Learning Efficient Video Representation with Video Shuffle Networks

Pingchuan Ma[1*] Yao Zhou[2*] Yu Lu[3] Wei Zhang[3]
[1] MIT [2] Tencent YouTu Lab [3] SenseTime Research
pika7ma@gmail.com yoosan.zhou@gmail.com {luyu,wayne.zhang}@sensetime.com

Abstract

3D CNN shows its strong ability in learning spatiotemporal representation in recent video recognition tasks. However, inflating 2D convolution to 3D inevitably introduces additional computational costs, making it cumbersome in practical deployment. We consider whether there is a way to equip the conventional 2D convolution with temporal vision no requiring expanding its kernel. To this end, we propose the video shuffle, a parameter-free plug-in component that efficiently reallocates the inputs of 2D convolution so that its receptive field can be extended to the temporal dimension. In practical, video shuffle firstly divides each frame feature into multiple groups and then aggregate the grouped features via temporal shuffle operation. This allows the following 2D convolution aggregate the global spatiotemporal features. The proposed video shuffle can be flexibly inserted into popular 2D CNNs, forming the Video Shuffle Networks (VSN). With a simple yet efficient implementation, VSN performs surprisingly well on temporal modeling benchmarks. In experiments, VSN not only gains non-trivial improvements on Kinetics and Moments in Time, but also achieves state-of-the-art performance on Something-Something-V1, Something-Something-V2 datasets.

1. Introduction

End-to-end learning methods have achieved great improvements over previous hand-crafted features [15, 17, 34], and become the mainstream in video recognition area. The design of video recognition models enjoys great benefits of the prior art of still image recognition models. On one hand, many works utilize successful 2D CNNs, such as Inception [31] and ResNet [10] architectures, to extract spatial features of individual frames and then perform temporal aggregation using pooling strategies [13, 6, 36], feature encoding functions [7, 24, 45], recurrent neural networks [3, 41, 19], and even optical flow-guided methods [4, 30, 27]. These approaches incorporate a learnable module into 2D CNNs that captures temporal dependency and motion information. On the other hand, some works directly inflate 2D CNNs into 3D CNNs by replacing $k \times k$ convolution filters with $k \times k \times k$ [1, 32], and then add non-local block [37] to grasp long-range temporal dependency or separate $k \times k \times k$ kernel into $1 \times k \times k$ and $k \times 1 \times 1$ kernels [28, 40, 33] to reduce computational costs. The expanded temporal filters in 3D CNNs thus can conveniently model the temporal information from videos. Although these improved 3D CNNs show their effectiveness for action recognition, they usually introduce additional computational cost. This may limit the usage of 3D CNNs in real-world applications requiring low latency.

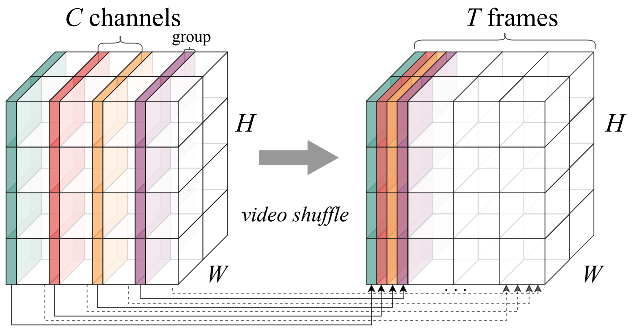


Figure 1. The proposed *video shuffle* first divides channels of each frame feature into several groups with equal sizes, and then stacks the grouped features at same index along temporal dimension into a new frame feature. Through video shuffle, spatial information is exchanged across all frames.

In conventional 2D video models, each frame is independently fed into a 2D CNN to extract feature and then a temporal pooling function aggregates all frame features to video level. We consider the probability of performing temporal modeling by reallocating the inputs of 2D convolution, instead of introducing the temporal integrating methods. To this end, we equip 2D convolution with spatio-temporal receptive field by employing the recent group [16, 40] and shuffle operations [43]. Specifically, we propose video shuffle, an efficient and generic plug-in component for modeling temporal dependency in 2D CNNs with zero cost. As shown in Figure 1, video shuffle first di-

*Equal contribution.

vides channels of each frame into several groups with equal size, and then aggregates all of grouped features with same group index into a new frame feature. The reallocated frame feature contains spatial information of all frames and therefore the following 2D convolutions can conveniently learn both spatial and temporal representation. Video shuffle is superiority efficient since there are no additional parameters and FLOPs (addition or multiplication) introduced. The computation time of the proposed video shuffle comes only from the data movement in memory, which hardly affects the inference latency.

Video shuffle can be easily incorporated into 2D video models. In this work, we adopt temporal segment networks (TSN) [36] as our basic model, and take ResNet-50 and ResNet-101 [10] as the backbones. In implementation, we plug video shuffle and its inverse operation, which restores the original spatial representation of each frame, into ResNet, before and after 2D convolutions inside residual blocks respectively. To demonstrate the effectiveness of the proposed VSN, we conduct experiments on several popular video action recognition datasets, including large-scale Kinetics and Moments in Time as well as the temporal-sensitive Something-Somethings. In experiments, VSN outperforms its counterpart on all datasets at the cost of zero parameters and zero FLOPs. Moreover, VSN surpass it by a large margin and further achieves state-of-the-art performance on the challenging Something-Something-V1, Something-Something-V2 datasets.

2. Related Work

2.1. Video Recognition Models

The conventional 2D CNNs learn video representation using 2D CNNs as spatial features extractor for frames and then performing temporal aggregation over frame features. In [13], they made use of 2D CNNs to extract features from individual frames and then integrated temporal features into a fixed-size video representation using various fusion methods. Many works focused on designing temporal aggregation methods to improve the recognition accuracy. Pooling approaches [24, 6, 36], feature encoding functions [7, 45] and recurrent neural networks [3, 41, 19] were usually preformed on high-level features, while optical flow-guided methods computed motion information on low and middle-level features [4, 30]. Two-stream framework introduced by [29] is a widely-used approach to capture the motion information. It fused deep features extracted from optical flows and traditional features computed from RGB inputs.

On the other hand, a video can be viewed as a cube stacked of many images. That is to say, a 3D convolution can process video directly. Previous works demonstrate that 3D CNNs [32] can straightforward learn the spatio-temporal features. In order to take benefits of the successful 2D CNNs and ImageNet pretraining, Carreira and Zisserman [1] introduces the Inflated 3D ConvNets (I3D) based on the Inception architecture [31, 12] and show its superior performance on a large human action recognition benchmark [14]. Meanwhile, several 3D variants are proposed [28, 40, 33, 9]. Qiu *et al*. [28] introduces a Pseudo-3D ResNet architecture. Tran *et al*. [33] presents the R(2+1)D model based on ResNet. Xie *et al*. [40] investigates where "deflating" 3D convolution are more suitable and then presents the separable-3D model built upon I3D. These mixed 2D and 3D networks are constructed by replacing $k \times k \times k$ filter to $1 \times k \times k$ followed by a $k \times 1 \times 1$ filter. Additionally, non-local neural network [37] and its improved version [38], trajectory convolution [44], energy-based models [35] are also introduced.

In parallel, some works focused on efficient model design in video understanding. The most related approaches were ECO [46] and TSM [20]. ECO [46] employed a 3D-net stacking on the 2D feature extractors to model the temporal relationships, and they further proposed an online video understanding algorithm for fast video inference. Although ECO achieves a good runtime-accuracy trade-off, it still increased the computational costs compared to 2D CNNs. TSM [20] introduced a zero-cost temporal shift module which *shift*s part of channels along temporal dimension by ± 1 to fuse temporal information. TSM not only had $2.7\times$ fewer FLOPs than ECO family but also achieved finer recognition performance, especially on temporal-sensitive datasets.

2.2. Group and Shuffle Operation

The idea of splitting channels into several groups was first presented in AlexNet [16] for distributing the model over two GPUs to handle the memory issue, and then widely used for designing tiny and efficient network architectures [11]. In ResNeXt [39], they further developed the group convolution to reduce the number of parameters and computational complexity by dividing input channels into several groups, then performing regular convolution on each group and concatenating all group results as the outputs. Experiments demonstrated that group convolution can lead to performance improvement on image recognition task. But when stacking group convolution multiple layers, the outputs from a certain channel were only derived from a small fraction of input channels. To address this weakness, ShuffleNet [43] utilized the channel shuffle operation, by which the resulting channels of each group were collected from all input groups, to enable information interaction across groups. It not only greatly reduced computational costs but also maintained accuracy. Besides, Zhang $etal$ [42] proposed an interleaved group convolutions which consists of a primary group convolution for handling spatial correlation and a secondary group convolution for blend-

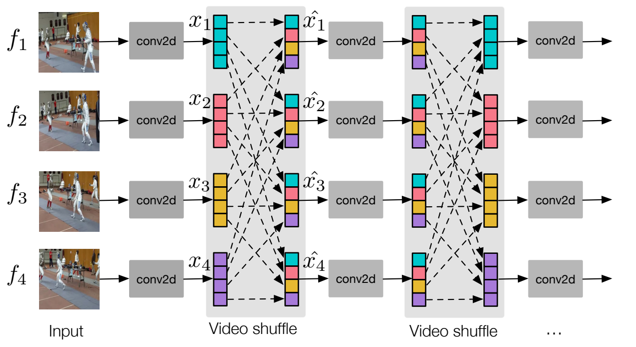


Figure 2. Graphical representation of *video shuffle networks*. Isolated frame features x_i are aggregated into video-level feature $\hat{x}_i$ via *video shuffle* operations. In this manner, instead of focusing on an instant moment, subsequent 2D convolutions are endowed with non-local perception and thus learn video representation more effectively and efficiently.

ing the channels and show its effectiveness. A special case of group convolutions was channel-wise convolution where the number of groups is equal to the number of channels, this is also very similar to the separable convolution [2, 11]. The basic idea of group and shuffle operations are adopted in this work.

3. Video Shuffle Networks

In this section, we first introduce the design criteria of video shuffle, and show how to incorporate it into the building block of ResNet. Then, we present the overall network architecture of VSN, followed by implementation details.

3.1. The Design of Video Shuffle

The design motivation of video shuffle lies in the fact that though recent 3D CNNs have improved recognition performance, they could hardly be deployed into real-world video recognition systems due to its heavy computational cost. The conventional 2D CNNs enjoy low latency, but they learn spatial feature from isolated frames without temporal modeling, leading to accuracy gap against state-of-the-arts models. Namely, there is an accuracy-speed trade-off in video recognition models. To address it, we propose to equip 2D CNNs with temporal receptive field by reallocating the inputs of 2D convolution. In order to enable 2D convolution learn both spatial and temporal feature without modifying its structure, there are two prerequisites: 1) the input should contain spatial representation from all frames and 2) its input size should not be changed. Group operation divides input channels into several groups to make each contain partial representation and shuffle operation further facilitates information exchanging across different groups. These features make them perfect choices in this work.

Figure 2 shows a graphical representation of the proposed video shuffle. A video with T frame features is shown as an example and each one of them is a $[C, H, W]$ tensor extracted by 2D convolutions, where C indicates the channel size, H and W are spatial dimensions. For each frame feature, we first divide channels into several groups with equal sizes. In this work, number of groups is heuristically set to number of frames T, in consequence, channel size of each grouped feature equals C/T. In this way, each grouped feature with shape of $[C/T, H, W]$ contains a part of spatial feature. We aggregate all of grouped features with same group index into a new frame feature by temporal shuffle operation, which allows spatial features exchanging across different frames. As illustrated in Figure 1 right: the reallocated feature at the first frame is a stack of all first grouped features in Figure 1 left (before video shuffle applied).

Denoting a feature at i-th frame as x_i, video shuffle transforms the original x_i to a new feature $\hat{x}_i$ by the following equation:

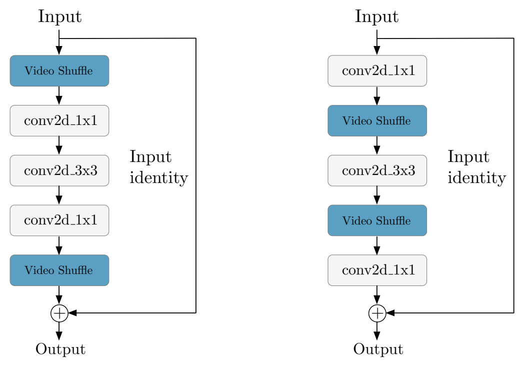
$$\hat{x}_i = [x_1^{(i-1)\eta:i\eta}, x_2^{(i-1)\eta:i\eta}, \dots, x_T^{(i-1)\eta:i\eta}], \quad (1)$$

where $1 \leqslant i \leqslant T$, η is the channel size of each group and $\eta = C/T$ in this setting, the symbol : denotes the tensor slicing operation along channel dimension. The index i plays the role of both frame and group index. For instance, $x_1^{0:\eta}$ indicates the first grouped feature at the first frame (masked green in Figure 1 left). As a result, the new frame feature $\hat{x}_i$ contains the spatial information of all sequential frames and further serves as the inputs of the following 2D convolutions. The proposed video shuffle has three advantages: first, it allows spatial features interacting across different frames; second, the following 2D convolutions in 2D CNNs can handily perform both spatial and temporal modeling; and third, video shuffle is easy to implement via data movement in memory, not introducing additional parameters or theoretical FLOPs at all.

3.2. Video Shuffle in Residual Block

Since we have obtained a parameter-free video shuffle that is able to model temporal information cooperating with 2D convolutions, we consider inserting it into conventional 2D CNNs. In this work, we mainly study on the ResNet architectures. As a result, we attempt to plug video shuffle and its reverse operation, which restores the original spatial representation for each frame, into the primary building block of ResNet. We investigated two positions to insert video shuffle units and obtain two variants: the *headtail (residual) block* and *compact (residual) block*.

We first consider placing video shuffle at the head and tail of a residual block, namely *headtail block*. Before any convolutional layer, the inputs go though the first video shuffle directly, and each new frame is consequently composed with partial spatial features from all sampled frames.



(a) headtail (b) compact

Figure 3. A headtail block places video shuffle at the head and tail of a residual block while a compact block performs video shuffle before and after conv2d_3x3 immediately.

The following convolutions in bottleneck block end-to-end learn both spatio-temporal feature consequently. After them, to guide subsequent convolutions focusing on spatial reasoning, we restore the original spatial feature of each frame by inserting an inverse video shuffle. As for *compact block*, video shuffle units are similarly configured with a "paired" setting but compactly performed before and after conv2d_3x3 (shown in Figure 3).

Given that a bottleneck performs compression on channel dimension where the spatial and temporal information blend, we argue that a weakness of headtail residual block could be an information loss along with dimensional reduction. We further empirically verify this assumption in experiments (ablation studies). Results demonstrate that compact block is stronger than headtail in temporal modeling. Unless specified, we always use the compact block in following experiments.

3.3. Network Architectures

layer name	ResNet-50	ResNet-101	output size
conv_1	7×7, stride 2		$64\times8\times112\times112$
pool_1	3×3 max, stride 2		$64\times8\times56\times56$
res_2	standard block $\times2$ compact block $\times1$	standard block $\times2$ compact block $\times1$	$256\times8\times56\times 56$
res_3	standard block $\times3$ compact block $\times1$	standard block $\times3$ compact block $\times1$	$512\times8\times28\times 28$
res_4	standard block $\times5$ compact block $\times1$	standard block $\times22$ compact block $\times1$	$1024\times8\times14\times 14$
res_5	standard block $\times2$ compact block $\times1$	standard block $\times2$ compact block $\times1$	$2048\times8\times7\times 7$
global average pooling			$2048\times1\times1\times1$

Table 1. The proposed VSN-ResNet-50 and VSN-ResNet-101 architectures for video recognition.

Table 1 presents VSN with ResNet-50 and ResNet-101 backbone for video recognition. In this work, we have not added or modified any convolution or pooling layer. Instead of incorporating video shuffle into all building blocks, we heuristically insert it into the last building block in each ResNet layer. For example, in res_2 of ResNet-50, we replace the third block with the compact video shuffle and retain the first two blocks. That is, there are **totally four blocks** equipped with video shuffle in overall ResNet architectures, and more related ablation studies are presented in experiments.

In TSM [20], they insert the temporal shift module into the residual block and show its effectiveness in video recognition. We argue that TSM only allows temporal information to be interchanged between neighbor frames, which models temporal information locally. As a result, it fails to take advantages of non-local details in long range. Compared with local-field TSM, video shuffle broadens its horizon to all frames and models temporal dependency in a global version. As TSM is orthogonal to video shuffle, we further combine them together. In terms of implementation, we use their *residual temporal shift module* with zero padding to replace the building blocks which have not be incorporated with video shuffle e.g. the first two blocks in res_2 of ResNet-50. Experiments are conducted to show the superiority of such combination in temporal modeling. In practice, we replace the last block of res_2, res_3, res_4 and res_5 with our compact block and add temporal shift module to the other residual blocks. We denote our video model as *VSN-ResNet-L* (*VSN-RL* for simplicity) if the backbone is *ResNet-L*, where L indicates the number of layers, e.g. *VSN-ResNet-50 (VSN-R50) and VSN-ResNet-101 (VSN-R101).*

3.4. Implementation Details

In this work, all models are implemented in PyTorch. ResNet architectures and ImageNet pre-trained models are derived from *torchvision* package.

Training. We sample 8 frames from an entire video using the sparse segment-based sampling [36]. For data augmentation, our implementation follows the practice in [36] to alleviate negative effects of overfitting. The images are first resized with shorter side to 256 and then applied by corner cropping and scale-jittering. We also apply random left-right flipping consistently for all videos except actions are horizontal-order-sensitive in Something-Something. e.g. *Pushing something from left to right*. Finally, the cropped images are resized to 224×224 pixels for network training. We distribute totally 64 videos into 8 TITANXP GPUs and each GPU has 8 videos in a mini-batch. We adopt SGD with momentum as optimizer and set its initial learning rate to 0.01. We utilize both the multi-step learning rate decaying and cosine learning rate schedule [23] with warm-up depending on dataset. The momentum value, weight decay and dropout rate are set to 0.9, 5e-4 and 0.8 respectively. We freeze all batch normalization except the first convolution layer.

Model	#Frame	#Params	FLOPs	Sth-V1 val	Sth-V1 test	Sth-V2 val	Sth-V2 test
TSN [36]	8	10.7M	16G	19.5	-	33.4	-
TRN-Multiscale [45]	8	18.3M	16G	34.4	33.6	48.8	50.9
Two-stream TRN [45]	8+8	36.6M	-	42.0	40.7	55.5	56.2
ECO [46]	16	47.5M	64G	41.4	-	-	-
ECO$_{En}$_*Lite* [46]	92	150M	267G	46.4	42.3	-	-
ECO$_{En}$_*Lite*$_{\text{R+F}}$ [46]	92+92	300M	-	49.5	43.9	-	-
I3D [37]	64	28.0M	306G	41.6	-	-	-
NL-I3D [37]	64	35.3M	335G	44.4	-	-	-
NL-I3D + GCN [38]	64	62.2M	605G	46.1	45.0	-	-
TrajectoryNet [44]	32	33.3M	-	47.8	-	-	-
TSM [20]	8	24.3M	33G	43.4	-	59.1	-
TSM [20]	16	24.3M	65G	44.8	-	59.4	60.4
Two-stream TSM [20]	16+16	48.6M	-	50.2	47.0	64.0	64.3
VSN-R50$_{\text{RGB}}$	8	24.3M	33G	46.6	-	60.6	-
VSN-R50$_{\text{Flow}}$	8	24.3M	33G	38.8	-	53.8	-
Two-stream VSN-R50	8+8	48.6M	-	51.6	-	65.3	-
VSN-R101$_{\text{RGB}}$	8	42.9M	63G	**47.8**	-	**61.6**	-
VSN-R101$_{\text{Flow}}$	8	42.9M	63G	41.4	-	56.7	-
VSN-R{50+101}$_{\text{RGB}}$	8	67.2M	96G	49.2	46.8	63.2	64.6
Two-stream VSN-R101	8+8	85.8M	-	**52.7**	**49.9**	**65.8**	**66.1**

Table 2. Comparison of the proposed VSN with the previous state-of-the-art models on Sth-V1 and Sth-V2 (Top-1 accuracy).

Inference. In the inference phase, TSN [36] takes the average predictions of 25×10 crops as the video prediction. I3D and S3D [39] densely sample all frames and take center crops for evaluation. In this work, we take the same pre-processing as non-local neural network [37], which performs spatially fully-convolutional inference on videos whose shorter side is re-scaled to 256. For temporal domain, we also sample total 8 frames during evaluation.

4. Experiments

In this paper, extensive experiments are performed on four popular and challenging video recognition benchmarks. We first introduce these experimental benchmarks and then show that the proposed VSN can not only perform very well on Kinetics, but also achieve state-of-the-art performance on Something-Something-V1, Something-Something-V2 and Moments in Time datasets.

4.1. Datasets

We conduct experiments on various video datasets with great diversity, whose sources range from YouTube to crowdsourcing videos and durations range from three seconds to tens of seconds, covering human daily activities, human actions to sports and events.

Kinetics [14] is a large human action recognition dataset, which contains around 240k training videos and 20k validation videos, involving 400 human cation classes.

Moments in Time (Moments) [25] includes a collection of 1 million trimmed 3s-video clips, corresponding to 339 dynamic event categories.

Something-Something-V1 (Sth-V1) [8] is a temporal-sensitive dataset, containing 108,499 videos. The total 174 categories are basic actions with objects.

Something-Something-V2 (Sth-V2) [8] increases its number of videos to 220,847 and further improves the annotation quality and pixel resolution.

Model	Backbone	Top-1	Top-5
TSN [36]	Inception-V3	71.5	90.2
Attention-Cluster [22]	Inc-Res-v2	75.0	91.9
NL-C2D [37]	ResNet-101	75.1	91.6
CPNet [21]	ResNet-101	75.3	92.4
I3D [1]	Inception	72.1	90.3
R(2+1)D [33]	ResNet-34	74.3	91.4
S3D-G [40]	Inception	74.7	93.4
CoST [18]	ResNet-101	77.5	93.2
NL-I3D [37]	ResNet-101	77.7	93.3
SlowFast+NL [5]	ResNet-101	**79.8**	**93.9**
VSN-R50	ResNet-50	73.5	91.3
VSN-R101	ResNet-101	75.4	92.2
Two stream VSN-R101	ResNet-101	77.6	93.7

Table 3. Comparison of VSN and the previous state-of-the-art models on the validation set of Kinetics.

4.2. Results on Something-Something

Something-Something-V1. We first show a comparison of the performance between VSN and previous state-of-the-art methods in Table 2, on both validation and test set of Something-Something-V1. The top-1 accuracy as well as the statistics of computational costs are reported.

Previous results are presented in the first group. [45]

found that TSN fails to reason temporal relation and thus proposed the temporal relation networks (TRN) to learn temporal dependencies between video frames at multiple time scales. They show that TRN-multiscale can improve TSN by 14.7% and fusing optical flow gives another 7.6% improvement. In [46], they introduced the efficient video understanding model ECO by leveraging the 3D-net stacking on 2D features. Their best single model achieved an accuracy of 41.4%. Some works attempted to use pure 3D models. Both I3D [1] and its improved version NL-I3D [37] obtained good performance, but their computational costs (FLOPs) are too huge to deploy. Furthermore, to explicitly model relationships between humans and objects, NL-I3D+GCN [38] used a object detector to extract region proposals and composed these regions from different frames by the graph convolution network. Although NL-I3D+GCN achieved a very competitive accuracy of 46.1%, the introduced computational cost is non-negligible. TrajectoryNet [44] allows visual features to be aggregated along motion paths by a trajectory convolution, achieving a higher accuracy of 47.8%. The recent temporal shift module (TSM) achieved 43.4% when taking 8 RGB frames as inputs. As number of frames was increased to 16, it gained another 1.4% boosts. The previous state-of-the-art performance was held by $\text{TSM}_{\text{RGB+Flow}}$, which fused 16-frames RGB model with another optical flow stream.

Our results are presented at the last two groups. Taking 8 RGB frames as inputs, our VSN-R50 achieves 46.6% accuracy, outperforming TSN and TSM by 27.1% and 3.2% respectively. This demonstrates that video shuffle performs outstanding for temporal modeling. When fused with optical flow stream, two-stream VSN-R50 achieves 51.6% accuracy, which is 1.4% higher than two-stream TSM. Going deeper with network architecture from ResNet-50 to ResNet-101 gives notable 2.2% improvements. The best single model VSN-R101 gets an accuracy of 47.8%. Our ensemble model, which averages the predictions of VSN-R101 and VSN-R50, achieves top-1 accuracy of 49.2%. Furthermore, the two-stream VSN-R101 gets a new state-of-the-art 52.7% performance on Something-Something-V1 dataset. We also submit test predictions to the evaluation server and report test results. The trend of improvement is basically consistent with the validation set and the best performance 49.9% is held by our two-stream VSN-R101.

Something-Something-V2. It distinguishes from previous Sth-V1 dataset with increased training examples, better annotation quality and higher video resolution. The comparison results of VSN with the state-of-the-art methods also listed in Table 2. The previous models have been introduced in the above experiments. Similar to Sth-V1 dataset, our models are able to outperform both 2D and 3D counterparts. VSN-R50 and VSN-R101 achieve 60.6% and 61.6% respectively. With VSN going deeper, the improvement gains constantly climb higher. VSN-R101 outperforms TSM by 2.5%. In accord with our expectation, the ensemble models ($\text{VSN-R}\{50+101\}_{\text{RGB}}$) show big advantages over the ones fed with single modality. Evaluated on the validation and test set, our two-stream VSN-R101 establishes the new state-of-the-art on both sets.

4.3. Results on Kinetics and Moments in Time

Model	Backbone	Top-1	Top-5
TSN [36]	BN-Inception	24.11	-
Two-stream TSN [36]	BN-Inception	25.32	50.10
TRN [45]	BN-Inception	25.97	-
Two-stream TRN [45]	BN-Inception	28.27	53.87
ResNet50-ImageNet [25]	ResNet-50	27.16	51.68
Two-stream I3D [25]	Inception-V1	29.51	56.06
CoST_{8F} [18]	ResNet-50	30.10	57.20
CoST_{8F} [18]	ResNet-101	31.50	57.90
CoST_{32F} [18]	ResNet-101	32.40	60.00
VSN-R50_{8F}	ResNet-50	31.74	58.66
VSN-R101_{8F}	ResNet-101	**32.65**	**61.47**

Table 4. Comparison with state-of-the-arts on Moments in Time dataset. Our models are trained only on RGB inputs.

Kinetics and Moments Our VSN models are also evaluated on both Kinetics and Moments in Time, featuring huge size and tough task. Table 3 and Table 4 compares our VSN-R50 and VSN-R101 models against the previous state-of-the-art models on Kinetics and Moments respectively. First, it is observed that VSN outperforms TSN by a considerable margin, which verifies the effectiveness of the proposed video shuffle component. Second, compared with 2D attention-based models, such as Attention-Cluster and NL-C2D listed in Table 3, our VSN-R101 achieves a very close performance. It demonstrates video shuffle indeed can act as a non-local feature integrator. Third, VSN-R101 can outperform 3D variants, such as I3D, R(2+1)D and S3D-G. Even comparing to huge 3D counterparts, our two-stream VSN-R101 is also competitive. Although both 2D and 3D models are not well-performed on the challenging Moments, VSN can outperform these counterparts. VSN-R101 beats all other models and stands as a new state-of-the-art.

4.4. Generalize to Optical Flow

We also verify whether VSN can generalize to optical flow. For these experiments, we follow the standard setup as described in [29] and extract optical flow with the TV-L1 algorithm [26]. All models are trained on the Kinetics and Sth-V1 and report the top-1 accuracy. We sample 8 segments in training optical flows like RGB. In [29, 36], they stack 5 or 10 consecutive optical flows for capturing the long-term temporal dependency in videos. We consider that VSN have ability to learn long-range temporal dependency and verify it by training models using only 1 optical flow in a segment.

Model	#Flow	Kinetics	Sth-V1
ResNet-50 [36]	8×1	47.5	27.0
ResNet-50 [36]	8×5	54.8	34.3
ResNet-101 [36]	8×1	49.7	29.2
ResNet-101 [36]	8×5	56.5	34.3
VSN-R50	8×1	53.0	33.7
VSN-R50	8×5	56.7	37.5
VSN-R101	8×1	56.0	36.1
VSN-R101	8×5	60.1	41.4

Table 5. Comparison of VSN models against TSN counterparts on Kinetics and Sth-V1, trained on optical flow inputs.

Model	Latency	Throughput	Sth Acc(%)
I3D [1]	165.3ms	6.1vps	41.6
TSN-R50 [36]	15.8ms	80.8vps	20.2
TSN-R101 [36]	26.7ms	48.8vps	22.7
ECO_{16F} (Zolfaghari et.al 2018)	30.6ms	45.6vps	41.4
TSM_{8F} [20]	17.4ms	77.4vps	43.4
VSN-R50	16.5ms	79.5vps	44.5
VSN-R101	28.7ms	47.2vps	46.5

Table 6. Comparison in latency of VSN against the others.

The results are shown in Table 5. The first group presents the performance of TSN baseline with different backbones while the second one presents ours. Trained with 8×1 flow as inputs on Kinetics and Sth-V1, our VSN-R50 outperforms its counterpart by 5.5% and 6.7% respectively. By increasing the number of inputting flows from 1 to 5, both baseline and our model yield considerable gains. Furthermore, our VSN-R50 trained on 8×1 optical flows is able to achieve performance close to the TSN ResNet-50 with 8×5 flows, whose inputs are $5\times$ more than ours. Going deeper with VSN, the improvement grows considerably. VSN-R101 outperforms VSN-R50 models by around 3%-4% accuracy.

4.5. Inference Latency

To measure the latency and throughput, we perform inference on one NVIDIA Tesla P100 GPU and use the average value of 500 times batch inference with batch size of 16. Following [20], we provide the speed of VSN-R50 and VSN-101. The *vps* indicates the videos per second. It is clearly observed from Table 6 that our VSN models act superior not only by high accuracy but also by low latency and high throughput. Compared to I3D, VSN gets $13\times$ speedup along with higher accuracy. It is also illustrated that video shuffle can hardly hurt the runtime speed: VSN has almost the same latency and throughput as TSN, but it brings 20%+ improvement.

4.6. Ablation Studies

Which residual block is better for temporal modeling? Table 7 shows top-1 accuracies of compact and headtail residual block both on Kinetics and Sth-V1. In this setting, we train our models with RGB inputs and replace all last blocks at different ResNet layers with video shuffle blocks. Although both of two variants outperform the baseline, compact residual block clearly outperforms headtail counterpart, no matter testing on temporal-sensitive dataset or using backbones network with different depth.

Backbone	Block	Kinetics	Sth-V1
ResNet-50	baseline	71.5	20.2
	headtail	$72.1^{+0.6}$	$43.2^{+23.0}$
	compact	$\mathbf{73.5^{+2.0}}$	$\mathbf{46.6^{+26.4}}$
ResNet-101	baseline	72.8	22.7
	headtail	$74.0^{+1.2}$	$44.5^{+21.8}$
	compact	$\mathbf{75.4^{+2.6}}$	$\mathbf{47.8^{+25.1}}$

Table 7. Comparison with different residual blocks.

#Blocks	Kinetics	Sth-V1
0	71.5	20.2
1	$72.2^{+0.7}$	$40.7^{+20.5}$
2	$73.0^{+1.5}$	$42.2^{+22.0}$
3	$72.8^{+1.3}$	$43.9^{+23.7}$
4	$\mathbf{73.5^{+2.0}}$	$\mathbf{46.6^{+26.4}}$

Table 8. Optimal number of compact blocks.

Model	Kinetics	Sth-V1
baseline	71.5	20.2
+ temporal shift	$72.4^{+0.9}$	$43.4^{+23.2}$
+ video shuffle	$73.3^{+1.8}$	$46.0^{+25.8}$
+ combination	$\mathbf{73.5^{+2.0}}$	$\mathbf{46.6^{+26.4}}$

Table 9. Comparison of video shuffle and temporal shift.

How many blocks are replaced with video shuffle block? As discussed above, the last block of ResNet layer is replaced by our compact residual block. We conduct experiments to verify whether our model can capture temporal information more effectively using more video shuffle blocks. Since a video shuffle block could be place at arbitrary ResNet layer, e.g. res_2, we average scores achieved by models whose number of video shuffle blocks is same. The results are reported in Table 8. It is obvious that increasing the number leads to better accuracy. Each ResNet layer having one video shuffle block (totally four) performs best.

Comparison of temporal shift and video shuffle. Table 9 presents the respective temporal modeling ability of temporal shift and video shuffle, as well as their combined version. In comparison to temporal shift module, video shuffle is more competitive on both Kinetics and Sth-V1. Furthermore, combining these two component gains higher scores and shows superior temporal modeling capability.

5. Conclusion

In this paper, we introduced the video shuffle network, an efficient video recognition model that can conveniently learn spatio-temporal representation by inserting video shuffle into 2D CNNs. VSN not only enables 2D convolutions performing temporal modeling, but also hardly increases the overall latency. In experiments, VSN outperforms its counterparts by a great margin and further achieves state-of-the-art performance on Something-Something-V1, Something-Something-V2 and Moments in Time. We hope that our insights will inspire new efficient network designs concentrating computation and accuracy trade-off in video recognition.

References

[1] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4733. IEEE, 2017. 1, 2, 5, 6, 7

[2] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 3

[3] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 1, 2

[4] L. Fan, W. Huang, C. Gan, S. Ermon, B. Gong, and J. Huang. End-to-end learning of motion representation for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6016–6025, 2018. 1, 2

[5] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. *arXiv preprint arXiv:1812.03982*, 2018. 5

[6] R. Girdhar and D. Ramanan. Attentional pooling for action recognition. In *Advances in Neural Information Processing Systems*, pages 34–45, 2017. 1, 2

[7] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 971–980, 2017. 1, 2

[8] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, volume 2, page 8, 2017. 5

[9] K. Hara, H. Kataoka, and Y. Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018. 2

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2, 3

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 2

[13] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. 1, 2

[14] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2, 5

[15] A. Klaser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association, 2008. 1

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1, 2

[17] I. Laptev. On space-time interest points. *International journal of computer vision*, 64(2-3):107–123, 2005. 1

[18] C. Li, Q. Zhong, D. Xie, and S. Pu. Collaborative spatio-temporal feature learning for video action recognition. *arXiv preprint arXiv:1903.01197*, 2019. 5, 6

[19] F. Li, C. Gan, X. Liu, Y. Bian, X. Long, Y. Li, Z. Li, J. Zhou, and S. Wen. Temporal modeling approaches for large-scale youtube-8m video understanding. *arXiv preprint arXiv:1707.04555*, 2017. 1, 2

[20] J. Lin, C. Gan, and S. Han. Temporal shift module for efficient video understanding. *arXiv preprint arXiv:1811.08383*, 2018. 2, 4, 5, 7

[21] X. Liu, J.-Y. Lee, and H. Jin. Learning video representations from correspondence proposals. In *IEEE CVPR*, 2019. 5

[22] X. Long, C. Gan, G. de Melo, J. Wu, X. Liu, and S. Wen. Attention clusters: Purely attention based local feature integration for video classification. In *IEEE CVPR*, 2018. 5

[23] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 4

[24] A. Miech, I. Laptev, and J. Sivic. Learnable pooling with context gating for video classification. *arXiv preprint arXiv:1706.06905*, 2017. 1, 2

[25] M. Monfort, A. Andonian, B. Zhou, K. Ramakrishnan, S. A. Bargal, Y. Yan, L. Brown, Q. Fan, D. Gutfreund, C. Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 5, 6

[26] J. S. Pérez, E. Meinhardt-Llopis, and G. Facciolo. Tv-l1 optical flow estimation. *Image Processing On Line*, 2013:137–150, 2013. 6

[27] A. Piergiovanni and M. S. Ryoo. Representation flow for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1

[28] Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5534–5542. IEEE, 2017. 1, 2

[29] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 2, 6

[30] S. Sun, Z. Kuang, L. Sheng, W. Ouyang, and W. Zhang. Optical flow guided feature: a fast and robust motion representation for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1390–1399, 2018. 1, 2

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 1, 2

[32] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 4489–4497. IEEE, 2015. 1, 2

[33] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 1, 2, 5

[34] H. Wang and C. Schmid. Action recognition with improved trajectories. *2013 IEEE International Conference on Computer Vision*, pages 3551–3558, 2013. 1

[35] L. Wang, W. Li, W. Li, and L. Van Gool. Appearance-and-relation networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1430–1439, 2018. 2

[36] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016. 1, 2, 4, 5, 6, 7

[37] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018. 1, 2, 5, 6

[38] X. Wang and A. Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417, 2018. 2, 5, 6

[39] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1492–1500, 2017. 2, 5

[40] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv preprint arXiv:1712.04851*, 2017. 1, 2, 5

[41] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702, 2015. 1, 2

[42] T. Zhang, G.-J. Qi, B. Xiao, and J. Wang. Interleaved group convolutions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4373–4382, 2017. 2

[43] X. Zhang, X. Zhou, M. Lin, and J. Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018. 1, 2

[44] Y. Zhao, Y. Xiong, and D. Lin. Trajectory convolution for action recognition. In *Advances in Neural Information Processing Systems*, pages 2208–2219, 2018. 2, 5, 6

[45] B. Zhou, A. Andonian, A. Oliva, and A. Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 803–818, 2018. 1, 2, 5, 6

[46] M. Zolfaghari, K. Singh, and T. Brox. Eco: Efficient convolutional network for online video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 695–712, 2018. 2, 5, 6