Learning and Using the Arrow of Time

Donglai Wei[1], Jospeh Lim[2], Andrew Zisserman[3] and William T. Freeman[4,5]
[1]Harvard University [2]University of Southern California
[3]University of Oxford [4]Massachusetts Institute of Technology [5]Google Research
donglai@seas.harvard.edu, limjj@usc.edu, az@robots.ox.ac.uk, billf@mit.edu

Figure 1: Seeing these ordered frames from videos, can you tell whether each video is playing forward or backward? (answer below[1]). Depending on the video, solving the task may require (a) low-level understanding (e.g. physics), (b) high-level reasoning (e.g. semantics), or (c) familiarity with very subtle effects or with (d) camera conventions. In this work, we learn and exploit several types of knowledge to predict the arrow of time automatically with neural network models trained on large-scale video datasets.

Abstract

We seek to understand the arrow of time in videos – what makes videos look like they are playing forwards or backwards? Can we visualize the cues? Can the arrow of time be a supervisory signal useful for activity analysis? To this end, we build three large-scale video datasets and apply a learning-based approach to these tasks.

To learn the arrow of time efficiently and reliably, we design a ConvNet suitable for extended temporal footprints and for class activation visualization, and study the effect of artificial cues, such as cinematographic conventions, on learning. Our trained model achieves state-of-the-art performance on large-scale real-world video datasets. Through cluster analysis and localization of important regions for the prediction, we examine learned visual cues that are consistent among many samples and show when and where they occur. Lastly, we use the trained ConvNet for two applications: self-supervision for action recognition, and video forensics – determining whether Hollywood film clips have been deliberately reversed in time, often used as special effects.

1. Introduction

We seek to learn to *see* the arrow of time – to tell whether a video sequence is playing forwards or backwards. At a small scale, the world is reversible–the fundamental physics equations are symmetric in time. Yet at a macroscopic scale, time is often irreversible and we can identify certain motion patterns (e.g., water flows downward) to tell the direction of time. But this task can be challenging: some motion patterns seem too subtle for human to determine if they are playing forwards or backwards, as illustrated in Figure 1. For example, it is possible for the train to move in either direction with acceleration or deceleration (Figure 1d).

Furthermore, we are interested in how the arrow of time manifests itself visually. We ask: first, can we train a reliable arrow of time classifier from large-scale natural videos while avoiding artificial cues (i.e. cues introduced during video production, not from the visual world); second, what does the model learn about the visual world in order to solve this task; and, last, can we apply such learned commonsense knowledge to other video analysis tasks?

[1]Forwards: (b), (c); backwards: (a), (d). Though in (d) the train can move in either direction.

Regarding the first question on classification, we go beyond previous work [14] to train a ConvNet, exploiting thousands of hours of online videos, and let the data determine which cues to use. Such cues can come from high-level events (e.g., riding a horse), or low-level physics (e.g., gravity). However, as discovered in previous self-supervision work [4], ConvNet may learn artificial cues from still images (e.g., chromatic aberration) instead of a useful visual representation. Videos, as collections of images, have additional artificial cues introduced during creation (e.g. *camera motion*), compression (e.g. *inter-frame codec*) or editing (e.g. *black framing*), which may be used to indicate the video's temporal direction. Thus, we design controlled experiments to understand the effect of artificial cues from videos on the arrow of time classification.

Regarding the second question on the interpretation of learned features, we highlight the observation from Zhou *et al.* [26]: in order to achieve a task (scene classification in their case), a network implicitly learns what is necessary (object detectors in their case). We expect that the network will learn a useful representation of the visual world, involving both low-level physics and high-level semantics, in order to detect the forward direction of time.

Regarding the third question on applications, we use the arrow-of-time classifier for two tasks: video representation learning and video forensics. For representation learning, recent works have used temporal ordering for self-supervised training of an image ConvNet [6, 13]. Instead, we focus on the motion cues in videos and use the arrow of time to pre-train action recognition models. For video forensics, we detect clips that are played backwards in Hollywood films. This may be done as a special effect, or to make an otherwise dangerous scene safe to film. We show good performance on a newly collected dataset of films containing time-reversed clips, and visualize the cues that the network uses to make the classification. More generally, this application illustrates that the trained network can detect videos that have been tampered in this way. In both applications we exceed the respective state of the art.

In the following, we first describe our ConvNet model (Section 2), incorporating recent developments for human action recognition and network interpretation. Then we identify and address three potential confounds to learning the arrow of time discovered by the ConvNet (Section 3), for example, exploiting prototypical camera motions used by directors. With the properly pre-processed data, we train our model using two large video datasets (Section 4): a 147k clip subset of the Flickr100M dataset [22] and a 58k clip subset of the Kinetics dataset [10]. We evaluate test performance and visualize the representations learned to solve the arrow-of-time task. Lastly, we demonstrate the usefulness of our ConvNet arrow of time detector for self-supervised pre-training in action recognition and for identifying clip from Hollywood films made using the reverse-motion film technique (Section 5).

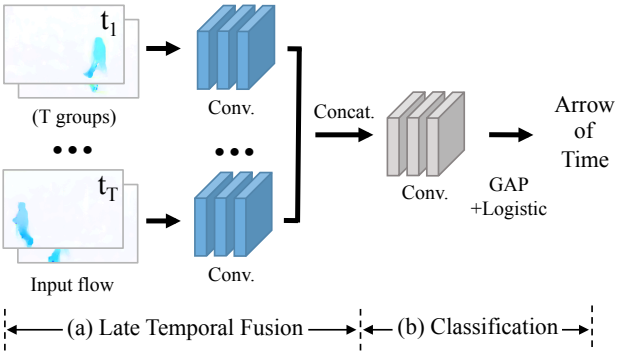


Figure 2: Illustration of our Temporal Class-Activation-Map Network (T-CAM) for the arrow of time classification. Starting from the traditional VGG-16 architecture [18] for image recognition, (a) we first concatenate the $conv5$ features from the shared convolutional layers, (b) and then replace the fully-connected layer with three convolution layers and global average pooling layer (GAP) [11, 20, 21, 27] for better activation localization.

1.1. Related Work

Several recent papers have explored the usage of the temporal *ordering* of images. Basha *et al.* [1, 3] consider the task of photo-sequencing – determining the temporal order of a collection of images from different cameras. Others have used the temporal ordering of frames as a supervisory signal for learning an embedding [15], for self-supervision training of a ConvNet [6, 13], and for construction of a representation for action recognition [7].

However, none of these previous works address the task of detecting the direction of time. Pickup *et al.*[14] explore three representations for determining time's arrow in videos: asymmetry in temporal behaviour (using hand-crafted SIFT-like features), evidence for causality, and an auto-regressive model to determine if a cause influences future events. While their methods work on a small dataset collected with known strong arrow of time signal, it is unclear if the method works on generic large-scale video dataset with different artificial signals. The study of the arrow of time is a special case of causal inference, which has been connected to machine learning topics, such as transfer learning and covariate shift adaptation [16].

In terms of ConvNet architectures, we borrow from recent work that has designed ConvNets for action recognition in videos with optical flow input to explicitly capture motion information [17, 24]. We also employ the Class Activation Map (CAM) visualization of Zhou *et al.* [27].

2. ConvNet Architecture

To focus on the time-varying aspects of the video, we only use optical flow as input to the ConvNet, and not its RGB appearance. Below, we first motivate the architecture, and then describe implementation details.

Model design. Our aim is to design a ConvNet that has an extended temporal footprint, and that also enables the learned features to be visualized. We also want the model to have sufficient capacity to detect subtle temporal signals. To this end, we base our model on three prior ConvNets: the VGG-16 network [18] as the backbone for the initial convolutional layers, for sufficient capacity; the temporal chunking in the model of Feichtenhofer *et al.* [5] to give an extended temporal footprint; and the CAM model of Zhou *et al.* [27] to provide the visualization.

The resulting architecture is referred to as "Temporal Class-Activation Map Network" (T-CAM) (Figure 2). For the temporal feature fusion stage (Figure 2a), we first modify the VGG-16 network to accept a number of frames (e.g. 10) of optical flow as input by expanding the number of channels of $conv1$ filters [24]. We use T such temporal chunks, with a temporal stride of τ. The $conv5$ features from each chunk are then concatenated. Then for the classification stage (Figure 2b), we follow the CAM model design to replace fully-connected layers with three convolution layers and global average pooling (GAP) before the binary logistic regression. Batch-Normalization layers [9] are added after each convolution layer.

Implementation details. To replace the fully-connected layers from VGG-16, we use three convolution layers with size $3\times3\times1024$, stride 1×1 and pad 1×1 before the GAP layer. For input, we use TV-L1 [25] to extract optical flow.

For all experiments in this paper, we split each dataset 70%-30% for training and testing respectively, and feed both forward and backward versions of the video to the model. The model is trained end-to-end from scratch, using fixed five-corner cropping and horizontal flipping for data augmentation. Clips with very small motion signals are filtered out from the training data using flow. Given a video clip for test, in addition to the spatial augmentation, we predict AoT on evenly sampled groups of frames for temporal augmentation. The final AoT prediction for each video is based on the majority vote of confident predictions (i.e. score $|x - 0.5| > 0.1$), as some groups of frames may be uninformative about AoT.

Verification on synthetic videos. Before testing on real world videos which may have comfounding factors (e.g. temporal codec, or cinematographer bias) to tell the time direction, we first examine the effectiveness of our T-CAM model on computer graphics videos where we have full control of the AoT signal. In the arXiv version of the paper, we train models on *three-cushion billiard game* videos simulated with different physical parameters (e.g. friction coefficient) by the physics engine in [8] with our extension to handle multiple balls. Trained only with the AoT signal on the synthetic videos, our model can not only learn video features to cluster test synthetic videos by their physical parameters, but also achieves 85% AoT classification accuracy on a collection of *real* three-cushion tournament videos (167 individual shots) from Youtube.

Figure 3: Illustration of artificial signals from videos in UCF101 dataset. (a) The black framing of the clip has non-zero intensity value (left), and a vertical slice over time displays an asymmetric temporal pattern (right). After training, we cluster the learned last-layer feature of top-confident test clips. We find some clusters have consistent (b) tilt-down or (c) zoom-in camera motion. We show two frames from two representative clips for each cluster.

3. Avoiding Artificial Cues from Videos

A learning-based algorithm may "cheat" and solve the arrow-of-time task using artificial cues, instead of learning about the video content. In this section, we evaluate the effect of three artificial signals, black framing, camera motion and inter-frame codec, on ConvNet learning and the effectiveness of our data pre-procession to avoid them.

3.1. Datasets regarding artificial cues

We use the following two datasets to study artificial cues.

UCF101 [19]. To examine the black framing and camera motion signal, we use this popular human action video dataset (split-1). Through automatic algorithms (i.e. black frame detection and homography estimation) and manual pruning, we find that around 46% of the videos have black framing, and 73% have significant camera motion (Table 1).

MJPEG Arrow of Time Dataset (MJPEG-AoT). To investigate the effect of inter-frame codec, we collect a new video dataset containing 16.9k individual shots from 3.5k videos from Vimeo[2]with diverse content. The collected

		Black frame	+Camera motion
Percent of videos		46%	73%
Acc.	before removal	98%	88%
	after removal	90%	75%

Table 1: AoT classification results to explore the effect of black framing and camera motion on UCF101 dataset. AoT test accuracy drops around 10% after removing black framing and drops another 10% after removing camera motion.

Train/Test	Original	H.264-F	H.264-B
Original	59.1%	58.2%	58.6%
H.264-F	58.1%	58.9%	58.8%
H.264-B	58.3%	59.0%	58.8%

Table 2: AoT classification results to explore the effect of the inter-frame codec on MJPEG-AoT dataset. We train and test on three versions of the data: original (no temporal encoding), encoded with H.264 in forward (H.264-F) and backward (H.264-B) direction. Similar AoT test accuracy suggests that the common H.264 codec doesn't introduce significant artificial signals for our model to learn from.

videos are either uncompressed or encoded with intra-frame codecs (e.g. MJPEG and ProRes) where each frame is compressed independently without introducing temporal direction bias. We can then evaluate performance with and without inter-frame codecs by using the original frames or the extracted frames after video compression with an inter-frame codec (e.g. H.264). The details of the dataset are in the arXiv version of the paper.

3.2. Experiments regarding artificial cues

We choose the T-CAM model to have two temporal segments and a total of 10 frames. More experimental details are in the arXiv version of the paper.

Black framing. Black frame regions present at the boundary may not be completely black after video compression (Figure 3a). The resulting non-zero image intensities can cause different flow patterns for forward and backward temporal motion, providing an artificial cue for the AoT.

For control experiments, we train and test our model on UCF101 before and after black framing removal, i.e., zero out the intensity of black frame regions. The test accuracy of the AoT prediction drops from 98% to 90% after the removal. This shows that black frame regions provides artificial cues for AoT and should be removed.

Camera motion. To understand the visual cues learned by our model after black framing removal, we perform K-means (K=20) clustering on the extracted feature before the logistic regression layer for the top-1K confidently classified test videos (foward or backward version). We estimate the homography for each video's camera motion with RANSAC, and compute the average translation and zoom in both horizontal and vertical directions. We find some video clusters have consistently large vertical translation motion (Figure 3b), and some have large zoom-in motion (Figure 3c). Such strong correlation among the confident clips between their learned visual representation and the camera motion suggests that cinematic camera motion conventions can be used for AoT classification.

For control experiments, we use a subset of UCF101 videos that can be well-stabilized. The test accuracy of the AoT prediction further drops from 88% to 75% before and after stabilization. Thus, we need to stabilize videos to prevent the model from using camera motion cues.

Inter-frame codec. For efficient storage, most online videos are compressed with temporally-asymmetric video codecs, e.g. H.264. They often employ "Forward prediction", which may offer an artificial signal for the direction of time. As it is almost impossible to revert the codecs, we train and test on our specially collected MJPEG-AoT dataset, where videos are not subject to this artificial signal.

We first remove black framing from these videos and choose individual shots that can be well-stabilized, based on the discoveries above. Then we create different versions of the downloaded MJPEG-AoT dataset (Original) by encoding the videos with the H.264 codec in either the forward (H.264-F) or backward direction (H.264-B), to simulate the corruption from the inter-frame codec. In Table 2 we show results where the model is trained on one version of the MJPEG-AoT dataset and tested on another version. Notably, our model has similar test accuracy, indicating that our model can not distinguish videos from each dataset for the AoT prediction. This finding offers a procedure for building a very large scale video dataset starting from videos that have been H.264 encoded (e.g. Youtube videos), without being concerned about artificial signals.

Conclusion. We have shown that black framing and camera motion do allow our model to learn the artificial signals for the AoT prediction, while the inter-frame codec (e.g. H.264) does not introduce significant signals to be learned by our model. For the experiments in the following sections we remove black framing and stabilize camera motion to pre-process videos for the AoT classification.

4. Learning the Arrow of Time

After verifying our T-CAM model on simulation videos and removing the known artificial signals from real world videos, we benchmark it on three real world video datasets and examine the visual cues it learns to exploit for the AoT.

[2] http://vimeo.com

# chunks	T=1			T=2		T=4
# frame	10	20	40	10	20	20
0% overlap	65%	62%	67%	79%	**81%**	71%
50% overlap	N/A			75%	76%	73%

Table 3: Empirical ablation analysis of T-CAM on Flickr-AoT. We compare the AoT test accuracy for models with a different number of input chunks (T), total number of frames, and overlap ratio between adjacent chunks. The best model takes in a total 20 frames of flow maps as input, and divides them into two 10-frame chunks without overlap to feed into the model.

Data (#clip)	[14]	T-CAM		Human
		Flickr	Kinetics	
TA-180 [14] (165)	82%	**83%**	79%	93%
Flickr-AoT (147k)	62%	**81%**	73%	81%
Kinetics-AoT (58k)	59%	71%	**79%**	83%

Table 4: AoT classification benchmark results on three datasets. We compare the T-CAM model, trained on either Flickr-AoT or Kinetics-AoT, with the previous state-of-the-art method [14] and with human performance. The T-CAM models outperform [14] on the large-scale datasets and achieves similar results on the previous TA-180 benchmark [14] (for test only).

4.1. Datasets

The previous AoT classification benchmark [14] contains only a small number of videos that are manually selected with strong AoT signals. To create large-scale AoT benchmarks with general videos, we pre-process two existing datasets through automated black framing removal and camera motion stabilization within a footprint of 41 frames. We use a fixed set of parameters for the data pre-processing, with the details in the arXiv version of the paper. We then use the following three video datasets to benchmark AoT classification.

TA-180 [14]. This dataset has 180 videos manually selected from Youtube search results for specific keywords (e.g. "dance" and "steam train") that suggest strong low-level motion cues for AoT. As some videos are hard to stabilize, in our experiments we only use a subset of 165 videos that are automatically selected by our stabilization algorithm.

Flickr Arrow of Time Dataset (Flickr-AoT). The Flickr video dataset [22, 23] is unlabeled with diverse video content, ranging from natural scenes to human actions. Starting from around 1.7M Flickr videos, we obtain around 147K videos after processing to remove artificial cues.

Kinetics Arrow of Time Dataset (Kinetics-AoT). The Kinectics video dataset [10] is fully labeled with 400 categories of human actions. Starting from around 266K train and validation videos, we obtain around 58K videos after processing to remove artificial cues. To balance for the AoT classification, we re-assign train and test set based on a 70-30 split for each action class.

4.2. Empirical ablation analysis

On the Flickr-AoT dataset, we present experiments to analyze various design decisions for our T-CAM model. With the same learning strategies (e.g. number of epochs and learning schedule), we compare models trained with (i) a different number of temporal segments (chunks); (ii) differing total number of input frames of flow; and (iii) varying overlap ratio between adjacent temporal segments.

In Table 4, we find that the best T-CAM model on Flickr-AoT has two temporal segments with 20 frames total without overlap. We use this model configuration for all the experimental results in this section.

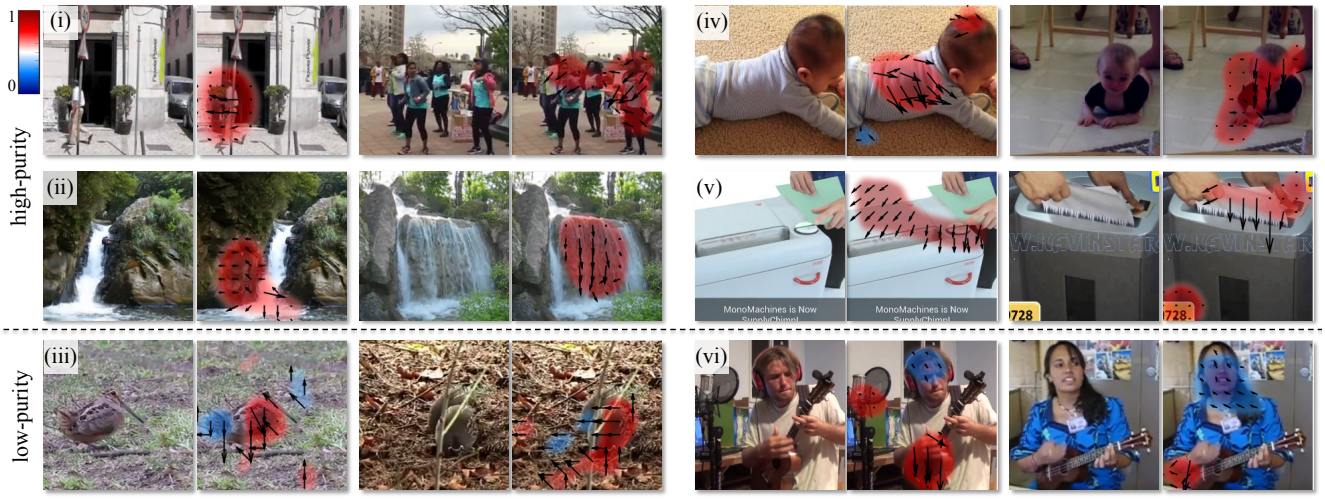
4.3. Experiments

In the following, we benchmark AoT classification results on all three datasets above.

Setup. For the baseline comparison, we implement the previous state-of-the-art, statistical flow method [14], and achieve similar 3-fold cross-validation results on the TA-180 dataset. To measure human performance, we use Amazon Mechanical Turk (AMT) for all three benchmark datasets (using random subsets for the large-scale datasets), where input videos have the same time footprint (i.e. 20 frames) as our T-CAM model. More details about the AMT study are in the arXiv version of the paper.

Classification results. On the previous benchmark TA-180 [14], we only test with models trained on Flickr-AoT or Kinetics-AoT dataset, as the dataset is too small to train our model. As shown in Table 4, the performance of the T-CAM models on TA-180, without any fine-tuning, are on-par with [14], despite being trained on different datasets. Testing on the large-scale datasets, Flickr-AoT and Kinetics-AoT, our T-CAM models are consistently better than [14] and are on par with human judgment.

Localization results. We localize regions that contribute most to the AoT prediction using techniques in Zhou *et al.* [27]. Given the 14×14 class activation map, we normalize it to a 0-1 probability heatmap p and resize it back to the original image size. Image regions are considered important for AoT prediction if their probability value is away from the random guess probability 0.5, i.e. $|p-0.5| > 0.2$. To visualize these important regions, we compute both the color-coded heatmap with a "blue-white-red colormap", where time forward evidence is red (close to 1) and backward is blue (close to 0), and also the sparse motion vectors on the middle frame of the input. In Figure 4, for each example we



(a) Clusters in Flickr-AoT (b) Action classes in Kinetics-AoT

Figure 4: Examples of T-CAM localization results on test clips from (a) Flickr-AoT and (b) Kinetics-AoT dataset. For each input clip, we compute its class activation map (CAM) from the model trained on the same dataset. We show its middle frame on the left, and overlay color-coded CAM (red for high probability of being forward, blue for backwards) and sparse motion vector on regions with confident AoT classification. For each dataset, we show localization results for two high-purity clusters (i.e., most clips have the same AoT label within the cluster) and one low-purity cluster. All the examples here are played in the forward direction and AoT in regions with red CAM are correctly classified. Notice that examples from low-purity clusters have a mix of red and blue regions.

show its middle frame and that with the heatmap and motion vector overlay for regions with confident predictions.

For each large dataset, we show localization results for three visual concepts from the cluster analysis. For each cluster, we define its "purity" as the average of the cluster samples' AoT value. A high-purity cluster means that its samples share the common feature that is indicative for AoT prediction. For the Flickr-AoT dataset, the two high-purity visual concepts (confident AoT prediction) correspond to "human walk" and "water fall" (Figure 4a). For the Kinetics-AoT dataset, the two AoT-confident action classes are "crawling baby" and "shredding paper", while the AoT-unsure action class is "playing ukulele" (Figure 4b).

5. Using the Arrow of Time

In this section, we describe two applications of the arrow of time signal: self-supervised pre-training for action recognition, and reverse film detection for video forensics.

5.1. Self-supervised pre-training

Initialization plays an important role in training neural networks for video recognition tasks. Self-supervised pre-training has been used to initialize action classification networks for UCF101, for example by employing a proxy task such as frame order, that does not require external labels [6, 12, 13]. For image input (i.e. the spatial stream), these approaches show promising results that are better than random initialization. However, their results are still far from the performance obtained by pre-training on a supervised task such as ImageNet classification [17, 24]. Further, there has been little self-supervision work on pre-training for the flow input (the temporal stream). Below we first show that the AoT signal can be used to pre-train flow-based action recognition models to achieve state-of-the-art results on UCF101 and HMDB51. Then to compare with previous self-supervision methods, we explore the effects of different input modalities and architectures on self-supervision with the AoT signal for UCF101 split-1.

Results with T-CAM model. To benchmark on UCF101 split-1, we pre-train T-CAM models with three different datasets and fine-tune each model with three different sets of layers. For pre-training, we directly re-use the models trained in the previous sections: one on UCF101 (on the subset that can be stabilized with black framing removed) from section 3, and also those trained on Flickr-AoT and Kinetics-AoT. To fine-tune for action classification, we replace the logistic regression for AoT with classification layers (i.e., a fully-connected layer + softmax loss), and fine-tune the T-CAM model with action labels. To understand the effectiveness of the AoT features from the different layers, we fine-tune three sets of layers separately: the last layer only, all layers after temporal fusion, and all layers. To compare with Wang *el al.* [24], we redo the random and

Initialization		Fine-tune		
		Last layer	After fusion	All layers
Random	[24]	-	-	81.7%
	T-CAM	38.0%	53.1%	79.3%
ImageNet	[24]	-	-	85.7%
	T-CAM	47.9%	68.3%	84.1%
AoT (ours)	UCF101	58.6%	**81.2%**	**86.3%**
	Flickr	57.2%	79.2%	84.1%
	Kinetics	55.3%	74.3 %	79.4%

Table 5: Action classification on UCF101 split-1 with flow input for different pre-training and fine-tuning methods. For random and ImageNet initialization, our modified T-CAM model achieves similar result to the previous state-of-the-art [24] that uses a VGG-16 network. Self-supervised pre-training of the T-CAM model using the arrow of time (AoT) consistently outperforms random and ImageNet initialization, i.e. for all three datasets and for fine-tuning on three different sets of levels.

	UCF101			HMDB51
	split1	split2	split3	
ImageNet [24]	85.7%	88.2%	87.4%	55.0%
AoT (ours)	**86.3%**	**88.6%**	**88.7%**	**55.4%**

Table 6: Action classfication on UCF101 (3 splits) and HMDB51 with flow input. We compare T-CAM models pre-trained with AoT to VGG-16 models pre-trained with ImageNet [24]. All models are pre-trained on the respective action recognition data and fine-tuned for all layers.

ImageNet initialization with the T-CAM model instead of the VGG-16 model, use 10 frames' flow maps as input, and only feed videos played in the original direction.

In Table 5, we compare self-supervision results for different initialization methods that use flow as input. First, it can be seen from the random and ImageNet initializations, that a VGG-16 model [24] has similar performance to the T-CAM model when fine-tuned on all layers. Second, self-supervised training of the T-CAM model with AoT on each of the three datasets outperforms random and ImageNet initialization for fine-tuning tasks at *all three* different levels of the architecture. Third, our AoT self-supervision method exceed the state-of-the-art when pre-trained on UCF101.

To benchmark on UCF101 other splits and HMDB51 dataset, we choose the best setting from above, that is to pre-train T-CAM model on the action recognition data and fine-tune for all layers. As shown in Table 6, our AoT self-supervision results outperform ImageNet pre-training [24] by around 0.5% consistently.

Comparison with other input and architectures. To further explore the effect of backbone architectures and modalites, we compare T-CAM with VGG-16 to ResNet-50, and stacked frames of flow to those of RGB and RGB difference (D-RGB) for action recognition on UCF101 split-1 dataset (Table 7). All models are pre-trained on UCF101 split-1 with 20-frame input and fine-tuned with all layers. To pre-train AoT with RGB and D-RGB input, we modify the number of channels of *conv1* filters correspondingly. In terms of the backbone architecture, the ResNet-50 models consistently outperform VGG-16 for each input modality. In terms of the input modality, all three modalities have similar performance for action recognition using ResNet-50 with our AoT pre-training and also with ImageNet pre-training as shown in Bilen *et al.* [2].

Model/Input	Flow	RGB	D-RGB
VGG-16	86.3%	78.1%	85.8%
ResNet-50	87.2%	86.5%	86.9%

Table 7: Action classification on UCF101 split-1, using AoT self-supervision but with other input and architectures. We compare results using VGG-16 and ResNet-50 backbone architectures, and flow, RGB and D-RGB input.

	Rand.	Shuffle [13]	Odd-One [6]	AoT
RGB	38.6%	50.9%	-	**55.3%**
D-RGB	-	-	60.3%	**68.9%**

Table 8: Action classification on UCF101 split-1, using AlexNet architecture but different self-supervision methods. We compare our results pre-trained with AoT to previous self-supervision methods using RGB or D-RGB input.

Comparison with other self-supervision methods. To compare with previous self-supervision methods [6, 13] that have used AlexNet as the backbone architecture and fine-tuned with all layers, we include fine-tuning results for models pre-trained using AoT on UCF101 split-1 for AlexNet with RGB or D-RGB inputs. In Table 8, our AoT results significantly outperform the prior art.

5.2. Video forensics: reverse film detection

Reverse action is a type of special effect in cinematography where the action that is filmed ends up being shown backwards on screen. Such techniques not only create artistic scenes that are almost impossible to make in real life (e.g. broken pieces coming back together), but also make certain effects easier to realize in the reverse direction (e.g. targeting a shot precisely). Humans can often detect such techniques, as the motion in the video violates our temporal structure prior of the world (e.g. the way people blink their eyes or steam is emitted from an engine). For this video forensics task, we tested the T-CAM model trained on the Flickr-AoT and Kinetics-AoT datasets with 10 frames of flow input, as some clips have fewer than 20 frames.

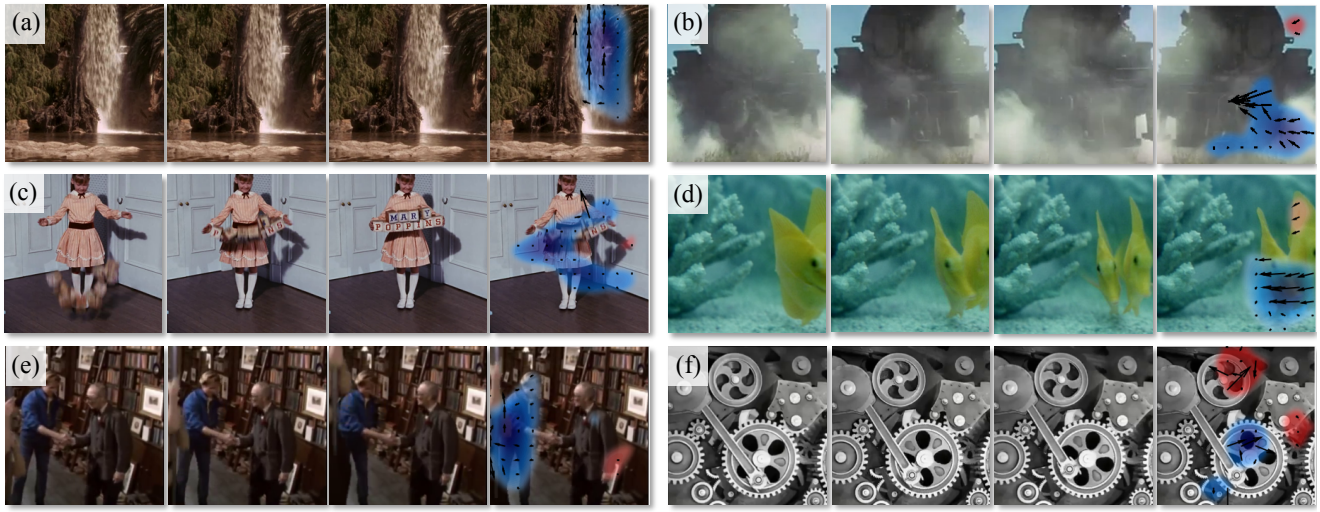


Figure 5: Example results from our Reverse Film dataset–short clips appearing time-reversed in Hollywood movies. For each example, we show four images: three frames from the input clip for our T-CAM model in their displayed order in the movie, and the class activation map with sparse motion field overlaid on the middle frame. As all the clips are played in the reverse direction, the ground truth class activation map color is blue. On examples that our T-CAM model classifies AoT correctly and confidently, the model exploits both low-level physical cues, e.g. (a) water falls, (b) smoke spreads, and (c) block falls and spreads; and high-level cues, e.g. (d) fish swim forwards, and (e) human action. The T-CAM model is unconfident about a motion that can be intrinsically symmetric, e.g. (f) wheel rotation.

Reverse Film Dataset. We collected clips from Hollywood films which are displayed in reverse deliberately. Thanks to the "trivia" section on the IMDB website, shots that use reverse action techniques are often pointed out by the fans as Easter eggs. With keyword matching (e.g. "reverse motion") and manual refinement on the trivia database, we collected 67 clips from 25 popular movies, including 'Mary Poppins', 'Brave Heart' and 'Pulp Fiction'. See the project page for the movie clips and more analysis of the common cues that can be used to detect the arrow of time.

Classification and localization results. As can be seen in Table 9, the overall test accuracy of the T-CAM model is 76% (trained on Flickr-AoT) and 72% (trained on Kinetics-AoT), where human performance (using Amazon Mechanical Turk) is 80%, and the baseline model [14] achieves 58%. In Figure 5, we visualize both successful and failure cases, and show the T-CAM heatmap score of being backward in time. The successful cases are consistent with our earlier finding that the model learns to capture both low-level cues such as gravity (Figure 5a,c) and entropy (Figure 5b), as well as high-level cues (Figure 5d-e), and . For the failure cases, some are due to the symmetric nature of the motion, e.g. wheel rotation (Figure 5f).

	Chance	[14]	T-CAM (ours)		Human
			Flickr	Kinetics	
Acc.	50%	58%	76%	72%	80%

Table 9: AoT test accuracy on the Reverse Film dataset. The T-CAM model pre-trained on either Flicker-AoT or Kinetics-AoT outperforms Pickup *et al.* [14], and is closer to human performance.

6. Summary

In this work, we manage to learn and use the prevalent arrow of time signal from large-scale video datasets. In terms of *learning* the arrow of time, we design an effective ConvNet and demonstrate the necessity of data pre-processing to avoid learning artificial cues. We develop two large-scale arrow of time classification benchmarks, where our model achieves around 80% accuracy, significantly higher than the previous state-of-the-art method at around 60%, and close to human performance. In addition, we can identify the parts of a video that most reveal the direction of time, which can be high- or low-level visual cues.

In terms of *using* the arrow of time, our model outperforms the previous state-of-the-art on the self-supervision task for action recognition, and achieves 76% accuracy on a new task of reverse film detection, as a special case for video forensics.

Acknowledgments. This work was supported by NSF Grants 1447476 and 1212849 (Reconstructive Recognition), and by the EPSRC Programme Grant Seebibyte EP/M013774/1.

References

[1] T. Basha, Y. Moses, and S. Avidan. Photo sequencing. In *ECCV*, 2012. 2

[2] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould. Dynamic image networks for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3034–3042, 2016. 7

[3] T. Dekel (Basha), Y. Moses, and S. Avidan. Photo sequencing. *IJCV*, 2014. 2

[4] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 2

[5] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 3

[6] B. Fernando, H. Bilen, E. Gavves, and S. Gould. Self-supervised video representation learning with odd-one-out networks. In *CVPR*, 2017. 2, 6, 7

[7] B. Fernando, E. Gavves, J. M. Oramas, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *CVPR*, 2015. 2

[8] K. Fragkiadaki, P. Agrawal, S. Levine, and J. Malik. Learning visual predictive models of physics for playing billiards. In *ICLR*, 2015. 3

[9] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 3

[10] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2, 5

[11] M. Lin, Q. Chen, and S. Yan. Network in network. In *ICLR*, 2013. 2

[12] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, 2017. 6

[13] I. Misra, L. Zitnick, and M. Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV*, 2016. 2, 6, 7

[14] L. C. Pickup, Z. Pan, D. Wei, Y. Shih, C. Zhang, A. Zisserman, B. Scholkopf, and W. T. Freeman. Seeing the arrow of time. In *CVPR*, 2014. 2, 5, 8

[15] V. Ramanathan, K. Tang, G. Mori, and L. Fei-Fei. Learning temporal embeddings for complex video analysis. In *ICCV*, 2015. 2

[16] B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij. On causal and anticausal learning. In *ICML*, 2012. 2

[17] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 2, 6

[18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 2, 3

[19] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In *arXiv preprint arXiv:1212.0402*, 2012. 3

[20] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR*, 2014. 2

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 2

[22] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 2, 5

[23] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *NIPS*, 2016. 5

[24] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *ECCV*, 2016. 2, 3, 6, 7

[25] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l 1 optical flow. In *JPRS*, 2007. 3

[26] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene CNNs. In *ICLR*, 2014. 2

[27] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016. 2, 3, 5