Birkbeck
UNIVERSITY OF LONDON

BIROn - Birkbeck Institutional Research Online

Yang, H. and Yuan, C. and Li, B. and Du, Y. and Xing, J. and Hu, W. and Maybank, Stephen J. (2018) Asymmetric 3D Convolutional Neural Networks for Action Recognition. Pattern Recognition 85 , pp. 1-12. ISSN 0031-3203.

Downloaded from: https://eprints.bbk.ac.uk/id/eprint/23342/

Usage Guidelines:
Please refer to usage guidelines at https://eprints.bbk.ac.uk/policies.html or alternatively contact lib-eprints@bbk.ac.uk.

Asymmetric 3D Convolutional Neural Networks for Action Recognition

Hao Yang[a,c], Chunfeng Yuan[a,*], Bing Li[a], Yang Du[a,c], Junliang Xing[a], Weiming Hu[a,b,c], Stephen J. Maybank[d]

[a] *National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, PR China*
[b] *CAS Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Beijing 100190, PR China*
[c] *University of Chinese Academy of Sciences, Beijing 100190, PR China*
[d] *Department of Computer Science and Information Systems, Birkbeck College, London WC1E 7HX, United Kingdom*

Abstract

Convolutional Neural Network based action recognition methods have achieved significant improvements in recent years. The 3D convolution extends the 2D convolution from operating on one single frame to a video clip, so it is able to extract effective spatial-temporal features for better analysis of human activities in videos. The 3D convolution, however, involves many more parameters than 2D convolution. Thus, it is very expensive on computation, costly on storage, and difficult to learn. In this work, we propose efficient asymmetric one-directional 3D convolutions to approximate the traditional 3D convolution. To improve the feature learning capacity of asymmetric 3D convolutions, we design a set of local 3D convolutional networks, *i.e. MicroNets*, to incorporate multi-scale 3D convolution branches. Then, we design an asymmetric 3D-CNN deep model which is constructed by *MicroNets* for the action recognition task. Moreover, to avoid training two networks on RGB and optical flow fields separately as most works do, we propose a simple but effective multi-source enhanced input, which fuses the useful information of the RGB frame and the optical flow field at the

*Corresponding author
Email addresses: hao.yang@nlpr.ia.ac.cn (Hao Yang), cfyuan@nlpr.ia.ac.cn (Chunfeng Yuan), bli@nlpr.ia.ac.cn (Bing Li), duyang2014@ia.ac.cn (Yang Du), jlxing@nlpr.ia.ac.cn (Junliang Xing), wmhu@nlpr.ia.ac.cn (Weiming Hu), sjmaybank@dcs.bbk.ac.uk (Stephen J. Maybank)

pre-processing stage.

We evaluate our asymmetric 3D-CNN models on two of the most challenging action recognition benchmarks, UCF-101 and HMDB-51. Our model outperforms all the traditional 3D-CNN models in both effectiveness and efficiency, and is comparable with the recent state-of-the-art action recognition methods on both benchmarks.

Keywords: Asymmetric 3D Convolution; MicroNets; 3D-CNN; Action Recognition.

1. Introduction

In recent years, Convolutional Neural Networks (CNNs) have achieved great success and become the mainstream method in many computer vision tasks, such as image classification [1, 2, 3, 4], object detection [5, 6, 7], semantic segmentation [8, 9, 10], and human action recognition [11, 12, 13, 14]. From these great improvements, several practices have been drummed in designing deep convolutional networks. First, information bottlenecks should be avoided when the representation size slowly decreases from the input to the output and the number of feature channels should be increased with the depth of the network. Second, the receptive fields at the end of the network should be large enough so that the processing units can base their operations on larger regions of the input. Large receptive fields can be achieved by stacking many small filters or by using large filters. Notably, the first choice can be implemented with fewer parameters and operations, and also allows inclusion of complex nonlinearity. Third, dimension reduction before aggregating filter is supported by the fact that outputs of neighboring filters are highly correlated and therefore the activation can be reduced before aggregation.

To accelerate the training and inference of 2D convolutional networks, many methods [15, 16, 17, 18, 19, 20] have been proposed in recent years. The linear structure in convolutional filters is exploited in the construction of approximations to the convolutional filters [15, 16, 17]. These methods [15, 18] flatten

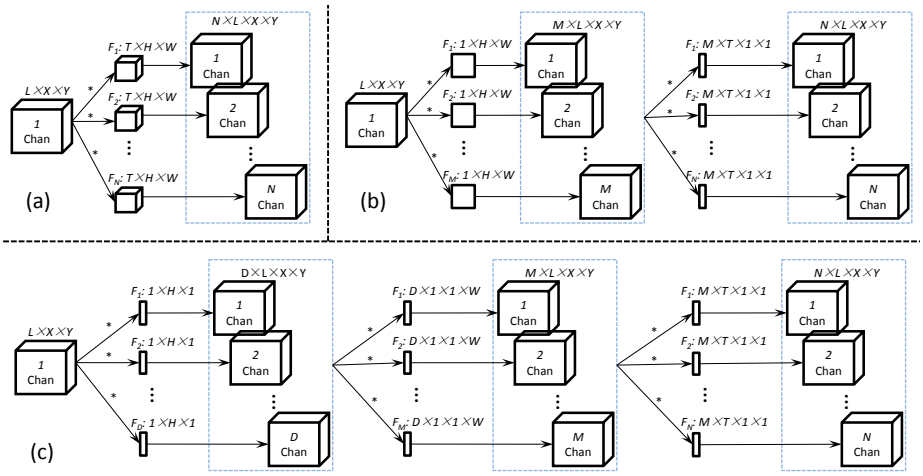


Figure 1: Illustration of three types of 3D convolutional operations: (a) the traditional 3D convolution, (b) the factorized spatial-temporal convolution proposed in FstCN [21], and (c) our asymmetric 3D convolutions. Our asymmetric 3D convolutions have many fewer parameters and operations than (a) and (b).

2D convolutional filters into a sequence of one-dimensional filters across spatial domain and channels. In [19], the 2D convolution is divided into two phases, namely in-channel convolution and across-channel linear projection. Moreover, the sparse regularity is introduced in training by [20] to remain the sparsity in convolutional filters, which decreases the computational cost of 2D convolution.

The 3D convolutional networks [22, 23] naturally extend the 2D convolutional network to the 3D spatial-temporal domain to better analyze human activities. The traditional 3D convolution is illustrated in Figure 1(a). However, 3D convolution is very expensive to compute, because a 3D convolution with k parameters in each direction requires one order more weights to be learned than 2D convolution (k^3 VS k^2). Additionally, a 3D convolutional deep model requires much more training data than a 2D convolutional deep model to be effectively trained, and obtaining the annotations of video data is much more costly than that of images. Last but not least, the 3D convolutional networks cannot be fine-tuned from a model pre-trained on the large-scale ImageNet dataset [24] as 2D-CNN based action recognition methods [11, 25, 26]. To decrease the number

of parameters and the computational cost of the 3D convolution, the FstCN [21] approximates a 3D convolutional layer by several 2D convolutional layers. The former 2D convolutional layers operate on the spatial domain and the last one operates on the temporal domain, as simply illustrated in Figure 1(b). The FstCN model is the first attempt to accelerate the traditional 3D convolution and it reduces the parameters and computational cost effectively of traditional 3D convolution from cubic to quadratic ($2k^2$ VS k^3). But there is a large space to reduce the parameters and the computational cost of the traditional 3D convolution further.

In this paper, inspired by the linear approximation of 2D convolution [15, 18], we exploit three cascaded asymmetric one-directional 3D convolutions to approximate a traditional 3D convolution with the same size of receptive field, to accelerate the traditional 3D convolution further, as shown in Figure 1(c). These asymmetric 3D convolutions decrease the number of parameters and computational cost significantly ($3k$ VS k^3). Moreover, we propose several local 3D convolutional networks, referred as *MicroNets*, which incorporate the asymmetric 3D convolutional layers with traditional 3D convolutional layers in multi-scale branches, to improve the representational ability of the asymmetric 3D convolutional layers without increasing the computational cost. Finally, following the practices in designing deep networks, we construct an efficient and effective 3D-CNN deep model by stacking several *MicroNets*. Our asymmetric 3D-CNN deep model has fewer weights, lower computational complexity, and stronger representative ability than traditional 3D-CNN models. Thus, it is more easily trained on video datasets which are usually too small for training traditional 3D-CNN models.

Additionally, we further propose an effective multi-source enhanced input for action recognition. Previous 2D-CNN based action recognition methods [11, 26, 27, 12] usually train two deep convolutional networks individually: the SpatialNet is trained on RGB frame and the TemporalNet is trained on stacked optical flow fields (Flow). The softmax scores of the two deep convolutional networks are fused to achieve a better classification performance. However, training

two convolutional deep models individually not only is costly on computation, but also does not allow end-to-end training of the whole model to better exploit the correlations between the appearance features and motion features. To overcome these limitations, we propose an effective multi-source enhanced input by incorporating useful information of the RGB and Flow frames. The enhanced input decreases computational cost by avoiding training two networks separately. It improves classification performance significantly from individual network fed with RGB and Flow respectively and achieves competitive performance with the results obtained by fusing the outputs of SpatialNet and TemporalNet.

The main contributions of this work are summarized as follows:

- We propose asymmetric 3D convolutions to approximate the traditional 3D convolution. The asymmetric 3D convolutions decrease parameters and computational cost significantly.
- To improve the feature learning capacity of asymmetric 3D convolutional layers, we propose the local 3D convolutional *MicroNets* which incorporate multi-scale convolutional features.
- Based on the *MicroNets*, we design asymmetric 3D convolutional deep model which outperforms the tradition 3D-CNN models on both effectiveness and efficiency.
- We propose the multi-sources enhanced input to decrease the computational cost further by avoiding training two deep networks individually.

2. Related Works

Action recognition, classifying actions from trimmed videos, is a very challenging domain and has long been an active research topic in computer vision, with many applications such as intelligent surveillance, human-computer interaction, robot, etc. In recent decades, researchers have proposed many methods to analyze human actions. In particular, deep learning based methods have outperformed traditional action recognition methods by a wide margin in recent

years. We review briefly the traditional action recognition methods and deep learning based action recognition methods.

2.1. Traditional Methods for Action Recognition

In the past decades, researchers have elaborately designed many handcrafted features to represent videos or actions [28, 29, 30, 31, 32, 33]. The Spatial-Temporal Interest Points (STIPs) [28] extend the Harris corner detector [34] to 3D spatial-temporal domain, which is appropriate to search for interest points in videos. The Histogram of Gradients (HOG) [35] and the Histogram of Optical Flows (HOF) [36] features are widely used to describe the STIPs. The Sparse Spatio-Temporal Features [29] improve the 3D Harris detector by applying Gabor filtering in the spatial and temporal dimensions separately. The scale-invariant STIPs [30] generalize the SURF descriptors [37] by computing a weighted sum of space-time Haar-wavelets in grid cells. Similarly, the 3D-SIFT descriptor [31] is the spatial-temporal extension of the SIFT [38] for human action recognition. The dense trajectories methods [32, 33] handle space and time differently by tracking densely sampled interest points through video sequences. The resulting trajectories and the aligned space-time volumes are used to represent the videos. These methods, combined with local HOG, HOF and MBH descriptors have achieved the best performance among handcrafted features. However, it is difficult to transfer these handcrafted features from one training dataset to another.

2.2. CNN based Methods for Action Recognition

Inspired by the great success of deep convolutional models in many computer vision tasks [3, 4, 6, 8], many CNN based methods have also been proposed for action recognition [39, 11, 12, 13]. The Slow Fusion model [39] is proposed to fuse spatial and temporal information at multiple semantic levels. Although this model is fed with multiple continuous RGB frames, it cannot learn the dynamic motion features, because the temporal information collapses after the first 2D convolutional layer according to [23]. The Two-stream model

[11] learns the appearance features and dynamic motion features using two independent networks, *i.e.* SpatialNet which is trained on single RGB frame to extract appearance features, and TemporalNet which is trained on ten continuous Flow frames to extract dynamic motion features. The confidence scores of SpatialNet and TemporalNet are fused in order to classify actions. The Fusion Two-stream model [27] demonstrates that the appearance features and the motion features being fused after the last convolutional layer achieves the best performance. The very deep two-stream model [25] exploits a very deep convolutional network, *i.e.* VGG-16 [2], as a backbone of SpatialNet and TemporalNet, to improve the performance of action classification. Based on the very deep two-stream model [25], the Temporal Segment Network (TSN) [12] splits each input video into three segments in the temporal domain and trains a very deep SpatialNet and a TemporalNet for each segment. The final fusion of the six networks achieves the current state-of-the-art performance in action recognition. Trajectory-Pooled Deep-Convolutional Descriptors (TDD) [40] combine deep convolutional features extracted from Two-stream model [11] with dense trajectory features using a trajectory centered pooling method. Although these 2D-CNN models are good at extracting spatial appearance features from subjects and backgrounds, it is difficult to learn the motion features required for action recognition.

2.3. RNN based Methods for Action Recognition

Recurrent Neural Network (RNN) models [41, 42] are effective in capturing temporal information because the current prediction is not only based on the current observation but the past information stored in hidden states. For this reason, RNN models are widely applied in action recognition to model the motion features in videos. The general pipeline for RNN based action recognition methods [43, 44, 26, 45] begin with extraction of frame-wise features using a CNN model. Then the frame-wise features are fed to LSTM layers for classification. Following this pipeline, Baccouche et al. [44] utilize a 3D-CNN model to extract spatial-temporal features. The Beyond Short Snippets [45] models the

full length content of the videos to produce large performance improvements over previously results. The Two-stream LSTM [46] stacks multiple LSTM layers to capture dynamic information in a hierarchical manner. The two feature streams, *i.e.*, convolutional feature stream and pooled feature stream, communicate with each other in training. LSTM based attention models [47, 48] are capable of capturing where the model should pay attention to in each frame of the video sequence when classifying actions. A soft attention mechanism is utilized in [47]. The focus of attention varies throughout the video sequence. Learning such attention weights through back-propagation is a computationally demanding task, because all possible combinations of input and output have to be checked.

2.4. 3D-CNN based Methods for Action Recognition

The 3D-CNN model [22] extends a 2D convolution to include temporal domain, to extract spatial-temporal features for action recognition. The 3D-CNN model abstracts spatial-temporal information naturally at multiple semantic levels from videos. The C3D model [23] is pre-trained on a large-scale video dataset to learn general features which are used to train a linear SVM for action classification. The T-CNN [49] extends the R-CNN [5] from detecting objects in images to detecting actions in videos. It replaces the last Max-pooling layer of the C3D model [23] with TOI pooling layer and improves the action recognition performance obviously. These 3D convolutional deep models are typically learned within a short snippet of the video, so they fail to model actions over their full temporal extent. The LTC-CNN model [50] operates on longer temporal extents of videos in order to improve the accuracy of action recognition. The I3D [51] proposes a very deep Inflated 3D-CNN model by extending the Inception model [3] to 3D to extract spatial-temporal features of actions. The I3D model is pre-trained on the very large and well-trimmed Kinetics video dataset and achieves a great improvement for action recognition. To avoid using computationally expensive 3D convolutions, the Factorized spatial-temporal Convolutional Network (FstCN) [21] extracts spatial-temporal features by in-

troducing a Transformation-Permutation layer to convert the 3D convolutional layer into several 2D convolutional layers operating on spatial domain followed by one 2D convolutional layer operating on temporal domain. However, the FstCN model may be difficult to learn effective spatial-temporal features, given that there is only one temporal convolutional layer followed behind several spatial convolutional layers.

3. Proposed 3D Convolutional Model

In this section, we propose a video-friendly asymmetric 3D convolutional deep model. Firstly, we introduce an asymmetric one-directional 3D convolutional layer, and compare it with traditional 3D convolutional layer taking into account the number of parameters and computational cost. Then the efficient asymmetric 3D convolutional layers are used to construct local 3D convolutional networks which are the building blocks of our asymmetric 3D-CNN deep model. Subsequently, the architecture of the asymmetric 3D-CNN deep model is presented. Finally we proposed the multi-source enhanced input to train the 3D-CNN deep model easily.

3.1. Asymmetric 3D Convolution

The 3D convolution is very effective in extracting spatial-temporal features from videos for action recognition [22, 52, 44, 23, 50]. The weights of a 3D convolution are denoted as 5-dimensional filters: $F \in \mathcal{R}^{N\times C\times T\times H\times W}$, where C is the number of input channels, T, H and W are the temporal length, height and width of the 3D convolutional kernel respectively, and N is the number of convolutional filters or output channels. The input video volume or internal convolutional feature volumes is denoted as $V \in \mathcal{R}^{C\times L\times X\times Y}$, where L, X and Y are the temporal length and spatial height and width of input volume. The operation of each 3D convolutional filter $F_f \in \mathcal{R}^{C\times T\times H\times W}, f = 1, ..., N$ is

formulated as:

$$\Phi_f(l,x,y) = V * F_f \tag{1}$$

$$= \sum_{c=1}^{C}\sum_{t=1}^{T}\sum_{h=1}^{H}\sum_{w=1}^{W} V(c,l-t,x-h,y-w)F_f(c,t,h,w), \tag{2}$$

where $l = 1,...,L$, $x = 1,...,X$ and $y = 1,...,Y$, as shown in Figure 1(a). Without loss of generality, we suppose the output feature volumes as $V' \in \mathcal{R}^{N\times L\times X\times Y}$. So the number of parameters of the traditional 3D convolutional filters is $C(THW)N$ and the number of multiplications is $C(THW)N(LXY)$, which are dramatically larger than the corresponding numbers for 2D convolutional filters. As a result, 3D convolutional networks have a much higher computational cost and require many more training videos than 2D convolutional networks.

In order to alleviate the drawbacks of traditional 3D convolution, we propose an efficient asymmetric 3D convolution by approximating each traditional 3D convolutional filter using three cascaded asymmetric 3D convolutional filters operating on three different directions, as shown in Figure 1(c). Corresponding to Equation 2, the operations of asymmetric 3D convolutional filters are formulated as:

$$\Phi_{\alpha_f}(l,x,y) = V * F_{\alpha_f} = \sum_{c=1}^{C}\sum_{h=1}^{H} V(c,l,x-h,y)F_{\alpha_f}(c,1,h,1), \tag{3}$$

$$\Phi_{\beta_f}(l,x,y) = \Phi_{\alpha_f} * F_{\beta_f} = \sum_{\alpha_f=1}^{D}\sum_{w=1}^{W} \Phi_{\alpha_f}(l,x,y-w)F_{\beta_f}(\alpha_f,1,1,w), \tag{4}$$

$$\Phi_{\gamma_f}(l,x,y) = \Phi_{\beta_f} * F_{\gamma_f} = \sum_{\beta_f=1}^{M}\sum_{t=1}^{T} \Phi_{\beta_f}(l-t,x,y)F_{\gamma_f}(\beta_f,t,1,1), \tag{5}$$

$$\widehat{\Phi}_f(l,x,y) = V * \widehat{F}_f = ((V * F_{\alpha_f}) * F_{\beta_f}) * F_{\gamma_f}, \tag{6}$$

where $\alpha_f = 1,...,D$, $\beta_f = 1,...,M$ and $\gamma_f = 1,...,N$. In Equation 6, $\widehat{\Phi}_f(l,x,y)$ denotes the approximated 3D convolutional feature volume. Equations 3, 4 and 5 define the asymmetric 3D convolution operating on height, width and temporal directions respectively. The numbers of parameters and multiplications of our

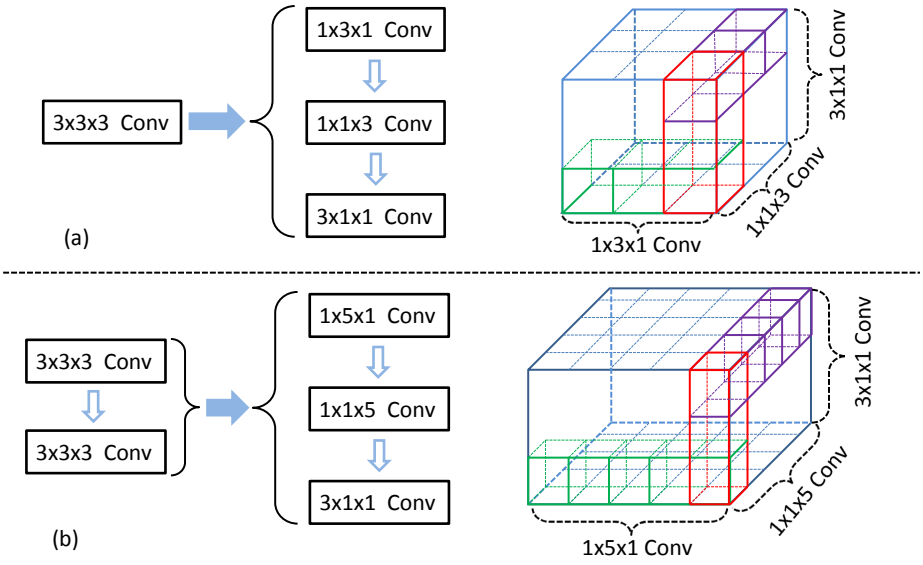


Figure 2: Illustration of (a) approximating the $3\times3\times3$ 3D convolutional layer to three asymmetric 3D convolutional layers with same size of receptive field and (b)factorizing the group of two $3\times3\times3$ 3D convolutional layers to three asymmetric 3D convolutional layers without reducing the receptive field of 3D-CNN deep model.

approximated 3D convolution are sums of the three asymmetric 3D convolutions defined in Equation 3, 4 and 5, respectively. The sums are $CHD + DWM + MTN$ convolutional parameters and $(CHD+DWM+MTN)(LXY)$ multiplications. If $D = M = N$, then the total number of convolutional parameters is $C(H+W+T)N$ and the number of multiplications is $C(T+H+W)N(LXY)$, which are decreased by two orders, compared with the totals $C(THW)N$ and $C(THW)N(LXY)$ for traditional 3D convolution.

More specifically, we approximate the traditional $3\times3\times3$ 3D convolutional layer which is widely used in the previous 3D-CNN models [22, 23, 50] using the proposed asymmetric 3D convolutions, *i.e.* three cascaded asymmetric 3D convolutional layers with kernel sizes of $1\times3\times1$, $1\times1\times3$ and $3\times1\times1$. As illustrated in Figure 2(a), three cascaded asymmetric 3D convolutional layers have same size of receptive field with the traditional 3D convolutional layer, but they are more efficient due to much fewer convolutional parameters.

Then, if we approximate each 3×3×3 3D convolutional layer in a traditional 3D-CNN deep models with three asymmetric 3D convolutional layers in the same way, for example, the C3D model will be equivalent to a very deep 3D convolutional network with 24 asymmetric 3D convolutional layers and it will be very efficient in computation. However, it is difficult to train the very deep asymmetric 3D convolutional network. In this paper, we group together adjacent two traditional 3×3×3 3D convolutional layers and then approximate the group of convolutional layers by three asymmetric 3D convolutional layers with kernel sizes of 1×5×1, 1×1×5 and 3×1×1 respectively. From the Figure 2 (b), it can be seen that the group of two traditional 3D convolutional layers and the three asymmetric 3D convolutional layers have the same size of receptive field.

Why not using $5\times1\times1$ asymmetric 3D convolutional layer in the temporal domain? Generally, we expect the receptive field in the last layer of a 3D convolutional network as large as possible. To compare with 3D-CNN models [21, 23, 49] fairly, we feed a clip of 16 frames into the deep model. Using the kernel sizes of $5\times1\times1$ and $3\times1\times1$, the receptive fields at the end of the 3D deep network are both larger than 16 in the temporal domain, but the former will introduce more parameters and computational cost. So the kernel size of $5\times1\times1$ is unnecessary for our asymmetric 3D convolutional network. In our experiments, the asymmetric 3D convolutional layers with kernel sizes of 1×5×1, 1×1×5 and 3×1×1 are appropriate to an input video volume in which the spatial dimensions are much larger than temporal duration. In theory, we could go even further that the sizes of k_1, k_2 and k_3 in asymmetric 3D convolutional layers could differ from each other, *i.e.* $k_1 \neq k_2 \neq k_3$, and they depend on the dimension of input data. Besides, this approximation method could be extended to higher dimensional convolutions easily.

3.2. 3D Convolutional MicroNets

Inspired by the Inception architecture [3], we design several local asymmetric 3D convolutional networks, *i.e. MicroNets*, to enhance the effectiveness of the asymmetric 3D convolutional layers. These local networks concatenate multi-

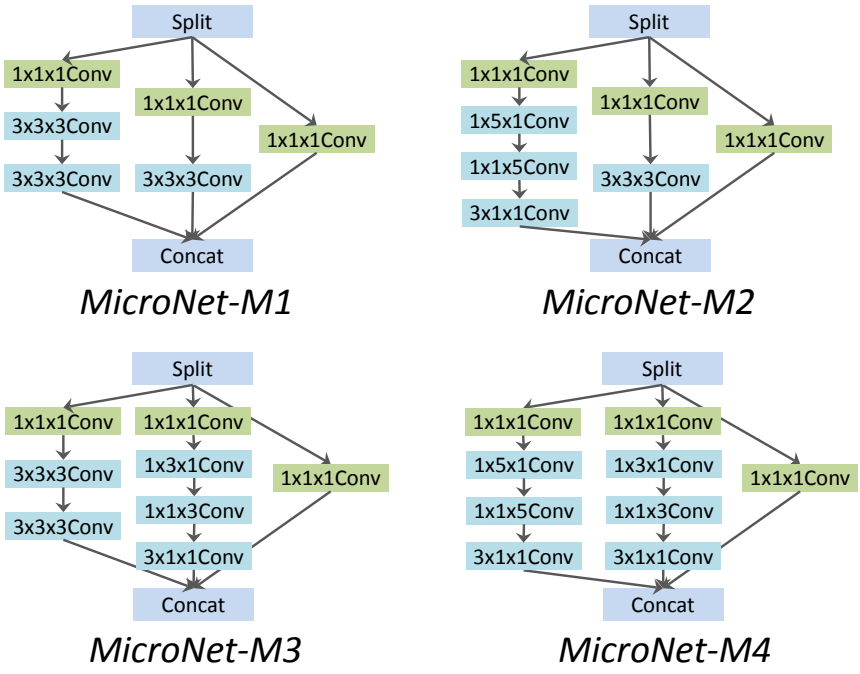


Figure 3: The four variants of local 3D convolutional *MicroNets*.

scale 3D convolution paths to handle the difference scales of spatial-temporal features in videos. Four variants of *MicroNets* are shown in Figure 3. The *MicroNet-M1* is the base model of local 3D convolutional networks. It only involves traditional 3D convolutional layers following 3D linear layers with a kernel size of $1\times1\times1$. The 3D linear layers decrease the dimension of the last concatenated layer and avoid introducing representational bottlenecks. The *MicroNet-M1* is more powerfully representative without increasing computational cost compared with the traditional 3D convolutional layer.

We further design other variants of local 3D convolutional *MicroNets* by incorporating the proposed asymmetric 3D convolutional layers to improve the effectiveness and efficiency, as shown in Figure 3. *MicroNet-M2* replaces the two traditional $3\times3\times3$ 3D convolutional layers in the left column of *MicroNet-M1* by three cascaded asymmetric 3D convolutional layers with kernel sizes

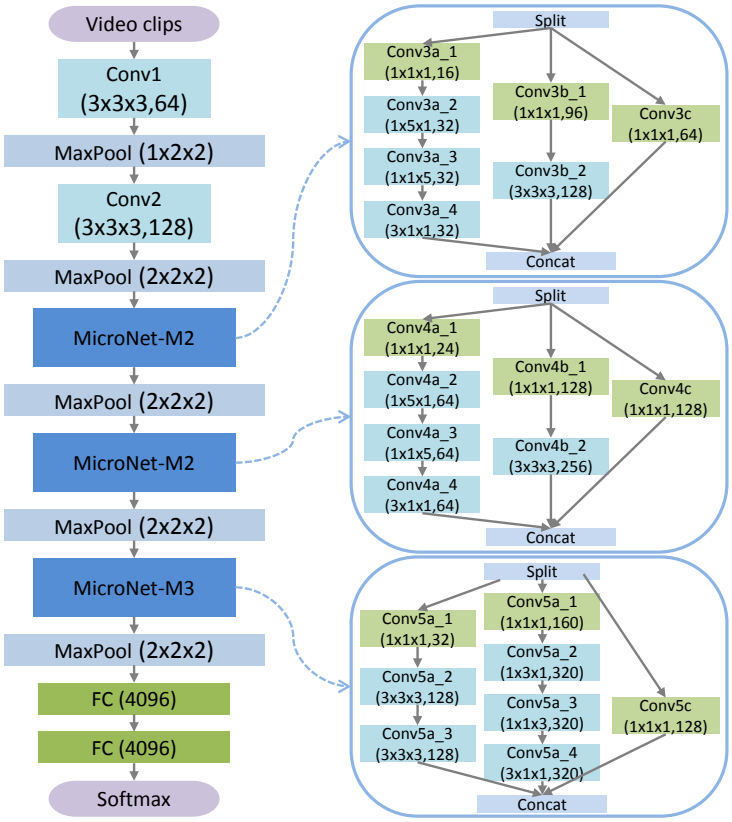


Figure 4: The layout of the asymmetric 3D-CNN deep architecture. The details of *MicroNets* structure and parameters are shown on the right column.

of $1\times5\times1$, $1\times1\times5$ and $3\times1\times1$. *MicroNet-M3* replaces the traditional $3\times3\times3$ 3D convolutional layer in the middle column of *MicroNet-M1* by three asymmetric 3D convolutional layers with kernel sizes of $1\times3\times1$, $1\times1\times3$ and $3\times1\times1$. To compare with *MicroNet-M2* and *MicroNet-M3*, we design the *MicroNet-M4*, which replaces both the left and middle column of *MicroNet-M1* with asymmetric 3D convolutional layers respectively. These *MicroNets* are representative and efficient for action recognition and they will be used as the building blocks of our 3D convolutional deep model.

Table 1: The numbers of parameters and multiplications in each convolutional layer of the C3D [23] and its proportion in all the 3D convolutional layers.

Layers	Parameters/proportion	multiplications/proportion
Conv1	0.005M (0.02%)	1.04B (1.96%)
Conv2	0.22M (0.80%)	11.09B (20.93%)
Conv3	2065M (9.48%)	31.12B (58.75%)
Conv4	10.62M (38.41%)	8.33B (15.73%)
Conv5	14.16M (51.21%)	1.39B (2.62%)
Total	27.65M	52.97B

3.3. Asymmetric 3D Convolutional Deep Model

Based on the asymmetric 3D convolutional *MicroNets*, we design an asymmetric 3D-CNN model. It stacks several local 3D convolutional *MicroNets* on traditional 3D convolutional layers, followed by two fully connected layers. The layout of our asymmetric 3D-CNN deep architecture is shown in Figure 4.

As shown in Table 1, the number of parameters the *Conv1* and *Conv2* layers is only 0.82% in that of all convolutional layers and the multiplications of the two layers are about 23% in that of all convolutional layers. So the numbers of parameters and multiplications of the first two layers only occupy a little part in the 3D-CNN deep model. To achieve a good trade-off between accuracy and complexity, we use two traditional $3 \times 3 \times 3$ 3D convolutional layers with the input size of $16 \times 112 \times 112$ in front of the asymmetric 3D convolutional deep network. The first two layers are followed by two local 3D convolutional *MicroNet-M2* with input sizes of $8 \times 28 \times 28$ and $4 \times 14 \times 14$ respectively. The final local convolutional network is *MicroNet-M3* with the input size of $2 \times 7 \times 7$. Following each group of 3D convolutional layers, a 3D Max-pooling layer is used to decrease the spatial-temporal resolution of feature volumes as well as double the number of feature maps to avoid introducing representational bottleneck into the deep network. Following the last Max-pooling layer, two fully connected layers and one *Softmax* layer are employed for prediction.

Additionally, the output channels and kernel size of convolutional layers are shown in Figure 4. The non-linear activation function is set as the rectified linear unit in our deep architecture. The stride of each convolutional layer in our model is set as one, and an appropriate padding is used for each 3D convolutional layer to keep the output size of each convolutional layer same with the input size. The kernel sizes and strides of all 3D Max-pooling layers are set as $2 \times 2 \times 2$ except the first Max-pooling layer with the kernel size of $1 \times 2 \times 2$.

3.4. Multi-source Enhanced Input

In order to effectively model the dynamic information of actions and avoid training two deep networks on RGB and Flow frames separately, we propose a multi-source enhanced representation of each video frame, called RGBF, which fuses the useful information in the RGB and Flow frames. Firstly, we pre-compute the Flow from successive RGB frames by the method in [53]. The Flow is recorded in the form of traditional images, using the horizontal and vertical components of Flow as the first two channels of the frame and the magnitude of Flow as the third channel of the Flow frame. Then, we linearly re-scale the Flow magnitude to a range of $[0, 1]$ to give a movement confidence map which indicates the possibility of where the movement occurs. *i.e.*, if the value of Flow magnitude is large, there is a great possibility of movement occurring, vice versa. The RGBF frame is generated by multiplying each channel of the RGB frame with the corresponding movement confidence map. Formally, the RGBF input at the pixel (x, y) in a frame is computed as:

$$RGBF_{x,y,c} = RGB_{x,y,c} \times \frac{|F|_{x,y} - |F|_{min}}{|F|_{max} - |F|_{min}}, \tag{7}$$

where $c = 1, 2, 3$. $|F|_{x,y}$ denotes the Flow magnitude at pixel (x, y). $|F|_{max}$ and $|F|_{min}$ denote the maximum and minimum of Flow magnitude in the frame.

We show several examples of RGB, Flow, and the enhanced RGBF frames on the UCF-101 and HMDB-51 benchmarks in Figure 5. For each dataset, there are three rows which present the RGB, Flow and RGBF frames from top to bottom

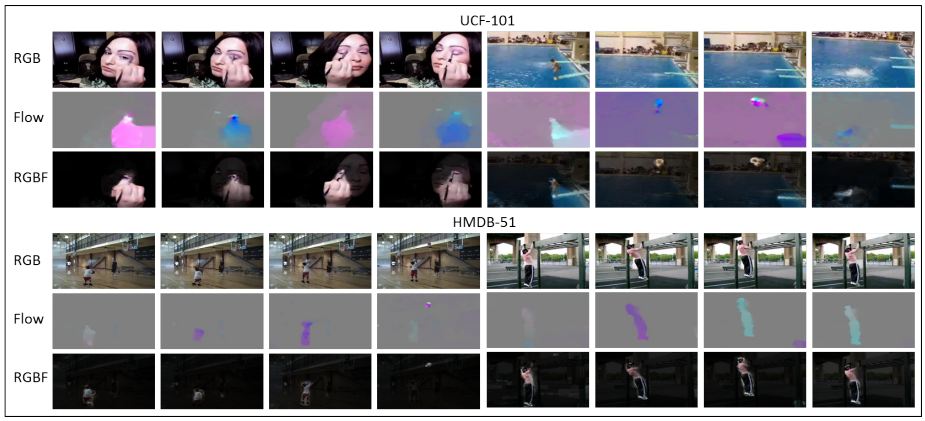


Figure 5: The comparison of the RGB, Flow and enhanced RGBF frames from the UCF-101 and HMDB-51 datasets.

respectively. It can be seen that the RGBF frame highlights the motion related parts and restrains the redundant information compared with the RGB frame. The RGBF frame also contains useful appearance information about motion regions in the image compared with the Flow frame. The enhanced RGBF frames enable the 3D-CNN deep model easily learn effective spatial-temporal features for action recognition.

4. Experiments

4.1. Datasets

The asymmetric 3D-CNN deep models were tested on two of the most challenging datasets, UCF-101 and HMDB-51. UCF-101 [54] is a dataset of realistic action videos, collected from YouTube, having 101 action categories with 13320 videos (27 hours in total). UCF-101 dataset has the largest diversity in terms of actions and large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc. We reported the average accuracy of the three standard splits provided in [54]. HMDB-51 dataset [55] is a large realistic collection of videos from movies and the web. It contains 6849 clips divided into 51 action categories. We used

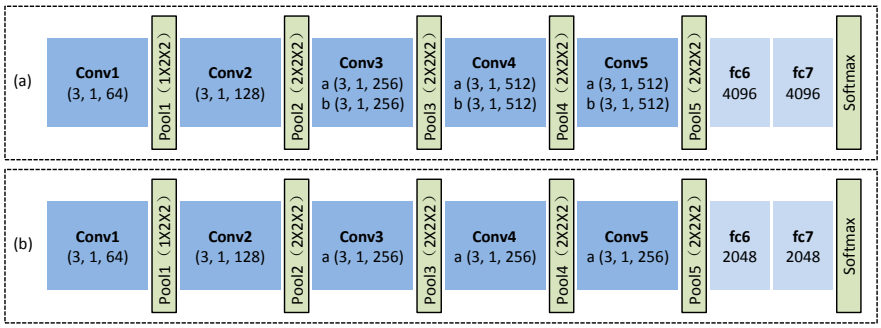


Figure 6: The two 3D-CNN baseline models to compare with the proposed asymmetric 3D-CNN model, where the baseline model (a) is denoted as *c3d-b8* and the baseline model (b) is denoted as *c3d-b5*.

the raw videos without stabilization and reported the average accuracy of the three splits provided by [55]. Additionally, to initialize our 3D-CNN deep model carefully, we pre-trained our model on a large-scale FCVID [56] dataset, which contains 91223 web videos annotated manually into 239 categories. We discarded categories, such as places, animals and scenes, which do not involve obvious movements. Finally, we used about 75K videos distributing over 170 categories.

4.2. Baselines

we used two 3D-CNN deep models constructed with only traditional 3D convolutional layers as the baselines to compare with our asymmetric 3D-CNN models. The first baseline is the C3D model [23], which involves 8 3D convolutional layers, 5 Max-pooling layers, and 2 fully connected layers, referred as *c3d-b8*. The architecture and super-parameters of the *c3d-b8* are shown in Figure 6(a). The second baseline is a reduced 3D-CNN model obtained by deleting the *conv3b*, *conv4b* and *conv5b* convolutional layers from the C3D model and halving the number of channels of last two convolutional layers and fully connected layers. The reduced baseline model is denoted as *c3d-b5* and its architecture and super-parameters are shown in Figure 6(b).

4.3. Experimental Settings

The asymmetric 3D-CNN deep models are trained using randomly selected clips with 16 frames in each clip. Each frame in the clip is resized to 128×171 and is cropped into 112×112 spatially. Horizontal flipping and corner cropping are used to prevent over-fitting. We train the models by Stochastic Gradient Descent (SGD) with batch size of 16. The momentum and weight-decay are set as 0.9 and 0.0005 respectively. In pre-training on the FCVID video dataset, all frames are cropped and resized to 256×320 spatial size. The base learning rate is set as 0.001 and it is divided by 5 for every 15 epochs. The training is stopped at 50 epochs. In finetuning on the UCF-101 and HMDB-51 datasets, the base learning rate is set as 0.0001 and is divided by 10 for every 8 epochs. The deep models are trained for 20 epochs in total.

In tests, firstly, 25 test clips are sampled from each test video with equal intervals and each clip consists of continuous 16 frames. Then, each frame in the clip is resized to 128×171 and then is cropped to obtain the ten standard 112×112 crops, i.e. one center and four corners with horizontal flippings, in the spatial domain. Finally, the average score of all the 250 crops from one video is used to predict the action label.

4.4. Evaluation of Asymmetric 3D Convolution

We designed two groups of experiments to evaluate the effectiveness and efficiency of the asymmetric 3D convolution. In the first group experiments, we converted the third, fourth and fifth 3D convolutional layers of the baseline *c3d-b5* to three cascaded asymmetric 3D convolutional layers, *i.e.* 1×3×1, 1×1×3 and 3×1×1, to obtain seven asymmetric 3D-CNN variants. The names of these asymmetric 3D-CNN variants are prefixed with "b5", and the last numbers in the model names denote the traditional 3D convolutional layers which are replaced by asymmetric 3D convolutional layers, as reported in Table 2. For example, the "*b5-asyConv34*" denotes the network in which the third and fourth 3D convolutional layers in the baseline *c3d-b5* are replaced by asymmetric 3D convolutional layers. Similarly, to compare with the deeper 3D-CNN baseline

Table 2: Two groups of comparison results between the traditional 3D-CNN with their four asymmetric 3D-CNN variants on the UCF-101 dataset.

Models	Parameter Numbers	Accuracy	Speed ($s/iter$)
c3d-b5	17.44M	45.4%	1.10
b5-asyConv3	17.05M	46.7%	0.85
b5-asyConv4	16.26M	46.5%	0.94
b5-asyConv5	16.26M	**47.2**%	0.94
b5-asyConv34	15.87M	45.7%	0.82
b5-asyConv35	15.87M	46.2%	0.82
b5-asyConv45	15.08M	46.9%	0.83
b5-asyConv345	**14.69M**	45.2%	**0.81**
c3d-b8	78.40M	42.3%	1.28
b8-asyConv3	76.43M	44.3%	0.99
b8-asyConv4	70.53M	44.7%	1.00
b8-asyConv5	67.65M	**45.3**%	1.03
b8-asyConv34	68.56M	42.9%	0.95
b8-asyConv35	65.68M	43.2%	0.97
b8-asyConv45	59.78M	44.1%	0.97
b8-asyConv345	**57.82M**	42.3%	**0.93**

model *c3d-b8*, the third, fourth and fifth groups of 3D convolutional layers of the *c3d-b8* are replaced with three cascaded asymmetric 3D convolutional layers, *i.e.* $1\times5\times1$, $1\times1\times5$ and $3\times1\times1$. As reported in Table 2, there are seven asymmetric 3D-CNN models extended from the baseline *c3d-b8*. The names of these seven asymmetric 3D-CNN variants are prefixed with "b8", and the last numbers in the model names denote the traditional 3D convolutional layers which are replaced by asymmetric 3D convolutional layers.

All the models above were trained from scratch on the UCF-101 dataset. The classification accuracy, the number of parameters and the training speed of the models are reported in Table 2. 1) The proposed asymmetric 3D-CNN mod-

els outperform their baseline *c3d-b5* and *c3d-b8* models. Particularly, the *b5-asyConv5* asymmetric 3D-CNN model outperforms the *c3d-b5* model by 1.8% and the *b8-asyConv5* achieves the improvement of 2% over the *c3d-b8* model. It indicates that replacing the traditional 3D convolutional layers with the proposed asymmetric 3D convolutional layers is effective for action recognition. 2) The models using asymmetric 3D convolutional layers in higher layers are usually better than these models which use the asymmetric 3D convolutional layers in lower layers. For example, the *b5-asyConv5* outperforms the *b5-asyConv3* and *b5-asyConv4* models, and the *b8-asyConv5* outperforms the *b8-asyConv3* and *b8-asyConv4* models. These results prove that using the asymmetric 3D convolutional layers in higher layers is more effective than using them in lower layers. 3) Increasing the asymmetric 3D convolutional layers in the 3D-CNN models cannot improve the performance of the asymmetric 3D-CNN further. For example, the *b5-asyConv345* model cannot outperform its baseline model. It is probably because that increasing the asymmetric 3D convolutional layers will increase the depth of the models and the deeper models are more difficult to train from scratch.

Additionally, the average training times for each iteration of the models are reported in Table 2. All models were trained with the same GPU and batch size. The only difference is the model architecture. The training speed of the asymmetric 3D-CNN models is much higher than that of the baseline models. For example, the *b5-asyConv3* is faster than the *c3d-b5* model by 29% and the best performance *b8-asyConv5* model improves the speed of the baseline model by 25%. So our asymmetric 3D convolutional layers are more efficient than the traditional 3D convolutional layer.

4.5. Evaluation of Asymmetric 3D Convolutional MicroNets

To evaluate the effectiveness of the proposed four local 3D convolutional *MicroNets*, we designed four 3D-CNN deep models by converting the *conv5* layer of the *c3d-b5* model to one of the *MicroNets*. The resulting asymmetric 3D convolutional networks are denoted by *b5-M1*, *b5-M2*, *b5-M3* and *b5-M4*. All

Table 3: Evaluating the effectiveness of the proposed four 3D convolutional *MicroNets* on the UCF-101 dataset.

Models	Paramter Numbers	Accuracy
c3d-b5	17.43M	45.7%
b5-asyConv5	16.26M	47.2%
b5-M1	16.07M	46.6%
b5-M2	16.04M	**48.1**%
b5-M3	15.87M	47.5%
b5-M4	**15.84M**	45.8%

the models are trained from scratch on the UCF-101 dataset. As shown in Table 3, the four asymmetric 3D-CNN models outperform the baseline *c3d-b5* model. The *b5-M2* model achieves the best performance among the four 3D-CNN variants and it outperforms the *c3d-b5* model by over 2%. Therefore, adding the 3D convolutional *MicroNets* to the baseline model yields an obvious improvement. Additionally, compared with the best performance *b5-asyConv5* model in Table 2, the *b5-M2* and *b5-M3* models both outperform the *b5-asyConv5* model. The *MicroNets* are thus more effective than a simply cascaded of asymmetric 3D convolutional layers. The *MicroNet-M2* and *MicroNet-M3* local networks will be used in the asymmetric 3D-CNN deep models later.

Table 4: Evaluating the performance of three asymmetric 3D-CNN variants, finetuned on the UCF-101 dataset.

Models	Accuracy
c3d-b8	84.6%
Asymmetric 3D-CNN (M2)	86.2%
Asymmetric 3D-CNN (M3)	85.1%
Asymmetric 3D-CNN	**86.4**%

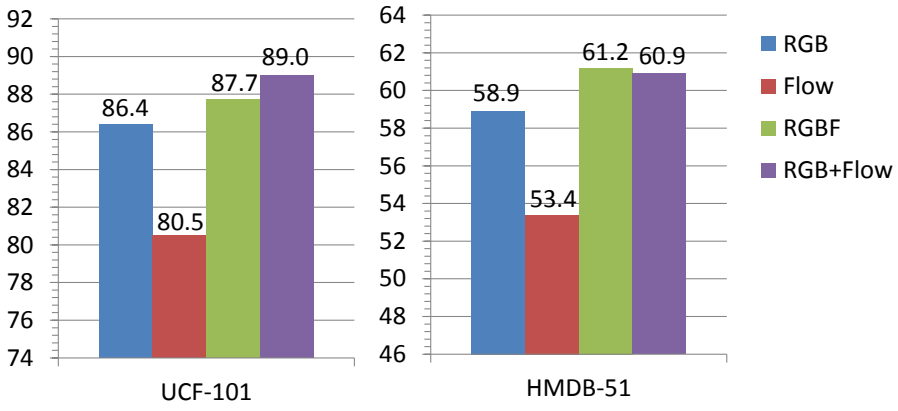


Figure 7: Evaluating the performance of the asymmetric 3D-CNN deep model fine-tuned on UCF-101 and HMDB-51 datasets with three kinds of inputs.

4.6. Evaluation of Our 3D-CNN Model and the RGBF Input

To avoid over-fitting of our 3D-CNN deep models caused by training on the limited quantity of videos, we pre-trained our 3D-CNN models on the large-scale FCVID video dataset. Subsequently, all the layers were fine-tuned on the target dataset with a ten times higher learning rate for the last fully connected layer.

Firstly, based on the conclusion in previous experiments, we designed our asymmetric 3D-CNN deep model, denoted as *Asymmetric 3D-CNN*, which has been described in detail in Section 3 and Figure 4. We compared the *Asymmetric 3D-CNN* with two variants of asymmetric 3D-CNN deep models, *i.e. Asymmetric 3D-CNN (M2)* and *Asymmetric 3D-CNN (M3)*, which are similar to the *Asymmetric 3D-CNN* architecture. The *Asymmetric 3D-CNN (M2)* model only used the local 3D convolutional *MicorNet-M2* and the *Asymmetric 3D-CNN (M3)* model only used the *MicorNet-M3*. As shown in Table 4, the *Asymmetric 3D-CNN (M2)* deep model achieves better performance than the *Asymmetric 3D-CNN (M3)*. The *Asymmetric 3D-CNN* outperforms both the *Asymmetric 3D-CNN(M2)* and *Asymmetric 3D-CNN(M3)* models. It demonstrates that diverse local networks allow the 3D-CNN deep model to learn complementary spatial-temporal features from videos for action recognition.

Additionally, we evaluated the performance of the *Asymmetric 3D-CNN* deep model on two most challenging benchmarks fed with three kinds of inputs, *i.e.* the RGB, Flow and enhanced RGBF frames. The results of the experiments are shown in Figure 7. The performance of the enhanced RGBF frames is better than that of the RGB and Flow inputs, and it even outperforms the fusion result of two networks fed with RGB and Flow frames on the HMDB-51 dataset. It indicates that the simply enhanced input is very effective. The results obtained from the Flow input are much worse than those obtained from the RGB input, which is different from 2D-CNN based action recognition methods [11, 25, 26]. It indicates that the 3D-CNN is more appropriate for extracting spatial-temporal features from raw videos compared with 2D-CNN. Moreover, fed with the RGB frames, the *c3d-b8* baseline model, also pre-trained on the FCVID dataset, achieves 84.6% and 56.7% accuracy on the UCF-101 and HMDB-51 datasets respectively, and the *Asymmetric 3D-CNN* deep model outperforms it by 1.8% and 2.2% respectively, which demonstrates the effectiveness of the *Asymmetric 3D-CNN* model.

4.7. Comparison with the State-of-the-arts

Our asymmetric 3D-CNN models are compared with current state-of-the-art methods on the UCF-101 dataset in Table 5. The *Asymmetric 3D-CNN* model fed with RGBF frames achieves a better performance than the most current state-of-the-art models, even thought many of them fuse two networks of SpatialNet and TemporalNet [11, 26]. Fusing the *Softmax* scores of the two networks fed with RGB and RGBF frames respectively can further improve the performance of our 3D-CNN deep model. The *Asymmetric 3D-CNN (RGB+RGBF)* outperforms all of the traditional methods, such as the Improved Dense Trajectories (IDT) [33] and the Multi-skIp Feature Stacking (MIFS) [58]. Compared with 2D-CNN based methods, our model outperforms Slow Fusion [39] by over 24% and outperforms Two-stream (fusion by averaging) [11] by 2.6%. Compared with the most comparable 3D-CNN based action recognition models, our asymmetric 3D-CNN model outperforms the C3D model [23] by

Table 5: Comparison with current state-of-the-art methods on UCF-101 dataset.

IDT [33]	85.9%
IDT (higher-dimension) [57]	87.9%
MIFS (L=3) [58]	89.1%
Slow Fusion [39]	65.4%
VGG16+Images on Web [59]	83.5%
Two-stream (fusion by averaging) [11]	86.9%
Two-stream (fusion by SVM) [11]	88.0%
Fusion Two-stream [27]	91.8%
Two-stream (VGG-16) [25]	91.4%
LRCN (weighted average) [26]	82.9%
C3D (1 net+SVM) [23]	82.3%
C3D (3 net+SVM) [23]	85.2%
C3D+IDT [23]	90.4%
T-CNN [49]	87.5%
FstCN (averaging fusion) [21]	87.9%
Asymmetric 3D-CNN (RGBF)	87.7%
Asymmetric 3D-CNN (RGB+RGBF)	89.5%
Asymmetric 3D-CNN (RGB+RGBF+IDT)	92.6%

over 4% and outperforms the FstCN model [21] by 1.6%. Meanwhile, it outperforms the T-CNN model [49] by 2.0%. Additionally, we combined the deep features which were extracted by our 3D-CNN deep models fed with RGB and RGBF individually with the widely used traditional IDT [33] features, and classified actions with a multi-class linear SVM. The resulting *Asymmetric 3D-CNN (RGB+RGBF+IDT)* outperforms the newest state-of-the-art methods [27, 25] on the UCF-101 dataset.

In Table 6, the *Asymmetric 3D-CNN* model is compared with current state-of-the-art methods on the HMDB-51 dataset. The *Asymmetric 3D-CNN* model

Table 6: Comparison with current state-of-the-art methods on HMDB-51 dataset.

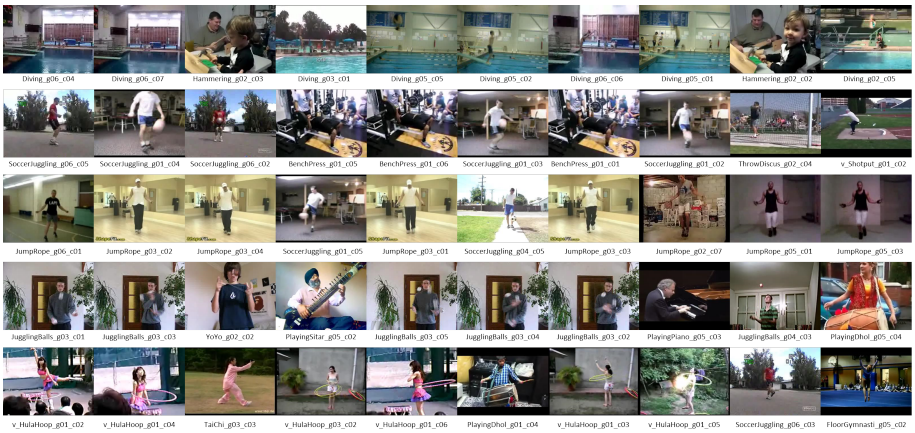
IDT [33]	57.2%
IDT (higher-dimension) [57]	61.1%
MIFS (L=3) [58]	65.1%
TDD [40]	63.2%
KVMF [60]	63.3%
Two-stream (fusion by SVM) [11]	59.4%
Fusion Two-stream [27]	64.6%
Action-Transformations [61]	63.4%
FstCN (averaging fusion) [21]	58.6%
FstCN (SCI fusion) [21]	59.1%
Asymmetric 3D-CNN (RGBF)	61.2%
Asymmetric 3D-CNN (RGB+RGBF)	63.5%
Asymmetric 3D-CNN (RGB+RGBF+IDT)	**65.4**%

fed with the enhanced RGBF frames outperforms most current state-of-the-art methods. The fusion model, *Asymmetric 3D-CNN (RGB+RGBF)*, fed with RGB and RGBF frames outperforms the Two-stream [11] by 4.1% and outperforms the FstCN [21] by 4.4%. Finally, the deep features which were extracted from RGB and RGBF frames were combined with the traditional IDT [33] features, which are used to train a linear SVM to classify actions. The resulting *Asymmetric 3D-CNN (RGB+RGBF+IDT)* outperforms the newest state-of-the-art methods [27, 61, 60] on the HMDB-51 dataset.

4.8. Visualization of Model Learning Results

To get an intuitive understanding on what is learnt by the *Asymmetric 3D-CNN* model, we visualized the learned convolutional features and the clustering ability of the last convolutional layer in Figure 8. Firstly, the *Asymmetric 3D-CNN* deep model was pre-trained on the large-scale FCVID dataset to get

(a) Visualizing the learned features of multiple convolutional layers on the HMDB-51 dataset.



(b) Visualizing the clustering ability of the last convolutional layer on the UCF-101 dataset.

Figure 8: Visualizing what the asymmetric 3D-CNN model learns from the UCF-101 and HMDB-51 datasets.

a better initialization. Then, it was finetuned on the UCF-101 or HMDB-51 datasets respectively.

In Figure 8(a), we visualized the learned features of several convolutional layers of the *Asymmetric 3D-CNN* deep model finetuned on the HMDB-51 dataset. Specifically, we randomly chose three examples from the *test* set of the HMDB-51 dataset and fed them into our 3D-CNN deep model to extract the learned features at multiple convolutional layers. One channel of feature maps is presented for the *Conv1*, *Conv2* and *Conv3a_4* convolutional layers for each example. From bottom to top (by depth order) of each example, the *Asymmetric 3D-CNN* model focuses on the appearance at first, then the appearance becomes obscure, and later the motion regions are highlighted.

To visualize the clustering ability of the *Asymmetric 3D-CNN* deep model, we randomly chose one clip for each video in the UCF-101 *test* set. All the clips were fed to the*Asymmetric 3D-CNN* model to compute the activation of the last convolutional layer. We singled out a specific neuron of the last convolutional layer and sorted the clips from highest to the lowest activation. We presented the top 10 clips of five neurons in Figure 8(b). Each row corresponds to one neuron. It can be seen that each particular neuron is only fired by some similar actions. In the top row, most clips are "Diving" actions from different subjects and scenes, but two clips of "Hammering" action are included. Actually, the configuration of the person is like a hammer and the "Diving" action shows a person jumping from a diving board, which is like a hammer "jumping" from a bred. In the second row, a soccer running up and down in the "SoccerJuggling" action has a similar configure and motion with a bench running up and down in the "BenchPress" action. In the third row, persons jumping on the ground in the "JumpRope" and "SoccerJuggling" actions are very similar. In the last two rows, most entries show the same or similar actions carried out by a number of different actors in different scenes.

5. Conclusion

In this paper, we have proposed an efficient 3D convolution method by approximating traditional 3D convolution with three cascaded one-directional asymmetric 3D convolutions. Then, we have designed multiple local asymmetric 3D convolutional *MicroNets* to further improve the effectiveness of asymmetric 3D convolutional layers. Finally, the asymmetric 3D-CNN deep model has been built by stacking the local 3D convolutional *MicroNets* with traditional 3D convolutional layers. Additionally, the proposed multi-source enhanced RGBF input has produced a large improvement in action recognition, compared with the performance obtained from the RGB and Flow inputs. Our 3D-CNN deep model outperforms most comparable 3D-CNN baselines and many state-of-the-art methods on two challenging datasets. In the future work, we expect to increase the depth of our 3D-CNN model by stacking more 3D convolutional *MicroNets* to improve the performance of action recognition further.

References

[1] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[2] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representations, 2015, pp. 1–14.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[4] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[5] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.

[6] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, 2015, pp. 91–99.

[7] E. Ohn-Bar, M. M. Trivedi, Multi-scale volumes for deep object detection and localization, Pattern Recognition 61 (2017) 557–572.

[8] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[9] Y. Wang, J. Liu, Y. Li, J. Fu, M. Xu, H. Lu, Hierarchically supervised deconvolutional network for semantic video segmentation, Pattern Recognition 64 (2017) 437–445.

[10] R. Hou, C. Chen, M. Shah, An end-to-end 3d convolutional neural network for action detection and segmentation in videos, arXiv preprint arXiv:1712.01111.

[11] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: Advances in Neural Information Processing Systems, 2014, pp. 568–576.

[12] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, L. Van Gool, Temporal segment networks: towards good practices for deep action recognition, in: Springer European Conference on Computer Vision, 2016, pp. 20–36.

[13] E. P. Ijjina, K. M. Chalavadi, Human action recognition using genetic algorithms and convolutional neural networks, Pattern Recognition 59 (2016) 199–212.

[14] M. Ma, N. Marturi, Y. Li, A. Leonardis, R. Stolkin, Region-sequence based six-stream cnn features for general and fine-grained human action recognition in videos, Pattern Recognition 76 (2018) 506–521.

[15] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, R. Fergus, Exploiting linear structure within convolutional networks for efficient evaluation, in: Advances in Neural Information Processing Systems, 2014, pp. 1269–1277.

[16] M. Jaderberg, A. Vedaldi, A. Zisserman, Speeding up convolutional neural networks with low rank expansions, in: British Machine Vision Conference, 2014, pp. 1–13.

[17] Y. Ioannou, D. Robertson, J. Shotton, R. Cipolla, A. Criminisi, Training CNNs with low-rank filters for efficient image classification, Journal of Asian Studies 62 (3) (2015) 952–953.

[18] J. Jin, A. Dundar, E. Culurciello, Flattened convolutional neural networks for feedforward acceleration, arXiv preprint arXiv:1412.5474.

[19] W. Min, L. Baoyuan, F. Hassan, Factorized convolutional neural networks, arXiv preprint arXiv:1606.0433.

[20] B. Liu, M. Wang, H. Foroosh, M. Tappen, M. Penksy, Sparse convolutional neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 806–814.

[21] L. Sun, K. Jia, D.-Y. Yeung, B. E. Shi, Human action recognition using factorized spatio-temporal convolutional networks, in: IEEE International Conference on Computer Vision, 2015, pp. 4597–4605.

[22] S. Ji, W. Xu, M. Yang, K. Yu, 3D convolutional neural networks for human action recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (1) (2013) 221–231.

[23] T. Du, L. Bourdev, R. Fergus, L. Torresani, Learning spatiotemporal features with 3D convolutional networks, in: IEEE International Conference on Computer Vision, 2015, pp. 4489–4497.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, F.-F. Li, Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[25] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, Towards good practices for very deep two-stream convnets, arXiv preprint arXiv:1507.02159.

[26] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2625–2634.

[27] C. Feichtenhofer, A. Pinz, A. Zisserman, Convolutional two-stream network fusion for video action recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1933–1941.

[28] I. Laptev, On space-time interest points, International Journal of Computer Vision 64 (2-3) (2005) 107–123.

[29] P. Dollár, V. Rabaud, G. Cottrell, S. Belongie, Behavior recognition via sparse spatio-temporal features, in: IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005, pp. 65–72.

[30] G. Willems, T. Tuytelaars, L. Van Gool, An efficient dense and scale-invariant spatio-temporal interest point detector, in: Springer European Conference on Computer Vision, 2008, pp. 650–663.

[31] P. Scovanner, S. Ali, M. Shah, A 3-dimensional SIFT descriptor and its application to action recognition, in: ACM International Conference on Multimedia, 2007, pp. 357–360.

[32] H. Wang, A. Kläser, C. Schmid, C.-L. Liu, Action recognition by dense trajectories, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 3169–3176.

[33] H. Wang, C. Schmid, Action recognition with improved trajectories, in: IEEE International Conference on Computer Vision, 2013, pp. 3551–3558.

[34] C. Harris, A combined corner and edge detector, Proc Alvey Vision Conf 1988 (3) (1988) 147–151.

[35] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision Pattern Recognition, 2013, pp. 886–893.

[36] N. Dalal, B. Triggs, C. Schmid, Human detection using oriented histograms of flow and appearance, in: Springer European Conference on Computer Vision, 2006, pp. 428–441.

[37] H. Bay, T. Tuytelaars, L. Van Gool, Surf: Speeded up robust features, in: Springer European conference on computer vision, 2006, pp. 404–417.

[38] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.

[39] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, F.-F. Li, Large-scale video classification with convolutional neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1725–1732.

[40] L. Wang, Y. Qiao, X. Tang, Action recognition with trajectory-pooled deep-convolutional descriptors, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4305–4314.

[41] J. L. Elman, Finding structure in time, Cognitive science 14 (2) (1990) 179–211.

[42] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[43] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, A. Baskurt, Action classification in soccer videos with long short-term memory recurrent neural

networks, in: Springer International Conference on Artificial Neural Networks, 2010, pp. 154–159.

[44] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, A. Baskurt, Sequential deep learning for human action recognition, in: Springer International Workshop on Human Behavior Understanding, 2011, pp. 29–39.

[45] Y.-H. N. Joe, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, G. Toderici, Beyond short snippets: Deep networks for video classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4694–4702.

[46] H. Gammulle, S. Denman, S. Sridharan, C. Fookes, Two stream LSTM: A deep fusion framework for human action recognition, in: IEEE Winter Conference on Applications of Computer Vision, 2017, pp. 177–186.

[47] S. Sharma, R. Kiros, R. Salakhutdinov, Action recognition using visual attention, arXiv preprint arXiv:1511.04119.

[48] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, A. Courville, Describing videos by exploiting temporal structure, in: IEEE international conference on computer vision, 2015, pp. 4507–4515.

[49] R. Hou, C. Chen, M. Shah, Tube convolutional neural network (T-CNN) for action detection in videos, arXiv preprint arXiv:1703.10664.

[50] G. Varol, I. Laptev, C. Schmid, Long-term temporal convolutions for action recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence PP (99) (2017) 1–1.

[51] J. Carreira, A. Zisserman, Quo vadis, action recognition? a new model and the kinetics dataset, arXiv preprint arXiv:1705.07750.

[52] Z. Liu, C. Zhang, Y. Tian, 3D-based deep convolutional neural network for action recognition with depth sequences, Image and Vision Computing 55 (2016) 93–100.

[53] T. Brox, A. Bruhn, N. Papenberg, J. Weickert, High accuracy optical flow estimation based on a theory for warping, in: Springer European conference on computer vision, 2004, pp. 25–36.

[54] K. Soomro, A. R. Zamir, M. Shah, UCF101: A dataset of 101 human actions classes from videos in the wild, arXiv preprint arXiv:1212.0402.

[55] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, HMDB: a large video database for human motion recognition, in: International Conference on Computer Vision, 2011, pp. 2556–2563.

[56] Y.-G. Jiang, Z. Wu, J. Wang, X. Xue, S.-F. Chang, Exploiting feature and class relationships in video categorization with regularized deep neural networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1) (2017) 1–14.

[57] X. Peng, L. Wang, X. Wang, Y. Qiao, Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice, Computer Vision and Image Understanding 150 (2016) 109–125.

[58] Z. Lan, M. Lin, X. Li, A. G. Hauptmann, B. Raj, Beyond gaussian pyramid: Multi-skip feature stacking for action recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 204–212.

[59] S. Ma, S. A. Bargal, J. Zhang, L. Sigal, S. Sclaroff, Do less and achieve more: Training cnns for action recognition utilizing action images from the web, Pattern Recognition 68 (2017) 334–345.

[60] W. Zhu, J. Hu, G. Sun, X. Cao, Y. Qiao, A key volume mining deep framework for action recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1991–1999.

[61] X. wang, F. Ali, G. Abhinav, Actions t̃ransformations, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2658–2667.