Recur, Attend or Convolve? Frame Dependency Modeling Matters for Cross-Domain Robustness in Action Recognition

Sofia Broomé\textsuperscript{1}  Ernest Pokropek\textsuperscript{1}  Boyu Li\textsuperscript{1}  Hedvig Kjellström\textsuperscript{1,2}

\textsuperscript{1} KTH, Sweden sbroome,pokropek,boyul,hedvig@kth.se  \textsuperscript{2} Silo AI, Sweden

Abstract. Most action recognition models today are highly parameterized, and evaluated on datasets with predominantly spatially distinct classes. Previous results for single images have shown that 2D Convolutional Neural Networks (CNNs) tend to be biased toward texture rather than shape for various computer vision tasks\textsuperscript{17}, reducing generalization. Taken together, this raises suspicion that large video models learn spurious correlations rather than to track relevant shapes over time and infer generalizable semantics from their movement. A natural way to avoid parameter explosion when learning visual patterns over time is to make use of recurrence across the time-axis. In this article, we empirically study the cross-domain robustness of models with different frame dependency modeling (recurrent, attention-based or 3D convolutional). In order to enable a light-weight and systematic assessment of the ability to capture temporal structure, not revealed from single frames, we provide the \textit{Temporal Shape} dataset. We find that when controlling for performance and layer structure, convolutional-recurrent models show better out-of-domain generalization ability on the Temporal Shape dataset than 3D convolution- and attention-based models. Moreover, our experiments indicate that convolution- and attention-based models exhibit more texture bias on Diving48 than convolutional-recurrent models.

1 Introduction

One of the most fundamental questions when it comes to video understanding in computer vision is how to model the dependency over time. The method should ideally be able to associate between frames at different timesteps, and provide hypotheses about how regions in each frame correspond to shapes and objects in the world that move, change, but persist over time. With this knowledge, it can start to infer relationships at a higher level, such as object-object or agent-object relationships, and, for example, distinguish a suddenly appearing fine-grained motion. In spite of the essentiality of this question, it has almost disappeared from action recognition articles, in the race to improve on classification benchmarks. In this paper, we perform an empirical study of the cross-domain generalization abilities of video models, comparing models that recur, attend or convolve across the temporal dimension. The aim of the study is to investigate whether there are inherent differences in what these different models learn; our
hypothesis is that a model that represents a motion class focusing on the relevant temporal shapes in the video will be better at generalizing than a model which focuses more on spatial correlations.

The deep learning subfield of action recognition, i.e., learning to recognize action classes that extend over time, is steadily evolving. The best models are increasingly learning features which truly include the temporal dimension, instead of aggregating per-frame predictions. This is important, since there lies information in the inter-dependency of the frames, which might be lost when down-sampling each frame of the sequence before the temporal processing. Despite this positive development, video models lack robustness to domain shift [56,57]. It has been repeatedly shown [6,55] that the action recognition datasets which were most frequently cited during the 2010s (UCF-101 [46], HMDB [30], Kinetics [20], AVA [22], and YouTube8M [1]) exhibit significant spatial biases. This is likely connected to poor domain shift generalization, since overly relying on spatial cues rather than motion cues intuitively results in overfitting to one domain (e.g., certain backgrounds, viewpoints or similar actor appearances).

Contemporary state-of-the-art approaches to action recognition are predominantly either fully convolutional [5,15,16,18,55], combine convolutions with temporal sampling and fusion strategies [53,54,58], or, more recently, attention-based (Video Transformers) [4,20,28,48]. The sheer size of the models, typically more than 50M trainable parameters, gives them a strong capacity to learn in-domain patterns. As models grow larger, ever more resources are spent to train them. Furthermore, state-of-the-art models should display competitive benchmarking numbers on large-scale datasets, such as Kinetics-400 and Kinetics-600. It is questionable whether these benchmarks are suitable for temporal modeling, or rather for how large amounts of YouTube clips efficiently can be stored as weight representations.

At the same time, the reciprocal dependency between the hardware and software of standard graphics processing units (GPUs), on the one hand, and models requiring massive parallel computation for their training, on the other hand, are becoming ever more intertwined [38,27]. The question looms whether we have cornered ourselves in action recognition, in the expectancy to work on ever larger models, in industry as well as in academia.

Theoretical works [3,36,45] have indicated that overparametrization helps generalization, in that the local minima of the loss landscape for such models often are global. These studies are made on held-out data, but never on data with significant domain shift, to the best of our knowledge.

In our work, we investigate the empirical consequences of these trends for time-critical video modeling. Although less efficient to train on GPUs, recurrent video models have a more parameter-efficient approach per timestep, which may hinder over-reliance on texture cues, and promote learning the temporally relevant motion cues. For Video Transformers, increasing the sequence length is quadratic in memory, which may become prohibitive for long videos. When convolving across the time-axis, there is no non-linearity between timesteps: for a long enough sequence, relevant patterns risk simply being blurred out.
Advantages of recurrence have repeatedly been pointed out in neuroscience \cite{50}, as well as in recent work in machine learning \cite{33}. Recurrent models are critical in the only visual system that has been ‘solved’ to date – biological vision \cite{21,35,11,12,13,29,31,42,47}. The need to be economical with the use of trainable parameters, we hypothesize, creates incitement to learn better shape representations instead of texture representations. In turn, this allows for better generalization across datasets and in the wild. For contour detection, it was found that a model with recurrent dynamics was more sample-efficient and generalized better than a feed-forward model \cite{34}. This trend has, to the best of our knowledge, not yet been studied for the case of time-critical action recognition. The primary contributions of our paper are as follows:

- We conduct the first empirical study on how the choice of frame dependency modeling in action recognition affects cross-domain robustness. Along with this, we open the discussion on whether inductive biases in a spatiotemporal back-bone can have consequences for cross-domain performance.
- We introduce a lightweight dataset which allows for investigation of both temporal shape modeling ability and domain generalization, called the Temporal Shape dataset.
- We provide the first discussion and experiments on shape vs. texture bias (following the nomenclature by Geirhos et al. \cite{17}) in deep video models.
- We make segmentation based shape and texture versions of the Diving48 dataset public, allowing studies on whether a video model has learned to rely more on (temporal) shape or on texture.

2 Related Work

**Domain Shift in Action Recognition.** In \cite{7,56}, cross-domain datasets are introduced to study methods for video domain adaptation. Chen et al. \cite{7} propose to align temporal and spatial features across the domains, whereas Yao et al. \cite{56} propose to improve the generalizability of so called local features instead of global features, and use a novel augmentation scheme. Strikingly, however, all experiments in \cite{7,56} are based on features extracted frame-by-frame, by a ResNet \cite{24}, and aggregated after-the-fact, which means that they in effect do not handle spatiotemporal features. In the Kinetics-Gameplay benchmark introduced by \cite{7}, only such per-frame features are released, showing that the focus lies on curing domain-bias symptoms of CNNs rather than searching for better architectures from the ground up. Using frame-wise features saves large amounts of time and computation, but it avoids an essential aspect of video modeling. Our work is novel in that we study cross-domain robustness for different kinds of spatiotemporal features, and whether the nature of the features themselves affects this robustness. Different from the field of Domain adaptation, we are not proposing methods on top of base architectures to reduce existing domain shift, but rather study empirically which types of fundamental video models inherently seem to be more robust to it. In an important work by Yi et al. \cite{57}, benchmarks are introduced to study robustness against common video corruptions, evaluated
for spatiotemporal attention- and convolution-based models. Different from our work, the domain shift is restricted to data corruptions rather than the same classification task in a new domain, and recurrent models are not evaluated.

**Emphasis on Temporality in Action Recognition.** Many works have started to emphasize the importance of temporal modeling in deep learning, e.g. Hara et al. [23], as the field of video understanding is growing. Tran et al. [49] study what kinds of convolutional layers (2D, (2+1)D or 3D) are most optimal for training and performance. Early work by Pickup et al. [40] investigates whether a video model can discriminate clips played forward from those played backward. In [55], it is shown that the arrow of time can matter more or less on different datasets, and that an inflated convolutional model, the I3D [5], will learn or ignore this, depending on the dataset at hand. In [19], 3D CNNs are compared to classical LSTMs and to the authors’ proposed model, the Time-Aligned DenseNet, which has properties of both sequential and hierarchical models but does not share parameters across time. Three tasks to measure temporal modeling abilities are introduced: forward/backward prediction, classifying the next most likely frame, and classification of action templates (Something-something-v2 [21]). In [37], video features are compared between 3D CNNs and convolutional LSTMs (ConvLSTMs [44]), in terms of both spatial and temporal focus. Qualitative differences are found between what the two model types tend to use as evidence for classification decisions. Another work on explainability for video models is by Price et al. [41], but only one type of model, and its decisions, is studied (TRN [58]). We are connected to the work of Sevilla-Lara et al. [43], who discuss the risk that models with strong image modeling abilities may prioritize those cues over the temporal modeling cues. Analogously to Geirhos et al. [17], Sevilla-Lara et al. find that inflated convolutions tend to learn classes where motion is less important better, and that generalization can be helped by training on more temporally focused data (in correspondence to training on shape-based data in [17]). Different from our work, however, only fully convolutional models are studied and the focus is not on comparing models with fundamentally different approaches to frame dependency modeling.

### 3 Experiment Design

In this section, we give an overview of the methodology and experiments conducted on the two datasets of this study: Temporal Shape and Diving48.

**Main Idea.** In all of our experiments, we begin by training on one specific domain, and validate on a held-out dataset from the same domain. We select the model that performs best on the validation set, and then proceed to evaluate it on other domains that are different in some respects but share the same task. Following [2], the domain we train on will be referred to as the source domain, and the unseen domains that we evaluate on, as the target domains.

The aim of the paper is to study how principally different ways of modeling temporal dependency affect the out-of-domain generalization ability. In a nutshell, we will empirically compare different video models on a number of domain
transfer tasks. To measure cross-domain robustness, we heuristically define the **robustness ratio (rr.)** for one model as the ratio between its accuracy on a target domain and its best validation accuracy on the source domain. When the target domain corresponds to the true task, this number should ideally be close to one (higher is better).

**Method Common to All Experiments.** We ensure that all sequences are of equal length by uniform re-sampling of the frames, since a fixed input size is required for the input to both 3D CNNs and attention-based models. There are other frame sampling methods, which can be used as augmentation, or as informed priors (e.g., the TimeSformer only samples the middle frames during inference in [4]), but in our study we are comparing the basic functionality of models. No pre-training, dropout, or data augmentation is applied in our experiments, except for 50% chance of horizontal flipping of the clips on Diving-48. Code related to neural networks was written in PyTorch [39] using Lightning [14]. All experiments have been run using the same random seed.

### 3.1 Models

We will compare ConvLSTMs, 3D CNNs and Video Transformers, since these present three principally different temporal modeling approaches with varying types and degrees of inductive biases. As Video Transformer, we will use the TimeSformer [4], because it recently achieved state-of-the-art results on a number of action recognition benchmarks.

It is a challenging task to compare neural network models which have principally different architectures. In our work, we decided on controlling for three different factors: the layer structure (i.e., the number and expressivity of hierarchical abstractions), the performance on a particular dataset and the number of trainable parameters. The experiments were designed prior to running them, to keep the process as unbiased as possible.

**Convolutional LSTMs.** The ConvLSTM [44] layer functions like an LSTM layer [25], but with matrix multiplication replaced with 2D convolutions. This crucially means that they allow for the input to maintain its spatial structure, contrary to classical recurrent layers which require a flattened input. Frame dependency is modeled using recurrence, which introduces non-linearities between timesteps. Time can only flow forward, and parameters are shared across the time dimension. A ConvLSTM video model, in this work, is a model fully based on these types of layers, with a classification head on top.

**TimeSformer.** The TimeSformer (hereon, TimeSf) [4] is a Transformer [51] for video, relying entirely on self-attention mechanisms to model frame dependency. As in [10], each frame is first divided into patches, which are flattened. In their best performing model, used in our experiments, the attention is divided in space and time. Self-attention is applied both among the patches of one frame (spatial attention) and across patches located in the same positions across the temporal axis (temporal attention).
Fig. 1: The videos can be displayed on click in Adobe Reader or in the supplementary material. Example clip showing the four domains of the Temporal Shape dataset, for the class circle. In 2Dot and 5Dot, the circle is drawn with a square of width 2 and 5 pixels. In MNIST and MNIST-bg, the circle is drawn with a MNIST digit, w/ and w/o a Perlin noise background.

We use the TimeSformer-PyTorch library [52], mainly with the standard settings given in the repository. We note again that we do not use pre-training, advanced data augmentation, nor averaging over multiple predictions, which results in a lower performance on Diving48 for this model than its state-of-the-art results.

3D Convolutional Neural Networks. In a 3D CNN, time is treated as space, and thus the input video as a tube, across which we convolve smaller filter tubes. Convolution is a linear operation, and, in that sense, the order of frames that the 3D filter traverses does not matter. Instead, all non-linearities are applied hierarchically, between layers, which is how this model still can learn the arrow of time. Except for the 3D filters, its layer structure is typically similar to a 2D CNN, including batch normalization and pooling. This is also the case for the instances used in our study.

3.2 Experiments on the Temporal Shape Dataset

To be able to study both temporal modeling abilities and cross-domain-robustness in a light-weight manner, we propose the Temporal Shape dataset. It is a synthetically created dataset for classification of short clips showing either a square dot or a MNIST digit tracing shapes with their trajectories over time (Fig. 1). The dataset has five different trajectory classes (i.e., temporal shapes): circle, line, arc, spiral and rectangle. The task is to recognize which class was drawn by the moving entity across the frames of the sequence. The spatial appearance of the moving object is not correlated with the temporal shape class, and can thus not be employed in the recognition. In the first three domains from the left in Fig. 1 (2Dot, 5Dot, MNIST), the background is black, and in the last domain, the background contains white Perlin noise (MNIST-bg). The Perlin noise can be more or less fine-grained; scale is regulated by a random parameter $\sigma \in [1, 10]$. The dataset can be thought of as a heavily scaled-down version of an action template dataset, such as 20BN-Something-something-v2 [21], entirely stripped of appearance cues.
The dataset was generated using Python code which can be found in the supplementary and will be made public. The sequences consisted of 20 64x64 frames, in grey scale. Each of the five classes has different amounts of possible variation in their states. The shapes can have varying starting positions, starting angles, direction, size and speeds. In the experiments, 4000 clips were used for training and 1000 for validation (model selection), and 500 clips for evaluation only. The classes are sampled so as to obtain class balance. More details can be found in the supplementary.

**Experiments.** In the Temporal Shape experiments, we evaluate a range of models of different sizes for each main architecture. Overall, we use small models, and control for layer structure by letting the compared models have three layers each of analogous blocks, where the number of hidden units varies in each experiment. One block for the ConvLSTM and 3D CNN consists of a model-specific layer including activations (tanh for the ConvLSTM layer and ReLU for the 3D convolutional layer), a max pooling layer (2D pooling for ConvLSTM and 3D pooling for the 3D CNN), followed by batch normalization. The models have the same spatial filter sizes in the three layers (3x3). For TimeSf, we use one TimeSf layer as one block, and the latent dimension for each attention head, $D_h$, as the number of hidden units. Further details can be found in the code.

For each experiment, we increase the number of hidden units, $h$, in each of the model-specific layers, where $h \in \{1, 2, 3, 5, 8, 12, 16, 24, 32, 48\}$. For each of the ten experiments varying model sizes, we train the models on the five-class task on the source domain for 100 epochs, with ten epochs of early stopping patience. For TimeSf, these numbers were 300 and 100, because it was larger and required more epochs to train than the other two types of models. Then, we evaluate the model which performed the best on the source domain validation set on different target domains. Experiments were conducted in two 'directions', from less texture to more texture (training on 2Dot), and from more texture to less texture (training on MNIST-bg). In the TimeSf implementation, the $D_h$ parameter had to be an even number; TimeSf was thus excluded where $h \in \{1, 3, 5\}$.

Training on the temporal shape data is light-weight compared to real video data, and runs fast (in the minutes-range for the model sizes we evaluated) on one GPU card. We train with a batch size of 64.

### 3.3 Experiments on Diving48

Diving48 is a well-known dataset for fine-grained and time-critical action recognition. It consists of 18k short clips depicting dives from 48 classes. Successfully classifying these dives requires temporal modeling, since one needs to keep track of the movements of the divers and their order. The dataset is varied appearance-wise, in terms of both backgrounds and viewpoints, which may contain unknown biases. The same competition sites are present in both the training and test split, "to avoid biases for certain competitions", according to the authors. Instead, in our view, this in fact increases the risk for bias, since the ability to recognize a dive at an unseen site is never tested.
Modified Domains of Diving48. We modify the dataset into three new domains; two based on shape and one based on texture (following Geirhos et al. [17], Fig. 2). In the shape domains, we blur the background and only maintain the segmented diver(s) (S1), or their bounding boxes (S2). In the texture domain (T), we conversely mask out bounding boxes where the diver(s) are, and only keep the background. The masked boxes are filled with the average Imagenet [9] pixel value, following [8]. The class evidence should lie only in the divers’ movement; hence, the texture version should not contain any relevant signal, and the accuracy should drop to random performance. Thus, we can study how different models drop in score when tested on the shape or texture domain, indicating both cross-domain robustness (for S1 and S2) and texture bias (for T).

Instance Segmentation of Diving48. The procedure of segmenting divers in Diving48 is detailed in the supplementary. We release 303 manually labeled frames with instance segmentation (for single or double dives), since off-the-shelf networks fail at this task for the class Person, presumably because of the unusual shapes assumed in the air by the diver, or include people in the audience.

Training. We are deliberately avoiding bells and whistles when training models on Diving48. All three models are trained with the same SGD optimizer, cross-entropy loss, and a constant learning rate of 0.001. Each model is trained for 500 epochs maximally, with an early stopping patience of 30 epochs if the validation performance does not improve. The only data augmentation used is horizontal flipping of 50% probability for the entire clip. The models are trained using PyTorch Lightning’s ddp parallelization scheme across eight A100 GPUs, with a batch size of 8 and a clip length of 32 uniformly sampled frames, at 224×224.

Given that the purpose of our experiments is not to optimize classification performance, we evaluate the models at different levels of performance, ranging from 30% to 50% accuracy. Some of the advanced state-of-the-art methods today, including pre-training and heavy data augmentation, obtain up to 80% performance, but when the dataset was introduced in 2018, and standard video methods were tested off-the-shelf on it, the best result was 27% accuracy [32].
Thus, the range of 30-50% is reasonably well-performing, and well above random, which is at 2.1%.

**Experiments.** We conduct three different kinds of experiments on the Diving48 dataset across different performance levels, namely: control for layer structure and performance (experiments a-c), and control for performance for the best performing variants (experiment d), and control for number of parameters and performance (experiments e-h).

**a-c. Controlling for layer structure and performance.** In this experiment, we let the models have four layers (blocks) of abstraction, with 128 hidden units in each. As in the Temporal Shape experiments, we treat $D_h$ as the hidden unit analogy for the TimeSformer, since it is most similar in scale. We evaluate checkpoints of the models at different performance levels: 30%, 35%, and 38.3% accuracy. The last, and very specific accuracy, was chosen because it was the limiting, highest performance by the 3D CNN in this experiment. Having the same layer structure gives rise to a varying number of parameters for each type of model. Here, the 3D CNN has 10.6M parameters, ConvLSTM has 14.3M parameters, and TimeSformer has 85M.

**d. Controlling for performance only.** Here, we compare models at their best performance, after hyperparameter search. Since it was not possible to train TimeSformer to a higher accuracy than 39.7% in all variants we tried, this experiment was only conducted with the 3D CNN and ConvLSTM. A complete list of the variants we attempted with the TimeSformer can be found in the supplementary. The ConvLSTM had four blocks of 128 hidden ConvLSTM units each (14.3M parameters). The 3D CNN was an 11-layer VGG-style model (23.3M parameters). The checkpoints used were both at exactly 50.07% validation accuracy.

**e-h. Controlling for number of parameters and performance.** Here, we have chosen models with a similar amount of trainable parameters, in this case 14M. To arrive at this number of parameters for TimeSformer, its depth was reduced from 12 to 11, and $D_h$ and $D$ were halved, to 32 and 256, respectively, relative to the default model. The ConvLSTM has four blocks with 128 units in each, and the 3D CNN has six blocks with 128 units in each. Further details can be found in the supplementary.

## 4 Results and Discussion

Next, we discuss our empirical findings, first on the Temporal Shape dataset (Section 4.1), and then on Diving48 and its modified domains (Section 4.2).

### 4.1 Temporal Shape

**Training on 2Dot.** The results are shown in Fig. 3a (with details in the supplementary). Although the 3D CNN generally obtains higher results on the source domain validation set, the rr. is generally higher for ConvLSTM; it drops less compared to its original result when tested on unseen domains with increasing noisy texture.
Robustness ratio vs. model size. In Fig. 4 we have plotted the rr. for the three target domains when training on 2Dot. For 5Dot, the rr. for ConvLSTM decreases slightly with the model complexity, whereas the 3D CNN and TimeSf, in contrast, increase the rr. along with increased model complexity. In the MNIST domain, which is further from the validation domain, the upward trend for the 3D CNN and TimeSf is broken. For the most challenging domain, MNIST-bg, the rr. becomes very low for all three models with increased size. The development in Figs. 4 a-c points to how a larger model size can potentially be an obstacle for models to be robust to domain shift, when combined with a lack of sound spatiotemporal inductive biases.

Training on MNIST-bg. In this experiment, TimeSf was the most robust to the domain shifts (Fig. 3 b). Among the three models, it had the largest expressive capacity, ranging from 20k parameters for the smallest model, and to 9M for the largest model. The corresponding numbers for the other two models were from 600 to 433k (ConvLSTM), and from 700 to 157k (3D CNN). We hypothesize that this allowed TimeSf to learn to fully disregard the Perlin noise (which is highly stochastic and demanding to model) and learn the true temporal shapes, using its ability for long-term dependencies in space and time, and that this, in turn, allowed it to be unbiased in the other domains, since the training data contains no bias at all. In real-world data, however, the training data will always contain biases, and it is therefore best to construct models which inherently model as little bias as possible, regardless of the training data. Overall, when comparing Figs. 3 a and b, we observe that training on data with more texture (MNIST-bg) makes for a larger drop in accuracy when the domain shifts.

4.2 Diving48: Sensitivity to Shape and Texture

Table 4 shows the average results for the eight Diving48 experiments. Overall, we note that ConvLSTM drops the most for T, both relative to the validation...
Recur, Attend or Convolve?

![Graphs showing robustness ratio (rr.)](image)

Fig. 4: Robustness ratio (rr.) (↑) when training on 2Dot, vs. # hidden per layer. The target domain is progressively further away from the source in subplots a-c.

| Model          | S1/V ↑ | S2/V ↑ | T/V ↓ | T/S1 ↓ |
|----------------|--------|--------|-------|--------|
| 3D CNN         | 0.255 ± 0.026 | 0.260 ± 0.034 | 0.221 ± 0.028 | 0.878 ± 0.15 |
| ConvLSTM       | 0.230 ± 0.028 | 0.266 ± 0.026 | 0.185 ± 0.042 | 0.807 ± 0.16 |
| TimeSformer     | 0.175 ± 0.028 | 0.176 ± 0.026 | 0.190 ± 0.037 | 1.10 ± 0.17 |

Table 1: Average results for experiments a-h.

(T/V) and to the S1 (T/S1) accuracies. ConvLSTM is also most robust to the S2 dataset, whereas the 3D CNN is most robust to the S1 shape dataset.

**Experiments a-d.** In experiments a-c (Fig. 5), where we vary the validation accuracy on the source domain between 30% and 38.3% on a fixed architecture of four layers with 128 units each, both TimeSf and the 3D CNN performs better on T than on S1 and S2, which contain actual class evidence. Tables 4-6 show that T/S1 > 1 in all three experiments for these two models, also visible in Fig. 5. In Tables 5-6, TimeSf drops the least for T relative to the validation set, despite T/S1 > 1, explained by its overall large drops, rather than being robust to texture domain shift. In contrast, ConvLSTM clearly drops for T. TimeSf is large here, at 85M parameters, whereas the 3D CNN is interestingly quite small at 10.6M parameters. This suggests that it is not only the parameter count that causes susceptibility for overfitting, but possibly that there are innate tendencies to overfitting in the choice of spatiotemporal (frame dependency) modeling. A recurrent model necessarily takes each timestep into account, in the sense that each timestep is non-linearly registered in the hidden state. We hypothesize that this enables it to register motion changes over time more carefully, and count these as salient, when that is the case (as it should be for Diving48).

Tables 2 show a breakdown of the models’ predictions on five randomly selected clips from a randomly chosen class (34). Top-1 accuracy being equal between ConvLSTM and the 3D CNN, we note that ConvLSTM has 100% top-5 accuracy for both S1 and S2, whereas the 3D CNN has 80% and 60%. This becomes interesting when looking at the texture (T) results, where the top-5 acc. of the 3D CNN remains at 80%, whereas ConvLSTM drops to 60%. ConvLSTM is also more calibrated with lower loss results than the 3D CNN and TimeSf.
Each label of Diving48 has four attributes: takeoff, somersault, twist and flight position. Looking closer at the top-1 predictions (Table 3), we study how many attributes are correct in the misclassifications for each model. Class 34 has the attribute values inward takeoff, 2.5 somersault, no twist and tuck flight position. For ConvLSTM, the misclassifications of class 34 are 8, 20, 35 and 44, where 8, 35 and 44 contain 3/4 correct attributes, and 20 contains 1/4 correct attributes (no twist). For the 3D CNN, only two predictions (32, 35) obtain three correct attributes, and for TimeSf, the best misclassification has only two correct attributes. This suggests that the 3D CNN and TimeSf have modeled the classes in terms of the true attributes to a lesser extent than ConvLSTM, i.e., ConvLSTM has learned more relevant temporal patterns. More details and further examples can be found in the supplementary.

| Model  | S1 | Top1 | Top5 | Loss | S2 | Top1 | Top5 | Loss | T  |
|--------|----|------|------|------|----|------|------|------|-----|
| ConvLSTM | 0.4 | 1.0  | 1.55 | 0.4  | 1.0 | 1.77 | 0.0  | 0.6  | 2.34|
| 3D CNN  | 0.4 | 0.8  | 3.03 | 0.4  | 0.6 | 3.33 | 0.0  | 0.8  | 4.18|
| TimeSf  | 0.4 | 0.4  | 4.26 | 0.2  | 0.4 | 4.50 | 0.0  | 0.2  | 6.85|

Table 2: Qualitative example with predictions five random clips from class 34, made by the model instances from experiment c) (38.3% acc.).

| Model  | S1  | Top1 | S2  | Top1 | Misclassifications | Correct attr. |
|--------|-----|------|-----|------|-------------------|----------------|
| ConvLSTM | [34, 34, 35, 8, 20] | [34, 34, 44, 8, 20] | [34, 34, 44, 8, 20] | 8, 20, 35, 44 | 10/16 |
| 3D CNN  | [34, 19, 21, 35, 34] | [34, 32, 21, 34] | [34, 32, 21, 34] | 19, 21, 32, 35 | 8/16 |
| TimeSf  | [34, 12, 34, 47, 20] | [31, 12, 34, 47, 20] | [31, 12, 34, 47, 20] | 12, 20, 31, 47 | 5/16 |

Table 3: Top-1 predictions and misclassifications by the models, same qualitative example as in Table 2. Each class has four attributes, and the Correct attributes column shows how many attributes were correct for the misclassifications.

In experiment d, where we compare a ConvLSTM and a 3D CNN at 50% validation accuracy, the 3D CNN does not improve on the texture dataset relative to S1 and S2, but the drop is larger for ConvLSTM. To increase its validation performance, it was necessary to scale up the size of the 3D CNN – here, it has 23.3M trainable parameters, compared to the 14.3M parameters of ConvLSTM. The larger number of parameters might contribute to its relatively overfitted tendencies. However, this is the most realistic scenario to evaluate a 3D CNN, since the state-of-the-art versions typically are deep (>10 layers) and have more than 20M parameters (25M for I3D, 60M for SlowFast). This raises concern that when a large model optimizes its performance on one domain, it may have difficulties to be robust in others.

**Experiments e-h.** The results for experiments e-h, where the number of trainable parameters and performance are fixed, are shown in Fig. 6, with accompanying results tables in the supplementary. Here, the 3D CNN is the most robust
out of the three, although ConvLSTM approaches the 3D CNN and drops more steeply for the texture dataset in experiment h), where the performance is highest (45% acc.). TimeSf drops from 30% and 35% to almost 5% accuracy in experiments e-f, but the result is not better for the texture dataset here. This suggests that TimeSf is more likely to display texture bias when it has a larger amount of parameters, as it does in experiments a-d.
5 Conclusions and Future Work

Our study shows indications that using more inductive biases for temporal modeling, such as recurrence or convolutions, helps with cross-domain robustness in temporally challenging video classification tasks. An explanation supported by our findings is that such models display less texture bias than the TimeSformer. In particular, the convolutional-recurrent ConvLSTM model was the most cross-domain robust on the two datasets used for this study. Sharing parameters across timesteps, as recurrent models do, narrows the parameter space, possibly incentivizing it to prioritize which patterns to learn. Another reason to use smaller models is that they require less data to train, which is desirable from an ethical point of view, in that the data can be inspected more easily.

Our results suggest that the fundamental low-level choices we make for our architectures matter, and can have important consequences for certain high-level behaviors. We believe that it is worthwhile to revisit the basics of temporal modeling – how we extract spatiotemporal features, and ask ourselves which method seems to learn the most sensible patterns for a given task, before we resort to curing the symptoms of our video architectures with various sophisticated operations added on top, such as contrastive learning, advanced regularization schemes, methods for domain adaptation or training on ever larger datasets.

Our study is not exhaustive on this topic – the space in which these types of models can be compared is vast, and there likely are important modes of comparisons that we have left out. For instance, the influence of pre-training on different spatiotemporal backbones is left for future work. A study including pre-training is suitable to combine with evaluation on domain adaptation benchmarks such as [7,56], where the splits often are too small to train from scratch. We encourage other researchers to continue the investigation of what different spatiotemporal backbones learn; hopefully, this has been made easier by our provided datasets. Our code, the Temporal Shape dataset, segmented diving frames and the shape and texture versions of Diving48 will be made public.
Acknowledgements. The computations were enabled by the supercomputing resource Berzelius provided by National Supercomputer Centre at Linköping University and the Knut and Alice Wallenberg foundation. We further thank Marcus Klasson, Taras Kucherenko and Ci Li for helpful feedback and discussions.

A Implementation Details

Sampling with Replacement in the Temporal Shape Dataset. In the experiments, 4000 clips were used for training and 1000 for validation. The number of samples was chosen so as to be able to sample randomly with replacement, while still keeping the risk low that an identical clip occurs in both the training and the validation set. For the 2Dot-domain, each class has more than 30k possible variations (lower bounds: 31k circle, 34k line, 51k rectangle, 150k arc), except the spiral class which has 7200 as a lower bound on the possible variations. When the training set consists of 5000 samples in total, we generate around 1000 samples per class. For the spiral class, a frequentist estimation gives that \( \frac{800}{7200} = 0.11 \) of the 200 spiral validation samples might be present in the training split (22 clips). However, this is still an over-estimation, since the spirals sometimes bounce against the sides of the frame which gives rise to extra variation. We decided to consider this as acceptable noise of the dataset. Some amount of data leakage can be considered interesting since this may occur in standard datasets as well.

Instance Segmentation of Diving48. To segment divers, it did not suffice to apply a pre-trained network and use the class "Person", which we first attempted (DeeplabV3 pre-trained on MS-COCO, provided by PyTorch). First of all, the off-the-shelf model could often not recognize the divers in the air as the "Person" class – they can be up-side down, or assume strange shapes in the air. Secondly, the model would often detect pixels of the "Person" class in the audience, when there was audience visible, which we, naturally, did not want to include.

Thus, we resorted to labelling our own segmented frames from the dataset (no segmentation masks were available online). We manually labelled 303 frames from the dataset containing one or two divers, picked from 303 randomly chosen videos of the training split. When there were two divers, we segmented each as its own instance. The segmentation masks will be made public.

We fine-tuned a MaskRCNN on our labeled dataset, using a random split of 290 frames as training set and 13 frames to validate, and monitored the bounding box IoU on the validation set. The best model achieved 93% validation bounding box IoU, which we used to segment the frames of the entire dataset (at 32 frames per clip). We used the confidence of the mask predictions as a threshold. The non-zero predictions were mostly confined to a bounding box surrounding the diver(s). When the threshold was \( t = 0 \), bounding boxes around the divers were used as crops (S2). When increased to \( t = 0.4 \), we obtained proper segments of the diver shape (S1). The frames contain a lot of motion blur which made the segmentation more challenging, and the segmentation at \( t = 0.4 \) is not perfect.
– Sometimes parts of for example an arm or foot is missing. The performance of the segmentation at \( t = 0.4 \) was deemed sufficient after manual inspection of 100 randomly chosen videos, where all videos had enough evidence to recognize the development of the dive. The segmentation at \( t = 0 \) (bounding boxes, S2) was satisfactory in all 100 clips inspected.

B Further Qualitative Examples

Additional qualitative examples were chosen from three more randomly selected classes: 12, 22 and 45. As in the main article, for each class, we select five random clips for evaluation of the three different models (with their checkpoints from experiment c, at 38.3% accuracy). We note their top-1 and top-5 accuracy, their loss, and how many attributes they are correct in for each clip. In Tables 9 and 11 we see that the ConvLSTM achieves the largest proportion of correct attributes in the misclassifications also for these three classes. Similarly to the results in the main article, the 3D CNN comes second, and TimeSf last.

B.1 Class 12

See Tables 8-9.

| Model     | S1       | S2       | T         |
|-----------|----------|----------|-----------|
|           | Top1     | Top5     | Loss      | Top1     | Top5     | Loss      | Top1     | Top5     | Loss      |
| ConvLSTM  | 0.0      | 0.2      | 6.21      | 0.0      | 0.2      | 6.02      | 0.0      | 0.0      | 5.40      |
| 3D CNN    | 0.2      | 0.2      | 5.99      | 0.2      | 0.2      | 6.35      | 0.2      | 0.2      | 8.68      |
| TimeSf    | 0.2      | 0.6      | 4.67      | 0.2      | 0.4      | 4.81      | 0.0      | 0.2      | 6.22      |

Table 8: Qualitative example with predictions five random clips from class 12, made by the model instances from experiment c) (38.3% acc.).

| Model     | S1       | S2       | T         |
|-----------|----------|----------|-----------|
|           | Top-1 predictions | Misclassifications | Correct attr. |
| ConvLSTM  | [35, 26, 45, 26, 21] | [27, 26, 45, 14, 21] | 14, 21, 26, 27, 35, 45 | 14/24 |
| 3D CNN    | [3, 20, 12, 5, 44] | [3, 20, 12, 5, 34] | 3, 5, 20, 34, 44 | 8/20 |
| TimeSf    | [22, 33, 12, 31, 14] | [22, 33, 12, 31, 14] | 14, 22, 31, 33 | 5/16 |

Table 9: Top-1 predictions and misclassifications by the models, same qualitative example as in Table 8. Each class has four attributes, and the Correct attributes column shows how many attributes were correct for the misclassifications.

B.2 Class 22

See Tables 10-11.
Table 10: Qualitative example with predictions five random clips from class 22, made by the model instances from experiment c) (38.3% acc.).

Table 11: Top-1 predictions and misclassifications by the models, same qualitative example as in Table 10. Each class has four attributes, and the Correct attr. column shows how many attributes were correct for the misclassifications.

Table 12: Qualitative example with predictions five random clips from class 45, made by the model instances from experiment c) (38.3% acc.).

C Additional Experiments

C.1 Results with One Attentional Head for the TimeSformer

Fig. 7 and 8 show additional results when the TimeSformer is run with the number of attention heads, $A$, set to 1 (meaning that $D = D_h$). This makes the model considerably smaller than in the main article (where $A = 8$), with the number of parameter ranging from 819 for the smallest model to 153k for the largest model instead of from 20k to 8M (Table 17).

In Fig. 8, there is an outlier for the TimeSformer in each subplot (the green dot to the left). This one is around 1 for each experiment, simply because the small TimeSformer (877 parameters) could not obtain more than random performance on the validation set when training (Table 16). Hence, the results for all
Table 13: Top-1 predictions and misclassifications by the models, same qualitative example as in Table 12. Each class has four attributes, and the Correct attributes column shows how many attributes were correct for the misclassifications.

four domains were around random (20%), which makes the rr. unusually high, but not for a valid reason.

The interpretation of these results, where we stress that the results for the 3D CNN and the ConvLSTM are identical to Fig. 3 in the main article, is that the smaller TimeSformer was less robust than the others when training on 2Dot, but still more robust than the others when training on MNIST-bg, though its results were generally lower and with more variance when running with the smaller model.

D Detailed results for the Temporal Shape experiments

In Tables 14 and 15 all the best validation results and the target domain results are shown for each of the experiments. These results correspond to Fig. 3 in the main article.

Table 14: Test results (% accuracy) when training on 2Dot (2-v is the best validation result on the source domain 2Dot) and evaluating on the three unseen domains, while increasing the number of hidden units per layer.

| #h | 3D CNN | ConvLSTM | TimeSf |
|----|--------|----------|--------|
|    | 2-v    | M        | M-bg   |
| 1  | 70.7   | 53.6     | 46.6   |
| 2  | 81.0   | 33.6     | 27.0   |
| 3  | 85.3   | 45.2     | 28.6   |
| 5  | 86.0   | 63.8     | 33.2   |
| 8  | 93.2   | 81.2     | 48.6   |
| 12 | 93.5   | 70.6     | 45.2   |
| 16 | 94.1   | 75.4     | 44.0   |
| 24 | 93.9   | 82.2     | 48.4   |
| 32 | 93.8   | 70.0     | 41.8   |
| 48 | 92.4   | 84.6     | 31.2   |
| Avg. | 88.4 | 67.0 | 40.2 |
| Std. | ± 7.5 | ± 15.2 | ± 7.58 |

Avg. rr. ↑: 0.753 ± 0.458 ± 0.312
Std. rr.: ± 0.13 ± ± 0.097 ± 0.11

0.770 0.587 0.368
Table 15: Test results (% accuracy) when training and validating on MNIST-bg (M-bg-v is the best validation result on the source domain MNIST-bg), and on the three unseen domains, while increasing the number of hidden units per layer.

| #h | 3D CNN | ConvLSTM | TimeSformer |
|----|--------|----------|-------------|
|    | M-bg  | M-bg-v   | M-bg  | M-bg-v | M-bg  | M-bg-v | M-bg  | M-bg-v |
| 1  | 86.6  | 86.0     | 26.6  | 26.0   | 81.2  | 82.0   | 27.0  | 23.6   |
| 2  | 90.7  | 91.0     | 28.0  | 24.0   | 84.3  | 86.8   | 28.8  | 22.2   |
| 3  | 94.1  | 94.4     | 29.8  | 27.2   | 85.0  | 83.4   | 29.6  | 24.6   |
| 5  | 93.9  | 93.6     | 28.0  | 26.0   | 92.4  | 91.0   | 30.2  | 24.8   |
| 8  | 95.5  | 96.8     | 28.6  | 25.4   | 92.8  | 91.6   | 31.2  | 25.2   |
| 12 | 96.5  | 96.4     | 27.8  | 27.0   | 95.5  | 96.6   | 30.0  | 25.8   |
| 16 | 97.6  | 98.0     | 32.2  | 26.6   | 90.7  | 89.4   | 27.8  | 23.0   |
| 24 | 97.3  | 97.6     | 28.8  | 24.2   | 93.8  | 91.2   | 30.0  | 24.0   |
| 32 | 96.9  | 97.8     | 29.0  | 27.0   | 93.6  | 91.2   | 32.8  | 24.4   |
| 48 | 96.7  | 96.8     | 30.4  | 25.6   | 93.8  | 92.6   | 31.0  | 22.4   |
| Avg. | 94.6 | 94.8     | 28.9  | 25.9   | 90.3  | 89.6   | 29.8  | 24.0   |
| Std. | ±3.50 | ±3.82    | ±1.57 | ±1.13  | ±4.95 | ±4.39  | ±1.68 | ±1.19  |

Table 16: Results for the TimeSformer with $A = 1$, both training on 2Dot and training on MNIST-bg. The boldface digits means that they were the best robustness ratios when training on MNIST-bg (better than the 3D CNN, ConvLSTM and the TimeSformer-8 in Table 15). The asterisk, for the result when training on 2Dot and validation on MNIST-bg (5th column), is there to treat the result with caution, because the result is better (higher) from the random results for $\#h = 2$

| #h | TimeSformer-1 | TimeSformer-1 |
|----|---------------|---------------|
|    | 2-v | 5 | M | M-bg | M-bg-v | 5 | 2 |
| 2  | 20.5 | 22.6 | 20.2 | 22.4 | 22.1 | 19.2 | 19.2 | 18.4 |
| 8  | 77.7 | 26.2 | 23.8 | 17.6 | 76.1 | 75.8 | 30.6 | 24.6 |
| 12 | 81.6 | 40.6 | 27.2 | 17.6 | 80.2 | 80.4 | 29.2 | 24.4 |
| 16 | 71.7 | 34.2 | 30.0 | 19.4 | 85.3 | 82.8 | 38.6 | 34.6 |
| 24 | 75.8 | 44.6 | 36.0 | 21.0 | 88.8 | 85.8 | 40.0 | 34.8 |
| 32 | 71.8 | 41.6 | 41.2 | 20.0 | 88.2 | 85.6 | 34.8 | 24.6 |
| 48 | 80.0 | 37.8 | 29.2 | 18.0 | 89.6 | 90.2 | 39.4 | 37.6 |
| Avg. | 68.4 | 35.4 | 29.7 | 19.4 | 75.8 | 74.3 | 33.1 | 28.4 |
| Std. | ±21.5 | ±8.22 | ±7.11 | ±1.84 | ±24.2 | ±24.7 | ±7.47 | ±7.18 |

| Avg. rr. | 0.579 | 0.494 | 0.369* | 0.969 | 0.482 | 0.422 |
| Std. rr. | ±0.245 | ±0.235 | ±0.32 | ±0.047 | ±0.17 | ±0.19 |
Table 17: List of the number of trainable parameters for each model at each of the ten experiments on Temporal Shape, where the model complexity was increased (the number of hidden units per layer, for three-layer models). TimeSformer-8 and TimeSformer-1 designates $A = 8$ or $A = 1$, respectively, i.e., the number of attention heads per layer.

| Nb. parameters | # hidden per layer | 3D CNN | ConvLSTM | TimeSformer-8 | TimeSformer-1 |
|----------------|--------------------|--------|-----------|---------------|---------------|
| 1              |                    | 735    | 571       | -             | -             |
| 2              |                    | 1573   | 1497      | 20451         | 877           |
| 3              |                    | 2519   | 2783      | -             | -             |
| 5              |                    | 4735   | 6435      | -             | -             |
| 8              |                    | 8869   | 14613     | 265989        | 6373          |
| 12             |                    | 15893  | 30557     | 583301        | 12437         |
| 16             |                    | 24645  | 52261     | 1023493       | 20421         |
| 24             |                    | 47333  | 112949    | 2272517       | 42149         |
| 32             |                    | 76933  | 196677    | 4013061       | 71557         |
| 48             |                    | 156869 | 433253    | 8968709       | 153413        |

Table 18: List of the model variants used in the experiments a-h for Diving48. For the 3D CNN and ConvLSTM, the [x,y,z] lists designate the number of hidden units per layer (x for the first layer, y for the second, z for the third, etc), and the filter sizes lists similarly correspond to the filter size per layer.

| Experiment | 3D CNN | ConvLSTM | TimeSformer |
|------------|--------|----------|-------------|
| a          | Hidden [128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=4, $D = 1024$, $D_h = 128$ |
| b          | Hidden [128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=4, $D = 1024$, $D_h = 128$ |
| c          | Hidden [128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=4, $D = 1024$, $D_h = 128$ |
| d          | Hidden [32,64,128,128,128,256,256,512,512,512], Filter sizes [5,3,3,3,3,3,3,3,3,3,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=4, $D = 1024$, $D_h = 128$ |
| e          | Hidden [128,128,128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=11, $D = 256$, $D_h = 32$ |
| f          | Hidden [128,128,128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=11, $D = 256$, $D_h = 32$ |
| g          | Hidden [128,128,128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=11, $D = 256$, $D_h = 32$ |
| h          | Hidden [128,128,128,128,128,128], Filter sizes [7,7,5,3] | Hidden [128,128,128], Filter sizes [7,7,5,3] | Depth=11, $D = 256$, $D_h = 32$ |
Fig. 7: Average results (% acc.) across trials with varying numbers of hidden units per layer. The results for the 3D CNN and ConvLSTM are the same as in Fig. 3 in the main article, only the TimeSformer results are new, with a smaller model ($A = 1$). The shaded area corresponds to standard deviation across the trials. Detailed data for each experiment part of this figure for the 3D CNN and ConvLSTM can be found in Tables 14 and 15 and in Table 16 for the TimeSformer.

E TimeSformer variants attempted for training

Table 19 lists the different variants we tested when training on Diving48 from scratch. In all variants, the number of heads was 8 ($A = 8$), the patch size was $16 \times 16$, the learning rate was fixed at 0.001, and the weight decay was 0.00001. When SGD was used, the momentum was always 0.9.

F Model specifications for the Diving48 experiments

Table 18 lists the different model specifications for each of the eight experiments a-h on Diving48 in the main article. For further details on the models, this is described in the main article and in the code repository (included in the supplementary zip-file).

References

1. Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, A., Toderici, G., Varadarajan, B., Vijayanarasimhan, S.: Youtube-8m: A large-scale video classification benchmark. ArXiv abs/1609.08675 (2016)
Fig. 8: Robustness ratio (rr.) when training on 2Dot, vs. number of hidden units per layer. X-axis and legend are shared across the three plots. Higher is better.

2. Angelucci, A., Bressloff, P.C.: Contribution of feedforward, lateral and feedback connections to the classical receptive field center and extra-classical receptive field surround of primate V1 neurons. Progress in brain research 154, 93–120 (2006)

3. Belkin, M., Hsu, D., Ma, S., Mandal, S.: Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proceedings of the National Academy of Sciences 116(32), 15849–15854 (2019). https://doi.org/10.1073/pnas.1903070116, https://www.pnas.org/content/116/32/15849

4. Bertasius, G., Wang, H., Torresani, L.: Is Space-Time Attention All You Need for Video Understanding? In: Proceedings of the International Conference on Machine Learning (ICML) (July 2021)

5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)

6. Chen, C.F., Panda, R., Ramakrishnan, K., Feris, R.S., Cohn, J.M., Oliva, A., Fan, Q.: Deep analysis of cnn-based spatio-temporal representations for action recognition. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 6161–6171 (2021)
Table 19: List of attempted TimeSformer variants, trained from scratch on Diving48. $D$ and $D_h$ are parameters in the TimeSformer [4] architecture, attn. do. and ff.do are attention dropout and feed-forward network dropout, $T$ is the number of uniformly sampled frames that constitute the clip, and additioanl ll. means an additional linear layer on top of the predictions output from the TimeSformer model.

| Best val. Ep. | $D$ | $D_h$ | Depth | Attn. do. | Ff. do. | $T$ | Batch size | Optimizer | Additional ll. | Patience |
|--------------|-----|-------|-------|-----------|--------|-----|------------|------------|----------------|----------|
| 32.7         | 88  | 512   | 64    | 12        | 0      | 0   | 8          | 8          | SGD            | 1        | 30       |
| 31.5         | 84  | 512   | 64    | 12        | 0      | 0   | 8          | 8          | SGD            | 0        | 30       |
| 36.1         | 78  | 512   | 64    | 3         | 0      | 0   | 32         | 8          | SGD            | 0        | 30       |
| 39.7         | 122 | 1024 | 128   | 4         | 0      | 0   | 32         | 8          | SGD            | 0        | 30       |
| 31.1         | 76  | 512   | 64    | 12        | 0.1    | 0   | 8          | 8          | SGD            | 0        | 30       |
| 31.7         | 71  | 256   | 32    | 11        | 0      | 0   | 8          | 8          | SGD            | 0        | 30       |
| 36.5         | 85  | 256   | 32    | 11        | 0      | 0   | 32         | 8          | SGD            | 0        | 30       |
| 19.0         | 79  | 256   | 32    | 11        | 0      | 0   | 32         | 8          | Adam           | 0        | 30       |
| 31.7         | 75  | 256   | 32    | 11        | 0      | 0   | 8          | 32         | Adam           | 0        | 30       |
| 32.4         | 133 | 256   | 32    | 11        | 0      | 0   | 8          | 48         | SGD            | 0        | 30       |
| 36.5         | 85  | 256   | 32    | 11        | 0      | 0   | 32         | 8          | SGD            | 0        | 75       |

7. Chen, M.H., Kira, Z., Al-Regib, G., Yoo, J., Chen, R., Zheng, J.: Temporal attentive alignment for large-scale video domain adaptation. 2019 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 6320–6329 (2019)
8. Choi, J., Gao, C., Messou, J.C., Huang, J.B.: Why can’t i dance in the mall? learning to mitigate scene bias in action recognition. In: NeurIPS (2019)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition pp. 248–255 (2009)
10. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021). [https://openreview.net/forum?id=YicbFdNTTy]
11. Douglas, R., Koch, C., Mahowald, M.A., Martin, K., Suarez, H.E.O.: Recurrent excitation in neocortical circuits. Science 269, 981 – 985 (1995)
12. Douglas, R.J., Martin, K.A.C.: Recurrent neuronal circuits in the neocortex. Current Biology 17, R496–R500 (2007)
13. Fahrenfort, J.J., Scholte, H.S., Lamme, V.A.F.: Masking disrupts reentrant processing in human visual cortex. Journal of Cognitive Neuroscience 19, 1488–1497 (2007)
14. Falcon, W., Cho, K.: A framework for contrastive self-supervised learning and designing a new approach. arXiv preprint arXiv:2009.00104 (2020)
15. Feichtenhofer, C., Fan, H., Malik, J., He, K.: SlowFast networks for video recognition. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
16. Feichtenhofer, C., Fan, H., Xiong, B., Girshick, R.B., He, K.: A large-scale study on unsupervised spatiotemporal representation learning, 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 3298–3308 (2021)
17. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: Imagenet-trained CNNs are biased towards texture; increasing shape bias improves
accuracy and robustness. In: International Conference on Learning Representations (2019). https://openreview.net/forum?id=Bygh9j09KX
18. Ghadiyaram, D., Feiszli, M., Tran, D., Yan, X., Wang, H., Mahajan, D.K.: Large-scale weakly-supervised pre-training for video action recognition. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 12038–12047 (2019)
19. Ghodrati, A., Gavves, E., Snoek, C.G.M.: Video time: Properties, encoders and evaluation. In: British Machine Vision Conference (2018). https://ivi.fnwi.uva.nl/isis/publications/2018/GhodratiBMVC2018
20. Girdhar, R., Carreira, J., Doersch, C., Zisserman, A.: Video action transformer network. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 244–253 (2019)
21. Goyal, R., Kahou, S.E., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fründ, I., Yianilos, P.N., Mueller-Freitag, M., Hoppe, F., Thurau, C., Bax, I., Memisevic, R.: The “something something” video database for learning and evaluating visual common sense. 2017 IEEE International Conference on Computer Vision (ICCV) pp. 5843–5851 (2017)
22. Gu, C., Sun, C., Vijayanarasimhan, S., Pantofaru, C., Ross, D.A., Toderici, G., Li, Y., Ricco, S., Sukthankar, R., Büchner, C., Malik, J.: Ava: A video dataset of spatio-temporally localized atomic visual actions. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 6047–6056 (2018)
23. Hara, K., Ishikawa, Y., Kataoka, H.: Rethinking training data for mitigating representation biases in action recognition. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) pp. 3344–3348 (2021)
24. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition pp. 770–778 (2016)
25. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. 9(8), 1735–1780 (Nov 1997). https://doi.org/10.1162/neco.1997.9.8.1735
26. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Nasev, P., Suleyman, M., Zisserman, A.: The kinetics human action video dataset. CoRR abs/1705.06950 (2017). http://arxiv.org/abs/1705.06950
27. Kobie, N.: NVIDIA and the battle for the future of AI chips. Wired (2021). https://www.wired.co.uk/article/nvidia-ai-chips
28. Koot, R., Lu, H.: Videolightformer: Lightweight action recognition using transformers (2021)
29. Kreiman, G., Serre, T.: Beyond the feedforward sweep: feedback computations in the visual cortex. Annals of the New York Academy of Sciences 1464 (2020)
30. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: Hmdb: A large video database for human motion recognition. In: 2011 International Conference on Computer Vision. pp. 2556–2563 (2011). https://doi.org/10.1109/ICCV.2011.6126543
31. Lamme, V.A.F., Roelfsema, P.R.: The distinct modes of vision offered by feedforward and recurrent processing. Trends in Neurosciences 23, 571–579 (2000)
32. Li, Y., Li, Y., Vasconcelos, N.: Resound: Towards action recognition without representation bias. In: Proceedings of the European Conference on Computer Vision (ECCV) (September 2018)
33. Linsley, D., Ashok, A.K., Govindarajan, L.N., Liu, R., Serre, T.: Stable and expressive recurrent vision models. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan,
34. Linsley, D., Kim, J., Ashok, A., Serre, T.: Recurrent neural circuits for contour detection. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020), https://openreview.net/forum?id=H1gB4RVKvB
35. di Lollo, V., Enns, J.T., Rensink, R.A.: Competition for consciousness among visual events: the psychophysics of reentrant visual processes. Journal of experimental psychology. General 129 4, 481–507 (2000)
36. Ma, S., Bassily, R., Belkin, M.: The Power of Interpolation: Understanding the Effectiveness of SGD in Modern Over-parametrized Learning. In: ICML (2018)
37. Mänttäri*, J., Broome*, S., Folkesson, J., Kjellström, H.: Interpreting Video Features: a Comparison of 3D Convolutional Networks and Convolutional LSTM Networks. In: Computer Vision - ACCV 2020, 15th Asian Conference on Computer Vision. (*Joint first authors) (December 2020)
38. Milesi, A.: Accelerating SE(3)-Transformers Training Using an NVIDIA Open-Source Model Implementation. https://developer.nvidia.com/blog/accelerating-se3-transformers-training-using-an-nvidia-open-source-model-implementation/, accessed: 2021-11-01
39. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
40. Pickup, L.C., Pan, Z., Wei, D., Shih, Y., Zhang, C., Zisserman, A., Schölkopf, B., Freeman, W.T.: Seeing the arrow of time. In: IEEE Conference on Computer Vision and Pattern Recognition (2014)
41. Price, W., Damen, D.: Play fair: Frame contributions in video models. In: Proceedings of the Asian Conference on Computer Vision (ACCV) (December 2020)
42. Salin, P.A., Bullier, J.: Corticocortical connections in the visual system: structure and function. Physiological reviews 75 1, 107–54 (1995)
43. Sevilla-Lara, L., Zha, S., Yan, Z., Goswami, V., Feiszli, M., Torresani, L.: Only time can tell: Discovering temporal data for temporal modeling. 2021 IEEE Winter Conference on Applications of Computer Vision (WACV) pp. 535–544 (2021)
44. Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., chun Woo, W.: Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In: NeurIPS (2015)
45. Soltanolkotabi, M., Javanmard, A., Lee, J.: Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. IEEE Transactions on Information Theory 65, 742–769 (2019)
46. Soomro, K., Zamir, A.R., Shah, M., Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. CoRR (2012)
47. Supèr, H., Spekreijse, H., Lamme, V.A.F.: Two distinct modes of sensory processing observed in monkey primary visual cortex (v1). Nature Neuroscience 4, 304–310 (2001)
48. Tan, H., Lei, J., Wolf, T., Bansal, M.: VIMPAC: Video Pre-Training via Masked Token Prediction and Contrastive Learning (2021)

49. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 6450–6459 (2018)

50. van Bergen, R.S., Kriegeskorte, N.: Going in circles is the way forward: the role of recurrence in visual inference. Current Opinion in Neurobiology 65, 176–193 (2020). https://doi.org/https://doi.org/10.1016/j.conb.2020.11.009, https://www.sciencedirect.com/science/article/pii/S0959438820301768 whole-brain interactions between neural circuits

51. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 6000–6010. NIPS’17, Curran Associates Inc., Red Hook, NY, USA (2017)

52. Wang, H.: TimeSformer-PyTorch. Implementation of TimeSformer from Facebook AI, a pure attention-based solution for video classification. https://github.com/lucidrains/TimeSformer-pytorch, accessed: 2021-11-13

53. Wang, L., Tong, Z., Ji, B., Wu, G.: Tdn: Temporal difference networks for efficient action recognition. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1895–1904 (2021)

54. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Gool, L.V.: Temporal segment networks for action recognition in videos. IEEE Transactions on Pattern Analysis and Machine Intelligence 41, 2740–2755 (2019)

55. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.P.: Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In: ECCV (2018)

56. Yao, Z., Wang, Y., Wang, J., Yu, P., Long, M.: VideoDG: Generalizing Temporal Relations in Videos to Novel Domains. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–1 (2021). https://doi.org/10.1109/TPAMI.2021.3116945

57. Yi, C., Yang, S., Li, H., Tan, Y.P., Kot, A.C.: Benchmarking the robustness of spatial-temporal models against corruptions. In: NeurIPS (2021)

58. Zhou, B., Andonian, A., Oliva, A., Torralba, A.: Temporal relational reasoning in videos. European Conference on Computer Vision (2018)