Exploring Temporal Differences in 3D Convolutional Neural Networks

Gagan Kanojia, Sudhakar Kumawat, and Shanmuganathan Raman

Indian Institute of Technology Gandhinagar, Gandhinagar, India
{gagan.kanojia,sudhakar.kumawat,shanmuga}@iitgn.ac.in

Abstract. Traditional 3D convolutions are computationally expensive, memory intensive, and due to large number of parameters, they often tend to overfit. On the other hand, 2D CNNs are less computationally expensive and less memory intensive than 3D CNNs and have shown remarkable results in applications like image classification and object recognition. However, in previous works, it has been observed that they are inferior to 3D CNNs when applied on a spatio-temporal input. In this work, we propose a convolutional block which extracts the spatial information by performing a 2D convolution and extracts the temporal information by exploiting temporal differences, i.e., the change in the spatial information at different time instances, using simple operations of shift, subtract and add without utilizing any trainable parameters. The proposed convolutional block has same number of parameters as of a 2D convolution kernel of size $n \times n$, i.e. n^2, and has n times lesser parameters than an $n \times n \times n$ 3D convolution kernel. We show that the 3D CNNs perform better when the 3D convolution kernels are replaced by the proposed convolutional blocks. We evaluate the proposed convolutional block on UCF101 and ModelNet datasets. All the codes and pretrained models are publicly available at `https://github.com/GaganKanojia/SSA-ResNet`.

Keywords: Deep learning · 3D convolution neural networks

1 Introduction

Lately, 3D convolutional neural networks are gaining popularity over the 2D CNNs when the task is to deal with 3D data representations which could be videos, shapes or other formats [6,17]. This is because 2D CNN lack in exploiting the temporal information. 3D CNNs are more proficient than 2D CNNs in extracting temporal information and utilizing it to perform specific tasks. It has been shown that a 3D CNN of same depth as that of a 2D CNN performs better on tasks like action recognition [6,18]. However, this proficiency comes with a cost in terms of the number of learnable parameters, memory requirements, and risks of overfitting. For example, 3D ResNet (18 layers) [6] has around 3 times more parameters than the 2D ResNet (18 layers) [7].
In this work, our focus is on acquiring both spatial and temporal structure of the 3D data while reducing the cost in terms of trainable parameters. We propose

a convolutional block which exploits both the spatial information and the temporal information by utilizing a 2D convolution and temporal differences, i.e., the change in the spatial information at different time instances, using simple operations of shift, subtract and add. We have also incorporated temporal max pooling in order to downsample the temporal depth of the feature maps along the depth of the network. None of the operations other than 2D convolution require trainable parameters which makes the number of trainable parameters of the proposed convolutional block equal to the 2D convolution kernel with same kernel size. The major contributions of the work are as follows. **(a)** We propose a novel convolutional block which captures spatial information by performing a 2D convolution and captures temporal information using simple operations of shift, subtract and add. **(b)** We reduce the number of parameters by n times by replacing the 3D convolution kernel of size $n \times n \times n$ with the proposed convolution block comprising a 2D convolution kernel of size $1 \times n \times n$. **(c)** We show that the proposed convolutional block helps the 3D CNNs to perform better while utilizing lesser parameters than the 3D convolution kernels.

2 Related work

In recent years, 2D CNNs have been dominating several applications of computer vision like object detection [7] and image classification[7]. However, they lack in extracting the temporal information present in the spatio-temporal data [18]. There are works which extend the 2D CNNs on videos by processing the video frames individually and then combining the extracted information along the temporal dimension to obtain the output [25,5]. Recently, 3D CNNs have shown great potential in dealing with the spatio-temporal data or 3D CAD models as inputs [16,27,9]. It has been observed that 3D CNNs are much better in exploiting the temporal information than 2D CNNs[18]. However, 3D CNNs are computationally expensive and they are prone to overfit due to their large number of parameters. Hence, the researchers moved on to find better and more efficient ways of mimicking 3D convolutions. There has been notable advances in the separable convolutions in 2D CNNs to reduce the space-time complexity [2,22]. In many works, the idea of separable convolutions has been extended to 3D CNNs [15,23,11,18]. In [11], the authors proposed the idea of replacing the 3D convolution kernel by a 2D convolution kernel to capture the spatial information followed by a 1D convolution kernel to convolve along the temporal direction. They showed that the proposed technique has several advantages, like parameter reduction and better performance, over the 3D convolutions, which has been further explored in [18]. Temporal differences has been explored in few recent works [19,8]. Wang *et al.* [19] use difference in two frames as the approximation of motion information. Similarly, Lee *et al.* [8] propose a motion block which extracts features using spatial and temporal shifts. In this work, we only rely on the temporal differences. Instead of relying on only the adjacent frames, we compute aggregated temporal differences over several frames. The proposed SSA Layer does not involve any trainable parameter to extract temporal information

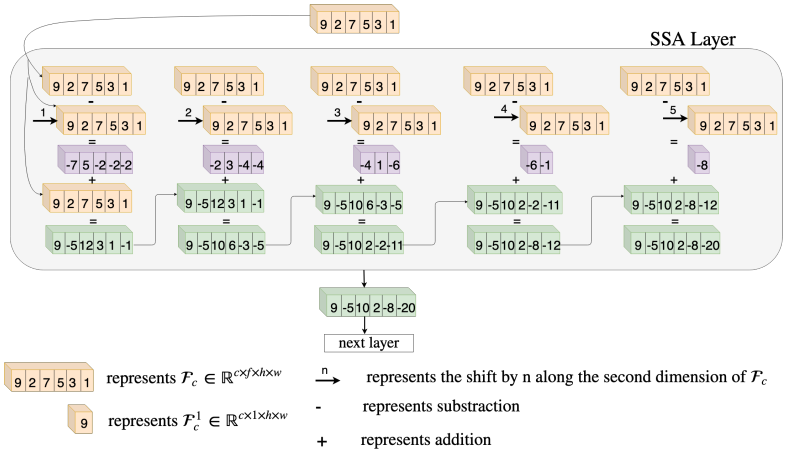


Fig. 1. An illustration of SSA Layer

via temporal differences. Our focus is to propose an efficient alternative to the 3D convolution filters which utilizes lesser parameters without compromising the performance.

3 Proposed Approach

In this section, we discuss the proposed convolutional block which extracts both spatial and temporal information. The proposed convolutional block has three parts: 2D convolution kernel, SSA layer, and temporal pooling layer. Here, SSA stands for Shift, Subtract and Add. Let the input to the proposed convolutional block be $\mathcal{X} \in \mathbb{R}^{c \times f \times h \times w}$. Here, $\mathcal{X}$ is the output feature maps of the previous convolutional block or layer, c is number of channels, f corresponds to the temporal depth, and h and w are the height and width of $\mathcal{X}$, respectively.

2D convolution. In traditional 3D CNNs, the feature maps are convolved with a 3D filter $\hat{g} \in \mathbb{R}^{c \times k \times k \times k}$ with c channels and kernel size $k \times k \times k$ [6]. In the proposed framework, first we obtain $\mathcal{X}_c = \mathcal{X} \star g$. Here, $\star$ stands for convolution, and g is a 2D filter of kernel size $1 \times k \times k$ and c channels. The purpose of the 2D convolution is to extract the spatial information present in the input feature maps[26]. We, then, pass $\mathcal{X}_c$ through the proposed SSA layer to obtain the temporal structure of the feature maps.

SSA Layer. SSA stands for Shift, Subtract and Add operations performed in the SSA layer. The purpose of the SSA layer is to extract the temporal information present in the spatio-temporal data. For example, in action recognition, motion features extracted from the videos can hold important information. In order to capture the motion information, optical flow techniques can be used [4]. However, capturing optical flow is in itself a computationally expensive task which can require a dedicated network [4]. In the proposed SSA layer, we rely on

temporal differences, i.e., the change in the spatial information at different time instances, to extract the necessary temporal information present in the spatio-temporal data.In the case of action recognition, temporal differences can provide the rough extimate of the location of moving objects or non-rigid bodies [10]. However, there is a possibility that there has not been enough change occurred in the adjacent frames. Hence, we take multiple frames into the consideration. The difference could be due to motion like in the case of action recognition or due to the structure of the input, like in the case of shapes. This makes the SSA layer to be used in a more general sense.

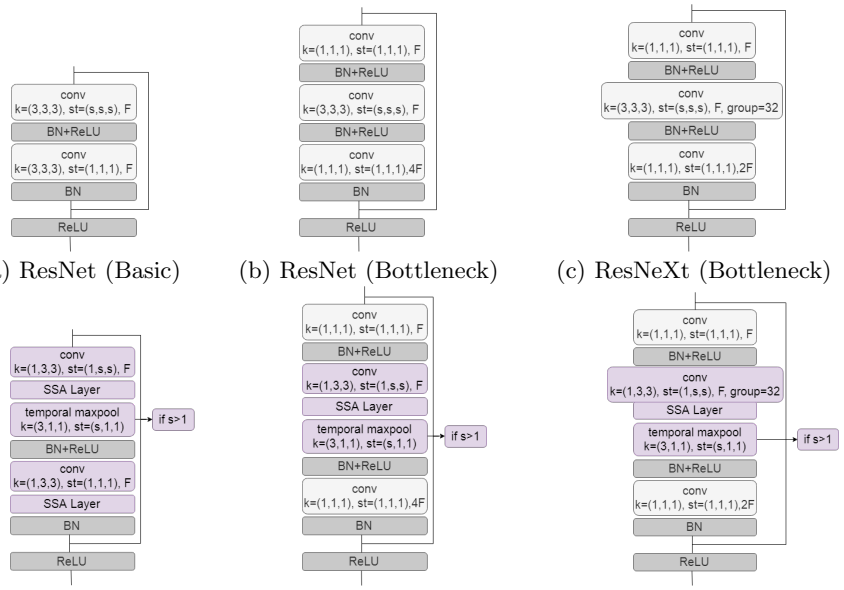
Let the input to the SSA layer be $\mathcal{X}_c \in \mathbb{R}^{c \times f \times h \times w}$. Here, c is the number of channels, f is the temporal depth, and h and w are the height and width of $\mathcal{X}_c$, respectively. We obtain the temporal differences between the volumes of the feature map $\mathcal{X}_c$ along the temporal depth. Let $\{\mathcal{X}_c^1, \mathcal{X}_c^2, \mathcal{X}_c^3, \ldots, \mathcal{X}_c^f\} \in \mathbb{R}^{c \times 1 \times h \times w}$ be the volumes of the feature map $\mathcal{X}_c$ along the temporal depth. $\mathcal{X}_c$ is passed through the SSA layer to obtain $\mathcal{X}_s \in \mathbb{R}^{c \times f \times h \times w}$ as shown in Eq. 1. Here, $\mathcal{X}_s$ is obtained by concatenating $\{\mathcal{X}_s^1, \mathcal{X}_s^2, \mathcal{X}_s^3, \ldots, \mathcal{X}_s^f\} \in \mathbb{R}^{c \times 1 \times h \times w}$ along the temporal dimension.

$$\mathcal{X}_s^i = \mathcal{X}_c^i + \frac{1}{f} \sum_{k=1}^{i-1} \frac{f-(i-k)}{f} (\mathcal{X}_c^i - \mathcal{X}_c^k), \quad \forall i = 2, \ldots, f \tag{1}$$

Here, k is the shift and i is a location along the temporal direction. If $i = 1$, $\mathcal{X}_s^i = \mathcal{X}_c^i$. Since, the nearby frames can have more contextual relation, the term $\frac{f-(i-k)}{f}$ is to ensure that the larger shifts get smaller weights than smaller shifts. Instead of computing for each temporal volume separately, X_s can be computed in a cumulative manner as illustrated in Fig. 1. However, the mathematical formulation and the illustration shown in Fig. 1 lead to the same output. Since, $\mathcal{X}_c$ is a four dimensional volume, it would be hard to provide a clean illustration. We have also omitted the multiplicative constants from the illustration to keep it clean. Hence, we have used 1-D representation to illustrate its operations visually. In Fig. 1, each column refers to a single shift. It can be seen that the input feature map $\mathcal{X}_c$ is subtracted from its shifted version and then, the difference is added to it in the corresponding locations to obtain $\hat{\mathcal{X}}_s$. Then, we again shift the feature map $\mathcal{X}_c$ by one more step, subtract it from its original version and then add the difference to the corresponding locations of $\hat{\mathcal{X}}_s$. At the end of $f-1$ steps, we obtain $\mathcal{X}_s$.

Temporal pooling. As, we move along the depth of the 3D convolution networks, the temporal depth of the feature maps keeps reducing as we perform 3D convolutions of stride more than one. In our case, we are not performing convolution along the temporal depth. Hence, to reduce the temporal depth, we perform max pooling along the temporal direction whenever we want to reduce the temporal depth of the feature maps.

Parameter Analysis. A standard 3D convolutional kernel of size $n \times n \times n$ and c channels contains cn^3 parameters. The proposed convolution block comprises a standard 2D-convolution kernel, an SSA layer and temporal max pooling. A standard 2D-convolution kernel of size $n \times n$ and c channels contains cn^2 pa-



(a) ResNet (Basic) (b) ResNet (Bottleneck) (c) ResNeXt (Bottleneck)

(d) SSA-ResNet (Basic) (e) SSA-ResNet (Bottleneck) (f) SSA-ResNeXt (Bottleneck)

Fig. 2. (a) and (b) show the basic and bottleneck blocks used in 3D ResNet architecture [6]. (c) shows the bottleneck bock used in 3D ResNeXT architecture [6]. (d), (e) and (f) show the residual blocks in which 3D convolution kernel is replaced by the proposed convolutional block.

rameters and an SSA layer consists of shift, subtract and add operations which do not require any trainable parameters. Also, temporal max pooling does not require any trainable parameters. Hence, the overall number of trainable parameters used in the proposed convolution block is cn^2 which is n times less than the standard 3D convolution kernel.

4 Experiments and Discussions

In this section, we show that the 3D CNNs perform better when the standard 3D convolution kernels are replaced by the proposed convolutional block. Our focus is mostly on the residual networks. We evaluate their performances on two types of 3D data: spatio-temporal image sequences and 3D CAD models.

4.1 Spatio-Temporal Image Sequences

Dataset. UCF101 [14] is a benchmark action recognition dataset containing complex real world videos which has been used in several works[16,17,3]. The videos of the dataset cover 101 action categories. We use UCF101 split-1 for all our experiments regarding spatio-temporal image sequences.

Network	Layers	Parameters (Millions)	SSA Layer	Temporal pooling	Accuracy(%)
3D ResNet[6,17] (baseline)	18	≈ 33			45.6
SSA-ResNet (ours)	18	≈ 11		✓	52.8
SSA-ResNet (ours)	18	≈ 11	✓	✓	**55.7**
3D ResNeXT[6] (baseline)	50	≈ 26			49.3
SSA-ResNeXT (ours)	50	≈ 23		✓	54.9
SSA-ResNeXT (ours)	50	≈ 23	✓	✓	**56.9**
3D WideResNet[6] (baseline)	50	≈ 157			46.8
SSA-WideResNet(ours)	50	≈ 67		✓	50.7
SSA-WideResNet(ours)	50	≈ 67	✓	✓	**52.9**
C3D[16] (baseline)	5	≈ 18			44
SSA-C3D (ours)	5	≈ 14		✓	50
SSA-C3D (ours)	5	≈ 14	✓	✓	**51.6**
3D ResNet[3,6] (baseline)	101	≈ 88			46.7
SSA-ResNet (ours)	101	≈ 43		✓	52.1
SSA-ResNet (ours)	101	≈ 43	✓	✓	**54.4**

Table 1. Comparisons with baselines. The comparison of the test accuracies obtained by the baseline 3D models with the networks obtained by replacing the 3D convolution kernel by the proposed convolution block in the baseline 3D models on UCF101 split-1 when trained from scratch.

Network Architectures We employ deep 3D residual networks to evaluate the proposed convolutional block [6]. Fig. 2 (a) and (b) show the basic and bottleneck block used in 3D ResNet architecture [6]. Fig. 2 (c) shows the bottleneck block used in ResNeXT architecture [6]. We replace the 3D convolution kernel of the residual blocks by the proposed convolutional block as shown in Fig. 2 (d), (e) and (f). In Fig. 2 (d), (e) and (f), it can be seen that we preserve the overall structure of the blocks while replacing the 3D convolution kernel by the proposed convolutional block. This is done to show the true effect of the proposed block on the existing networks. We have also experimented with the WideResNet architecture with a widening factor of 2 [6]. The structure of bottleneck block of WideResNet is same as the bottleneck block of ResNet. The only difference is the number of channels of the feature maps in the layers. To show that the proposed approach is not constrained to the residual networks, we have also done experiments with C3D network proposed in [16]. Similar to the residual networks, we replace the 3D convolution kernel with the proposed convolutional block. The training details are provided in the supplementary material.

Comparisons with baselines We perform our experiments by training the networks from scratch on UCF101 split-1. The test accuracies of 3D ResNeXT and 3D WideResNet when trained from scratch on UCF101 split-1 are not available in the previous works[6]. So, we train these networks on UCF101 from scratch to obtain them. For the other baseline networks, we mention the accuracies reported in [6,3,17]. SSA-ResNet, SSA-WideResNet, SSA-ResNeXT, and SSA-C3D are obtained by replacing the 3D convolution kernels in ResNet, WideResNet, ResNeXT, and C3D[16] by the proposed convolutional block. We train these networks from scratch on UCF101 with same hyperparameter settings. Table 1 shows that the accuracies obtained by the baseline 3D models when trained from scratch on the split-1 of UCF101 dataset. It also shows the

Network	Layers	Parameters (Millions)	Model Size (MB)	Accuracy
2D-ResNet[7,17]	18	≈11.2	-	42.2
2D-ResNet[7,17]	34	≈21.5	-	42.2
3D-ResNet[17]	18	≈33.2	254	45.6
3D-ResNet[17]	34	≈63.5	485	45.9
3D-ResNet[3]	101	≈86.06	657	46.7
3D STC-ResNet[3]	18	-	-	42.8
3D STC-ResNet[3]	50	-	-	46.2
3D STC-ResNet[3]	101	-	-	47.9
C3D[16]	5	≈18	139.6	44
R(2+1)D[18]	18	≈33.3	128	48.37
SSA-ResNet (ours)	18	≈11	88.5	55.7
SSA-ResNeXt (ours)	50	≈23	185.9	**56.9**

Table 2. Comparisons with the state-of-the-art. The comparison of the proposed approach with the state-of-the-art methods when trained from scratch on UCF101 dataset.

#Shift	Temporal pooling	Accuracy
0		46.3
0	✓	52.8
1	✓	52.6
2	✓	53.4
3	✓	53.9
f-1		51.3
f-1	✓	**55.7**

Table 3. Analysis of different shifts and temporal pooling. The comparison of test accuracies obtained on UCF101 split-1 using SSA-ResNet (18 layers) (when trained-from-scratch) with varying number of shifts along with the effect of temporal pooling.

accuracies obtained by replacing the 3D convolution kernel by the proposed convolution block. It can be seen that the networks perform significantly better with the proposed convolutional block while utilizing lesser trainable parameters.

Comparisons with the state-of-the-art Table 2 compares the proposed approach with the state-of-the-art methods when trained from scratch on UCF101 dataset. The test accuracy of R(2+1)D[18] when trained from scratch on UCF101 is not available in the previous works. So, we trained the network on UCF101 split-1 from scratch to obtain it using the same hyperparameter settings as ours. It can be observed that SSA-ResNeXT performs significantly better than the previous approaches. SSA-ResNet (18 layers) utilizes approximately 11 million parameters which is roughly equal to the parameters used in 2D-ResNet [7] (18 layers). Inspite of having almost equal parameters, SSA-ResNet (18 layers) outperforms 2D-ResNet (18 layers) by 13.5 % in terms of classification accuracy. Also, SSA-ResNet (18 layers) utilizes approximately 3 times less parameters than 3D-ResNet (18 layers)[17], 3D STC-ResNet (18 layers)[3], and R(2+1)D (18 layers)[18] and still outperforms them by 10.1%, 12.9 %, and 7.33%, respectively.

Analysis In the proposed convolutional block, apart from a standard 2D convolution kernel, there are two components: SSA layer and Temporal pooling.
SSA Layer. As shown in Fig.1, we perform the shift operation $f - 1$ times, where f is the temporal depth of the input feature map. We perform the exper-

Network	Framework	Augmentation	Parameters (Millions)	ModelNet40 (%)	ModelNet10 (%)
3D ShapeNets[21]	Volumetric	Az × 12	≈38	77	83.5
Beam Search[24]	Volumetric	Az × 12	≈0.08	81.26	88
3D-GAN[20]	Volumetric	Az × 12	≈11	83.3	91
VoxNet[9]	Volumetric	Az × 12	≈0.92	83	92
LightNet[27]	Volumetric	Az × 12	≈0.30	86.90	93.39
ORION[13]	Volumetric	Az × 12	≈.91	-	**93.8**
SSA-ResNeXT8 (ours)	Volumetric	Az × 12	≈3.38	**89.5**	93.3

Table 4. Comparisons with the state-of-the-art. The comparison of the SSA-ResNeXT8 with the state-of-the-art methods on the voxelized version of ModelNet40 and ModelNet10 datasets.

iments on SSA-ResNet (18 layers) with different values of shifts. The results are shown in Table 3. It can be seen that as we increase the fixed number of shifts from 1 to 3, the test accuracy increases and we obtain the highest accuracy when we perform $f - 1$ shifts.

Temporal Pooling. In Table 3, it can be observed that by using 2D-convolution kernel and only max temporal pooling, the network outperforms the baseline case, i.e. with only 2D convolution kernels. The same pattern can be observed in Table 1, in which the baseline 3D models are replaced with the proposed convolution block without SSA layer (second row for each network) and the networks performed significantly better than the baseline 3D CNNs.

4.2 3D CAD Models

Dataset. ModelNet[21] is a collection of 3D CAD models of objects. It has two subsets: ModelNet10 and ModelNet40. ModelNet10 and ModelNet40 contains 10 and 40 classes of objects, respectively, which are manually aligned to a canonical frame. In our experiments, we use the voxelized version of size $32 \times 32 \times 32$ and augmentation with 12 orientations [9]. Similar to [9,1], we add noise, random translations, and horizontal flips for data augmentation to the training data. Similar to [1], we scale the binary voxel range from $\{0, 1\}$ to $\{-1, 5\}$.

Network Architecture. To avoid overfitting on ModelNet40 and ModelNet10, we use a smal network SSA-ResNext8 to evaluate our approach on 3D CAD models. We use the SSA-ResNeXT bottleneck block in the architecture of the network. Let us denote the SSA-ResNeXT bottleneck block with $SSAR(k, F, s)$, where $1 \times k \times k$ is the kernel size of the 2D convolution filter, F is the number of channels in the input feature map and s is the value of stride passed to the block. The architecture of SSA-ResNext8 is as follows: $Conv2D(3, 1) \rightarrow MP(3, 2) \rightarrow SSAR(3, 64, 1) \rightarrow SSAR(3, 256, 1) \rightarrow SSAR(3, 256, 2) \rightarrow SSAR(3, 512, 1) \rightarrow SSAR(3, 512, 2) \rightarrow SSAR(3, 1024, 1) \rightarrow GP \rightarrow FC$. Here, $Conv2D(3, 1)$ is a 2D convolution kernel of size $1 \times 3 \times 3$ and stride of 1, followed by a batch normalization layer and ReLU, and $MP(3, 2)$ is the max-pooling layer with kernel size of $3 \times 3 \times 3$ and stride of 2. GP and FC stands for global average pooling and fully connected layer, respectively. The training details are provided in the supplementary material.

Comparisons with the state of the art. Table 4 shows the comparison of the

SSA-ResNeXT8 with the state-of-the-art methods that use voxelized/volumetric ModelNet datasets as input. For fair comparison, we only consider volumetric frameworks. It can be observed that the network with the proposed convolutional block performs better than the state-of-the-art on ModelNet40 and comparable on ModelNet10 in the case when the networks are trained with shapes augmented with 12 orientations. This shows that the proposed convolution block is not restricted to videos and can be further exploited in shapes.

5 Conclusion

We propose a novel convolutional block which is proficient in capturing both spatial and temporal structure of the 3D data while utilizing lesser parameters than the 3D convolution kernel. It comprises three components: a 2D-convolution kernel to capture the spatial information, a novel SSA layer to capture the temporal structure, and a temporal pooling layer to reduce the temporal depth of the input feature map. We show that the 3D CNNs perform better when the 3D convolution kernels are replaced by the proposed convolutional block. SSA-ResNet (18 layers) outperforms the state-of-the-art accuracy on the UCF101 dataset split-1 while utilizing lesser parameters when networks are trained-from-scratch. We have also evaluated the proposed convolutional block on 3D CAD models and we outperform the state-of-the-art on ModelNet40 among the volumetric framework, when the training data is augmented with 12 rotations.

Acknowledgments. Gagan Kanojia and Sudhakar Kumawat were supported by TCS Research Fellowships. Shanmuganathan Raman was supported by SERB Core Research Grant and Imprint 2 Grant.

References

1. Brock, A., Lim, T., Ritchie, J.M., Weston, N.: Generative and discriminative voxel modeling with convolutional neural networks. arXiv preprint arXiv:1608.04236 (2016)
2. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1251–1258 (2017)
3. Diba, A., Fayyaz, M., Sharma, V., Mahdi Arzani, M., Yousefzadeh, R., Gall, J., Van Gool, L.: Spatio-temporal channel correlation networks for action classification. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 284–299 (2018)
4. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2758–2766 (2015)
5. Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., Russell, B.: Actionvlad: Learning spatio-temporal aggregation for action classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 971–980 (2017)

6. 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. pp. 6546–6555 (2018)
7. 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)
8. Lee, M., Lee, S., Son, S., Park, G., Kwak, N.: Motion feature network: Fixed motion filter for action recognition. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 387–403 (2018)
9. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 922–928. IEEE (2015)
10. Park, D., Zitnick, C.L., Ramanan, D., Dollár, P.: Exploring weak stabilization for motion feature extraction. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2882–2889 (2013)
11. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: proceedings of the IEEE International Conference on Computer Vision. pp. 5533–5541 (2017)
12. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4510–4520 (2018)
13. Sedaghat, N., Zolfaghari, M., Amiri, E., Brox, T.: Orientation-boosted voxel nets for 3d object recognition. In: British Machine Vision Conference (BMVC) (2017)
14. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
15. Sun, L., Jia, K., Yeung, D.Y., Shi, B.E.: Human action recognition using factorized spatio-temporal convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4597–4605 (2015)
16. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 4489–4497 (2015)
17. Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M.: Convnet architecture search for spatiotemporal feature learning. arXiv preprint arXiv:1708.05038 (2017)
18. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 6450–6459 (2018)
19. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: European conference on computer vision. pp. 20–36. Springer (2016)
20. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: Advances in neural information processing systems. pp. 82–90 (2016)
21. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1912–1920 (2015)
22. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1492–1500 (2017)
23. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 305–321 (2018)

24. Xu, X., Todorovic, S.: Beam search for learning a deep convolutional neural network of 3d shapes. In: 2016 23rd International Conference on Pattern Recognition (ICPR). pp. 3506–3511. IEEE (2016)
25. Xu, Z., Yang, Y., Hauptmann, A.G.: A discriminative cnn video representation for event detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1798–1807 (2015)
26. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision. pp. 818–833. Springer (2014)
27. Zhi, S., Liu, Y., Li, X., Guo, Y.: Lightnet: a lightweight 3d convolutional neural network for real-time 3d object recognition. In: Proceedings of the Workshop on 3D Object Retrieval. pp. 9–16. Eurographics Association (2017)