In Defense of Image Pre-Training for Spatiotemporal Recognition

Xianhang Li[1], Huiyu Wang[2], Chen Wei[2], Jieru Mei[2], Alan Yuille[2],
Yuyin Zhou[1], and Cihang Xie[1]

[1]University of California, Santa Cruz   [2]Johns Hopkins University

Abstract. Image pre-training, the current de-facto paradigm for a wide range of visual tasks, is generally less favored in the field of video recognition. By contrast, a common strategy is to directly train with spatiotemporal convolutional neural networks (CNNs) from scratch. Nonetheless, interestingly, by taking a closer look at these from-scratch learned CNNs, we note there exist certain 3D kernels that exhibit much stronger appearance modeling ability than others, arguably suggesting appearance information is already well disentangled in learning. Inspired by this observation, we hypothesize that the key to effectively leveraging image pre-training lies in the decomposition of learning spatial and temporal features, and revisiting image pre-training as the appearance prior to initializing 3D kernels. In addition, we propose Spatial-Temporal Separable (STS) convolution, which explicitly splits the feature channels into spatial and temporal groups, to further enable a more thorough decomposition of spatiotemporal features for fine-tuning 3D CNNs.

Our experiments show that simply replacing 3D convolution with STS notably improves a wide range of 3D CNNs without increasing parameters and computation on both Kinetics-400 and Something-Something V2. Moreover, this new training pipeline consistently achieves better results on video recognition with significant speedup. For instance, we achieve +0.6% top-1 of Slowfast on Kinetics-400 over the strong 256-epoch 128-GPU baseline while fine-tuning for only 50 epochs with 4 GPUs. The code and models are available at `github.com/UCSC-VLAA/Image-Pretraining-for-Video`.

Keywords: Video Classification, ImageNet Pre-Training, 3D Convolution Networks

1 Introduction

Deep convolutional neural networks (CNNs) pre-trained on large-scale datasets (*e.g.*, ImageNet [8]) play a vital role in computer vision. The spatial feature representations acquired by such models can then be transferred to the downstream task of interest via fine-tuning, leading to significant performance improvements especially for small target datasets. This "*pre-training then fine-tuning*" paradigm has gradually become the de-facto standard and established
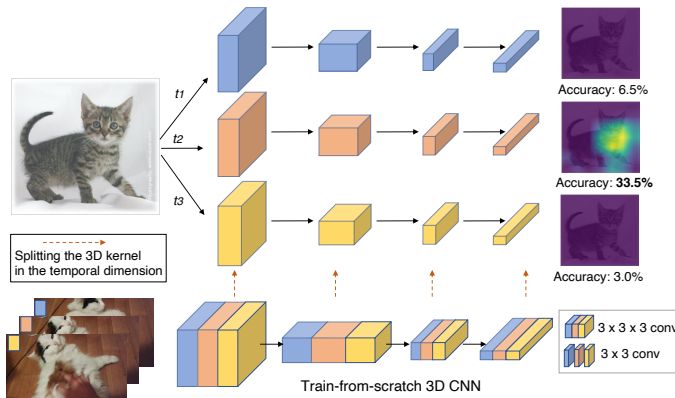
Fig. 1: **Motivation**. By splitting the 3D convolution into several 2D ones along the temporal dimension, we can easily transform a 3D CNN into 2D CNNs. Note the resting 2D parts in the 3D CNN remain the same. Interestingly, we note the 2D CNN along the center temporal direction attains much stronger linear-probing ImageNet accuracy than others, *i.e.*, 33.5% *vs.* 6.5% or 3.0%.

state-of-the-arts for a wide range of vision tasks, such as semantic segmentation [27,16] and object detection [13,29].

The "*pre-training then fine-tuning*" paradigm is readily used for image recognition, as the two stages usually share the same backbone. However, for video recognition, to model the temporal dynamics and build spatiotemporal features, recent research efforts are dedicated to building video-specific architectures (*e.g.*, Slowfast [12], X3D [11]). However, these structural changes prevent the model from getting image pre-training weights for *free*. One popular method to leverage image pre-training is to inflate 2D kernels pre-trained on ImageNet to 3D ones [4]. However, recent studies such as SlowFast and Multigrid [12,46] suggest that using ImageNet pre-training does not outperform (sometimes even produces worse results than) training from scratch. Hence, the field of video representation learning has witnessed a paradigm shift from pre-training then fine-tuning to training from scratch for today's advanced spatiotemporal 3D CNNs.

As directly training these 3D CNNs from scratch requires enormous computational resources [39,12,38,46,40,11], many recent research efforts are devoted to designing efficient operations [24,23,43] and architectures [39,40] for 3D video recognition. For instance, R2plus1D [39] is proposed to separate a 3D convolution into a 1D temporal convolution and a 2D spatial convolution. Tran et al. propose CSN [38] which operates 3D convolution in the depth-wise manner. This structural decomposition assumes that we should not treat space and time symmetrically. However, due to the end-to-end characteristic of training from scratch, the spatiotemporal features are still jointly learned in these 3D CNNs in an unconstrained manner. Thus, it remains unclear how spatial and temporal information is exploited in the network.

To better understand what these from-scratch trained 3D kernels learn, as shown in Figure 1, we split the 3D convolution trained on Kinetics-400 dataset [4]

using the CSN-50 network [38] into several 2D counterparts along the temporal dimension, and quantify the appearance modeling power by assessing the linear probing performance of these 2D CNNs on ImageNet-1K. Surprisingly, despite the domain gap between Kinetics-400 and ImageNet-1K, the 2D CNN along the center temporal direction (indicated by the orange branch in Figure 1) still achieves a non-trivial linear probing performance, indicating its strong appearance modeling ability. Moreover, using the gradients returned from logits in a category [30], we plot the corresponding heat map that highlights the important regions in the image used for prediction, which are also highly correlated with the appearance information (yellow and blue regions).

This interesting phenomenon further inspires us to rethink the value of image pre-training for 3D spatiotemporal recognition—if 3D kernels have such strong appearance modeling power, then the spatial features pre-trained from 2D images should be beneficial for training 3D CNNs. As the appearance information is already well disentangled in 3D CNNs, we hypothesize that the key to harnessing image pre-training lies in properly decomposing the spatiotemporal features learned from 3D kernels into spatial and temporal parts. Based on this principle, we design a new pre-training and fine-tuning pipeline to facilitate feature decomposition. Specifically, image pre-training is leveraged as the appearance prior for only initiating the 2D counterpart along the center temporal direction of the 3D convolution where the strongest appearance modeling power lies. Then in the fine-tuning stage, we propose 3D Spatial-Temporal Separable (STS) convolution which explicitly splits the feature channels into spatial and temporal groups. The former group learns the static appearance while the latter focuses on learning dynamic motion features, thus enabling a more thorough decomposition of spatiotemporal features for fine-tuning 3D CNNs.

Compared with prior solutions which train 3D CNNs from scratch, our method not only enjoys a significant speedup, but also improves 3D video representation learning by carefully tackling the appearance prior from image pre-training models. We evaluate our method on a wide range of advanced 3D CNNs, including SlowFast [12], CSN [38], R2plus1D [39], and X3D [11]. Without using more computation or parameters, our method brings notable improvement on two popular video benchmarks Kinetics-400 [4] (**+2.1%**) and Something-Something V2 [14] (**+5.3%**). On Kinetics-400, our proposed training pipeline is at least **2×** faster than training from scratch. The empirical analysis and design principles provided in our work will help researchers rethink the value of image pre-training and better understand its role in video recognition.

2 Related Works

Pre-training and fine-tuning for video classification. Pre-training on a large-scale dataset and then fine-tuning on a minor dataset for downstream tasks has been widely used to resist overfitting and get state-of-the-art performance [31,4,13,29,27,16]. This paradigm has also been employed and continuously discussed in video classification [4,6]. Due to the nature of temporal variance in

the video, two types of methods can be used to accelerate the learning of video classification algorithms: image-based pre-training and video-based pre-training.

Image pre-training. Video classification models tend to consider both temporal and spatial information, and image pre-training can contribute to preserving spatial properties. There is active research about designing a well-performed video architecture based on a 2D ImageNet pre-trained backbone [17,32,35]. TSN [42] is proposed to sample video frames sparsely that boost 2D image model performance on video classification. Revising a 2D block [26,23,24,18,20,33,22] or inserting attention-style block [43,44,52,5,3,49] into a 2D model can enhance the ability of spatiotemporal representation. In order to make the 3D model converge faster, I3D [4] is proposed to initialize 3D convolution by inflating the 2D kernels. The recent success of the vision transformer [9,36] has also made significant progress in video classification [2,1,25,10,21], and using larger image datasets such as ImageNet-21K [8], JFT-300M [34] significantly improves the model performance. Compared to these approaches, we find that the improvement from such pre-training strategies are still not well explored in 3D CNNs. Secondly, instead of using larger datasets, we focus on an more efficient and fair setting.

Efficient 3D Video Recognition. Expanding a 2D image model is widely adopted in video recognition. First, extending the kernel into 3D increases the spatial and temporal receptive field and brings more parameters and computation [37,28,47,19]. Second, extending the temporal dimension into different resolutions further improves the performance. Recent work X3D [11] considers all these expansions in a video-oriented perspective. Similarly, the decomposition can perform either from the model or the data perspective. R2plus1D [39] is proposed to separate the spatiotemporal convolution. There are other approaches which also separate the 3D convolution along the channel dimension for efficiency [38,20]. From the data perspective, a two-stream architecture [31,6,12,48] is a representative method that divides the input video into appearance and motion, *e.g.*, optical flow [31] and temporal difference [18,41]. In this work, we focus on separating the learning process into efficient image pre-training and fine-tuning. Furthermore, we proposed that the Spatiaotemporal Separable 3D convolution further improves the performance.

3   Methodology

In this section, we first reveal that in spatiotemporal recognition, 3D kernels still exhibit strong appearance modeling ability (Section 3.1). This observation further inspires us to design a novel pre-training & fine-tuning paradigm for video recognition, where 2D image pre-training serves the appearance prior, guiding the further spatiotemporal fine-tuning process (Section 3.2). Finally in Section 3.3, we provide two general training settings to conduct a fair comparison with training from scratch.

	t1-CNN	t2-CNN	t3-CNN
CSN-50 [38]	3.0	**33.5**	6.3
X3D-S [11]	5.7	**41.2**	5.5
R2plus1D-34 [39]	0.8	**32.3**	2.2
Slowonly-50 [12]	18.3	**45.1**	17.6

Table 1: ImageNet-1K **Linear Probing** Top1 Accuracy. The weights come from the models trained from scratch on Kinetics-400. We fine tune the linear classification layer for 100 epochs on ImageNet-1K.

3.1 Appearance Modeling in 3D Kernels

Despite numerous variants of 3D CNNs being proposed, it is unclear what these 3D kernels actually learn. In this paper, to better understand the properties of 3D kernels, we aim to examine the relationship between the 3D kernels and 2D appearance modeling by answering the following question: *Do 3D features equally model the 2D appearance across the temporal dimension?*

To answer this question, we propose to split the standard 3D convolution into several 2D ones along the temporal dimension, and quantify the appearance modeling of the 3D CNN by assessing the linear probing performance of these 2D CNNs instead. Specifically, we split a 3D convolution ($3 \times 3 \times 3$ or $3 \times 1 \times 1$) into three 2D convolutions (3×3 or 1×1) at different time stamps t1, t2, t3, as shown in Figure 1. We then evaluate these 2D CNNs on ImageNet-1k by freezing the weights and only fine-tuning the linear classification layer. As shown in Table 1, we examine four widely-used video classification models: 1) CSN-50 [38], 2) X3D-S [11], 3) R2plus1D-34 [39], 4) Slowonly-50 [12], all of which are trained from scratch on Kinetics-400. Surprisingly, though there is a domain gap between Kinetics-400 and ImageNet-1k, out of the three 2D CNNs, the middle one at t2 consistently achieves a non-trivial linear probing performance, whereas in the other two 2D networks at t1 and t3 time stamps, we observe severe performance degradation. This demonstrates that while spatiotemporal features are not equivalent in terms of appearance modeling, they do exhibit strong appearance modeling ability.

This phenomenon further motivates us to rethink the value of image pre-training in spatiotemporal recognition. Specifically, *can we achieve a better trade-off between performance and training efficiency*? Next, we will elaborate on how image pre-training can be leveraged to improve spatiotemporal recognition.

3.2 Image Pre-Training and Spatiotemporal Fine-tuning

The value of ImageNet pre-training is firstly challenged in He et al. [15]. Later, many recent studies have further demonstrated that compared with training from scratch, ImageNet pre-training even results in performance drop for video action recognition models [11,38,39,12]. However, as shown in Section 3.1, by closely examining these from-scratch learned 3D kernels, we find they still exhibit strong appearance modeling power. Based on this interesting observation, we hypothesize that image pre-training is indeed valuable for training video recognition
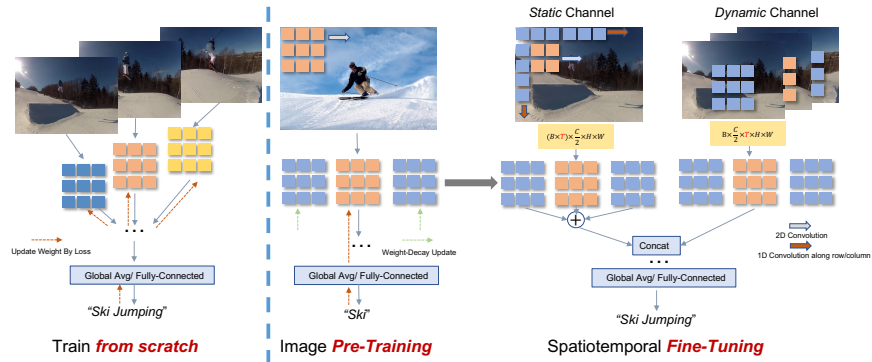
Fig. 2: **Overview of Image Pre-Training & Spatiotemporal Fine-Tuning.** Left: training from scratch learns features in a joint manner. Right: decompose the learning into image pre-training and spatiotemporal fine-tuning.

models, when properly decomposing the learning of spatiotemporal features. To this end, we propose a novel pre-training and fine-tuning paradigm. As shown in Figure 2, compared with the popular train-from-scratch pipeline which learns spatial and temporal features in a joint fashion, our proposed method thoroughly decomposes the spatial and temporal features via the following two steps: 1) image pre-training for appearance modeling, and 2) spatiotemporal fine-tuning.

Image pre-training for appearance modeling. Based on our observation in Section 3.1 that 3D kernels usually exhibit strong appearance ability along the center temporal direction, we propose to leverage image pre-training as the appearance prior by only initiating the middle kernel of the 3D convolution. As other parts of the 3D convolution do not have a strong relationship with appearance modeling, the rest of the weights will simply be updated by weight-decay regularization, as shown in Figure 2. Leveraging the appearance prior from image pre-training plays a key role in facilitating the following fine-tuning stage since 1) it provides better initialization for the 3D kernels, and 2) it helps significantly reduce the fine-tuning schedule. Given that the appearance modeling is already well established, the 3D CNN only needs to focus on learning the temporal information.

Spatiotemporal fine-tuning. After acquiring the appearance prior from image pre-training, we then present a novel spatiotemporal fine-tuning method for a more thorough decomposition of spatiotemporal features. Specifically, as shown in Figure 2, we propose 3D Spatiotemporal Separable (STS) convolution, which splits the 3D convolution along the output channel dimension into two groups, one for the static appearance modeling, and the other for dynamic motion modeling. The motion group builds the spatiotemporal feature by a 3D convolution, while the appearance group focuses on the spatial feature. In the appearance group, we divide the original 3D convolution along the temporal dimension into one 2D convolution (*e.g.*, 3×3) and two 1D convolutions (*e.g.*, 1×9 and 9×1), to enlarge the receptive field. The formal definition of STS can

	# Instance	# Images (frames)	Training Epochs	Total Training Images
ImageNet-1K	$1.28M$	$1.28M$	X	$X \times 1.28M$
Kinetics-400	$0.24M$	$68.89M$	Y	$Y \times T \times 0.24M$
Something-Something V2	$0.17M$	$18.98M$	Z	$Z \times T \times 0.17M$

Table 2: **Dataset Settings.** Each video clip is an instance. Total training images are total number of instances multiply the number of input frames T.
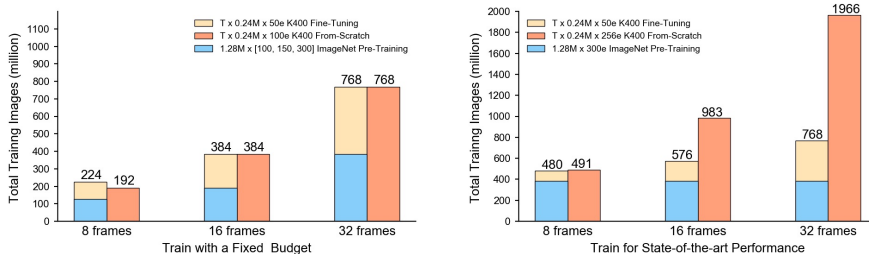

Fig. 3: **Computation Budget Towards Different Input Frames.** Total numbers of images/frames seen during all training epochs, for pre-training + fine-tuning (blue + yellow bars) *vs.* from scratch (orange bars). In Fixed-Budget setting (left), we pre-train for 100, 150 and 300 epochs to match total computation. In the train-for-state-of-the-art setting (right), the pre-train is 300 epoch. In both setting, we fine-tune for 50 epochs.

be found in the Appendix. The reason is that if we directly apply a 3D convolution on the spatial feature (*e.g.*, $3{\times}3{\times}3$), the receptive field is limited to the middle 2D spatial convolution ($1{\times}3{\times}3$). By contrast, in the proposed STS, the 1D convolutions can build the spatial feature along each row/column as shown in Figure 2, which enlarges the original receptive field in the 2D space ($3{\times}3$) to be ($3{\times}3{+}1{\times}9{+}9{\times}1$) without increasing parameters and computation. With a strong appearance prior, STS can significantly boost the performance during the fine-tuning stage benefiting from the more discriminative spatial features.

3.3 Training with a Fair Budget

Following the fair comparison setting [15] between pre-training & fine-tuning and training from scratch, we consider total training images as the condition for controlled experiments. We show the comparison on different datasets in Table 2. A typical training from scratch schedule on Kinetics-400 in recent advanced spatiotemporal 3D CNNs [12,11] involves $T \times 0.24M$ frames iterated for 256 epochs. For example, Slowfast[12] that take 32 frames as input for each training epoch will "see" over **7.68** million images, which is $\times$**6** more than one epoch ImageNet training. Compared with the training from scratch schedule, the pre-training & fine-tuning schedule (90 epochs pre-training + 100 epochs fine-tuning) consumes much less computation, which is usually unfair. As shown in Figure 3, we provide two general settings about how to fairly compare the performance of pre-training & fine-tuning and training from scratch.

Training with a fixed budget. Compared with training from scratch, we aim to maximize the performance of pre-training and fine-tuning with the same amount of computation. In this setting, we allocate around half of the total training budget to image pre-training, and another half to spatial-temporal fine-tuning, as shown in Figure 3. Surprisingly, we found that this simple strategy works well no matter the model consumes 8 frames, 16 frames, or 32 frames, according to our experiments.

Training for state-of-the-art. In this setting, we provide a more efficient training pipeline to match the previous state-of-the-art CNNs performance. Instead of scaling the image pre-training budget with respect to the total budget, we train our model with images for a long schedule (*e.g.*, 300 epochs) until convergence. As shown in Figure 3, we then fine-tune our model with a fixed short schedule (50 epochs) for all frame settings. We notice that the total computation cost is still lower than training on videos from scratch but the performance can be significantly improved over a wide range of advanced 3D CNNs on Kinetics-400 (**+2.1%**) and SS-V2 (**+4.1%**).

4   Experiments and Results

Datasets. We choose ImageNet-1K [8] as our primary image pre-training dataset. It contains around 1.28 million images for training and $50K$ images for validation. We also compare different pre-training datasets in the ablation study, *e.g.*, Kinetics-400. Unless otherwise specified, our default pre-training dataset is ImageNet-1K. Then we fine-tune our models on widely-used Kinetics-400 (abbreviated as K400) [4] and Something-Something V2 (abbreviated as SS-V2) [14]. Kinetics-400 contains about $260K$ videos of 400 different human action categories. We use the training split ($240K$ videos) for training and the validation split ($20K$ videos) for evaluating different models. SS-V2 contains $220K$ videos of 174 predefined human-object interactions with everyday objects.

Training. We pre-train all models for 100, 150, 300 epochs, respectively, based on different computation budgets. In the 100 and 150 epochs schedule, the model is trained using SGD with a learning rate of 0.4, a momentum of 0.9, a weight decay of 0.0001, and a batch size of 1024. The learning schedule is cosine annealing with a 5-epoch warm-up, and the initial learning rate is 1e-6. For augmentation, we enable the RandAugment [7], Mixup [51], and Cutmix [50]. We only use label smoothing as regularization. In the 300 epochs setting, we follow the training recipe in [45]. In the next fine-tuning stage, we use a dense-sampling strategy for Kinetics-400 [12,11] and an n-segments based method sampling strategy for SS-V2 [42,24]. During training, random resize cropping is utilized for data augmentation, and the cropped region is resized to 224×224 for each frame. For K400, the batch size, initial learning rate, weight decay, and dropout rate are set as 128 or 64 (based on input frames), 0.02 or 0.01, 1e-4, and 0.5 respectively. For SS-V2, we change the learning rate to 0.1 and the batch size is 64. Unless otherwise specified, the networks are fine-tuned for 50 epochs using SGD.

	+ Image Pre. & STS Conv	Pre-Train (hrs)	Fine-tune (hrs)	Total Time	Speedup	K400	Δ
Slowonly50-8*f* [12]	✗	-	389	389	×1	74.9	-
	✔	182	**76**	**258**	**×1.5**	**75.6**	**+0.7**
Slowfast50 (4x16) [12]	✗	-	696	696	×1	75.6	-
	✔	237	**136**	**373**	**×1.9**	**76.2**	**+0.6**
Slowfast50 (8x8) [12]	✗	-	840	840	×1	76.9	-
	✔	237	**164**	**401**	**×2.1**	**77.2**	**+0.3**
R2plus1D34-8*f* [39]	✗	-	778	778	×1	68.7	-
	✔	230	**152**	**382**	**×2.0**	**76.2**	**+7.5**
CSN50-32*f* [38]	✗	-	1106	1106	×1	73.6	-
	✔	176	**216**	**392**	**×2.8**	**76.7**	**+3.1**
X3D-S-13*f* [11]	✗	-	432	432	×1	73.2	-
	✔	121	**144**	**265**	**×1.6**	**73.5**	**+0.3**

Table 3: **Comparing Top-1 Accuracy of Common 3D CNNs on K400.** Training time is measured in GPU hours on the same hardware and software implementation for all methods. Baseline performance is obtained by testing the released model. We pre-train all models for 300 epochs on ImageNet.

Inference. For K400, following the common practice [43,12,11] we uniformly sample 10 clips from a video along its temporal axis. We scale the shorter spatial axis to 256 pixels for each clip and take 3 crops of 256×256 to cover the spatial dimensions. We average the softmax scores for prediction. For SS-V2, we only sample 1 clip based on the n-segments method and conduct the 3-crop testing following the common recipe in [24,20].

4.1   Comparison on Common 3D CNN Architectures

Main results on K400. We first demonstrate the evaluation of common 3D CNNs on K400 in Table 3. Firstly, the image pre-training (denoted by **Image Pre.**) and the proposed STS (denoted by **STS Conv**) both improve the video classification accuracy on a wide range of advanced video models, including Slowonly [12], Slowfast [12], R2plus1D [39], CSN [38] and X3D [11], by an average of **2.1 %**. Note that all models are trained with the same pre-training and fine-tuning pipeline (300 epochs pre-training + 50 epochs fine-tuning) described in Section 3.3, except for X3D-S. Due to its extremely tiny characteristic we extend the fine-tuning epochs for X3D-S to 100. These results demonstrate the strong generalization ability of our proposed pre-training and fine-tuning pipeline. Secondly, regarding the training efficiency, our method is ×**2** faster than training from scratch baseline on average. Note that to be as fair as possible, we take the pre-training time into account of the total training time. In addition, all methods compared in Table 3 use the same hardware and software implementation. More importantly, the training-from-scratch recipe usually includes a large batch size and long training schedule, which requires significant

	Parameters	GFLOPs	Image Pre.	STS Conv	SS-V2	Δ
Slowonly50-8f [12]	32.0M	54.9	✗	✗	58.4	-
			✔	✗	61.6	**+3.2**
			✔	✔	**62.7**	**+4.3**
Slowfast50 (4x16)-32f [12]	34.0M	36.4	✗	✗	49.9	-
			✔	✗	55.8	**+5.9**
			✔	✔	**57.2**	**+7.3**
R2plus1D34-8f [39]	63.6M	114.8	✗	✗	59.1	-
			✔	✗	62.3	**+3.2**
			✔	✔	**63.0**	**+3.9**
CSN50-8f [38]	12.7M	26.6	✗	✗	57.5	-
			✔	✗	60.4	**+2.9**
			✔	✔	**61.4**	**+4.0**
X3D-S-8f [11]	3.3M	3.2	✗	✗	51.9	-
			✔	✗	57.0	**+5.1**
			✔	✔	**58.3**	**+6.4**

Table 4: **Comparison on Common 3D CNNs on SS-V2.** We show the comparison on training from scratch, image pre-training and applying STS convolution respectively. We use test resolution (T×256×256) to measure GFLOPS.

computation resources. For example, X3D-S [11], Slowfast [12] are trained on 128-GPU and R2plus1D-34 [39], CSN-50 [38] are trained on 64-GPU cluster. In our experiments, with our pre-training and fine-tuning pipeline, we can fine-tune the model with a much smaller batch size and achieve better results. Thus, our method requires significantly fewer computational resources. All results are obtained on a single 4-GPU machine.

Main results on SS-V2. SS-V2 is relatively small, therefore training efficiency is not our goal on this dataset. Here we only use the fixed budget setting as described in Section 3.3 and fairly train all the methods in our machine within the same setting. We evaluate the temporal modeling ability of our proposed training pipeline and the STS convolution respectively over five common 3D video models in Table 4. First, SS-V2 is known as a temporal-related dataset. The appearance modeling is considered less important than temporal modeling [24,18,23]. It is interesting to see that the image pre-training and fine-tuning improve the training from scratch methods by a large margin of **4.1%**. The significant improvements demonstrate the superiority of our proposed training pipeline. Second, the STS convolution can replace any 3D convolution layer (*e.g.*, 3×3×3 or 3×1×1) without increasing parameters and computation. Upon image pre-training, using STS convolution can boost the performance by an average of **1.2%**, which clearly shows that the STS convolution is more suitable for temporal modeling. Equipped with both proposed methods, we in total improve the baseline methods by an average of **5.3%**, which indicates that the STS convolution benefits more from the proposed training pipeline.

layer name	output size	ResNet50-3×1×1	ResNet50-3×3×3*
conv1	T×112×112	1×7×7 or 3×7×7, 64, stride 1×2×2	
pool1	T×56×56	max, 1×3×3, stride 1×2×2	
stage 1	T×56×56	$\begin{bmatrix} 1×1×1,\ 256 \\ 1×3×3,\ 64 \\ 1×1×1,\ 256 \end{bmatrix} ×3$	$\begin{bmatrix} 1×1×1,\ 256 \\ 3×3×3,\ 64 \\ 1×1×1,\ 256 \end{bmatrix} ×3$
stage 2	$T×28×28$	$\begin{bmatrix} 1×1×1,\ 512 \\ 1×3×3,\ 128 \\ 1×1×1,\ 512 \end{bmatrix} ×4$	$\begin{bmatrix} 1×1×1,\ 512 \\ 3×3×3,\ 128 \\ 1×1×1,\ 512 \end{bmatrix} ×4$
stage 3	$T×14×14$	$\begin{bmatrix} 3×1×1,\ 1024 \\ 1×3×3,\ 256 \\ 1×1×1,\ 1024 \end{bmatrix} ×6$	$\begin{bmatrix} 1×1×1,\ 1024 \\ 3×3×3,\ 256 \\ 1×1×1,\ 1024 \end{bmatrix} ×6$
stage 4	$T×7×7$	$\begin{bmatrix} 3×1×1,\ 2048 \\ 1×3×3,\ 512 \\ 1×1×1,\ 2048 \end{bmatrix} ×3$	$\begin{bmatrix} 1×1×1,\ 2048 \\ 3×3×3,\ 512 \\ 1×1×1,\ 2048 \end{bmatrix} ×3$
pool5	1×1×1	spatial-temporal avg pool, fc layer with softmax	

Table 5: **ResNet3D architectures considered in our exploration experiments**. The backbone is ResNet50 [17]. Note that we set the temporal downsampling rate is 1 by default. *: to reduce computation and parameters, all the 3×3×3 convolutions are channel-wise.

5   Ablation Study

Next, we fairly perform an in-depth experimental analysis and complete ablation experiments on our proposed training pipeline and STS convolution. We use the fixed budget setting as default pipeline. All the from-scratch baselines are trained for 100 epochs. When evaluating our methods, we only fine-tune 50 epochs.
Base architecture. In general, 3D convolution can be divided into temporal 3D convolution and spatial-temporal 3D convolution. We chose two corresponding architectures to verify the generality and effectiveness of our method, presented in Table 5. Their backbone is ResNet 50 [17]. They are distinguished into ResNet50-3×1×1 and ResNet50-3×3×3 depending on whether the convolution of the core bottleneck block is $3 \times 1 \times 1$ or $3 \times 3 \times 3$. It is worth noting that all the 3D convolutions in ResNet50-3×3×3 are channel-wise to reduce the computation and the number of parameters. Following Slowfast and X3D [12,11], the temporal downsampling rate is always set to 1. Specifically, our model takes clips with a size of T×224×224 where $T = \{8, 16, 32\}$ is the number of frames.

5.1   Ablation on Image Pre-Training

Effectiveness. We first validate the effectiveness of the proposed method on K400 and SS-V2 in Table 6. First, we control the total training budget to be close to from-scratch (×1.16). On the K400, ResNet50-3×1×1 trained with our pipeline outperform the from-scratch by **1.8%**. For ResNet50-3×3×3 it improves

ResNet50-**3x1x1**	K400	SS-V2	Training Budget
From Scratch	73.1	58.4	192M
Pre-train&Fine-tune	**74.9**	**61.6**	×1.16
+ STS Conv	**75.0**	**62.7**	×1.16

ResNet50-**3x3x3**	K400	SS-V2	Training Budget
From Scratch	70.7	57.5	192M
Pre-train&Fine-tune	**74.3**	**60.4**	×1.16
+ STS Conv	**74.7**	**61.4**	×1.16

Table 6: **Effectiveness**. Ablation study on Image pre-training & fine-tuning and Spatiotemporal separable 3D convolution (STS). We use 8 frames as input and all the models are pre-trained on ImageNet-1K.

the baseline by **3.5%**. Under the tight training computation, image pre-training significantly improves the performance. It shows the importance of appearance modeling in video classification. In addition, replacing all the 3D convolution layer with STS convolution further improve the results. On the SS-V2, the proposed training pipeline can improve the accuracy by **3.2%** from **58.4%** to **61.6%** on ResNet50-3×1×1. The same-scale improvement of **2.9%** can be found on ResNet50-3×3×3. Moreover, on both backbones, STS convolution notably improve the pre-trained models by an average of **1.1%** All the results demonstrate that under the same training budget, image pre-training can remarkably boost performance, and the STS convolution can also benefit from the appearance modeling prior to further improve performance.

ResNet50-**STS 3x3x3**	Input Frames	Pre-Train	Fine-tune	Training Budget	K400 Top1
From Scratch	8	-	192M	192M	71.0
ImageNet Pre.	8	128M	96M	×1.16	**74.7**
From Scratch	16	-	384M	384M	73.5
ImageNet Pre.	16	192M	192M	×1	**76.1**
From Scratch	32	-	768M	768M	74.4
ImageNet Pre.	32	384M	384M	×1	**76.7**

Table 7: **More Frames *vs*. More Image Pre-training.** Following the fixed budget setting described in Section 3.3, we extend the pre-training epochs to match the total computation budget.

More input frames *vs*. more image pre-training epochs. In this experiment, we aim to compare the capacity of the pre-training schedule when scaling up to different input frames. We use the ResNet50-3×3×3 equipped with STS convolution as the backbone. To make sure our total computation is consistent, we extend the image pre-training epochs from 100 epochs to 150 or 300 for 16 frames and 32 frames. The fine-tuning epochs remain the same for all models (50 epochs). Surprisingly, this simple scaling works well for more input frames. As shown in Table 7, employing image pre-training improve performance by **3.7%**, **2.6%** and **2.4%** for 8, 16, and 32 frames, respectively. The stable performance gains demonstrate that our method can scale well over different input frames. Furthermore, these results may draw an important clue that why previous default pre-training pipeline does not work well: we should extend the pre-training schedule when our video models take more frames as input.

Source of image pre-training dataset. We next discuss the influence of the dataset used for pre-training. We conduct experiments on two different training

ResNet50-STS 3x3x3	Input Frames	Pre-Train	Fine-tune	Training Budget	K400 Top1
From Scratch	8	-	192M	192M	71.0
ImageNet Pre.	8	128M	96M	×1.16	**74.7**
K400/Image Pre.	8	128M	96M	×1.16	73.4

X3D-S	Input Frames	Pre-Train	Fine-tune	Training Budget	K400 Top1
From Scratch	13	-	798M	798M	73.2
ImageNet Pre.	13	384M	312M	×0.87	**73.5**
K400/Image Pre.	13	384M	312M	×0.87	72.8

Table 8: **Choice of Pre-Training Dataset: ImageNet *vs*. K400**

budgets (100 epochs *vs*. 256 epochs train from scratch). First, in the fixed budget setting, we use ResNet50-STS 3×3×3 as the backbone. Here we pre-train each frame of all videos in K400, and we match the total training computation with ImageNet by controlling the number of epochs, as shown in Table 8. The performance of the ImageNet pre-train model is better than K400-Image pre-train under the same pre-training setting (**74.7%** *vs*. **73.4%**). The k400-image pre-train model also has notable improvement over training from scratch from **71.0%** to **73.4%**, proving the effectiveness of image pre-training. In the train for state-of-the-art setting, the results obtained by pre-training on K400-image are slightly lower than from-scratch by 0.5%. In both settings, the ImageNet pre-training is consistently better than the K400-image. We assume that the label of the K400-image is still an action label, which can lead to confusion. Hence, we choose ImageNet as our source of image pre-training dataset by default.

5.2   Ablation on Spatiotemporal Fine-Tuning

ResNet50-**3x1x1**	K400	SS-V2
Inflation	74.5	61.8
+ STS Conv	**74.8**	**62.6**
Zero-Init	74.9	61.6
+ STS Conv	**75.0**	**62.7**

ResNet50-**3x3x3**	K400	SS-V2
Inflation	74.0	60.0
+ STS Conv	**74.6**	**61.7**
Zero-Init	74.3	60.4
+ STS Conv	**74.7**	**61.4**

ResNet50-**3x3x3**	K400	SS-V2
$(1/3, 1/3, 1/3)$	74.0	60.0
$(1/3, 1/3, 1/9)$	74.3	60.0
$(1/9, 1/3, 1/9)$	74.3	60.2
$(0, 1, 0)$	**74.3**	**60.4**

Table 9: **Different Initialization Strategies.**

Initialization strategy. Next, we explore how to initialize the model in the fine-tuning phase. Inflation averages the pre-trained 2D model parameters along the temporal dimension [4]. We can initialize the center kernel of 3D convolution with pre-trained weights named as zero-init. Remaining parts are zeros. As shown in Table 9, in most cases, the zero-init approach exceeds inflation. ResNet50-3×3×3 zero-init exceeds the inflation by **0.3%** on K400 and **0.2%** on SS-V2. ResNet50-3×1×1 zero-init outperforms the inflation by **0.4%** on K400 but slightly lower on SS-V2. These results suggest 3D CNNs prefer 2D image initialization. More importantly, our proposed STS convolution achieves a considerable improvement regardless of the initialization. For example, ResNet50-STS

$3\times3\times3$ improves the inflation baseline by **0.6%** on K400 and **1.7%** on SS-V2. Compared with zero-init, it boosts the performance by **0.4%** on K400 and **1.0%** on SS-V2. Same scale improvement can be found on ResNet50-STS $3\times1\times1$. It indicates that STS convolution can be optimized very well using different initialization. The performance improvement of STS convolution comes more from the specific design. In the right table, we also evaluate different inflation rates, closer (*e.g.*, $(1/9, 1/3, 1/9)$) to zero-init rate works better. It also indicates that the initialization strategy is less important than training pipeline setting.

ResNet50-3x3x3	ImageNet-1K	K400	SS-V2
Baseline	76.6	74.3	60.4
w/o splitting	76.6	74.0	60.8
w/ splitting	**77.1**	**74.7**	**61.4**

ResNet50-3x3x3	ImageNet-1K	K400	SS-V2
1:1	77.1	**74.7**	61.4
1:2	76.8	74.6	**61.8**
2:1	**77.2**	74.7	61.4

Table 10: **Details of Design in STS Convolution.**

Ablation on STS convolution. Next, we perform ablation on the STS convolution design. The experimental results are shown in Table 10. First of all, our baseline model is ResNet50-$3\times3\times3$. From the table on the left, we can see that the key factor in STS is whether to use splitting or not. Splitting operation utilizes all the weights in the original 3D convolution, increasing the spatial perception field. It can boost the video classification performance as same as image classification. With splitting operation, it improves the without reshaping counterpart on the ImageNet (+ **0.5%**), K400 (+**0.7%**) and SS-V2 (+**1.0%**). On the right table, we examine how many channels should be divided. For example, ratio of $1:2$ means $1/3$ channels are used for appearance modeling. The results are very close on the K400. Thus we use $1:1$ as the default ratio.

6   Conclusion

We argue that image pre-training has significant potential for 3D video recognition, and should be more commonly adopted since 1) image pre-training models which are widely and freely available, contain enriched appearance information for facilitating temporal reasoning 2) image pre-training enables a large speedup for existing 3D CNN training. Specifically, in this paper, we reveal that the key to exploiting image pre-training lies in properly decomposing the spatiotemporal features into spatial and temporal parts. Based on this principle, we design a new image pre-training and spatiotemporal fine-tuning strategy, which achieves notable performance improvements for a wide range of 3D CNNs on multiple video recognition tasks with significant speedup.

Acknowledgement

References

1. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: ICCV (2021)
2. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding. In: ICML (2021)
3. Cao, Y., Xu, J., Lin, S., Wei, F., Hu, H.: Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In: ICCVW (2019)
4. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017)
5. Chen, Y., Rohrbach, M., Yan, Z., Shuicheng, Y., Feng, J., Kalantidis, Y.: Graph-based global reasoning networks. In: CVPR (2019)
6. Christoph, R., Pinz, F.A.: Spatiotemporal residual networks for video action recognition. In: NeurIPS (2016)
7. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: CVPRW (2020)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
10. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. In: ICCV (2021)
11. Feichtenhofer, C.: X3d: Expanding architectures for efficient video recognition. In: CVPR (2020)
12. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: ICCV (2019)
13. Girshick, R.: Fast r-cnn. In: ICCV (2015)
14. Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The" something something" video database for learning and evaluating visual common sense. In: ICCV (2017)
15. He, K., Girshick, R., Dollár, P.: Rethinking imagenet pre-training. In: ICCV (2019)
16. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
18. Jiang, B., Wang, M., Gan, W., Wu, W., Yan, J.: Stm: Spatiotemporal and motion encoding for action recognition. In: ICCV (2019)
19. Kondratyuk, D., Yuan, L., Li, Y., Zhang, L., Tan, M., Brown, M., Gong, B.: Movinets: Mobile video networks for efficient video recognition. In: CVPR (2021)
20. Li, K., Li, X., Wang, Y., Wang, J., Qiao, Y.: Ct-net: Channel tensorization network for video classification. In: ICLR (2021)
21. Li, K., Wang, Y., Gao, P., Song, G., Liu, Y., Li, H., Qiao, Y.: Uniformer: Unified transformer for efficient spatiotemporal representation learning. In: ICLR (2022)
22. Li, X., Wang, Y., Zhou, Z., Qiao, Y.: Smallbignet: Integrating core and contextual views for video classification. In: CVPR (2020)
23. Li, Y., Ji, B., Shi, X., Zhang, J., Kang, B., Wang, L.: Tea: Temporal excitation and aggregation for action recognition. In: CVPR (2020)
24. Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: ICCV (2019)

25. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. arXiv (2021)
26. Liu, Z., Luo, D., Wang, Y., Wang, L., Tai, Y., Wang, C., Li, J., Huang, F., Lu, T.: Teinet: Towards an efficient architecture for video recognition. In: AAAI (2020)
27. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
28. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: ICCV (2017)
29. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NeurIPS (2015)
30. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Gradcam: Visual explanations from deep networks via gradient-based localization. In: ICCV (2017)
31. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NeurIPS (2014)
32. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
33. Sudhakaran, S., Escalera, S., Lanz, O.: Gate-shift networks for video action recognition. In: CVPR (2020)
34. Sun, C., Shrivastava, A., Singh, S., Gupta, A.: Revisiting unreasonable effectiveness of data in deep learning era. In: ICCV (2017)
35. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR (2015)
36. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: ICML (2021)
37. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV (2015)
38. Tran, D., Wang, H., Torresani, L., Feiszli, M.: Video classification with channel-separated convolutional networks. In: ICCV (2019)
39. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: CVPR (2018)
40. Wang, H., Tran, D., Torresani, L., Feiszli, M.: Video modeling with correlation networks. In: CVPR (2020)
41. Wang, L., Tong, Z., Ji, B., Wu, G.: Tdn: Temporal difference networks for efficient action recognition. In: CVPR (2021)
42. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Gool, L.V.: Temporal segment networks: Towards good practices for deep action recognition. In: ECCV (2016)
43. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018)
44. Wang, X., Gupta, A.: Videos as space-time region graphs. In: ECCV (2018)
45. Wightman, R., Touvron, H., Jégou, H.: Resnet strikes back: An improved training procedure in timm. arXiv (2021)
46. Wu, C.Y., Girshick, R., He, K., Feichtenhofer, C., Krahenbuhl, P.: A multigrid method for efficiently training video models. In: CVPR (2020)
47. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: ECCV (2018)
48. Yang, C., Xu, Y., Shi, J., Dai, B., Zhou, B.: Temporal pyramid network for action recognition. In: CVPR (2020)
49. Yin, M., Yao, Z., Cao, Y., Li, X., Zhang, Z., Lin, S., Hu, H.: Disentangled non-local neural networks. In: ECCV (2020)

50. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: ICCV (2019)
51. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: ICLR (2018)
52. Zhou, B., Andonian, A., Oliva, A., Torralba, A.: Temporal relational reasoning in videos. In: ECCV (2018)