Multi-Temporal Convolutions for Human Action Recognition in Videos

Alexandros Stergiou

Department of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
a.g.stergiou@uu.nl

Ronald Poppe

Department of Information and Computing Sciences
Utrecht University
Utrecht, The Netherlands
r.w.poppe@uu.nl

Abstract—Effective extraction of temporal patterns is crucial for the recognition of temporally varying actions in video. We argue that the fixed-sized spatio-temporal convolution kernels used in convolutional neural networks (CNNs) can be improved to extract informative motions that are executed at different time scales. To address this challenge, we present a novel convolution block that is capable of extracting spatio-temporal patterns at multiple temporal resolutions. Our proposed multi-temporal convolution (MTConv) blocks utilize two branches that focus on brief and prolonged spatio-temporal patterns, respectively. The extracted time-varying features are aligned in a third branch, with respect to global motion patterns through recurrent cells. The proposed blocks are lightweight and can be integrated into any 3D-CNN architecture. This introduces a substantial reduction in computational costs. Extensive experiments on Kinetics, Moments in Time and HACS action recognition benchmark datasets demonstrate competitive performance of MTConvs compared to the state-of-the-art with a significantly lower computational footprint[1].

I. INTRODUCTION

The variations in how humans execute tasks and how they interact with each other present significant challenges for the recognition of their actions in videos [1]. Differences in visual appearance can largely be captured by deep convolutional neural networks (CNNs). For action recognition in videos, 2D convolutions have been successfully extended to 3D convolutions to additionally extract informative patterns over time. Although connections between the three dimensions do exist [2], the symmetrical processing of temporal and spatial information significantly limits how the variations in the execution of actions over time can be modeled. For example, as shown in Figure 1, examples in the same action category can significantly differ in the type and duration of the performed movements.

Current efforts in action recognition are based on 3D convolutions [3] with fixed-sized kernels. We observe that variations in the duration of the performance of an action are typically not in the order of magnitude. Therefore, a greater number of variations in temporal movements can be captured with the extraction of different timescale spatio-temporal patterns. The flexible extraction of space-time features can improve the feature representation capabilities of space-time models.

Fig. 1. Three examples of basketball passes of different duration: (A) brief hand-off pass, (B) longer wing pass, and (C) full-court pass spanning the entire clip duration.

In this work, we introduce a convolutional block that explicitly models temporally variant motion patterns and their cross-feature dependencies. We observe that contributing factors relating to the temporal complexity of actions are primarily recording-related, such as camera motion and video frame rates, or performance-related corresponding to the actor's *pre-potent identity* of an action [4]. Motivated by this distinction, we believe that 3D convolutions are inherently constrained to capture only fixed-sized local patterns. To address the temporal complexity of human actions in videos, we propose a multi-temporal convolution (MTConv) block that captures spatio-temporal features variations within their representations. As shown in Figure 2, the blocks consist of three branches: a *local branch* ($\mathcal{L}$), a *prolonged branch* ($\mathcal{P}$), and a *global aggregated feature importance branch* ($\mathcal{G}$). The local and prolonged branches focus on spatio-temporal patterns that are performed in short and longer spatio-temporal windows, respectively. The global aggregated feature importance branch aligns the activations of these two branches based on the temporal dynamics of motions across the entire video. The novel design of MTConvs, and the resulting MTBlocks, enable the discovery of patterns across multiple time-scales as well as their combined temporal attention over the entire video sequence. Owing to this property, we can capture local feature dependencies within the (global) scale of the entire sequence.

We validate the proposed blocks on action classification for third person videos, on four large-scale benchmark datasets

[1]Our code is available at: https://git.io/JfuPi

and an additional fine-tuning dataset. We report performance on Kinetics-400 [5] demonstrating the descriptive quality of MTConv features over equivalent state-of-the-art approaches with similar or larger computational requirements. We further test our method on the Moments in Time [6] dataset that includes large motion variances across examples of the same action class. Equivalently, we present results on the recent Kinetics-700 [7] and HACS [8] datasets. The feature transfer capabilities are then tested with pre-trained models used on UCF-101 [9].

We discuss current progress in action recognition in Section II. We then introduce the proposed MTConv blocks in Section III. Evaluation of our work is shown in Section IV, and we conclude in Section V.

II. RELATED WORK

Two-stream networks. Optical flow is a widely used method for representing motion information across video frames. In two-stream networks [10], one stream is responsible for handling individual (RGB) frames sequentially, while the other processes the optical flow equivalent motions. Information from both streams is then fused at the end. Enhancements to the two-stream approach include the addition of lateral connections between the two streams, for the inclusion of temporal information within spatial patterns [11], division into temporal segments [12] and spatial-based and temporal-based encoding of segments [13]. The main limitation of the two-stream approach is the strong dependence on hand-coded optical flow inputs that prevents the joint learning of complex spatio-temporal features in an end-to-end manner.

3D convolutions. An alternative approach for utilizing temporal information is the use of 3D convolutions that operate over space-time volumes of stacked frames [14]. The use of 3D convolutions has shown improvements on modeling complex spatio-temporal features [15], [16]. Compared to their 2D counterparts, the large computational requirements of 3D convolutions is a disadvantage with much of the recent literature aimed at improving the efficiency of 3D-CNNs. Tran *et al.* [17] have considered decoupling 3D convolutions into temporal-only and spatial-only, thus strongly reducing the number of trainable parameters. 3D grouped convolutions [18], [19] have also shown benefits on performing convolutions in groups and decreasing the number of GFLOPs without a reduction in accuracy. Other works have focused on step-wise progressive network expansions [20] that improved the overall efficiency 3D CNNs. Lin *et al.* [21] have introduced temporal shifting activations, emulating the effect of 3D convolutions while relying on frame-based 2D convolutions. This temporal shift module (TSM) has recently been combined with a 3D convolution method to enable or disable shifting [22].

Temporal streams in 3D convolutions. Inspired by the two-steam 2D CNNs with optical flow, initial works on stream-based 3D CNNs have focused on using dual inputs of stacked RGB frames alongside optical flow [5]. Later works of Feichenhofer *et al.* [23] proposed a 3D model of dual adjacent input-based frame sampling of slow and fast frame rates.

With the same rationale, Qiu *et al.* [24] showed how global paths using entire videos as inputs and local paths with local spatio-temporal segments can be used in two separate network pathways. Others have considered block-based approaches with octave convolutions [25] to model temporal variation in the frequency domain.

Although these methods have shown great promise in extracting robust spatio-temporal features, they do not directly address the complex motion features and their relationships across varying temporal scales. In contrast, our tri-branch method aims to address within convolutional blocks the temporal disparities of actions through the extraction of different periodic spatio-temporal features and the discovery of their dynamics.

Spatio-temporal attention. Image-based methods *Squeeze and Excitation* [26] and *Gather and Excite* [27] consider self-attention mechanisms as calibration methods for convolutional features in images. Extensions to video have also been proposed based on attention clustering [28] or through the division of information in terms of appearance and spatial relations [29]. This has also led to the introduction of recurrent sub-networks to explore the dynamics of extracted patterns [30]. Recent works (e.g. [31]) further utilize self-attention for the creation of local importance maps and location-invariant feature weights.

Our method uses temporal attention through the alignment of activations with Squeeze and Recursion modules [30] on the *global aggregated feature importance* branch. This enforces coherence of the learned spatio-temporal patterns despite temporal variations.

III. MULTI-TEMPORAL NETWORKS

In this section, we first describe multi-temporal convolutions (MTConvs) and their inner workings in terms of how information is processed. We then detail the structure of the blocks (*MTBlocks*, shown in Figure 2) in which they operate in Section III-B. In Section III-C, we introduce the CNN architecture that employs these MTBlocks.

Formally, layer activations are denoted by $\mathbf{a}_{(C \times T \times H \times W)}$ with C channels, T frames, height H and width W. Branch activations are denoted by $\mathbf{a}^{\mathcal{L}}$ for the local branch ($\mathcal{L}$) and $\mathbf{a}^{\mathcal{P}}$ for the prolonged branch ($\mathcal{P}$). Layers are indexed with l and indicated as $\mathbf{a}^{[l]}$ or following $\mathbf{a}^{[l],\mathcal{L}}$, $\mathbf{a}^{[l],\mathcal{P}}$ in branch notation.

A. Multi-Temporal Convolutions (MTConv)

The local and prolonged branches each use a portion of the number of channels ($\widetilde{C}$) of the layer (l). To determine the number of channels for each branch, a channel ratio parameter δ is introduced. We denote the channel size of the input activations ($\mathbf{a}^{[l-1]}$) as C. Channels $C_{\mathcal{L}}$, based on ratio δ, are the lowest integer value approximation (through the homonym function denoted with $\lfloor floor \rfloor$). Respectively, the maximum integer value approximation ($\lceil ceil \rceil$) is used for channels $C_{\mathcal{P}}$:

$$C_{\mathcal{L}} + C_{\mathcal{P}} = \widetilde{C} \, , where$$
$$C_{\mathcal{L}} = \lfloor \delta * \widetilde{C} \rfloor \; and \; C_{\mathcal{P}} = \lceil (1-\delta) * \widetilde{C} \rceil \tag{1}$$
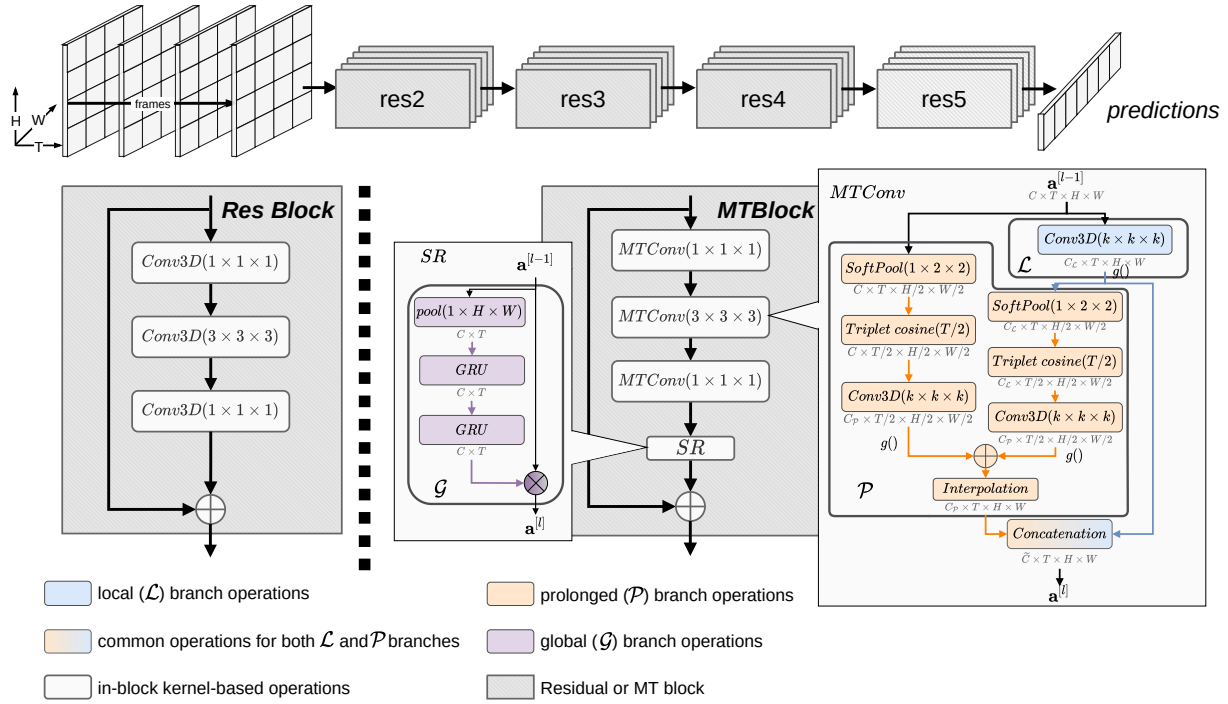
Fig. 2. **MTNet architecture**. Residual blocks (left) in the X3D [20] are replaced by the proposed MTBlocks (right). Blocks contain three consecutive multi-temporal convolutions (MTConv, center) followed by the Squeeze and Recursion (SR, left of center) feature alignment. We denote element-wise additions with $\oplus$ and element-wise multiplications with $\otimes$.

Inputs are first processed by the $\mathcal{L}$ and $\mathcal{P}$ branches. $\mathcal{L}$ processes a single input $\mathbf{a}^{[l-1]}$ of size $(C \times T \times H \times W)$. $\mathcal{P}$ is performed over input pair $(\mathbf{a}^{[l],\mathcal{L}}, \mathbf{a}^{[l-1]})$ with the first volume being the resulting activations from branch $\mathcal{L}$ of size $(C_{\mathcal{L}} \times T \times H \times W)$ and the second being the original layer input. Dual inputs are used in the prolonged branch ($\mathcal{P}$) as spatio-temporal patterns of elongated duration and spatial sizes are strongly correlated with corresponding more local features, which are extracted by $\mathcal{L}$. With prolonged features incorporating the complexity of local short-term ones, the $\mathcal{P}$ branch effectively operates over $C_{\mathcal{L}} + C_{in}$ channels addressing the added complexity over $\mathcal{L}$. The $\mathcal{L}$ branch and $\mathcal{P}$ branch feature extraction process is summarized in Equation (2):

$$\mathbf{a}^{[l]} = \mathcal{L}(\mathbf{a}^{[l-1]}) \frown (\mathcal{P}(\mathcal{L}(\mathbf{a}^{[l-1]}), \mathbf{a}^{[l-1]})) \qquad (2)$$

where $\frown$ denotes the concatenation of the outputs from the two branches into a single volume.

Local branch in MTConv. The local branch is used for the extraction of short-term local motions within the input activations. Given input $(\mathbf{a}^{[l-1]})$ we use a Conv3D followed by batch normalization (BN) [32] and compute feature volume $(\mathbf{z}^{[l],\mathcal{L}})$ of $C_{\mathcal{L}}$ channels followed by non-linearity ($g()$) with ReLU. Unless stated otherwise, $g()$ refers to a ReLU activation. The final branch output takes the form of $\mathbf{a}^{[l],\mathcal{L}} = g(\mathbf{z}^{[l],\mathcal{L}})$.

Prolonged branch in MTConv. The prolonged branch aims at the extraction of patterns of extended duration, incorporating information from the local branch ($\mathcal{L}$) and the layer input. To explore long-temporal features, both inputs are reduced by a factor of two across their spatio-temporal dimensions. Such a size reduction provides a balanced trade-of between accuracy and computation. More aggressive reduction strategies using larger factors lead to significant information loss. Both inputs are initially downsampled spatially by their per-frame regional exponential maximum with *SoftPool* [33] with the activations produced being of size $T \times H' \times W'$ (where $H' = H/2$ and $W' = W/2$). The activations are then downsampled temporally by a temporal triplet cosine frame selection to size $T' = T/2$. We provide detailed explanations for both methods later in the section. The inclusion of receptive fields twice the duration of those in $\mathcal{L}$ allows for the exploration of temporal movements of larger spatio-temporal regions without the increased computational requirements of kernels double the size. Extended temporal patterns for inputs $\mathbf{a}^{[l-1]}$ and $\mathbf{a}^{[l],\mathcal{L}}$ are extracted by Conv3D operations followed by BN. The complete process is formulated as follows:

$$\mathbf{a}^{[l],\mathcal{P}} = \mathcal{I}(g(\mathbf{z}^{[l],\mathcal{L} \to \mathcal{P}}) \oplus g(\mathbf{z}^{[l],\mathcal{P}})) \qquad (3)$$

in which $\oplus$ denotes element-wise addition and $\mathcal{I}()$ is the spatio-temporal tri-linear interpolation of the volume from size $(T' \times H' \times W')$ to original size $(T \times H \times W)$. The feature volume $\mathbf{z}^{[l],\mathcal{L} \to \mathcal{P}}$ corresponds to the extracted patterns from the reduced input $\mathbf{a}^{[l],\mathcal{L}}$, while $\mathbf{z}^{[l],\mathcal{P}}$ corresponds to features extracted from $\mathbf{a}^{[l-1]}$:

$$\mathbf{z}^{[l],\mathcal{L} \to \mathcal{P}} = \mathcal{T}(\bar{\mathbf{a}}^{[l],\mathcal{L}}) * \mathbf{w}^{\mathcal{L} \to \mathcal{P}} \; and \; \mathbf{z}^{[l],\mathcal{P}} = \mathcal{T}(\bar{\mathbf{a}}^{[l]}) * \mathbf{w}^{\mathcal{P}} \quad (4)$$

with $\mathcal{T}()$ the triplet cosine frame selection for a spatially pooled volume ($\overline{\mathbf{a}}$). The convolutional weight vectors for the respective inputs are denoted as $\mathbf{w}^{\mathcal{L}\to\mathcal{P}}$ and $\mathbf{w}^{\mathcal{P}}$.

Prolonged branch spatial downsampling. Downsampling blocks use soft-maximum approximation (*SoftPool*, [33]), to reduce the spatial dimensions of the input activations. The method uses the softmax weights of activations with each of the inputs within the kernel region having a proportional effect to the output. This is formulated given an input $\mathbf{a}$ and frame (t) region $\mathbf{R}$ for size $H \times W$:

$$\overline{\mathbf{a}}_{t,r} = \sum_{r\in\mathbf{R}} \frac{e^{\mathbf{a}_{t,r}} * \mathbf{a}_{t,r}}{\sum_{k\in\mathbf{R}} e^{\mathbf{a}_{t,k}}}, \ \forall\, t \in |T| \quad (5)$$
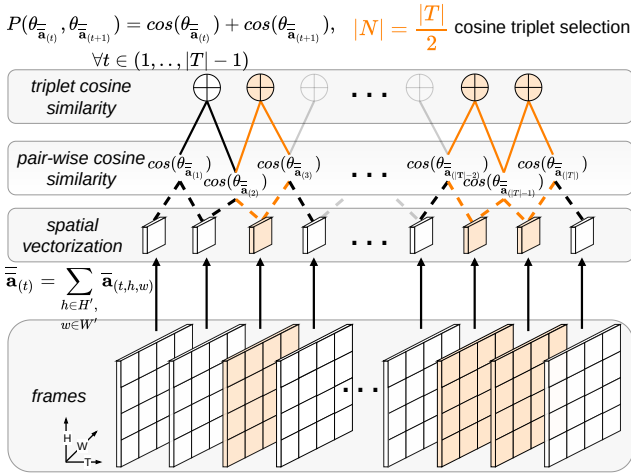

Fig. 3. **Temporal triplet cosine similarity frame selection**. Selection is based on the channel-wise sum ($\oplus$) of cosine similarities per pair of adjacent frames ($cos(\theta_{\overline{\overline{\mathbf{a}}}_{(t)}})$), calculated from their spatially-summed volumes ($\overline{\overline{\mathbf{a}}}$).

Prolonged branch temporal downsampling. The extension of image-based pooling methods to time-inclusive data comes at the expense of a decrease in spatial detail through the fusion of multiple frames. As the proposed method depends on the preservation of such features in order to extract their extended spatio-temporal patterns, we instead introduce a frame-selection sampling method to decrease the temporal dimensionality of the spatially-reduced activation volume ($\overline{\mathbf{a}}$). We termed this method *temporal triplet cosine frame selection*.

The frame selection process is shown in Figure 3. Initially, for each frame t all activations are summed spatially to produce a single value per frame. The spatially-summed activation ($\overline{\overline{\mathbf{a}}}$) contains frame-wise activation vectors. These vectors can be used to measure the feature-wise similarity per pair of frames using their respective dot product and magnitude:

$$cos(\theta_{\overline{\overline{\mathbf{a}}}_{(t)}}) = \frac{\sum_{c\in C} \overline{\overline{\mathbf{a}}}_{(t,c)} * \overline{\overline{\mathbf{a}}}_{(t+1,c)}}{\sqrt{\sum_{c\in C} \overline{\overline{\mathbf{a}}}^2_{(t,c)}} * \sqrt{\sum_{c\in C} \overline{\overline{\mathbf{a}}}^2_{(t+1,c)}}} \quad (6)$$

Their cosine similarity is then summed in similarity pairs ($P(\theta_{\overline{\overline{\mathbf{a}}}_{(t)}}, \theta_{\overline{\overline{\mathbf{a}}}_{(t+1)}}) = cos(\theta_{\overline{\overline{\mathbf{a}}}_{(t)}}) + cos(\theta_{\overline{\overline{\mathbf{a}}}_{(t+1)}})$) for the creation of triplets. This represents a concatenated view of the similarity in features for frame (t) in comparison to features of the preceding frame ($t-1$) and succeeding frame ($t+1$). Temporal pooling by triplets then takes the form of selecting the frame locations (N) with the lowest $|T|/2$ triplet cosine similarities. This cosine-based sampling can reduce feature redundancy across frames while focusing of frames that are found to be more informative:

$$\underset{\forall n\in N}{argmin}\, P(\theta_{\overline{\overline{\mathbf{a}}}_{(n)}}, \theta_{\overline{\overline{\mathbf{a}}}_{(n+1)}}) = cos(\theta_{\overline{\overline{\mathbf{a}}}_{(n)}}) + cos(\theta_{\overline{\overline{\mathbf{a}}}_{(n+1)}}),$$
$$where\ N \subset T, |N| = |T|/2 \quad (7)$$

As frames are selected based on their similarity instead of being temporally fused, the per-frame activations remain consistent over the produced decreased volume.

B. Multi-Temporal Blocks (MTBlocks)

Global aggregated feature importance. We align the concatenated activations of the local ($\mathcal{L}$) and prolonged ($\mathcal{P}$) branches based on the importance of each feature in the context of the entire video sequence. The role of the *global aggregated feature importance* branch ($\mathcal{G}$) is the creation of coherent activations based on averaged feature attention through *Squeeze and Recursion* [30] with GRU [34] recurrent cells. The branch operates over a vectorized version of the original volume pooled by its spatial dimensionality ($pool(\mathbf{a}^{[l-1]})$). The pooled volume is processed through a dual-layer recurrent sub-network for the discovery and amplification of globally-informative features. Initial refinement of salient features is achieved by the update gate ($\mathbf{z}_{(t)}$) that uses the per-frame (t) instance input ($pool(\mathbf{a}^{[l-1]})_{(t)}$) with state ($\mathbf{h}_{(t-1)}$) of the previous recurrent cell (for time $t-1$), through a sigmoid (σ) activation and weight $\mathbf{W}_z$ (with bias term $\mathbf{b}_z$):

$$\mathbf{z}_{(t)} = \{\sigma(\mathbf{W}_z * [\mathbf{h}_{(t-1)}, pool(\mathbf{a}^{[l-1]})_{(t)}] + \mathbf{b}_z)\} \quad (8)$$

Cell input $pool(\mathbf{a}^{[l-1]})_{(t)}$ and previous state outputs $\mathbf{h}_{(t-1)}$ also pass through a reset gate ($\mathbf{r}_{(t)}$), with weight ($\mathbf{W}_r$) and bias ($\mathbf{b}_r$) terms, to ignore temporally inconsistent features.

$$\mathbf{r}_{(t)} = \{\sigma(\mathbf{W}_r * [\mathbf{h}_{(t-1)}, pool(\mathbf{a}^{[l-1]})_{(t)}] + \mathbf{b}_r)\} \quad (9)$$

Both update and reset gates act in a complementary manner on the same inputs. Based on the activations produced by the reset gate, a candidate hidden state is computed ($\widetilde{\mathbf{h}}_{(t)}$) with a $tanh$ activation, and reduced influence from the previous state ($\mathbf{h}_{(t-1)}$) based on $\mathbf{r}_{(t)}$. The produced cell state is the fusion of a proportion of the previous state ($\mathbf{z}_{(t)} * \mathbf{h}_{(t-1)}$) and the supplementary portion of the candidate hidden state (($1 - \mathbf{z}_{(t)}) * \widetilde{\mathbf{h}}_{(t)}$), as summarized in Equations (10) and (11):

$$\widetilde{\mathbf{h}}_{(t)} = tanh(\mathbf{W}_h * [\mathbf{r}_{(t)} * \mathbf{h}_{(t-1)}, pool(\mathbf{a}^{[l-1]})_{(t)}] + \mathbf{b}_h) \quad (10)$$
$$\mathbf{h}_{(t)} = \mathbf{z}_{(t)} * \mathbf{h}_{(t-1)} + (1 - \mathbf{z}_{(t)}) * \widetilde{\mathbf{h}}_{(t)} \quad (11)$$

All cell outputs ($\mathbf{h}_{(t)}$) are concatenated to create a filtered activation map of intensities. As shown in Figure 2, this temporal excitation volume is used in conjunction with the original input ($\mathbf{a}^{[l-1]}$) through an element-wise multiplication operation. The produced activation map ($\mathbf{a}^{[l]}$) effectively incorporates the global feature dynamics for the discovered features of different spatio-temporal region sizes.

C. Multi-Temporal Networks (MTNet)

We introduce three MTNet architecture variants that use as backbones the corresponding X3D$_S$, X3D$_M$ and X3D$_L$ models which vary in size and GFLOP usage, and replace their Residual blocks and 3D Convs with the proposed MTBlocks and MTConvs, as shown in Figure 2. We denote our models as MTNet$_S$, MTNet$_M$ and MTNet$_L$. The architectures follow a step-wise network and block expansion as recently proposed in video [20] and image-based models [35]. Details of the three proposed models in terms of the number of parameters and GFLOPs appear in Table I.

IV. EXPERIMENTS AND RESULTS

We evaluate our MTBlock and the three MTNets on five popular action recognition benchmark datatsets and compare them against the current state-of-the-art in Section IV-D. In Section IV-E, we compare MTNets with different channel ratios (δ) as well as regular 3D convolutions in terms of classification accuracy and computational complexity. Finally, we evaluate the performance in a transfer learning setting (Section IV-F).

A. Datasets

For our main evaluation, we use four large-scale action recognition datasets. The results presented in this section are calculated on the validation sets of Kinetics-400 (K-400) [5], the extended Kinetics-700 (K-700) [7], Moments in Time (MiT) [6] and Human Action Clips Segments (HACS) Clips [8] datasets. For the transfer learning task, we evaluate on the smaller UCF-101 [9] dataset.

B. Training

For HACS models are trained from random initialization (*from scratch*) without pre-training. We set the mini-batch size to 16 clips per GPU, with the total mini-batch size of 64. All experiments were performed with half-precision (float16) for more effective utilization of memory. Similar to relevant works [20], [23], we use a cosine-based learning rate decay schedule [36] with the the learning rate lr_n for iteration n calculated as $lr_n = lr_0 * 0.5[cos(\frac{n}{n_{max}}\pi) + 1]$ in which n_{max} is the total number of iterations and lr_0 is the starting learning rate. We use $lr_0 = 1.16$ and learning rate warm-up for the first 8k iterations similar to [20], [23]. Batch sizes are determined by a multigrid method [37] with the initial batch size of 64. The multigrid learning rate follows the linear scaling rule [38] given the mini-batch size scaling. Unless specified otherwise, all experiments were performed over $n_{max} = 400$ epochs with momentum of 0.9 and weight decay of 5×10^{-5}.

Table I. **Comparison with K-400 state-of-the-art**. For consistency with previous testing methods, we report the model complexity as the GFLOPs per single clip view $\times$ the number of clips with spatial cropping of size 256×256.

| Model	Input	Backbone	top-1 (%)	top-5 (%)	GFLOPs $\times$ views	Params
R(2+1)D [17]	16×224^2	ResNet101	62.8	83.9	152×115	63.6M
I3D [5]	16×224^2	InceptionV1	71.6	90.0	$108 \times N/A$	12.0M
MF-Net [18]	16×224^2	ResNet50	72.8	90.4	11.1×50	**8.0M**
TAM [39]	$(24, 2) \times 224^2$	ResNet50	73.5	91.2	93.4×9	25.0M
SRTG-101 (3D) [30]	16×224^2	ResNet101	73.2	91.3	78.1×30	107.1M
SRTG-101 (2+1D) [30]	16×224^2	ResNet101	73.8	92.0	163.1×30	105.3M
TSM [21]	16×224^2	ResNet50	74.7	91.4	65×10	24.3M
ip-CSN-101 [19]	8×224^2	ResNet101	76.7	92.3	83×30	24.5M
ip-CSN-152 [19]	8×224^2	ResNet152	77.8	92.8	108.8×30	32.8M
SF-50 [23]	$(8, 8) \times 224^2$	ResNet50	77.0	92.6	65.7×30	34.4M
SF-101 [23]	$(8, 8) \times 224^2$	ResNet101	77.9	93.5	213×30	53.7M
SF-101+NL [23]	$(8, 8) \times 224^2$	ResNet101	78.7	93.5	116×30	59.9M
X3D-XL [20]	16×312^2	ResNet(X3D)	**79.1**	**93.9**	48.4×30	11.0M
MTNet$_S$ **(ours)**	16×256^2	ResNet(X3D)	74.8	92.1	**5.8** $\times 30$	25.8M
MTNet$_M$ **(ours)**	16×256^2	ResNet(X3D)	76.6	92.5	8.8×30	25.8M
MTNet$_L$ **(ours)**	16×256^2	ResNet(X3D)	78.1	93.2	17.6×30	50.1M
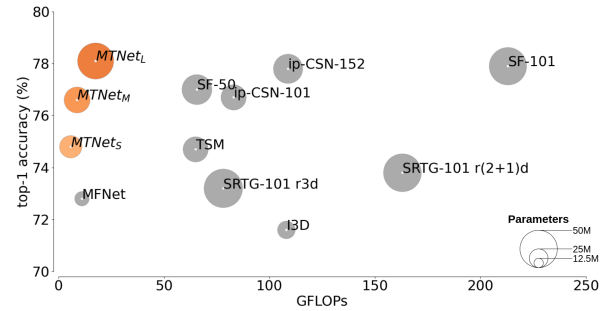
Fig. 4. **Top-1 accuracy to computational complexity tradeoff in K-400.** FLOP and parameter calculations are based on the inputs in Table I.

For K-400, K-700 and MiT, we use similar training parameters but initialize the networks weights from the models trained on HACS. For the transfer learning task on UCF-101, we reduce the start learning rate to 0.01 and do not use warm-up. We also decrease the number of epochs to 150 while including a learning rate multiplier for convolutional weights of value 0.1. Unless stated otherwise, our models use $\delta = 0.875$. We motivate this choice and experiment with other values in Section IV-E.

The input frames are uniformly randomly selected. Based on the average clip length, for each dataset, we use equivalently sized temporal strides when selecting frames. For HACS with an average clip length of 60 frames, we use a temporal stride of 2. The two Kinetics datasets have clips with 250 frames on average and we use a temporal stride of 5. For MiT, we use a temporal stride of 3 with an average clips length of 90 frames. For UCF-101 we use strides of 4. On the spatial domain, we randomly crop a region of size 256×256 pixels in resized video frames with the shortest side being 320.

C. Computational Inference

We report inference with two different measures. We first report computational costs (FLOPs) similar to [19], [20], [23], [39] by sampling 10 clips from a single video and perform 3 crops along the spatial dimensions ($10 \times 3 = 30$ views) of size

Table II. **Spatio-temporal block comparison on K-400.** Using a ResNet-50 as backbone, accuracy rates are reported for different spatio-temporal blocks. Numbers in parentheses are in comparison to 3D baseline.

Method	top-1 (%)	top-5 (%)	FLOPs (G)	Params (M)
3D [15]	61.3	83.1	53.2	36.7
(2+1)D [17]	61.8 (+0.5)	83.5 (+0.4)	56.0 (+2.8)	38.8 (+2.1)
Multi-Fiber [18]	72.8 (+11.5)	90.4 (+7.3)	**22.5** (−30.7)	**8.0** (−28.7)
Slow-only [23]	72.6 (+11.5)	90.3 (+7.2)	27.3 (−25.9)	26.6 (−10.1)
SlowFast [23]	74.3 (+13)	91.0 (+7.9)	39.8 (−13.4)	34.4 (−2.3)
MTConv (**ours**)	**74.8** (+13.5)	**91.3** (+8.2)	23.1 (−30.1)	35.7 (−1.0)

Table III. **Comparison with MiT state-of-the-art.** Models denoted with ‡ include additional optical flow input.

Model	Arch. size	top-1 (%)	top-5 (%)
EvaNet [40]	NAS [41]	31.8	N/A
AssembleNet [42]		34.3	**62.7**
TSN-Flow [6] ‡	Fixed	15.7	34.7
TSN-2stream [6] ‡		25.3	50.1
TRN-Multiscale [43]		28.3	53.9
I3D [5]		29.5	56.1
CoST [44]		32.4	60.0
SRTG-101 [30]		33.6	58.5
MTNet$_M$ (**ours**)		34.5	58.6
MTNet$_L$ (**ours**)		**35.2**	59.3

Table IV. **Comparison with K-700 state-of-the-art.** GFLOP calculation is similar to that in Table I.

Model	Pre-train	top-1 (%)	top-5 (%)	GFLOPs × views
I3D [5]	K-600	58.7	81.7	108×N/A
SRTG-101 (3D) [30]	HACS	56.5	76.8	78.1×30
SRTG-101 (2+1)D [30]	HACS	56.8	77.4	163.1×30
MTNet$_M$ (**ours**)	HACS	58.4	77.6	8.8×30
MTNet$_L$ (**ours**)	HACS	**63.3**	**84.1**	17.6×30

Table V. **Comparison with HACS state-of-the-art.** Weight initialization sources are denoted by their respective indicators.

Model	Pre-train	top-1	top-5	GFLOPs × views	Params
MF-Net [18]†	K-400	78.3	94.6	11.1×50	8.0M
TAM [39]†		82.2	95.2	93.4×9	25.0M
SF-101 [23]†		83.7	96.8	213×30	53.7M
X3D-L [20]†		85.8	96.1	24.8×30	6.1M
R3D-101 [16]*	K-700	80.5	95.8	78.0×30	69.0M
R(2+1)D-101 [16]†		82.9	95.6	163.0×30	72.1M
ir-CSN-101 [19]†	IG65	83.8	93.8	63.6×10	22.1M
ip-CSN-101 [19]†		84.1	93.9	63.6×10	24.5M
SRTG-101 (3D) [30]†	-	81.6	96.3	78.1×30	107.1M
SRTG-101 (2+1)D [30]†	-	84.3	**96.8**	163.1×30	105.3M
MTNet$_S$ (**ours**)	-	80.7	95.2	5.8×30	25.8M
MTNet$_M$ (**ours**)	-	83.4	95.9	8.8×30	25.8M
MTNet$_L$ (**ours**)	-	**86.6**	96.7	17.6×30	50.1M

† models and weights from authors' repositories.
* models and weights that we re-trained.

256×256. The inference time is then reported as the number of FLOPs per spatio-temporal view (clips times crops). This provides a standardized measure of computing inference when comparing across models as shown for Tables I, IV and V.

In the case of architectural changes such as in Table VIb we additionally report the inference time in terms of computational latency (in msecs.) for forward and backward passes, independently. Computational latency times are calculated on single clips of size 16×256×256.

D. Main Results

We discuss the comparisons of our MTNets to the current state-of-the-art for datasets K-400, MiT, K-700, and HACS.

Kinetics-400 (K-400). We present results in Table I. In comparison to the top performing X3D$_{XL}$ [20], our largest **MTNet$_L$** produces comparable performance (1.0% top-1 and 0.7% top-5 lower accuracies), despite a considerable reduction in computation of ×2.75 in terms of GFLOPs. When comparing **MTNet$_L$**, we observe performance on par with the significantly larger SlowFast-101 [23] which requires more than 12 times the number of computations. Notably, **MTNet$_L$** outperforms the similarly complex MFNet [18] with +5.3% top-1 and +2.8% top-5 accuracies. The trade-off between the achieved accuracies, computational complexity (GFLOPs) and parameter number is visualized in Figure 4.

For the smaller **MTNet$_M$** we report accuracies close to *Channel-Separated Network* (ip-CSN-101) [19] and *Temporal Adaptive Module* ResNet-50 [39], while being significantly more efficient than both. Considering its low number float-

ing point operations (FLOPs), **MTNet$_M$** can still outperform R(2+1)D ResNet101 [17], *Temporal Shift Module* (TSM) [21] and *Squeeze and Recursion Temporal Gates* (SRTG) [30].

Finally, our smallest network **MTNet$_S$** performs on par with TSM and SRTG, while having the lowest number of GFLOPs of all tested networks.

To better understand the relative contribution of network architecture and convolution operator, we compare a variety of spatio-temporal convolutional methods on the same ResNet-50 architecture. As shown in Table II, a direct replacement to MTConvs can yield a significant performance improvement over 3D convolutions with +13.5% in top-1 and +8.2% in top-5 accuracies. MTConvs also outperform other popular spatio-temporal convolution-based methods. MTConvs reduce the number of GFLOPs by 56% in comparison to regular 3D convolutions. The decrease in FLOPs for MTConvs does not come at a cost of parameters. MTNets include only a slightly reduced number of parameters in comparison to 3D Convs, which allows the models to preserve the level of complexity.

Moments in Time (MiT). Table III summarizes performance in terms of the top-1 and top-5 accuracies of current state-of-the-art models. Comparisons are performed on models with fixed-sized architectures as well as those that employ Neural Architecture Search (NAS) [41]. Our best performing architecture **MTNet$_L$** outperforms current state-of-the-art models with top-1 accuracy of 35.2%. Notably, these comparisons also include models with supplementary inputs optical flow [6] and audio [6] while both MTNet architectures are trained only on RGB frames. The smaller **MTNet$_M$**

Table VI. **Ablation studies on HACS.** We evaluate MTNet architectures under different training parameters and report top-1 and top-5 accuracies as well as the number of GFLOPs and parameters.

(a) **Branch channel ratio**: Varying channel ratio (δ) across MTNet$_M$ and MTNet$_L$ architectures.

Net.	δ setting	top-1	top-5	GFLOPs	Params (M)
MTNet$_M$	$\delta = 1.0$ (No $\mathcal{P}$)	82.2	93.6	10.8	29.7
	7/8	**83.4**	**95.9**	8.8	25.8
	3/4	83.1	95.6	6.7	21.8
	5/8	81.6	93.2	4.8	19.3
	1/2	79.7	91.8	3.6	**18.6**
	3/8	78.6	89.4	2.6	19.2
	1/4	77.1	88.6	**2.1**	21.0
MTNet$_L$	$\delta = 1.0$ (No $\mathcal{P}$)	84.9	95.7	20.6	53.5
	7/8	**86.6**	**96.7**	17.6	50.1
	3/4	86.1	96.2	12.5	45.3
	1/2	83.2	95.3	7.09	**42.7**
	3/8	82.1	93.9	5.2	45.3
	1/4	80.3	92.4	**4.1**	47.8

(b) **Recurrent cell configurations:** Alternative recurrent cells for the *global aggregated feature importance branch* ($\mathcal{G}$) with $\delta = 7/8$.

Net	Cell type	Params (M)	FLOPS (G)	Latency (msec) ↓F	Latency (msec) ↑B	top-1	top-5
MTNet$_S$	RNN [45]	24.3	5.8	58	78	78.8	93.7
	LSTM [46]	26.5	5.8	61	79	79.9	94.3
	LSTM (peephole) [47]	26.5	5.8	68	85	80.1	94.5
	GRU [34]	25.8	5.8	65	80	**80.7**	**95.2**
MTNet$_M$	RNN [45]	24.3	8.8	84	113	82.5	94.8
	LSTM [46]	26.5	8.8	86	109	83.1	95.4
	LSTM (peephole) [47]	26.5	8.8	94	120	83.2	95.6
	GRU [34]	25.8	8.8	90	111	**83.4**	**95.9**

(c) **Spatio-temporal pooling methodology:** Top-1 accuracy for different pooling methods in the *prolonged branch* ($\mathcal{P}$).

Net	Pooling strategies					
	Avg	Max	Stochastic [48]	SoftPool [33]	Avg + cos	SoftPool [33] + cos
MTNet$_S$	77.8	75.9	76.8	77.8	80.5	**80.7**
MTNet$_M$	79.8	77.6	78.2	80.7	82.6	**83.4**
MTNet$_L$	83.8	82.1	82.9	84.2	85.9	**86.6**

achieves a similar classification accuracy compared to learned architectures such as AssembleNet [40]. This comes with a reduction in terms of computations as there is no additional objective to permute the base model.

Kinetics-700 (K-700). We further evaluate our MTNets and their generalization capabilities on the recently introduced 700-class variant of Kinetics. As shown in Table IV, our architectures demonstrate similar performance trends as the accuracy values reported in Tables I and III. Specifically, we observe that **MTNet$_L$** outperforms other methods by a significant margin of +(4.7-6.8)% for top-1 accuracy and +(2.4-7.3)% for top-5. Again, **MTNet$_M$** performs similar to I3D [5] with a strongly reduced number of GFLOPs.

HACS. Finally, we present results on HACS in Table V. The datasets on which the models have been pre-trained are included in the table. Note that MTNets are trained on HACS from scratch. As shown, the use of MTConvs improves the overall accuracy. Notably, **MTNet$_S$** performs similarly to both R3D-101 [16] and SRTG-101 (3D) [30]. **MTNet$_M$** provides overall higher performance with additional +2.7% and +0.7% top-1 and top-5 accuracies over the smaller counterpart **MTNet$_S$**, while achieving similar accuracies as SlowFast-101 and ir-CSN-101. Finally, comparing **MTNet$_L$** to X3D-L shows an improvement of +0.8% for the top-1 and +0.6% top-5 accuracies while having $\sim$29% fewer GFLOPs.

E. Ablation Studies

In this section we present ablation studies on the HACS dataset. We compare different ratios (δ) used by the local ($\mathcal{L}$) and prolonged ($\mathcal{P}$) branches. We additionally evaluate the effect of different recurrent cells on the global aggregated feature importance branch ($\mathcal{G}$). Finally, we present results based on different spatio-temporal pooling methods in $\mathcal{P}$.

Branch channel ratio. As shown in Table VIa, the best performing ratios (δ) are within the range of (0.875 $\sim$ 0.75) with marginal differences in the range of $\pm(0.3 \sim 0.5)\%$ for both the top-1 and top-5 performances. These ratios also lead to a reduction in computational costs and the number of parameters. Improvements on number of computations (GFLOPs) based on these ratios are shown by the reduction of (25 $\sim$ 37)% when using both $\mathcal{L}$ and $\mathcal{P}$ branches, compared to using solely the local branch ($\mathcal{L}$) which is equivalent to a single standard 3D Conv. We attribute the loss in performance when using small ratios to the dependency of branch $\mathcal{P}$ on branch $\mathcal{L}$. Interestingly, the decrease in feature dimensionality of the local features with the use of smaller δ values corresponds to the inability of the prolonged features to encapsulate substantial video action details by themselves. In addition, decreases in δ do not directly relate to decreases in the number of parameters as seen in Table VIa. For $\delta < 1/2$, the number of parameters increases again with branch $\mathcal{P}$ employing a larger number of parameters. Therefore, the smallest number of parameters is observed when the ratio is split equally between the two channels ($\delta = 1/2$). This setting shows the largest combined reduction of GFLOPs (-66%) and number of parameters (-37%) for a standard 3D Conv. Lastly, we note that zero ratios $\delta = 0$ are not feasible as branch $\mathcal{P}$ includes the outputs of branch $\mathcal{L}$ which thus cannot be omitted.

Recurrent cell configuration. Next, we study the effect that the recurrent cell methodology used in $\mathcal{G}$ has on the accuracy. Recurrent layers are replaced in **MTNet$_S$** and **MTNet$_M$** with the changes only affecting branch $\mathcal{G}$. Latencies are calculated as the time (in msecs.) required for a full forward ($\downarrow$F) and backward ($\uparrow$B) pass for a single clip of size $16 \times 256 \times 256$. Results appear in Table VIb. The proposed use of GRUs [34] is motivated by the (slight) improvements over alternative

Table VII. **Transfer learning performance on UCF-101:**
Top-1 and top-5 accuracies after pre-training.

Model	Pre-training	top-1 (%)	top-5 (%)
I3D	K-400	92.4	97.6
TSM	K-400	92.3	97.9
ir-CSN-152	IG65M	95.4	99.2
MF-Net	K-400	93.8	98.4
SF-50	ImageNet	94.6	98.7
SF-101	ImageNet	95.8	99.1
SRTG-101 (2+1)D	HACS+K-700	97.2	99.1
SRTG-101 (3D)	HACS+K-700	97.3	**99.6**
MTNet$_S$ **(ours)**	HACS	94.2	98.0
MTNet$_M$ **(ours)**	HACS	95.4	98.1
MTNet$_L$ **(ours)**	HACS	**97.4**	99.2

recurrent cell structures. For **MTNet$_S$**, GRUs perform better than regular RNN cells [45] with +1.9% top-1 and +1.5% top-5 accuracies. However, the overall simplicity of RNNs can be more efficient in terms of parameter use with a -8% overall network parameter reduction as well as marginally lower forward and backward latency times. A similar observation is made for **MTNet$_M$** as GRU's top-1 and top-5 accuracies improve the RNN baseline by +0.9% and +1.1% respectively. Compared to LSTMs [46] and LSTMs with peephole variants [47], GRUs also show marginally better accuracy rates. We note that a property of GRUs is the merge of LSTM's *forget* and *input* states as well as their *cell* and *hidden* states. This simplifies the recurrent structure.

Spatio-temporal pooling methodology. We conclude our ablation studies by exploring the effect of different pooling methods used for $\mathcal{P}$ branch's inputs. Experiments were performed with temporal and spatial symmetric and asymmetric methods. In the first category, operations are performed similarly in all dimensions while the latter methods perform spatial and temporal pooling independently. In Table VIc, we report the top-1 accuracies for different pooling configurations. Frame selection with the proposed temporal triplet cosine (*cos*) similarity yields overall improvements over symmetric methods. For average pooling with triplet *cos*, accuracies are improved by +2.7% for $MTNet_S$, +2.8% for $MTNet_M$ and +2.1% for $MTNet_L$. Similarly, using SoftPool [33] and triplet *cos* increases top-1 accuracy, by +2.9%, +2.7% and +2.4% for each of the models respectively, in comparison to symmetric SoftPool. The improvement of SoftPool asymmetrically compared to average pooling is only marginal with +0.57% improvement on accuracy on average across the three architectures. We thus conclude that the temporal dimensionality reduction method has a significantly larger effect on the overall performance than the selection of a spatial pooling method. Temporal reductions with asymmetric methods such as our combined SoftPool with triplet cosine similarity show greater accuracy gains than symmetric methods that apply a pooling operation across all dimensions.

F. Feature Transferability with MTNets

Finally, we compare the transfer learning capabilities of MTNets with state-of-the-art video models on the smaller action recognition dataset UCF-101. To allow a fair comparison with other methods, all tested architectures are initialized with weights as in Table V. $MTNet_L$ achieves performance comparable to that of SRTG-101 (2+1)D which has been pre-trained on both large-scale HACS and K-700 datasets. The second model $MTNet_M$ can also perform as well as the ir-CSN that used a 65M dataset sourced from Instagram [49] and the 101-layer variant of SlowFast. Our smallest architecture $MTNet_S$ also shows good performance with accuracies above those of TSM and I3D while being similar to SlowFast-50. These findings further show the generalization capabilities of our spatio-temporal feature extraction approach.

V. CONCLUSIONS

We have introduced a novel multi-temporal convolution (MTConv) block that models variations in the performance in action videos by extracting and aligning spatio-temporal patterns across temporal scales. Our proposed convolution block uses two branches to address motions performed within a short and prolonged time span, respectively. A third global aggregated feature importance branch aligns the output activations of the first two branches based on the discovered feature dynamics. With this mechanism, we can extract salient spatio-temporal patterns despite potential differences in the temporal execution. We have also introduced MTNets that include MT-Convs in a X3D backbone. MTNets achieve comparable or, in many cases, higher classification accuracies than current state-of-the-art models on the most widely used action recognition benchmarks. Importantly, MTNets achieve a reduction in terms of computation costs. Based on these results, we believe that the modeling of variable-duration spatio-temporal patterns can be more widely exploited in future research in video action recognition for both real time and post-video processing.

VI. ACKNOWLEDGMENTS

REFERENCES

[1] A. Stergiou and R. Poppe, "Analyzing human-human interactions: A survey," *Computer Vision and Image Understanding*, vol. 188, p. 102799, 2019. 1

[2] D. W. Dong and J. J. Atick, "Statistics of natural time-varying images," *Network: Computation in Neural Systems*, vol. 6, no. 3, pp. 345–358, 1995. 1

[3] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013. 1

[4] R. R. Vallacher and D. M. Wegner, "Action identification theory," *Handbook of theories of social psychology*, vol. 1, pp. 327–349, 2011. 1

[5] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the Kinetics dataset," in *Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4724–4733. 2, 5, 6, 7

[6] M. Monfort, A. Andonian, B. Zhou, K. Ramakrishnan, S. A. Bargal, T. Yan, L. Brown, Q. Fan, D. Gutfreund, C. Vondrick, and A. Oliva, "Moments in time dataset: One million videos for event understanding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 502–508, 2019. 2, 5, 6

[7] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, "A short note on the Kinetics-700 human action dataset," *arXiv preprint arXiv:1907.06987*, 2019. 2, 5

[8] H. Zhao, A. Torralba, L. Torresani, and Z. Yan, "HACS: Human action clips and segments dataset for recognition and temporal localization," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 8668–8678. 2, 5

[9] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012. 2, 5

[10] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 568–576. 2

[11] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 3468–3476. 2

[12] 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 (ECCV)*, 2016, pp. 20–36. 2

[13] A. Diba, V. Sharma, and L. Van Gool, "Deep temporal linear encoding networks," in *Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2329–2338. 2

[14] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in *International Workshop on Human Behavior Understanding (HBU)*, 2011, pp. 29–39. 2

[15] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet?" in *Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 18–22. 2, 6

[16] H. Kataoka, T. Wakamiya, K. Hara, and Y. Satoh, "Would mega-scale datasets further enhance spatiotemporal 3D CNNs?" *arXiv preprint arXiv:2004.04968*, 2020. 2, 6, 7

[17] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6450–6459. 2, 5, 6

[18] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng, "Multi-fiber networks for video recognition," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 352–367. 2, 5, 6

[19] D. Tran, H. Wang, L. Torresani, and M. Feiszli, "Video classification with channel-separated convolutional networks," in *International Conference on Computer Vision (ICCV)*. IEEE, 2019, pp. 5552–5561. 2, 5, 6

[20] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 203–213. 2, 3, 5, 6

[21] J. Lin, C. Gan, and S. Han, "TSM: Temporal shift module for efficient video understanding," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 7083–7093. 2, 5, 6

[22] S. Sudhakaran, S. Escalera, and O. Lanz, "Gate-shift networks for video action recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 1102–1111. 2

[23] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 6202–6211. 2, 5, 6

[24] Z. Qiu, T. Yao, C.-W. Ngo, X. Tian, and T. Mei, "Learning spatio-temporal representation with local and global diffusion," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 056–12 065. 2

[25] Y. Chen, H. Fan, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, S. Yan, and J. Feng, "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 3435–3444. 2

[26] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141. 2

[27] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 9401–9411. 2

[28] 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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7834–7843. 2

[29] L. Wang, W. Li, W. Li, and L. Van Gool, "Appearance-and-relation networks for video classification," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1430–1439. 2

[30] A. Stergiou and R. Poppe, "Learn to cycle: Time-consistent feature discovery for action recognition," *Pattern Recognition Letters*, vol. 141, pp. 1–7, 2021. 2, 4, 5, 6, 7

[31] Z. Liu, L. Wang, W. Wu, C. Qian, and T. Lu, "TAM: Temporal adaptive module for video recognition," *arXiv preprint arXiv:2005.06803*, 2020. 2

[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456. 3

[33] A. Stergiou, R. Poppe, and K. Grigorios, "Refining activation downsampling with SoftPool," *arXiv preprint*, 2021. 3, 4, 7, 8

[34] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734. 4, 7

[35] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10 428–10 436. 5

[36] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," *International Conference on Learning Representations (ICLR)*, 2017. 5

[37] C.-Y. Wu, R. Girshick, K. He, C. Feichtenhofer, and P. Krähenbühl, "A multigrid method for efficiently training video models," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 153–162. 5

[38] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: training ImageNet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017. 5

[39] Q. Fan, C.-F. Chen, H. Kuehne, M. Pistoia, and D. Cox, "More is less: Learning efficient video representations by big-little network and depthwise temporal aggregation," *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 5, 6

[40] A. Piergiovanni, A. Angelova, A. Toshev, and M. S. Ryoo, "Evolving space-time neural architectures for videos," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 1793–1802. 6, 7

[41] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *Internation Conference on Learning Representations (ICLR)*, 2017. 6

[42] M. S. Ryoo, A. Piergiovanni, M. Tan, and A. Angelova, "Assemblenet: Searching for multi-stream neural connectivity in video architectures," *Internation Conference on Learning Representations (ICLR)*, 2020. 6

[43] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 803–818. 6

[44] C. Li, Q. Zhong, D. Xie, and S. Pu, "Collaborative spatiotemporal feature learning for video action recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 7872–7881. 6

[45] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985. 7, 8

[46] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 7, 8

[47] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *International Joint Conference on Neural Networks (IJCNN)*, vol. 3, 2000, pp. 189–194. 7, 8

[48] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," in *International Conference on Learning Representationsm (ICLR)*, 2013. 7

[49] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8