Capturing Temporal Information in a Single Frame: Channel Sampling Strategies for Action Recognition

Capturing Temporal Information in a Single Frame: Channel Sampling Strategies for Action Recognition

Kiyoon Kim
kiyoon.kim@ed.ac.uk
Shreyank N Gowda
s.narayana-gowda@sms.ed.ac.uk
Oisin Mac Aodha
oisin.macaodha@ed.ac.uk
Laura Sevilla-Lara
lsevilla@ed.ac.uk

School of Informatics
University of Edinburgh
Edinburgh, UK

Abstract

We address the problem of capturing temporal information for video classification in 2D networks, without increasing their computational cost. Existing approaches focus on modifying the architecture of 2D networks (*e.g.* by including filters in the temporal dimension to turn them into 3D networks, or using optical flow, *etc.*), which increases computation cost. Instead, we propose a novel sampling strategy, where we re-order the channels of the input video, to capture short-term frame-to-frame changes. We observe that without bells and whistles, the proposed sampling strategy improves performance on multiple architectures (*e.g.* TSN, TRN, TSM, and MVFNet) and datasets (CATER, Something-Something-V1 and V2), up to 24% over the baseline of using the standard video input. In addition, our sampling strategies do not require training from scratch and do not increase the computational cost of training and testing. Given the generality of the results and the flexibility of the approach, we hope this can be widely useful to the video understanding community. Code is available on our website: https://github.com/kiyoon/channel_sampling.

1 Introduction

Understanding temporal information is crucial in order to understand video. While 2D convolutional filters are usually the standard approach to capture spatial information, there is a wider variety of methods for capturing temporal information. These include, for example, using the transformer architecture [2] 3D networks [19], computing optical flow and feeding it to a 2D convolutional network [17], using relational networks [27], or using Recurrent Neural Networks (RNNs) [28], among others. All of these methods require changes in the underlying network architectures, additional computational cost compared to simple 2D networks, as well as the time-consuming process of pre-training from scratch.

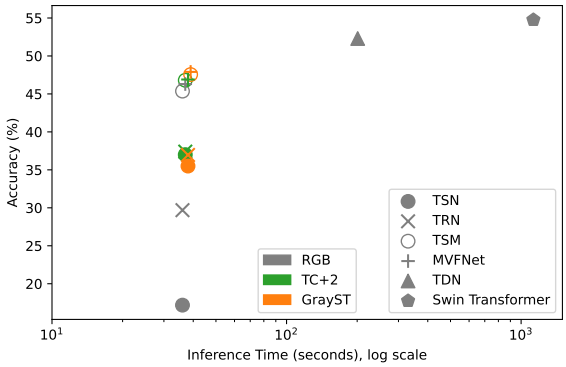


Figure 1: We show that it is possible to significantly increase the performance of lightweight action recognition networks on the challenging Something-Something-V1 dataset by simply adapting how the individual image channels are sampled. Our approaches, *TC+2* (in green) and *GrayST* (in orange), improve accuracy across several networks without increasing inference time. This enables us to narrow the gap between these efficient baselines and much more computationally demanding methods such as TDN [21] and Video Swin Transformers [14]. Note that we use a log scaling on the horizontal axis.

In this paper, we propose two simple and novel channel sampling strategies that improve the ability of a given 2D network to capture temporal information without changing the architecture. In particular, we re-order the channels of the input video in two ways: in the first, we re-order the channels of the video, so that each frame is composed of three channels: one that belongs to the frame before, one that belongs to the current frame, and one that belongs the frame after. These three channels are concatenated in the channel dimension as if it were a single frame. This procedure incorporates temporal information from neighboring frames, while keeping the dimensions of the input frame. In the second strategy, we compute a grayscale version of three neighboring frames at a time, and again concatenate them in the channel dimension, as if they were a single RGB image.

We test these extremely simple re-ordering strategies on five widely used 2D networks (TSN [20], TRN/MTRN [27], TSM [13], and MVFNet [24]). We observe that without any additional engineering these re-ordering strategies improve results up to 24% compared to the standard RGB channel ordering, across multiple challenging video datasets and different networks. We also observe that the improvement is particularly large in the datasets where temporal information is more important, in particular on CATER [7] and Something-Something [9]. Figure 1 illustrates the performance improvement which does not add additional significant computational overhead.

Our main contributions are:

- We improve the performance of simple 2D CNN-based action recognition models while incurring no additional computational complexity and with no modification to the underlying models. Our solutions can be used with most existing network architectures.

- Through extensive experimental evaluation on several common models and datasets, we show the efficiency of our proposed sampling methods and report superior performance compared to standard image sampling.

2 Related Work

Representations of Input Video. The most common representation of video frames for deep learning is 8-bit raw RGB, where each pixel has three color channels – red, green, and blue, each represented by 8 bits. This representation does not encode any temporal information in a frame. Another approach is to compute differential images by simply subtracting each pair of consecutive frames [20]. This gives the network a more explicit representation of temporal high-frequency changes. Unfortunately, this is not robust to spatial shifting, so as the camera pans, all pixel values change, which defeats the purpose of identifying the moving objects by subtracting the frames.

Optical flow is another common representation in action recognition [4, 17, 20]. Given two consecutive video frames, the optical flow is the direction and magnitude of the motion at each pixel. Using optical flow as a separate stream [17] generally improves performance, although it is often computationally expensive. Recent work has used the motion vectors from compressed video (*e.g*. H.264) to avoid the computational cost of optical flow [23, 26]. Given any of these representations, one of the standard approaches is to simply feed them as input to a 2D convolutional network, similar to those used in the image domain, where frames are processed independently, and the predictions are aggregated in the end. This family of approaches is called 2D, or frame-based networks. Despite their simplicity, they often work well on many datasets, and are fast to train and test.

Dynamic images [3] are a compact video representation that encode spatial and the entire motion information in a single image, so any image classification network can perform action recognition using the generated images. In contrast, our methods encode short-term temporal information per frame, resulting in space-time sequential representation (*i.e*. they do not compress an entire video into a single frame) and can thus use off-the-shelf 2D action recognition networks to better capture temporal information compared to only using a single image classification model as in [3].

3D Architectures. The main issue with 2D networks is that they are limited in their temporal receptive field size. 3D networks [19] on the other hand, benefit from learning spatio-temporal representations explicitly as they apply convolutions in both space and time. One of the most popular approaches is the I3D model [4] that inflates 2D models to 3D. As a result, they are able to make use of well established 2D architectures and pre-trained weights. Extensions to I3D include adding a non-local (NL) module [22] to help capture pixel-level correlations, resulting in improved performance. The SlowFast network [6] is also a widely used 3D network, and it uses two pathways, a slow and a fast one, to capture motion and fine temporal information. While 3D networks are helpful, they are much more expensive than their 2D counterparts. In this work we aim to introduce temporal information into 2D backbone-based models without requiring the computation of optical flow or without expensive 3D convolutions.

2D Architectures for Temporal Modeling. While 3D action recognition networks can perform better than their 2D counterparts, they have larger computational requirements for training and testing. This in turn, makes them challenging to deploy in resource-constrained settings, *e.g*. mobile and online scenarios. As a result, 2D networks for action recognition tasks is still common practice for certain application domains.

Temporal Segment Networks (TSN) [20] is an early and widely used 2D method. TSN simply extracts predictions for each frame sampled from the input video using a 2D backbone

network and then averages these predictions for the final output. Frames are sparsely sampled from longer videos so that the model can reason over long time spans. With an added optical flow stream, TSN achieves competitive performance on action datasets, even defeating some 3D networks at the time of release. This high performance is more obvious in datasets where there is a strong correlation between actions and static objects and scenes [11]. In contrast, TSN performs worse on datasets that require explicit temporal reasoning [9, 16].

Temporal Relation Networks (TRN) [27] aim to capture relational information across frames. For this, it also uses a 2D backbone to extract features from sparsely sampled frames, but then aggregates the frame-level features using a relational module [15]. The authors proposed a multi-scale relational module that learns 2-frame, 3-frame, and up to T-frame relationships (TRN-multiscale, or MTRN in short). This slightly improves performance over the single-scale TRN. This is an improvement over TSN, which is agnostic to the temporal ordering of the input frames. As a result, TRN improves performance on temporal datasets, *e.g.* [9]. However, their model is not very memory-efficient, as it requires $T-1$ relational modules (from 2 to T frame) using fully-connected layers, making it challenging to process many frames.

The Temporal Shift Module (TSM) [13] follows the same strategy as the TSN but modifies the ResNet [10] backbone so that the network can partially access information from one past and one future frame (*i.e.* temporal shift). In order to maintain the spatial feature learning capability, the temporal shift happens inside a residual branch, so the backbone choice is limited to those with residual connections. TSM achieved improved performance on the Something-Something-V2 dataset [9], showing that 2D models are not obsolete and are still competent in action recognition.

Multi-View Fusion Network (MVFNet) [24] adds channel-wise convolutions to model height-width, height-time, and width-time views instead of treating height-width-time video frames as a space-time signal. This approach performs better than TSM while still remaining computation efficient.

Finally, Temporal Difference Networks (TDN) [21] explicitly compute motion information by sampling five times as many input frames (*e.g.* 8-frame TDN uses 40 frames) and computes RGB differences for capturing short-term information and uses multi-scale attention for capturing long-term temporal information. TDN presented a new state-of-the-art on the Something-Something datasets, but at the cost of using significantly more input frames.

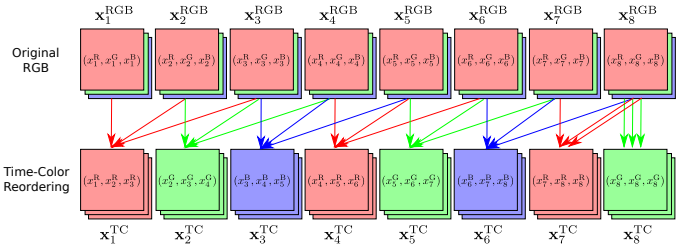
In this work, we take advantage of the efficiency and simplicity of 2D video models by proposing image channel sampling strategies that improve a model's ability to capture temporal information. This increases accuracy, all without introducing any additional computation during training or testing.

3 Method

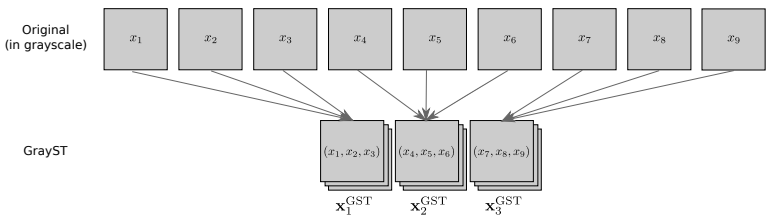
3.1 Overview

Given an input video v, the task of action recognition is to predict the action class label y of the video. Typically, the input video v can be sampled sparsely, which means that each video is evenly divided into T segments. Then, in the standard 2D paradigm, a frame $\mathbf{x}_t$ is chosen at random for each segment at training time. We call the set of sampled frames $\mathbf{X}$, *i.e.* $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$.

We let $f(\mathbf{x}_i)$ denote the 2D backbone model that takes a single image as input, and



(a) 8-frame TC Reordering



(b) 3-frame GrayST. The input video frames are converted to grayscale and the output is simply the concatenation of these frames into groups of three channels

Figure 2: Visualization of our TC Reordering (a) and GrayST (b) methods. In each case, the top row represents the original input frames, *i.e.* eight RGB frames in (a), and nine grayscale frames in (b). Note also that, GrayST gets to use three times as many input frames, *e.g.* to generate 8 output frames, one needs to sample 24 inputs.

outputs a feature map. We compute $f(\mathbf{X}) = \{f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_T)\}$. Finally, $g(\cdot)$ is a temporal aggregation module that takes the T feature maps and returns the output category prediction $\hat{y}$. The aggregation function $g(\cdot)$ can simply be an average of the individual predictions as in TSN [13, 20],

$$g(f(\mathbf{X})) = \frac{\sum_{t=1}^{T} h_\theta(f(\mathbf{x}_t))}{T}, \tag{1}$$

where h_θ is a per-frame classifier. In other cases, like TRN [27], the aggregation function reasons about the relationship among multiple frames by concatenating the features and applying a multi-layer perceptron (MLP) h_ϕ,

$$g(f(\mathbf{X})) = h_\phi(\text{concat}(f(\mathbf{X}))). \tag{2}$$

The multi-scale version of TRN (MTRN) works similarly but has several aggregation modules, each accounting for reasoning between 2-frame, 3-frame, ..., T-frame relationships.

Across variants of the 2D paradigm, the frames of a video are typically sampled sequentially with all color channels intact. Instead, we propose two alternative sampling strategies, *Time-Color Reordering* and *Grayscale Short-Term Stacking* that enables 2D backbone-based models to exploit temporal information by means of reordering the input channels. Despite their conceptual simplicity, these two sampling strategies significantly improve action recognition performance without requiring any structural changes to the backbone model.

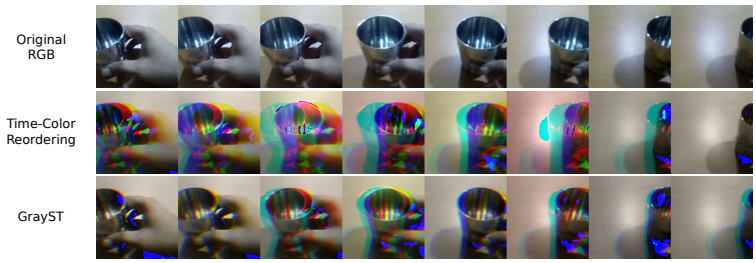


Figure 3: Visualization of the different sampling strategies. All frames are interpreted as if they were RGB. The video depicted is an instance of "Pulling something from left to right" from the Something-Something-V2 dataset. Our two sampling strategies make the motion and its direction clearly visible by utilizing three color channels.

3.2 Time-Color Reordering

We propose a simple yet powerful sampling technique for video frames called *Time-Color (TC) Reordering*. We allow the 2D backbone model $f(\mathbf{x}_i)$ to see temporal information by re-sampling the three color channels of the model. We do this by taking one color channel from the input clip (*i.e.* red) of 3 consecutive frames, and concatenating in the channel dimension to form an input "image". Then we repeat the process with the next color channel (*i.e.* green), and so on. Note that this is merely a different representation of the input video by means of changing the channel order, and does not require any modification to the backbone model or include additional information or processing. More specifically, the process is as follows. Let $\mathbf{X}^{\mathrm{TC}} = \{\mathbf{x}_1^{\mathrm{TC}}, \mathbf{x}_2^{\mathrm{TC}}, \ldots, \mathbf{x}_T^{\mathrm{TC}}\}$ denote a video clip sampled following the proposed TC Reordering sampling strategy, where

$$\mathbf{x}_i^{\mathrm{TC}} = \begin{cases} (x_i^{\mathrm{R}}, x_{i+1}^{\mathrm{R}}, x_{i+2}^{\mathrm{R}}), & \text{if } i \bmod 3 = 1. \\ (x_i^{\mathrm{G}}, x_{i+1}^{\mathrm{G}}, x_{i+2}^{\mathrm{G}}), & \text{if } i \bmod 3 = 2. \\ (x_i^{\mathrm{B}}, x_{i+1}^{\mathrm{B}}, x_{i+2}^{\mathrm{B}}), & \text{otherwise.} \end{cases} \tag{3}$$

Here, x_i^{R} is the red channel from the i-th frame, x_i^{G} for green, and x_i^{B} for blue. Since the last two TC frames try to access future frames (*i.e.* x_{T+1} and x_{T+2}) that are not available, we just use x_T with the corresponding color channel for this case. This simple process yields frames that contain information about the neighboring frames, and it is the core reason for the large advantage we observe in temporal tasks. Fig. 2(a) outlines the procedure in detail.

We alternate color channels to expose the network to more varied data. That is, different channels depict a particular object with different brightness and contrast, while keeping the original shape of the object. We observe that this color alternation produces better results than a single one, and speculate that it may work as a form of data augmentation.

We also propose an alternative method *TC+2* where we sample just two more frames to avoid the duplication of the last frames. The cost of sampling two more frames is almost negligible in many practical scenarios even with long-input networks, and we observe significant improvement of performance as the way the input data is formed is consistent throughout the frames.

3.3 Grayscale Short-Term Stacking

Here we introduce another sampling technique that we call *Grayscale Short-Term Stacking* (*GrayST*). This approach is designed to use more source frames with the same compact 2D networks by using grayscale images instead of RGB. The motivation is that in semantic understanding tasks, color information can be redundant, and we can use that capacity to include temporal information instead, by inputting more frames. Previous work [25] showed that there is only a 0.5% drop in accuracy on ImageNet [6], when training on grayscale images compared to RGB images. Thus, we replace the three color channels that are normally fed into a 2D backbone network with three grayscale frames, from three different sequential time steps. In effect, this allows the backbone model to see short temporal information at the expense of forgoing the ability to reason about color. This sampling strategy is visualized in Fig. 2(b).

When the input sequence length to a network is intended to be T frames, we simply sample $3 \times T$ frames in grayscale, and stack the three consecutive frames into one image, containing three grayscale channels. In offline scenarios, we can utilize higher temporal resolution per video clip without introducing any latency for training or testing. Some online scenarios may have a limited number of frames, so we also experiment with matching the number of input RGB frames, *i.e*. 8-frame GrayST (that sees 24 frames) vs 24-frame RGB.

We let $\mathbf{X}^{\text{GST}} = \left\{\mathbf{x}_1^{\text{GST}}, \mathbf{x}_2^{\text{GST}}, \ldots, \mathbf{x}_T^{\text{GST}}\right\}$ denote a GrayST video clip. We first sample $3 \times T$ grayscale frames following the same sparse sampling strategy as before,

$$\mathbf{X}^g = \left\{x_1^g, x_2^g, \ldots, x_{3T}^g\right\}, \tag{4}$$

where x_i^g is a grayscale image. Then, a GrayST frame is made of three neighboring temporal frames in $\mathbf{X}^g$ which is defined as

$$\mathbf{x}_i^{\text{GST}} = (x_{3i-2}^g, x_{3i-1}^g, x_{3i}^g). \tag{5}$$

A comparison between the two sampling techniques is visualized in Fig. 3.

4 Evaluation

In this section we evaluate our two channel sampling strategies on multiple different action recognition datasets using several different network architectures.

4.1 Experiment Details

Datasets. We experiment with challenging datasets that require extensive temporal reasoning. The datasets are chosen to span a wide range of situations, *e.g*. short and long range temporal reasoning, static and moving cameras, *etc*. CATER [7] is a synthetic action recognition dataset involving long-term temporal reasoning. CATER has two versions of the videos, with camera motion and without, and we experiment with both of them. CATER also consists of three different tasks: primitive multi-label action recognition (task 1), compositional multi-label action recognition (task 2), and localization of object of interest (task 3). We evaluate task 2, compositional action recognition, as it requires the longest temporal reasoning (from the beginning to the end of the videos consisting of 301 frames). This provides the biggest challenge for action recognition methods that only focus on short-term clips.

Model	Sampling	mAP	
		Static	Camera Motion
TSN	RGB	49.6	51.6
	TC	**73.7**	56.6
	TC+2	73.5	60.5
	GrayST	71.9	**61.9**
TRN	RGB	54.9	54.7
	TC	**72.4**	52.9
	TC+2	72.3	52.9
	GrayST	69.8	**57.6**
TSM	RGB	79.9	65.8
	TC	81.2	63.3
	TC+2	82.0	64.0
	GrayST	**82.2**	**74.7**
MVFNet	RGB	80.2	63.5
	TC	82.1	62.7
	TC+2	82.8	65.5
	GrayST	**83.4**	**67.8**

Table 1: Performance on CATER task 2 using 32-frame models ($T = 32$). We report last epoch's validation mAP using one clip. Note, GrayST uses three times as many frames as RGB or TC, which are then converted to grayscale, and TC+2 uses just two frames more. The different sampling strategies do not impact the size of the network, *i.e*. in the end, they all get the same number of input frames.

Specifically, CATER task 2 has 301 classes. Each class represents the ordering of two action pairs: an object performing an action before/during/after another object performing another action, for example, "cone slides before cylinder rotates". Furthermore, CATER allows containment of objects, and even recursion of it. That is, the action models have to remember which object is contained by which carrier, during which period, to successfully keep track of all actions happening in the scene.

We also evaluate on datasets that require temporal understanding such as Something-Something-V1 and Something-Something-V2 [9]. The Something-Something datasets consist of videos of pre-defined actions being performed using everyday objects. To focus on the action, and not the objects being used, these datasets removed object and scene bias by grouping the same action using various objects. The labels include "pushing something from right to left", "putting something on a surface", *etc*., which ensures the focus is on the action. Something-Something-V1 contains 108,499 videos with 174 class labels, while Something-Something-V2 consists of 220,847 videos of the same 174 classes.

Networks. We use a ResNet50 [10], pre-trained on ImageNet [5], as our 2D backbone model for all experiments. Both sampling strategies would likely benefit further from the backbone being pre-trained for their specific channel sampling strategies, but we leave this for future work. We mainly present results for five popular 2D models: TSN [20], TRN/MTRN [27], TSM [13], and MVFNet [24] with some additional experiments using I3D [4] pre-trained on Kinetics-400 [11] for completeness. Note, that we were not able to perform 32-frame MTRN experiments due to GPU VRAM limitations, as it has 31 temporal relational modules consisting of dense fully-connected layers. More details on the training settings can be found in the supplementary material.

(a) Something-Something-V1

Model	Sampling	Top1	Top5
TSN	RGB	17.2	42.7
	TC	36.8	65.3
	TC+2	**37.0**	**65.6**
	GrayST	35.5	65.4
TRN	RGB	29.7	57.6
	TC	35.9	65.2
	TC+2	**37.4**	**66.8**
	GrayST	37.0	66.1
MTRN	RGB	31.0	59.4
	TC	36.6	65.8
	TC+2	38.1	67.5
	GrayST	**38.4**	**67.8**
TSM	RGB	45.4	74.5
	TC	45.8	74.7
	TC+2	46.8	75.8
	GrayST	**47.6**	**76.7**
MVFNet	RGB	46.3	75.3
	TC	45.7	74.6
	TC+2	46.9	75.8
	GrayST	**47.9**	**76.6**
I3D*	RGB	40.5	68.5
	TC	39.1	68.6
	TC+2	40.8	69.2
	GrayST	**41.1**	**70.2**

(b) Something-Something-V2

Model	Sampling	Top1	Top5
TSN	RGB	30.2	60.5
	TC	**51.9**	**79.5**
	TC+2	51.3	79.4
	GrayST	50.9	79.4
TRN	RGB	45.7	73.6
	TC	51.3	78.9
	TC+2	**52.3**	**80.5**
	GrayST	52.1	79.9
MTRN	RGB	46.7	75.3
	TC	52.0	79.9
	TC+2	52.9	80.9
	GrayST	**53.0**	**81.1**
TSM	RGB	59.1	85.6
	TC	59.2	85.5
	TC+2	59.7	86.0
	GrayST	**59.8**	**86.2**
MVFNet	RGB	59.7	85.9
	TC	59.6	85.9
	TC+2	59.7	86.1
	GrayST	**60.8**	**86.6**

Table 2: Validation accuracy on Something-Something with 8-frame models ($T = 8$). Models marked with $*$ use 30 clips for testing (3 spatial $\times$ 10 temporal), otherwise we report 1-clip accuracy. Our focus is on making simple 2D networks better, and thus we only report the performance of the expensive I3D on Something-Something-V1 for comparison.

4.2 Results

We illustrate results for CATER Static/Camera Motion task 2 in Table 1 and Something-Something-V1/V2 in Table 2. Note that the GrayST method gets to use three times more frames and TC+2 uses just two more frames than RGB or TC, while maintaining the same computational cost of the model. We also report I3D results on Something-Something-V1, to show the impact of our sampling strategies when applied to 3D networks, see Table 2(a).

TC Reordering Performance. Using our TC Reordering, we observe that even the simple TSN model obtains much better performance compared to using conventional RGB frames with TRN and MTRN on all datasets. On TRN and MTRN, we observe significant improvement on most datasets, except on CATER Camera Motion task 2. However, it has less impact when combined with TSM and MVFNet, and sometimes hurts performance.

TC Reordering shifts the input video through its color channels in order to provide temporal information to 2D backbones. As a result, if a model already captures temporal information (*e.g.* TSM and I3D), TC Reordering is less effective, but still helps in many cases.

GrayST Performance. We observe consistent improvement on GrayST over RGB on most datasets and all models. The first obvious reason may be that it gets to see more

Model	Sampling	Model Size (T)	Top-1
TSN	RGB	8	17.2
	RGB	24	18.3
	GrayST	8	**35.5**
TRN	RGB	8	29.7
	RGB	24	31.7
	GrayST	8	**37.0**
TSM	RGB	8	45.4
	RGB	24	**51.1**
	GrayST	8	47.6

Table 3: Validation accuracy on Something-Something-V1 for 8 vs 24 frames. T refers to the temporal size of the model, and note that the GrayST gets to access three times as many frames without increasing the model size.

frames, *i.e*. three times as many as the other strategies. By combining grayscale information from different points in time it enables the 2D backbones to see temporal information. Despite utilizing more frames, this method does not increase training and inference times, with the exception of the time associated with loading the extra frames and converting them to grayscale. In comparison, using more RGB frames would make training and testing slower.

8-frame GrayST vs 24-frame RGB One might wonder if GrayST is better just because it 'sees' more frames. However, it has been reported that simply increasing the number of RGB frames does not necessarily improve performance or it is very marginal [27], and can even decrease performance [8] depending on the dataset. For example, [22] showed a baseline I3D-ResNet50 network with a 32-frame input obtained 73.3% on the Kinetics-400 dataset [11]. [6] later conducted an experiment with an 8-frame input following the same recipe for training and obtained 73.4%. We also conducted an ablation experiment in Table 3 to show that in many cases, the GrayST 8-frame (that looks at 24 frames and generates 8-frame 3-channel representation) is better than the RGB 24-frame. A GrayST frame not only increases the number of frames without increasing training and inference cost, but also allows 2D models to see more temporal information by discarding redundant color information. Note that TSM can make use of the information contained in the extra frames (see 8 versus 24 frames), but this benefit comes with increasing the training/testing cost by a factor of three.

5 Conclusion

We presented two novel video channel sampling strategies: TC Reordering and GrayST. The former re-orders RGB channels to increase temporal information, and the latter uses grayscale images to use more frames resulting in an increased temporal receptive field. In spite of the simplicity of our approaches, we observe a significant boost in performance on multiple challenging datasets. Importantly, these sampling strategies allow us to significantly increase the performance of existing lightweight video networks without increasing the computational cost or requiring any modifications to the underlying network architectures. Our hope is that this will pave the way for alternative sampling strategies. Future work includes developing a temporal aggregation module that is compatible with our sampling strategies for reasoning over much longer time scales.

Supplementary Material

A Training Details

On CATER task 2, CATER Camera Motion task 2, and Something-Something datasets, we used 2 NVIDIA RTX 3090 GPUs to train and test. We used 2 NVIDIA RTX 2080 Ti GPUs for the other datasets. For CATER, we used 32 frames due to the need for long-term temporal understanding. We set the total batch size to 24 and the initial learning rate to 0.0024. For Something-Something, we used 8 frames, a total batch size of 64, and an initial learning rate of 0.0064. As an exception, TRN and MTRN models are trained with one RTX 3090 GPU with half the total batch size and quarter the learning rate.

In all of the experiments, we kept the following protocols. The videos were resized so that the shorter side becomes 224 to 336 pixels, and we performed random cropping during training. For testing, we resized the input to have the shorter spatial side resolution of 256 and we used one center crop for CATER and Something-Something, and five crops (center and corners) and their horizontal flips for other datasets. For I3D experiments, we followed the common practice from [22] of densely sampling the video with a sampling stride of eight for RGB, three for GrayST because it samples more frames, and tested using ten evenly sampled clips throughout the video with three spatial crops of each, totaling 30 clips. The learning rate was decayed by 0.1 when validation metrics saturated for ten epochs. We stopped the experiments when the validation metrics saturated for 20 epochs. We used 16-bit precision to save memory, increase the batch size, and to train faster.

B Ablation Studies

TC variants. We explore some variants for ablation experiments: TC-Red, TC-RGB, and TC-ShortLong. TC-Red uses only red channels with the same frame ordering, *i.e.* $\mathbf{x}_i^{\mathrm{TC}} = (x_i^{\mathrm{R}}, x_{i+1}^{\mathrm{R}}, x_{i+2}^{\mathrm{R}})$. This baseline allows us to measure the effect of the diversity of color information using the TC Reordering. TC-RGB uses traditional RGB-like representation with the same temporal frame ordering as the TC Reordering: $\mathbf{x}_i^{\mathrm{TC}} = (x_i^{\mathrm{R}}, x_{i+1}^{\mathrm{G}}, x_{i+2}^{\mathrm{B}})$. Intuitively, this may seem to be the best representation as this is the closest representation to the RGB representation that the model is pre-trained on. However, in our experiments we observe that our TC Reordering actually outperforms TC-RGB. Finally, TC-ShortLong replaces the last two frames that consists of a lot of duplicates, with frames having longer sampling stride instead, *i.e.* $\mathbf{x}_7^{\mathrm{TC}} = (x_3^{\mathrm{R}}, x_5^{\mathrm{R}}, x_7^{\mathrm{R}})$ and $\mathbf{x}_8^{\mathrm{TC}} = (x_4^{\mathrm{G}}, x_6^{\mathrm{G}}, x_8^{\mathrm{G}})$ for the $T = 8$ case.

In Table 4 we contrast the different variants of TC Reordering. In the case of TSN, all variants of TC Reordering result in a significant performance increase compared to standard RGB. However, for TSM, only our proposed TC variant is superior. Perhaps counter-intuitively, TC-RGB, which maintains the RGB channel order but temporally shifts them, actually performs worse than our TC Reordering. TC frames consist of information from the same input color channel, which perhaps better enables it to capture local temporal changes, especially in cases where color is not informative for the task at hand.

Grayscale-only We report a grayscale-only experiment result on Something-Something-V1 in Table 5. We let $\mathbf{X}^{\mathrm{GO}} = \{\mathbf{x}_1^{\mathrm{GO}}, \mathbf{x}_2^{\mathrm{GO}}, \ldots, \mathbf{x}_T^{\mathrm{GO}}\}$ denote a GrayOnly video clip. We sample

Model	Sampling	Top-1	Top-5
TSN	RGB	17.2	42.7
	TC	**36.8**	**65.3**
	TC-RGB	33.6	61.4
	TC-Red	36.0	65.2
	TC-ShortLong	35.2	64.4
TSM	RGB	45.4	74.5
	TC	**45.8**	**74.7**
	TC-RGB	44.9	74.1
	TC-Red	44.7	73.8
	TC-ShortLong	44.8	73.6

Table 4: Ablation experiments of different TC Reordering strategies on Something-Something-V1. RGB refers to standard RGB channel sampling and TC is our proposed sampling strategy.

Model	Sampling	Top-1	Top-5
TSN	GrayOnly	16.1	41.2
	RGB	17.2	42.7
	TC	36.8	65.3
	TC+2	**37.0**	**65.6**
	GrayST	35.5	65.4

Table 5: Grayscale-only result on Something-Something-V1. Surprisingly, the performance is very close to that of RGB, and our methods utilize this fact to make the models further capture temporal information.

T grayscale frames following the sparse sampling strategy,

$$\mathbf{X}^g = \{x_1^g, x_2^g, \ldots, x_T^g\}, \tag{6}$$

where x_i^g is a grayscale image. Then, a GrayOnly frame is made by duplicating the same channel three times.

$$\mathbf{x}_i^{\mathrm{GO}} = (x_i^g, x_i^g, x_i^g). \tag{7}$$

We see very little difference in performance compared to RGB.

C Only Time Can Tell dataset

We show results on OnlyTimeCanTell-Temporal dataset [16] in Table 6. The dataset consists of 50 classes which require extensive temporal reasoning, taken from the Kinetics-400 and Something-Something-V1 dataset. Our methods outperformed the baselines by a significant margin on this dataset.

D Limitations

Despite the positive results, we also find some limitations of the proposed approaches, which we hope can lead to interesting future work.

Model	Sampling	Top-1	Top-5
TSN	RGB	50.7	84.6
	TC+2	65.9	92.0
	GrayST	**66.4**	**92.7**
TSM	RGB	71.7	94.2
	TC+2	73.3	94.0
	GrayST	**73.7**	**95.0**

Table 6: Evaluation of our methods on the OnlyTimeCanTell-Temporal dataset.

Model	Backbone	#frame	Sampling	Top-1
TSN	ResNet50	8	RGB	**75.4**
			TC	75.0
			GrayST	73.5
SlowFast*	ResNet101	16	RGB	77.6

Table 7: Evaluation on Diving-48-V2. The result marked with $*$ is from [1], which uses 30 clips for testing (3 spatial $\times$ 10 temporal).

Datasets Requiring Less Temporal Reasoning. Numerous datasets have significant object and scene bias making even TSN perform very similar to powerful 3D networks. In such cases, we found that the sampling strategies do not result in improved performance. Table 7 and 8 show such cases on Diving-48 [12] and UCF-101 [13] datasets.

Interestingly, Diving-48 V2 shows a decrease in performance with the GrayST input. The implication of this is that color information is important on this dataset, and with grayscale images it is difficult to distinguish divers of interest from the background. Despite the fact that the dataset is said to be "temporally-heavy", our experiment showed that the gap between 2D network and the state-of-the-art 3D network with double the number of frames and backbone depth is marginal.

The UCF-101 labels are biased towards object and scene information. As a result the performance of TSM is almost identical to that of TSN. The result did not show strong pattern but in general RGB seems to be preferable in this case.

Camera Motion. We found that TC Reordering combined with TRN and TSM negatively impacts performance on the CATER Camera Motion dataset. Note that the models make use of the temporal ordering of frames while TSN does not. Again, we still saw improvement with the GrayST method on this dataset. Judging from the fact that TC Reordering only hurts the temporal models, we think that this ordering plays a critical role in heavy camera motion scenarios.

Additionally, this dataset requires 3D-geometric understanding as the camera orbits around the objects substantially, making stationary objects look like they are sliding. The strength of TC Reordering is in cases where the model can analyze the motion information, but the large camera motion likely confuses the model when trying to understand what is the action of interest and what is the camera motion.

Model	Backbone	Sampling	Top-1	Top-5
TSN	ResNet50	RGB	83.6	**96.1**
		TC	83.6	96.0
		GrayST	**84.7**	95.9
TRN	ResNet50	RGB	**84.6**	**96.6**
		TC	81.6	95.3
		GrayST	82.2	95.5
TSM	ResNet50	RGB	**83.7**	96.0
		TC	81.4	95.5
		GrayST	83.6	**96.1**

Table 8: 8-frame evaluation on UCF-101 split 1.

References

[1] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? *arXiv preprint arXiv:2102.05095*, 2021.

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning (ICML)*, July 2021.

[3] Hakan Bilen, Basura Fernando, Efstratios Gavves, and Andrea Vedaldi. Action recognition with dynamic image networks. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2799–2813, 2017.

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019.

[7] Rohit Girdhar and Deva Ramanan. CATER: A diagnostic dataset for Compositional Actions and TEmporal Reasoning. In *ICLR*, 2020.

[8] Shreyank N Gowda, Marcus Rohrbach, and Laura Sevilla-Lara. Smart frame selection for action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1451–1459, 2021.

[9] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[12] Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[13] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7083–7093, 2019.

[14] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *CVPR*, 2022.

[15] Adam Santoro, David Raposo, David GT Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *NeurIPS*, 2017.

[16] Laura Sevilla-Lara, Shengxin Zha, Zhicheng Yan, Vedanuj Goswami, Matt Feiszli, and Lorenzo Torresani. Only time can tell: Discovering temporal data for temporal modeling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 535–544, January 2021.

[17] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[18] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[19] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[20] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2740–2755, 2018.

[21] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. Tdn: Temporal difference networks for efficient action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1895–1904, 2021.

[22] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[23] Chao-Yuan Wu, Manzil Zaheer, Hexiang Hu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Compressed video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6026–6035, 2018.

[24] Wenhao Wu, Dongliang He, Tianwei Lin, Fu Li, Chuang Gan, and Errui Ding. Mvfnet: Multi-view fusion network for efficient video recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2943–2951, 2021.

[25] Yiting Xie and David Richmond. Pre-training on grayscale imagenet improves medical image classification. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.

[26] Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, and Hanli Wang. Real-time action recognition with enhanced motion vector cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2718–2726, 2016.

[27] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 803–818, 2018.

[28] Linchao Zhu, Du Tran, Laura Sevilla-Lara, Yi Yang, Matt Feiszli, and Heng Wang. Faster recurrent networks for efficient video classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13098–13105, 2020.