Multi-fiber Networks for Video Recognition

Yunpeng Chen[1](✉), Yannis Kalantidis[2], Jianshu Li[1], Shuicheng Yan[1,3], and Jiashi Feng[1]

[1] National University of Singapore, Singapore, Singapore
{chenyunpeng,jianshu}@u.nus.edu,
{eleyans,elefjia}@nus.edu.sg

[2] Facebook Research, Menlo Park, USA
yannisk@fb.com

[3] Qihoo 360 AI Institute, Beijing, China

Abstract. In this paper, we aim to reduce the computational cost of spatio-temporal deep neural networks, making them run as fast as their 2D counterparts while preserving state-of-the-art accuracy on video recognition benchmarks. To this end, we present the novel *Multi-Fiber* architecture that slices a complex neural network into an ensemble of lightweight networks or *fibers* that run through the network. To facilitate information flow between fibers we further incorporate multiplexer modules and end up with an architecture that reduces the computational cost of 3D networks by an order of magnitude, while increasing recognition performance at the same time. Extensive experimental results show that our multi-fiber architecture significantly boosts the efficiency of existing convolution networks for both image and video recognition tasks, achieving state-of-the-art performance on UCF-101, HMDB-51 and Kinetics datasets. Our proposed model requires over 9× and 13× less computations than the I3D [1] and R(2+1)D [2] models, respectively, yet providing higher accuracy.

Keywords: Deep learning · Neural networks · Video · Classification Action recognition

1 Introduction

With the aid of deep convolutional neural networks, image understanding has achieved remarkable success in the past few years. Notable examples include residual networks [3] for image classification, FastRCNN [4] for object detection, and Deeplab [5] for semantic segmentation, to name a few. However, the progress of deep neural networks for video analysis still lags their image counterparts, mostly due to the extra computational cost and complexity of spatio-temporal inputs.

The temporal dimension of videos contains valuable motion information that needs to be incorporated for video recognition tasks. A popular and effective way of reasoning spatio-temporally is to use spatio-temporal or 3D convolutions [6,7] in deep neural network architectures to learn video representations. A 3D convolution is an extension of the 2D (spatial) convolution, which has three-dimensional kernels that also convolve along the temporal dimension. The 3D convolution kernels can be used to build 3D CNNs (Convolutional Neural Networks) by simply replacing the 2D spatial convolution kernels. This keeps the model end-to-end trainable. State-of-the-art video understanding models, such as Res3D [7] and I3D [1] build their CNN models in this straightforward manner. They use multiple layers of 3D convolutions to learn robust video representations and achieve top accuracy on multiple datasets, albeit with high computational overheads. Although recent approaches use decomposed 3D convolutions [2,8] or group convolutions [9] to reduce the computational cost, the use of spatio-temporal models still remains prohibitive for practical large-scale applications. For example, regular 2D CNNs require around 10s GFLOPs for processing a single frame, while 3D CNNs currently require more than 100 GFLOPs for a single clip[1]. *We argue that a clip-based model should be able to highly outperform frame-based models at video recognition tasks for the same computational cost, given that it has the added capacity of reasoning spatio-temporally.*

In this work, we aim to substantially improve the efficiency of 3D CNNs while preserving their state-of-the-art accuracy on video recognition tasks. Instead of decomposing the 3D convolution filters as in [2,8], we focus on the other source of computational overhead for 3D CNNs, the large input tensors. We propose a sparsely connected architecture, the *Multi-Fiber* network, where each unit in the architecture is essentially composed of multiple *fibers*, *i.e.* lightweight 3D convolutional networks that are independent from each other as shown in Fig. 1(c). The overall network is thus sparsely connected and the computational cost is reduced by approximately N times, where N is the number of fibers used. To improve information flow across fibers, we further propose a lightweight multiplexer module, that redirects information between parallel fibers if needed and is attached at the head of each residual block. This way, with a minimal computational overhead, representations can be shared among multiple fibers, and the overall capacity of the model is increased.

Our main contributions can be summarized as follows:

(1) We propose a highly efficient multi-fiber architecture, verify its effectiveness by evaluating it 2D convolutional neural networks for image recognition and show that it can boost performance when embedded on common compact models.
(2) We extend the proposed architecture to spatio-temporal convolutional networks and propose the Multi-Fiber network (MF-Net) for learning robust video representations with significantly reduced computational cost, *i.e.* about an order of magnitude less than the current state-of-the-art 3D models.

[1] *E.g.* the popular ResNet-152 [3] and VGG-16 [10] models require 11 GFLOPs and 15 GFLOPs, respectively, for processing a frame, while I3D [1] and R(2+1)D-34 [2] require 108 GFLOPs and 152 GFLOPs, respectively.

(3) We evaluate our multi-fiber network on multiple video recognition benchmarks and outperform recent related methods with several times lower computational cost on the Kinetics, UCF-101 and HMDB51 datasets.

2 Related Work

When it comes to video models, the most successful approaches utilize deep learning and can be split into two major categories: models based on spatial or 2D convolutions and those that incorporate spatio-temporal or 3D convolutions.

The major advantage of adopting 2D CNN based methods is their computational efficiency. One of the most successful approaches in this category is the Two-stream Network [13] architecture. It is composed of two 2D CNNs, one working on frames and another on optical flow. Features from the two modalities are fused at the final stage and achieved high video recognition accuracy. Multiple approaches have extended or incorporated the two-stream model [14–17] and since they are built on 2D CNNs are very efficient, usually requiring less than 10 GFLOPS per frame. In a very interesting recent approach, CoViAR [18] further reduces computations to 4.2 GFLOPs per frame in average, by directly using the motion information from compressed frames and sharing motion features across frames. However, as these approaches rely on pre-computed motion features to capture temporal dependencies, they usually perform worse than 3D convolutional networks, especially when large video datasets are available for pre-training, such as Sports-1M [19] and Kinetics [20].

On the contrary, 3D convolution neural networks are naturally able to learn motion features from raw video frames in an end-to-end manner. Since they use 3D convolution kernels that model both spatial and temporal information, rather than 2D kernels which just model spatial information, more complex relations between motion and appearance can be learned and captured. C3D [7] is one of the early methods successfully applied to learning robust video features. It builds a VGG [10] alike structure but uses $3 \times 3 \times 3$ kernels to capture motion information. The Res3D [23] makes one step further by taking the advantage of residual connections to ease the learning process. Similarly, I3D [1] proposes to use the Inception Network [24] as the backbone network rather than residual networks to learn video representations. However, all of the methods suffer from high computational cost compared with regular 2D CNNs due to the newly added temporal dimension. Recently, S3D [8] and R(2+1)D [2] are proposed to use one $1 \times 3 \times 3$ convolution layer followed by another $3 \times 1 \times 1$ convolutional layer to approximate a full-rank 3D kernel to reduce the computations of a full-rank $3 \times 3 \times 3$ convolutional layer while achieving better precision. However, these methods still suffer from an order of magnitude more computational cost than their 2D competitors, which makes it difficult to train and deploy them in practical applications.

The idea of using spare connections to reduce the computational cost is similar to low-power networks built for mobile devices [25–27] as well as other recent approaches that try to sparsify parts of the network either through group convolutions [28] or through learning connectivity [29]. However, our proposed network

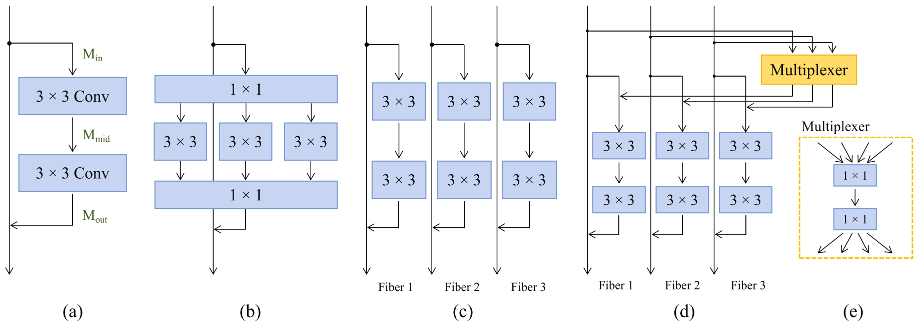


Fig. 1. From ResNet to multi-fiber. (a) A residual unit with two 3 × 3 convolution layers. (b) Conventional Multi-Path design, *e.g.* ResNeXt [28]. (c) The proposed multi-fiber design consisting of multiple separated lightweight residual units, called fibers. (d) The proposed multi-fiber architecture with a *multiplexer* for transferring information across separated fibers. (e) The architecture details of a multiplexer. It consists of two linear projection layers, one for dimension reduction and the other for dimension expansion.

is built for solving video recognition tasks and proposed different strategies that can also benefit existing low-power models, *e.g.* MobileNet-v2 [26]. We further discuss the differences of our architecture and compare against the most related and state-of-the-art methods in Sects. 3 and 4.

3 Multi-fiber Networks

The success of models that utilize spatio-temporal convolutions [1,2,7–9] suggests that it is crucial to have kernels spanning both the spatial and temporal dimensions. Spatio-temporal reasoning, however, comes at a cost: Both the convolutional kernels and the input-output tensors are multiple times larger.

In this section, we start by describing the basic module of our proposed model, *i.e.*, the multi-fiber unit. This unit can effectively reduce the number of connections within the network and enhance the model efficiency. It is generic and compatible with both 2D and 3D CNNs. For clearer illustration, we first demonstrate its effectiveness by embedding it into 2D convolutional architectures and evaluating its efficiency benefits for image recognition tasks. We then introduce its spatio-temporal 3D counterpart and discuss specific design choices for video recognition tasks.

3.1 The Multi-fiber Unit

The proposed multi-fiber unit is based on the highly modularized residual unit [3], which is easy to train and deploy. As shown in Fig. 1(a), the conventional residual unit uses two convolutional layers to learn features, which is straightforward but computationally expensive. To see this, let M_{in} denote the number of input channels, M_{mid} denote the number of middle channels, and M_{out} denote the number of output channels. Then the total number of connections between these two layers can be computed as

$$\# \text{ Connections} = M_{in} \times M_{mid} + M_{mid} \times M_{out}. \tag{1}$$

For simplicity, we ignore the dimensions of the input feature maps and convolution kernels which are constant. Equation (1) indicates that the number of connections is quadratic to the width of the network, thus increasing the width of the unit by a factor of k would result in k^2 times more computational cost.

To reduce the number of connections that are essential to the overall computation cost, we propose to *slice* the complex residual unit into N parallel and separated paths (called *fibers*), each of which is isolated from the others, as shown in Fig. 1(c). In this way, the overall width of the unit remains the same, but the number of connections is reduced by a factor of N:

$$\begin{aligned}\# \text{ Connections} &= N \times (M_{in}/N \times M_{mid}/N + M_{mid}/N \times M_{out}/N) \\ &= (M_{in} \times M_{mid} + M_{mid} \times M_{out})/N.\end{aligned} \tag{2}$$

We set $N = 16$ for all our experiments, unless otherwise stated. As we show experimentally in the following section, such a slicing strategy is intuitively simple yet effective. At the same time, however, slicing isolates each path from the others and blocks any information flow across them. This may result in limited learning capacity for data representations since one path cannot access and utilize the feature learned from the others. In order to recover part of the learning capacity, recent approaches that partially use slicing like ResNeXt [28], Xception [30] and MobileNet [25,26] choose to only slice a small portion of layers and still use fully connected parts. The majority of layers (>60%) remains unsliced and dominates the computational cost, becoming the efficiency bottleneck. ResNeXt [28], for example, uses fully connected convolution layers at the beginning and end of each unit, and only slices the second layer as shown on Fig. 1(b). However, these unsliced layers dominate the computation cost and become the bottleneck. Different from only slicing a small portion of layers, we propose to slice the entire residual unit creating multiple fibers. To facilitate information flow, we further attach a lightweight bottleneck component we call the *multiplexer* that operates across fibers, in a residual manner.

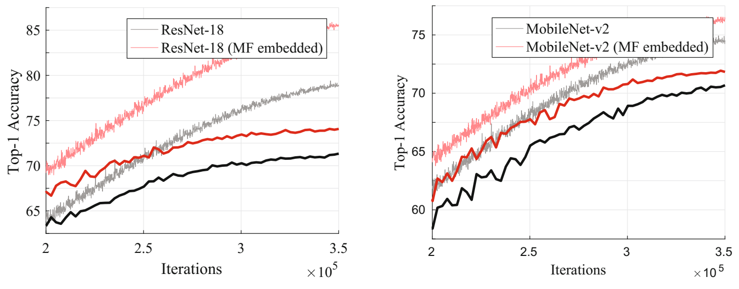
The multiplexer acts as a router that redirects and amplifies features from all fibers. As shown in Fig. 1(e), the multiplexer first gathers features from all fibers using a 1×1 convolution layer, and then redirects them to specific fibers using the following 1×1 convolution layer. The reason for using two 1×1 layers instead of just one is to lower the computational overhead: we set the number of the first-layer output channels to be k times smaller than its input channels, so that the total cost would be reduced by a factor of $k/2$ compared with using a single 1×1 layer. The parameters within the multiplexer are randomly initialized and automatically adjusted by back-propagation end-to-end to maximize the performance gain for the given task. Batch normalization and ReLU nonlinearities are used before each layer. Figure 1(d) shows the full multi-fiber network, where the proposed multiplexer is attached at the beginning of the multi-fiber unit for routing features extracted from other paralleled fibers.

We note that, although the proposed multi-fiber architecture is motivated to reduce the number of connections for 3D CNNs to alleviate high computational cost, it is also applicable to 2D CNNs to further enhance efficiency of existing 2D architectures. To demonstrate this and verify effectiveness of the proposed architecture, we conduct several studies on 2D image classification tasks at first.

3.2 Justification of the Multi-fiber Architecture

We experimentally study the effectiveness of the proposed multi-fiber architecture by applying it on 2D CNNs for image classification and the ImageNet-1k dataset [31]. We use one of the most popular 2D CNN model, residual network (ResNet-18) [3], and the most computationally efficient ModelNet-v2 [26] as the backbone CNN in the following studies.

Our implementation is based on the code released by [32] using MXNet [33] on a cluster of 32 GPUs. The initial learning rate is set to 0.5 and decreases exponentially. We use a batch size of 1,024 and train the network for 360,000 iterations. As suggested by prior work [25], we use less data augmentations for obtaining better results. Since the above training strategy is different from the one used in our baseline methods [3,26], we report both our reproduced results and the reported results in their papers for fair comparison.



(a) ResNet-18 (b) MobileNet-v2

Fig. 2. Training and validation accuracy on the ImagaNet-1k dataset for (a) ResNet-18 and (b) MobileNet-v2 backbones respectively. The red lines stand for performance of the model with our proposed multi-fiber unit. The black lines show performance of our reproduced baseline model using exactly the same training settings as our method. The line thickness indicates results on the validation set (the ticker one) or the training set (the thinner one). (Color figure online)

The training curves in Fig. 2 plot the training and validation accuracy on ImageNet-1k during the last several iterations. One can observe that the network with our proposed Multi-fiber (MF) unit can consistently achieve higher training and validation accuracy than the baseline models, with the same number of

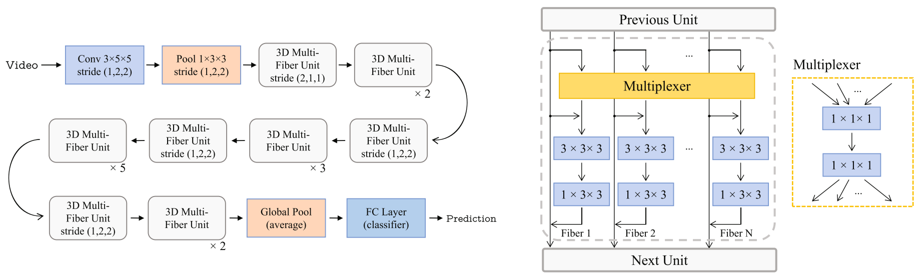
Table 1. Efficiency comparison on the ImageNet-1k validation set. "MF" stands for "multi-fiber unit", and Top-1/Top-5 accuracies are evaluated on a 224 × 224 single center crop [3]. "MF-Net" is our proposed network, with the architecture shown in Table 2. The ResNeXt row presents results for a ResNeXt-26 model of our design that has about the same number of FLOPS as MF-Net.

Model	Top-1 Acc.	Top-5 Acc.	#Params	FLOPs
ResNet-18 [3]	69.6%	89.2%	11.7 M	1.8 G
ResNet-18 (reproduced)	71.4%	90.2%	11.7 M	1.8 G
ResNet-18 (MF embedded)	74.3%	92.1%	9.6 M	1.6 G
ResNeXt-26 (8 × 16d)	72.8%	91.1%	6.3 M	1.1 G
ResNet-50 [3]	75.3%	92.2%	25.5 M	4.1 G
MobileNet-v2 (1.4) [26]	74.7%	–	6.9 M	585 M
MobileNet-v2 (1.4) (reproduced)	72.2%	90.8%	6.9 M	585 M
MobileNet-v2 (1.4) (MF embedded)	73.0%	91.1%	6.0 M	578 M
MF-Net (N = 12)	74.5%	92.0%	5.9 M	895 M
MF-Net (N = 16)	74.6%	92.0%	5.8 M	861 M
MF-Net (N = 24)	75.4%	92.5%	5.8 M	897 M
MF-Net (N = 16, w/o multiplexer)	70.2%	89.4%	4.5 M	600 M
MF-Net (N = 16, w/o multiplexer, deeper & wider)	71.0%	90.0%	6.4 M	897 M

iterations. Moreover, the resulted model has a smaller number of parameters and is more efficient (see Table 1). This demonstrates that embedding the proposed MF unit indeed helps reduce the model redundancy, accelerates the learning process and improves the overall model generalization ability. Considering the final training accuracy of the "MF embedded" network is significantly higher than the baseline networks and all the network models adopt the same regularization settings, the MF units are also demonstrated to be able to improve the learning capacity of the baseline networks.

Table 1 presents results on the validation set for Imagenet-1k. By simply replacing the original residual unit with our proposed multi-fiber one, we improve the Top-1/Top-5 accuracy by 2.9%/1.9% upon ResNet-18 with smaller model size (9.6M vs. 11.7M) and lower FLOPs (1.6G vs. 1.8G). The performance gain also stands for the more efficient low-complexity MobileNet-v2: introducing the multi-fiber unit also boosts its Top-1/Top-5 accuracy by 0.8%/0.3% with smaller model size (6.0M vs. 6.9M) and lower FLOPs (578M vs. 585M), clearly demonstrating its effectiveness. We note that our reproduced MobileNet-v2 has slightly lower accuracy than the reported one in [26] due to difference in the batch size, learning rate and update policy. But with the same training strategy, our reproduced ResNet-18 is 1.8% better than the reported one [3].

The two bottom sections of Table 1 further show ablation studies of our MF-Net, with respect to the number of fibers N and with/without the use of the multiplexer. As we see, increasing the number of fibers increases performance, while performance drops significantly when removing the multiplexer unit, demonstrating the importance of sharing information between fibers. Overall, we see



(a) 3D Multi-fiber Network (b) 3D Multi-fiber Unit

Fig. 3. Architecture of 3D multi-fiber network. (a) The overall architecture of 3D Multi-fiber Network. (b) The internal structure of each Multi-fiber Unit. Note that only the first 3×3 convolution layer has expanded on the 3rd temporal dimension for lower computational cost.

that our 2D multi-fiber network can perform as well as the much larger ResNet-50 [3], that has 25.5M parameters and requires 4.1 GFLOPS[2].

3.3 Spatio-Temporal Multi-fiber Networks

In this subsection, we extend out multi-fiber architecture to spatio-temporal inputs and present a new architecture for 3D convolutional networks and video recognition tasks. The design of our spatio-temporal multi-fiber network follows that of the "ResNet-34" [3] model, with a slightly different number of channels for lower GPU memory cost on processing videos. In particular, we reduce the number of channels in the first convolution layer, *i.e.* "Conv1", and increase the number of channels in the following layers, *i.e.* "Conv2-5", as shown in Table 2. This is because the feature maps in the first several layers have high resolutions and consume exponentially more GPU memory than the following layers for both training and testing.

The detailed network design is shown in Table 2, where we first design a 2D MF-Net and then "inflate" [1] its 2D convolutional kernels to 3D ones to build the 3D MF-Net. The 2D MF-Net is used as a pre-trained model for initializing the 3D MF-Net. Several recent works advocate separable convolution which uses two separate layers to replace one 3×3 layer [2,8]. Even though it may further reduce the computational cost and increase the accuracy, we do not use the separable convolution due to its high GPU memory consumption, considering video recognition application.

[2] It is worth noting that in terms of wall-clock time measured on our server, our MF-Net is only slightly (about 30%) faster than the highly optimized implementation of ResNet-50. We attribute this to the unoptimized implementation of group convolutions in CuDNN and foresee faster actual running times in the near future when group convolution computations are well optimized.

Table 2. Multi-fiber Network architecture. The "2D MF-Net" takes images as input, while the "3D MF-Net" takes frames, *i.e.* video clips, as input. Note, the complexity is evaluated with FLOPs, *i.e.* floating-point multiplication-adds. The stride of "3D MF-Net" is denoted by "(temporal stride, height stride, width stride)", and the stride of "2D MF-Net" is denoted by "(height stride, width stride)".

Layer	Repeat	#Channel	2D MF-Net		3D MF-Net	
			Output size	Stride	Output size	Stride
Input		3	224×224		$16 \times 224 \times 224$	
Conv1	1	16	112×112	(2,2)	$16 \times 112 \times 112$	(1,2,2)
MaxPool			56×56	(2,2)	$16 \times 56 \times 56$	(1,2,2)
Conv2	1	96	56×56	(1,1)	$8 \times 56 \times 56$	(2,1,1)
	2			(1,1)		(1,1,1)
Conv3	1	192	28×28	(2,2)	$8 \times 28 \times 28$	(1,2,2)
	3			(1,1)		(1,1,1)
Conv4	1	384	14×14	(2,2)	$8 \times 14 \times 14$	(1,2,2)
	5			(1,1)		(1,1,1)
Conv5	1	768	7×7	(2,2)	$8 \times 7 \times 7$	(1,2,2)
	2			(1,1)		(1,1,1)
AvgPooling			1×1		$1 \times 1 \times 1$	
FC			1000		400	
#Params			5.8 M		8.0 M	
FLOPs			861 M		11.1 G	

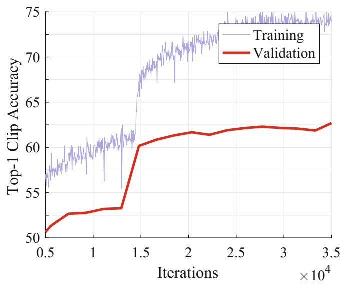
Figure 3 shows the inner structure of each 3D multi-fiber unit after the "inflation" from 2D to 3D. We note that all convolutional layers use 3D convolutions thus the input and output features contain an additional temporal dimension for preserving motion information.

4 Experiments

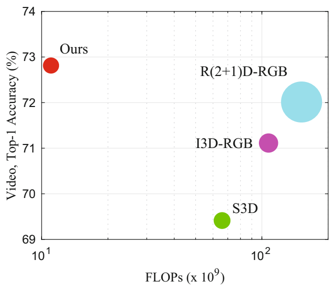
We evaluate the proposed multi-fiber network on three benchmark datasets, Kinetics [20], UCF-101 [34] and HMDB51 [35], and compare the results with other state-of-the-art models. All experiments are conducted using PyTorch [36] with input size of $16 \times 224 \times 224$ for both training and testing. Here 16 is the number of frames for each input clip. During testing, videos are resized to resolution 256×256, and we average clip predictions randomly sampled from the long video sequence to obtain the video predictions.

4.1 Video Classification with Motion Trained from Scratch

In this subsection, we study the effectiveness of the proposed model on learning video representations when motion features are trained from scratch. We use



(a)



(b)

Fig. 4. Results on the Kinetics dataset (RGB Only). (a) The training and validation accuracy for multi-fiber network. (b) Efficiency comparison between different 3D convolutional networks. The area of each circle is proportional to the total parameter number of the model.

Table 3. Comparison on action recognition accuracy with state-of-the-arts on Kinetics. The complexity is measured using FLOPs, *i.e.* floating-point multiplication-adds. All results are only using RGB information, *i.e.* no optical flow. Results with citation numbers are copied from the respective papers.

Method	#Params	FLOPs	Top-1	Top-5
Two-Stream [1]	12 M	–	62.2%	–
ConvNet+LSTM [1]	9 M	–	63.3%	–
S3D [8]	8.8 M	66.4 G	69.4%	89.1%
I3D-RGB [1]	12.1 M	107.9 G	71.1%	89.3%
R(2+1)D-RGB [2]	63.6 M	152.4 G	72.0%	90.0%
MF-Net (Ours)	**8.0 M**	**11.1 G**	**72.8%**	**90.4%**

the large-scale Kinetics [20] benchmark dataset for evaluation, which consists of approximately 300,000 videos from 400 action categories.

In this experiment, the 3D MF-Net model is initialized by inheriting parameters from a 2D one (see Sect. 3.3) pre-trained on the ImageNet-1k dataset. Then the 3D MF-Net is trained on Kinetics with an initial learning rate 0.1 which decays step-wisely with a factor 0.1. The weight decay is set to 0.0001 and we use SGD as the optimizer with a batch size 1,024. We train the model on a cluster of 64 GPUs. Figure 4(a) shows the training and validation accuracy curves, from which we can see the network converges fast and the total training process only takes about 36,000 iterations.

Table 3 shows video action recognition results of different models trained on Kinetics. The models pre-trained on other large-scale video datasets, *e.g.* Sports-1M [19], using substantially more training videos are excluded in the table for fair comparison. As can be seen from the results, 3D based CNN models significantly

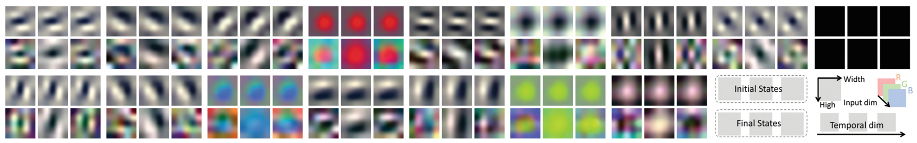


Fig. 5. Visualization of the learned filters. The filters initialized by the ImageNet pretrained model using inflating are shown on the top. The corresponding learned 3D filters on Kinetics are shown at the bottom. (upscaled by 15x). Best viewed in color. (Color figure online)

improve the Top-1 accuracy upon 2D CNN based models. This performance gap is because 2D CNNs extract features from each frame separately and thus are incapable of modeling complex motion features from a sequence of raw frames even when LSTM is used, which limits their performance. On the other hand, 3D CNNs can learn motion features end-to-end from raw frames and thus are able to capture effective spatio-temporal information for video classification tasks. However, these 3D CNNs are computationally expensive compared 2D ones.

In contrast, our proposed MF-Net is more computationally efficient than existing 3D CNNs. Even with a moderate number of fibers, the computational overhead introduced by the temporal dimension is effectively compensated and our multi-fiber network only costs 11.1 GFLOPs, as low as regular 2D CNNs. Regarding performance and parameter efficiency, our proposed model achieves the highest Top-1/Top-5 accuracy and meanwhile it has the smallest model size. Compared with the best $R(2+1)D\text{-}RGB$, our model is over $13\times$ faster with $8\times$ less parameters, yet achieving 0.8% higher Top-1 accuracy. We note that the proposed model also costs the lowest GPU memory for both training and testing, benefiting from the optimized architecture mentioned in Sect. 3.3.

To get further insights into what our network learns, we visualize all 16 spatio-temporal kernels of the first convolutional layer in Fig. 5. Each 2-by-3 block corresponds to two $3 \times 3 \times 5 \times 5$ filters, with the top and bottom rows showing the filter before and after learning, respectively. As the filters are initialized from a 2D network pretrained on ImageNet and inflated in the temporal dimension, all three sub-kernels are identical in the beginning. After learning, however, we see filters evolving along the temporal dimension with diverse patterns, indicating that spatio-temporal features are learned effectively and embedded in these 3D kernels.

4.2 Video Classification with Fine-Tuned Models

In this experiment, we evaluate the generality and robustness of the proposed multi-fiber network by transferring the features learned on Kinetics to other datasets. We are interested in examining whether the proposed model can learn robust video representations that can generalize well to other datasets. We use the popular UCF-101 [34] and HMDB51 [35] as evaluation benchmarks.

The UCF-101 contains 13, 320 videos from 101 categories and the HMDB51 contains 6, 766 videos from 51 categories. Both are divided into 3 splits. We

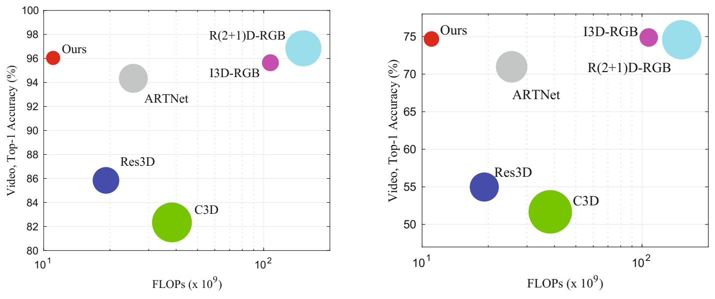
Table 4. Action recognition accuracy on UCF-101 and HMDB51. The complexity is evaluated with FLOPs, *i.e.* floating-point multiplication-adds. The top part of the table refers to related methods based on 2D convolutions, while the lower part to methods utilizing spatio-temporal convolutions. Column "+OF" denotes the use of Optical Flow. FLOPs for computing optical flow are not considered.

Method	FLOPs	+OF	UCF-101	HMDB51
ResNet-50 [37]	3.8 G		82.3%	48.9%
ResNet-152 [37]	11.3 G		83.4%	46.7%
CoViAR [18]	4.2 G		90.4%	59.1%
Two-Stream [13]	3.3 G	✓	88.0%	59.4%
TSN [38]	3.8 G	✓	94.2%	69.4%
C3D [7]	38.5 G		82.3%	51.6%
Res3D [23]	19.3 G		85.8%	54.9%
ARTNet [16]	25.7 G		94.3%	70.9%
I3D-RGB [1]	107.9 G		95.6%	74.8%
R(2+1)D-RGB [2]	152.4 G		96.8%	74.5%
MF-Net (Ours)	**11.1 G**		96.0%	74.6%

follow experiment settings in [2,7,8,23] and report the averaged three-fold cross validation accuracy. For model training on both datasets, we use an initial learning rate 0.005 and decrease it for three times with a factor 0.1. The weight decay is set to 0.0001 and the momentum is set to 0.9 during the SGD optimization. All models are fine-tuned using 8 GPUs with a batch size of 128 clips.

Table 4 shows results of the multi-fiber network and comparison with state-of-the-art models. Consistent with above results, the multi-fiber network achieves the state-of-the-art accuracy with much lower computation cost. In particular, on the UCF-101 dataset, the proposed model achieves 96.0% Top-1 classification accuracy which is comparable with the sate-of-the-arts, but it is significantly more computationally efficient (11.1 vs. 152.4 GFLOPs). Compared with Res3D [23] which is also based on ResNet backbone and costs about 19.3 GFLOPs, the multi-fiber network achieves over 10% improvement in Top-1 accuracy (96.0% v.s. 85.8%) with 42% less computational cost.

Meanwhile, the proposed multi-fiber network also achieves the state-of-the-art accuracy on the HMDB51 dataset with significantly less computational cost. Compared with the 2D CNN based models that also only use RGB frames, our proposed model improves the accuracy by more than 15% (74.6% v.s. 59.1%). Even compared with the methods that using extra optical information, our proposed model still improves the accuracy by over 5%. This advantage partially benefits from richer motion features that learned from large-scale video pre-training datasets, while 2D CNNs cannot. Figure 6 shows the results in details. It is clear that our model provides an order of magnitude higher efficiency than previous state-of-the-arts in terms of FLOPs but still enjoys the high accuracy.



(a) UCF-101 (b) HMDB51

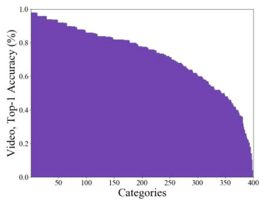
Fig. 6. Efficiency comparison between different methods. We use the area of each circle to show the total number of parameters for each model.

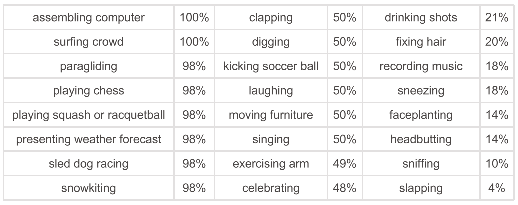
4.3 Discussion

The above experiments clearly demonstrate outstanding performance and efficiency of the proposed model. In this section, we discuss its potential limitations through success and failure case analysis on Kinetics.

We first study category-wise recognition accuracy. We calculate the accuracy for each category and sort them in a descending order, shown in Fig. 7(left). Among all 400 categories, we notice that 190 categories have an accuracy higher than 80% and 349 categories have an accuracy higher than 50%. Only 17 categories cannot be recognized well and have an accuracy lower than 30%. We list some examples along the spectrum in the right panel of Fig. 7. We find that in categories with highest accuracy there are either some specific objects/backgrounds clearly distinguishable from other categories or specific actions spanning long duration. On the contrary, categories with low accuracy usually do not display any distinguishing object and the target action usually lasts for a very short time within a long video.

To better understand success and failure cases, we visualize some of the video sequences in Fig. 8. The frames are evenly selected from the long video sequence.





assembling computer	100%	clapping	50%	drinking shots	21%
surfing crowd	100%	digging	50%	fixing hair	20%
paragliding	98%	kicking soccer ball	50%	recording music	18%
playing chess	98%	laughing	50%	sneezing	18%
playing squash or racquetball	98%	moving furniture	50%	faceplanting	14%
presenting weather forecast	98%	singing	50%	headbutting	14%
sled dog racing	98%	exercising arm	49%	sniffing	10%
snowkiting	98%	celebrating	48%	slapping	4%

Fig. 7. Statistical results on Kinetics validation dataset. Left: Accuracy distribution of the proposed model on the validation set of Kinetics. The category is sorted by accuracy in a descending order. Right: Selected categories and their accuracy.

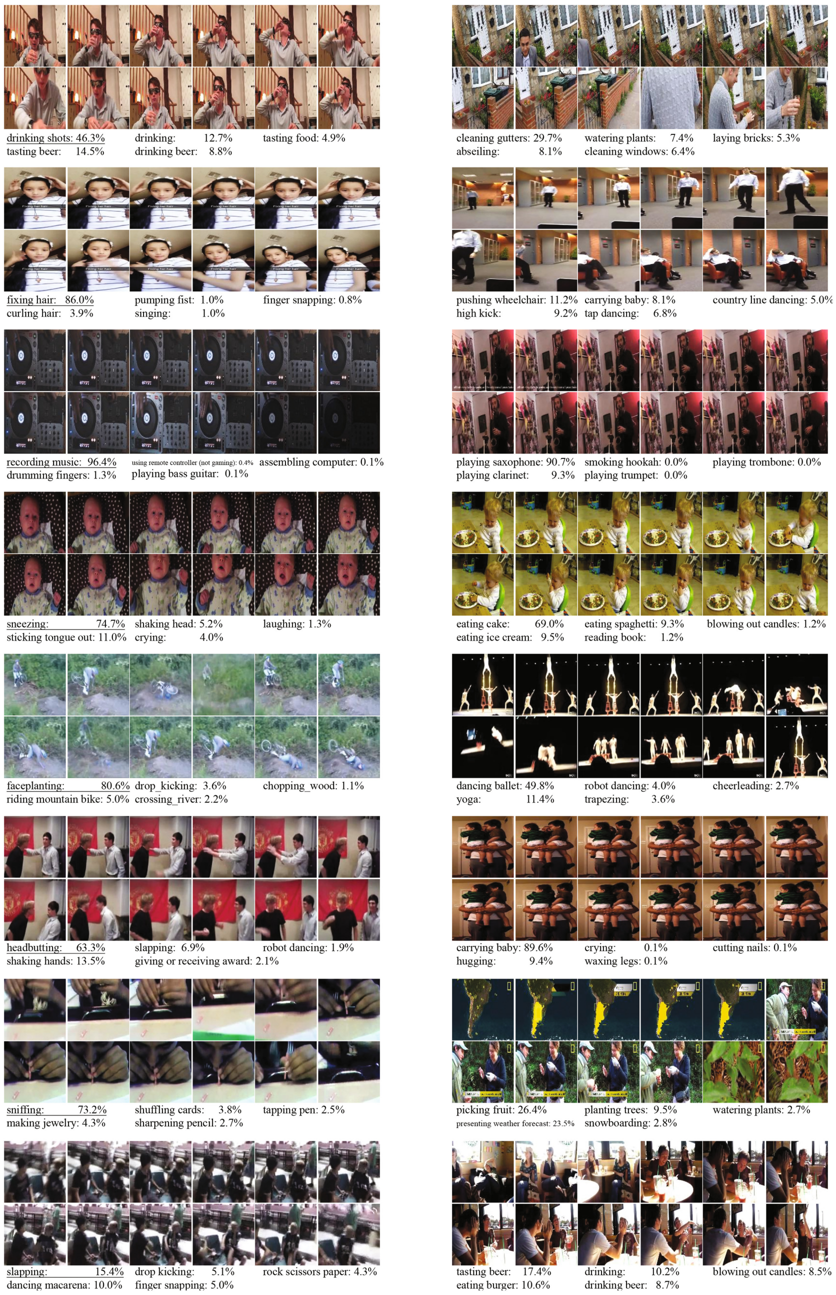


Fig. 8. Predictions made on the most difficult eight categories in Kinetics validation set. Left: Easy samples. Right: Hard samples. Top-5 confidence scores are shown below each video sequence. Underlines are used to emphasize correct prediction. Videos within the same row are from the same ground truth category.

As can be seen from the results, the algorithm is more likely to make mistakes on videos without any distinguishable object or containing an action lasting a relatively short period of time.

5 Conclusion

In this work, we address the problem of building highly efficient 3D convolution neural networks for video recognition tasks. We proposed a novel *multi-fiber* architecture, where sparse connections are introduced inside each residual block effectively reducing computations and a multiplexer is developed to compensate the information loss. Benefiting from these two novel architecture designs, the proposed model greatly reduces both model redundancy and computational cost. Compared with existing state-of-the-art 3D CNNs that usually consume an order of magnitude more computational resources than regular 2D CNNs, our proposed model costs significantly less resources yet achieves the state-of-the-art video recognition accuracy on Kinetics, UCF-101, HMDB51. We also showed that the proposed multi-fiber architecture is a generic method which can also benefit existing networks on image classification task.

Acknowledgements. Jiashi Feng was partially supported by NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112.

References

1. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4724–4733. IEEE (2017)
2. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition (2017). arXiv preprint: arXiv:1711.11248
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
4. Girshick, R.: Fast R-CNN (2015). arXiv preprint: arXiv:1504.08083
5. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs (2016). arXiv preprint: arXiv:1606.00915
6. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1725–1732 (2014)
7. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 4489–4497. IEEE (2015)
8. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning for video understanding (2017). arXiv preprint: arXiv:1712.04851

9. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and imagenet. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, pp. 18–22 (2018)
10. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). arXiv preprint: arXiv:1409.1556
11. Shou, Z., Wang, D., Chang, S.F.: Temporal action localization in untrimmed videos via multi-stage CNNs. In: CVPR (2016)
12. Shou, Z., Chan, J., Zareian, A., Miyazawa, K., Chang, S.F.: CDC: convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In: CVPR (2017)
13. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems, pp. 568–576 (2014)
14. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
15. Ng, J.Y.H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: deep networks for video classification. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4694–4702. IEEE (2015)
16. Wang, L., Li, W., Li, W., Van Gool, L.: Appearance-and-relation networks for video classification (2017). arXiv preprint: arXiv:1711.09125
17. Tran, A., Cheong, L.F.: Two-stream flow-guided convolutional attention networks for action recognition. In: International Conference on Computer Vision (2017)
18. Wu, C.Y., Zaheer, M., Hu, H., Manmatha, R., Smola, A.J., Krähenbühl, P.: Compressed video action recognition (2017). arXiv preprint: arXiv:1712.00636
19. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR (2014)
20. Kay, W., et al.: The kinetics human action video dataset (2017). arXiv preprint: arXiv:1705.06950
21. Shou, Z., Gao, H., Zhang, L., Miyazawa, K., Chang, S.F.: Autoloc: weakly-supervised temporal action localization in untrimmed videos. In: Ferrari, V. (ed.) ECCV 2018, Part XVI. LNCS, vol. 11220, pp. 162–179. Springer, AG (2018)
22. Shou, Z.: Online detection of action start in untrimmed, streaming videos. In: Ferrari, V. et al. (eds.) ECCV 2018, Part III. LNCS, vol. 11207, pp. 551–568. Springer, AG (2018)
23. Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M.: Convnet architecture search for spatiotemporal feature learning (2017). arXiv preprint: arXiv:1708.05038
24. Szegedy, C., et al.: Going deeper with convolutions
25. Howard, A.G., et al.: Mobilenets: efficient convolutional neural networks for mobile vision applications (2017). arXiv preprint: arXiv:1704.04861
26. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation (2018). arXiv preprint: arXiv:1801.04381
27. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: an extremely efficient convolutional neural network for mobile devices (2017). arXiv preprint: arXiv:1707.01083
28. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5987–5995. IEEE (2017)

29. Ahmed, K., Torresani, L.: Maskconnect: Connectivity learning by gradient descent. In: Ferrari, V., et al. (eds.) ECCV 2018, Part V. LNCS, vol. 11209, pp. 362–378. Springer, AG (2018)
30. Chollet, F.: Xception: deep learning with depthwise separable convolutions (2017). arXiv preprint: arXiv:1610.02357
31. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
32. Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S., Feng, J.: Dual path networks. In: Advances in Neural Information Processing Systems, pp. 4470–4478 (2017)
33. Chen, T., et al.: Mxnet: a flexible and efficient machine learning library for heterogeneous distributed systems (2015). arXiv preprint: arXiv:1512.01274
34. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: a dataset of 101 human actions classes from videos in the wild (2012). arXiv preprint: arXiv:1212.0402
35. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 2556–2563. IEEE (2011)
36. Paszke, A., Gross, S., Chintala, S., Chanan, G.: Pytorch (2017)
37. Feichtenhofer, C., Pinz, A., Wildes, R.P.: Spatiotemporal multiplier networks for video action recognition. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7445–7454. IEEE (2017)
38. Wang, L., et al.: Temporal segment networks: towards good practices for deep action recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016, Part VIII. LNCS, vol. 9912, pp. 20–36. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_2