by crowd workers. Because Kinetics datasets include hundreds of thousands of videos, which are sufficient for training deep 3D CNNs [26], Kinetics pretrained on CNNs are used as base models in various tasks. Similar to Kinetics, Moments in Time, which include a larger number of action instances and a smaller number of classes than Kinetics, has been released [40]. HACS Clips [65] consists of 1.5M annotated clips sampled from 504K untrimmed videos. Zhao et al. demonstrated that recognition models pretrained on HACS Clips achieved better performances compared with the models pretrained on Kinetics-600, Moments in Time, and Sports1M datasets [65]. FineGym focuses on fine-grained action recognition in gymnastic videos and provides class labels in a three-level semantic hierarchy.

In addition to the datasets above, which were collected primarily from YouTube, datasets that include videos captured by crowd workers exist as well. Whereas YouTube-based datasets comprise actions in various domains, most crowdsourced datasets focus on daily activities. STAIR-Actions [62] includes daily actions at home, whereas Something-Something comprises daily object manipulations. Because Something-Something [61], [64] includes action pairs, which are difficult to classify without temporal information such as *pull something* and *push something*, it is used in studies discussing temporal feature representation [78], [79].

5.2 Untrimmed Video Datasets with Temporal Annotations

In addition to the datasets mentioned in the previous section, video datasets that contain untrimmed videos involving multiple actions exist. Charades [59] is a video dataset collected by crowd workers and focuses on daily activities. One video in the dataset includes approximately 6.8 actions on the average. Temporal annotations that indicate the start

and end positions in time for each action are annotated. Similar to Charades, ActivityNet [58] is an untrimmed YouTube video dataset in various domains. One video in ActivityNet includes approximately 1.4 actions on the average. Recently, larger untrimmmed datset known as HACS Segments, which contains 139K action segments densely annotated in 50K untrimmed videos spanning 200 action categories, is released [65].

These datasets can be used not only for action recognition by trimming videos based on temporal annotations, but also for temporal action localization, which estimates begin and end positions in the time of actions, as well as for untrimmed action recognition, which classify actions in videos that include non-action frames.

5.3 Untrimmed Video Datasets with Video-Level Labels

Instead of detailed annotations, large-scale video datasets with video-level labels have been released. Sports-1M [57] includes one million videos in the sports domain. YouTube-8M [60] includes eight million videos in various domains. A video in both datasets has one/multiple labels without beginning and end positions in time.

Although few studies have used these datasets because they are extremely large to be utilized easily and include noisy labels, some studies have used these datasets as large-scale video data for training deep models [20], [80].

6. Conclusion

We introduced various 3D convolutional network architectures and presented their analyses and improvements in video action recognition. Video action recognition has developed rapidly in recent years, whereas numerous methods and large-scale video datasets have been proposed. Models and datasets become larger every year; consequently, the computational resources required for video research has increased. Meanwhile, efficient training and inference methods have been developed recently (although some of them require significant resources to develop their algorithms). Improving motion representation of 3D CNNs is also interesting topic. Future studies will be conducted to address these current shortcomings.

References

[1] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.
[2] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," Proc. International Conference on Machine Learning, pp.448–456, 2015.
[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.770–778, 2016.
[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," Proc. Advances in Neural Information Processing Systems (NIPS), pp.1–9, 2015.
[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," Proc. International Conference on Computer Vision (ICCV), pp.2961–2969, 2017.
[6] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," Proc. International Conference on Machine Learning (ICML), pp.2048–2057, 2015.
[7] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," Proc. Advances in Neural Information Processing Systems (NIPS), pp.568–576, 2014.
[8] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.4305–4314, 2015.
[9] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," Proc. Advances in Neural Information Processing Systems (NIPS), pp.3468–3476, 2016.
[10] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1933–1941, 2016.
[11] C. Feichtenhofer, A. Pinz, and R.P. Wildes, "Spatiotemporal multiplier networks for video action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[12] 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," Proc. European Conference on Computer Vision (ECCV), pp.20–36, 2016.
[13] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao, "Towards good practices for very deep two-stream convnets," arXiv preprint, arXiv:1507.02159, 2015.
[14] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the Kinetics dataset," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.4724–4733, 2017.
[15] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol.35, no.1, pp.221–231, 2013.
[16] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol.40, no.6, pp.1510–1517, 2018.
[17] J. Aggarwal and M. Ryoo, "Human activity analysis: A review," ACM Comput. Surv., vol.43, no.3, 2011.
[18] S. Herath, M. Harandi, and F. Porikli, "Going deeper into action recognition: A survey," Image and Vision Computing, vol.60, pp.4–21, 2017.
[19] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.6450–6459, 2018.
[20] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," Proc. International Conference on Computer Vision (ICCV), pp.4489–4497, 2015.
[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Proc. International Conference on Learning Representations (ICLR), pp.1–14, 2015.
[22] V. Nair and G.E. Hinton, "Rectified linear units improve restricted Boltzmann machines," Proc. International Conference on Machine Learning, pp.807–814, Omnipress, 2010.
[23] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," Proc. European Conference on Computer Vision (ECCV), pp.318–335, 2018.
[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," Proc. IEEE Conference on Computer Vision and Pattern

Recognition (CVPR), pp.1–9, 2015.
[25] K. Hara, H. Kataoka, and Y. Satoh, "Learning spatio-temporal features with 3D residual networks for action recognition," Proc. ICCV Workshop on Action, Gesture, and Emotion Recognition, 2017.
[26] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and imageNet?," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.6546–6555, 2018.
[27] D. Tran, J. Ray, Z. Shou, S. Chang, and M. Paluri, "ConvNet architecture search for spatiotemporal feature learning," arXiv preprint, arXiv:1708.05038, 2017.
[28] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The Kinetics human action video dataset," arXiv preprint, arXiv:1705.06950, 2017.
[29] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," Proc. European Conference on Computer Vision (ECCV), pp.630–645, 2016.
[30] S. Zagoruyko and N. Komodakis, "Wide residual networks," Proc. British Machine Vision Conference (BMVC), 2016.
[31] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1492–1500, 2017.
[32] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3D residual networks," Proc. International Conference on Computer Vision (ICCV), 2017.
[33] L. Sun, K. Jia, D.Y. Yeung, and B.E. Shi, "Human action recognition using factorized spatio-temporal convolutional networks," Proc. International Conference on Computer Vision (ICCV), pp.4597–4605, 2015.
[34] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.7794–7803, 2018.
[35] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," Proc. International Conference on Computer Vision (ICCV), pp.6202–6211, 2019.
[36] C. Feichtenhofer, "X3D: Expanding architectures for efficient video recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.203–213, 2020.
[37] D.A. Huang, V. Ramanathan, D. Mahajan, L. Torresani, M. Paluri, L. Fei-Fei, and J. Carlos Niebles, "What makes a video a video: Analyzing temporal information in video understanding models and datasets," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.7366–7375, 2018.
[38] H. Kataoka, T. Wakamiya, K. Hara, and Y. Satoh, "Would mega-scale datasets further enhance spatiotemporal 3D CNNs?," arXiv preprint, arXiv:2004.04968, 2020.
[39] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, "A short note on the Kinetics-700 human action dataset," arXiv preprint, arXiv:1907.06987, 2019.
[40] M. Monfort, C. Vondrick, A. Oliva, A. Andonian, B. Zhou, K. Ramakrishnan, S.A. Bargal, T. Yan, L. Brown, Q. Fan, and D. Gutfreund, "Moments in time dataset: One million videos for event understanding," IEEE Trans. Pattern Anal. Mach. Intell., vol.42, no.2, pp.502–508, 2020.
[41] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.12046–12055, 2019.
[42] K. Hara, H. Kataoka, and Y. Satoh, "Towards good practice for action recognition with spatiotemporal 3D convolutions," Proc. International Conference on Pattern Recognition (ICPR), pp.2516–2521, 2018.
[43] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L^1 optical flow," Proc. 29th DAGM conference on Pattern recognition, pp.214–223, 2007.
[44] A. Piergiovanni and M.S. Ryoo, "Representation flow for action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.9945–9953, 2019.
[45] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "FlowNet: Learning optical flow with convolutional networks," Proc. International Conference on Computer Vision (ICCV), pp.2758–2766, 2015.
[46] H. Wang, D. Tran, L. Torresani, and M. Feiszli, "Video modeling with correlation networks," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.352–361, 2020.
[47] Y. Zhu, Z. Lan, S. Newsam, and A. Hauptmann, "Hidden two-stream convolutional networks for action recognition," Proc. Asian Conference on Computer Vision (ACCV), pp.363–378, 2018.
[48] J.C. Stroud, D.A. Ross, C. Sun, J. Deng, and R. Sukthankar, "D3D: Distilled 3D networks for video action recognition," Proc. IEEE Winter Conference on Applications of Computer Vision (WACV), pp.614–623, 2020.
[49] J.Y. Ng, J. Choi, J. Neumann, and L.S. Davis, "Actionflownet: Learning motion representation for action recognition," Proc. IEEE Winter Conference on Applications of Computer Vision (WACV), pp.1616–1624, 2018.
[50] B. Jiang, M. Wang, W. Gan, W. Wu, and J. Yan, "STM: Spatiotemporal and motion encoding for action recognition," Proc. International Conference on Computer Vision (ICCV), pp.2000–2009, 2019.
[51] J. Zhao and C.G.M. Snoek, "Dance with flow: Two-in-one stream action detection," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.9935–9944, 2019.
[52] S. Sun, Z. Kuang, L. Sheng, W. Ouyang, and W. Zhang, "Optical flow guided feature: A fast and robust motion representation for video action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1390–1399, 2018.
[53] M. Lee, S. Lee, S. Son, G. Park, and N. Kwak, "Motion feature network: Fixed motion filter for action recognition," Proc. European Conference on Computer Vision (ECCV), pp.392–408, 2018.
[54] J. Lin, C. Gan, and S. Han, "TSM: Temporal shift module for efficient video understanding," Proc. International Conference on Computer Vision (ICCV), pp.7083–7093, 2019.
[55] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," Proc. International Conference on Computer Vision (ICCV), pp.2556–2563, 2011.
[56] K. Soomro, A. Roshan Zamir, and M. Shah, "UCF101: A dataset of 101 human action classes from videos in the wild," CRCV-TR-12-01, 2012.
[57] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1725–1732, 2014.
[58] B.G. Fabian C. Heilbron, V. Escorcia, and J.C. Niebles, "ActivityNet: A large-scale video benchmark for human activity understanding," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.961–970, 2015.
[59] G.A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, "Hollywood in homes: Crowdsourcing data collection for activity understanding," Proc. European Conference on Computer Vision (ECCV), pp.510–526, 2016.
[60] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "YouTube-8M: A large-scale video classification benchmark," arXiv preprint, arXiv:1609.08675, 2016.
[61] R. Goyal, S.E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, F. Hoppe, C. Thurau, I. Bax, and R. Memisevic, "The "something something" video database for learning and evaluating visual common sense," Proc. International Conference on Computer Vision (ICCV), pp.5843–5851, 2017.

[62] Y. Yoshikawa, J. Lin, and A. Takeuchi, "STAIR Actions: A video dataset of everyday home actions," arXiv preprint, arXiv:1804.04326, 2018.
[63] J. Carreira, E. Noland, A. Banki-Horvath, C. Hillier, and A. Zisserman, "A short note about Kinetics-600," arXiv preprint, arXiv:1808.01340, 2018.
[64] F. Mahdisoltani, G. Berger, W. Gharbieh, D. Fleet, and R. Memisevic, "On the effectiveness of task granularity for transfer learning," arXiv preprint, arXiv:1804.09235, 2018.
[65] H. Zhao, A. Torralba, L. Torresani, and Z. Yan, "HACS: Human action clips and segments dataset for recognition and temporal localization," Proc. International Conference on Computer Vision (ICCV), pp.8668–8678, 2019.
[66] D. Shao, Y. Zhao, B. Dai, and D. Lin, "FineGym: A hierarchical video dataset for fine-grained action understanding," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.2616–2625, 2020.
[67] C.Y. Wu, R. Girshick, K. He, C. Feichtenhofer, and P. Krahenbuhl, "A multigrid method for efficiently training video models," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.153–162, 2020.
[68] B. Korbar, D. Tran, and L. Torresani, "Scsampler: Sampling salient clips from video for efficient action recognition," Proc. International Conference on Computer Vision (ICCV), pp.6232–6242, 2019.
[69] C.Y. Wu, M. Zaheer, H. Hu, R. Manmatha, A.J. Smola, and P. Krähenbühl, "Compressed video action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.6026–6035, 2018.
[70] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1251–1258, 2017.
[71] C. Luo and A.L. Yuille, "Grouped spatial-temporal aggregation for efficient action recognition," Proc. International Conference on Computer Vision (ICCV), pp.5512–5521, 2019.
[72] D. Tran, H. Wang, L. Torresani, and M. Feiszli, "Video classification with channel-separated convolutional networks," Proc. International Conference on Computer Vision (ICCV), pp.5552–5561, 2019.
[73] C. Gu, C. Sun, D.A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, C. Schmid, and J. Malik, "AVA: A video dataset of spatio-temporally localized atomic visual actions," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.6047–6056, June 2018.
[74] S. Stein and S.J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp.729–738, 2013.
[75] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J.C. Niebles, "Dense-captioning events in videos," Proc. International Conference on Computer Vision (ICCV), pp.706–715, 2017.
[76] A. Miech, D. Zhukov, J.B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic, "HowTo100M: Learning a text-video embedding by watching hundred million narrated video clips," Proc. International Conference on Computer Vision (ICCV), pp.2630–2640, 2019.
[77] J. Ji, R. Krishna, L. Fei-Fei, and J.C. Niebles, "Action genome: Actions as compositions of spatio-temporal scene graphs," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.10236–10247, 2020.
[78] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," Proc. European Conference on Computer Vision (ECCV), pp.831–846, 2018.
[79] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.591–600, 2020.
[80] A. Piergiovanni, A. Angelova, and M.S. Ryoo, "Evolving losses for unsupervised video representation learning," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.133–142, 2020.

Kensho Hara received the B.E. degree in Information Engineering, the M.S. degree in Information Science, and the Ph.D. degree in Information Science from Nagoya Univeristy, in 2012, 2014 and 2017, respectively. He is currently a research scientist at the National Institute of Advanced Industrial Science and Technology. His research interests include video action recognition. He is a member of IEICE and IEEE.