Temporal Residual Networks for Dynamic Scene Recognition

Christoph Feichtenhofer *
Graz University of Technology
feichtenhofer@tugraz.at

Axel Pinz
Graz University of Technology
axel.pinz@tugraz.at

Richard P. Wildes
York University, Toronto
wildes@cse.yorku.ca

Abstract

This paper combines three contributions to establish a new state-of-the-art in dynamic scene recognition. First, we present a novel ConvNet architecture based on temporal residual units that is fully convolutional in spacetime. Our model augments spatial ResNets with convolutions across time to hierarchically add temporal residuals as the depth of the network increases. Second, existing approaches to video-based recognition are categorized and a baseline of seven previously top performing algorithms is selected for comparative evaluation on dynamic scenes. Third, we introduce a new and challenging video database of dynamic scenes that more than doubles the size of those previously available. This dataset is explicitly split into two subsets of equal size that contain videos with and without camera motion to allow for systematic study of how this variable interacts with the defining dynamics of the scene per se. Our evaluations verify the particular strengths and weaknesses of the baseline algorithms with respect to various scene classes and camera motion parameters. Finally, our temporal ResNet boosts recognition performance and establishes a new state-of-the-art on dynamic scene recognition, as well as on the complementary task of action recognition.

1. Introduction

Image-based scene recognition is a basic area of study in visual information processing. Humans are capable of recognizing scenes with great accuracy and speed [32]. Reliable automated approaches can serve to provide priors for subsequent operations involving object and action recognition, e.g., [25, 43]. Moreover, scene recognition could serve in browsing image databases, e.g. [45]. While early computational research in scene recognition was concerned with operating on the basis of single images, e.g., [8, 23, 28], more recently dynamic scene recognition from video has emerged as a natural progression, e.g., [5, 11, 34].

Beyond dynamic scene recognition, considerable research has addressed allied tasks in video-based recognition. Arguably, the most heavily researched has been action recognition [25, 27, 35]; although, a variety of additional video-based recognition tasks also have been considered, *e.g.* [29, 30, 31]. In response to the challenges these tasks pose, a wide variety of approaches have been developed. Here, it is worth noting that recent extensions of Convolutional Networks (ConvNets) to video have shown particularly strong results, *e.g.* [26, 44, 50]. While many of these approaches have potential to be generalized and applied to dynamic scene recognition, that avenue has been under researched to date. The current paper addresses this situation by applying a representative sampling of state-of-the-art video recognition techniques to dynamic scenes, including a novel ConvNet. This work extends our understanding of not only the individual techniques under evaluation, but also the nature of dynamic scenes as captured in video.

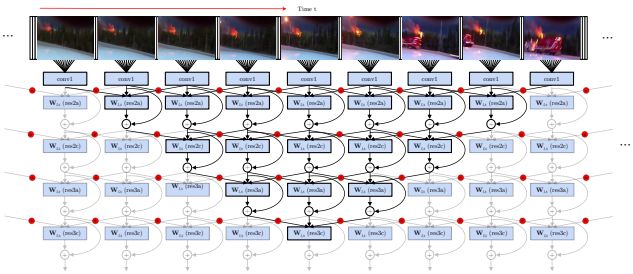

Figure 1: Temporal receptive field of a single neuron at the third conv block of our spatiotemporal ConvNet architecture. Our T-ResNet is *fully convolutional in spacetime* and performs temporal filtering at residual units to hierarchically inject temporal information as depth increases.

1.1. Related research

Currently, there are two standard databases to support the study of scene recognition from videos [5, 34]. Both of these databases capture a range of scene classes and natural variations within class (seasonal and diurnal changes as well as those of viewing parameters). A significant difference between the two datasets is that one includes camera motion [34], while the other does not [5]. Unfortunately, neither

*Christoph Feichtenhofer is a recipient of a DOC Fellowship of the Austrian Academy of Sciences at the Institute of Electrical Measurement and Measurement Signal Processing, Graz University of Technology.

database provides balanced scene samples acquired with and without camera motion to support systematic study of how scene dynamics can be disentangled from camera motion. Moreover, at this time performance on both datasets is at saturation [12, 44]. Correspondingly, research in dynamic scene recognition is at risk of stagnation, unless new, systematically constructed and challenging video databases relevant to this task are introduced.

Video-based dynamic scene classification has been approached based on linear dynamical systems [7], chaotic invariants [34], local measures of spatiotemporal orientation [5, 10, 11], slowly varying spatial orientations [42] and spatiotemporal ConvNets [44], with spatiotemporal orientation and ConvNets showing strongest recent performance [12].

Action recognition is a related task where scene context plays an important role [25]. The state-of-the-art in action recognition is currently dominated by ConvNets that learn features in an end-to-end fashion either directly in the spatiotemporal domain [18, 44], or in two streams of appearance and motion information [9, 13, 27, 35].

The most closely related work to ours is a spatiotemporal residual network, ST-ResNet [9] that is based on two-stream [35] and residual networks [15]. The ST-ResNet [9] architecture injects residual connections between the appearance and motion pathways of a two-stream architecture and transforms spatial filter kernels into spatiotemporal ones to operate on adjacent feature maps in time. Our work instead extends the spatial residual units with a temporal kernel that is trained from scratch and hence is able to learn complex temporal features as it is initialized to receive more informative temporal gradients.

1.2. Contributions

This paper makes the following contributions. First, a novel spatiotemporal ConvNet architecture, T-ResNet, is introduced that is based on transformation of a spatial network to a spatiotemporal ConvNet; see Fig 1. This transformation entails a particular form of transfer learning from spatial image classification to spatiotemporal scene classification. Second, the superiority in scene recognition from video of the newly developed T-ResNet is documented by comparing it to a representative sampling of alternative approaches. Results show that our spatiotemporally trained ConvNet greatly outperforms the alternatives, including approaches hand-crafted for dynamic scenes and other networks trained directly for large scale video classification. Third, a new dynamic scenes dataset is introduced. This dataset more than doubles the size of the previous collections in common use in dynamic scene recognition [5, 34], while including additional challenging scenarios. Significantly, for each scene class that is represented an equal number of samples is included with and without camera motion to support systematic investigation of this variable in scene recognition. Finally, we also show that our model generalizes for other tasks, by reporting results on two widely used video action recognition datasets. Our Code uses the MatConvNet toolbox [46] and is available at https://github.com/feichtenhofer/temporal-resnet and our novel dynamic scene recognition dataset is available at http://vision.eecs.yorku.ca/research/dynamic-scenes/.

2. Temporal residual networks

Various paths have been followed to extend ConvNets from the 2D spatial domain, (x, y), to the 3D spatiotemporal domain, (x, y, t), including building on optical flow fields [31, 35], learning local spatiotemporal filters [18, 41, 44] and modeling as recurrent temporal sequences [1, 6, 37]. To date, however, these approaches have not triggered the dramatic performance boost over hand-crafted representations (*e.g.* IDT [48]) that ConvNets brought to the spatial domain *e.g.* in image classification [20, 36]. This relative lack of impact has persisted even when large new datasets supported training of 3D spatiotemporal filters [18, 44]. This section documents a novel approach that proceeds by transforming a spatial ConvNet, ResNet [15], to a spatiotemporal kindred, T-ResNet, as outlined in Fig. 1. In empirical evaluation (Sec. 4) it will be shown that this approach yields a network with state-of-the-art performance.

2.1. Spatiotemporal residual unit

The main building blocks of the ResNet architecture are residual units [15]. Let the input to a residual unit be a feature map, $\mathbf{x}_l \in \mathbb{R}^{H\times W\times T\times C}$, where W and H are spatial dimensions, C is the feature dimension and T is time. Such maps can be thought of as stacking spatial maps of C dimensional features along the temporal dimension. At layer l with input $\mathbf{x}_l$, a residual block is defined as [15, 16]

$$\mathbf{x}_{l+1} = f\left(\mathbf{x}_l + \mathcal{F}(\mathbf{x}_l; \mathcal{W}_l)\right), \tag{1}$$

with $f \equiv \text{ReLU}$, $\mathcal{W}_l = \{\mathrm{W}_{l,k}|_{1\leq k\leq K}\}$ holding the K corresponding filters and biases in the unit, and $\mathcal{F}$ denoting the residual function representing convolutional operations. Formally, each of the K layers in the l^{th} residual unit performs the following filtering operation

$$\mathbf{x}_{l,k+1} = \mathrm{W}_{l,k}\mathbf{x}_{l,k}, \tag{2}$$

where $\mathrm{W}_{l,k}|_{1\leq k\leq K}$ are the convolutional filter kernels arranged as a matrix and batch normalization layers are omitted for simplicity. We use the original ResNet architecture [15] where $K = 3$, consisting of 1×1 dimensionality reduction, 3×3 spatial aggregation and 1×1 dimensionality restoration filtering operations. These choices lead to the residual unit

$$\mathcal{F} = f(\mathrm{W}_{l,3}f(\mathrm{W}_{l,2}f(\mathrm{W}_{l,1}\mathbf{x}_l))), \tag{3}$$

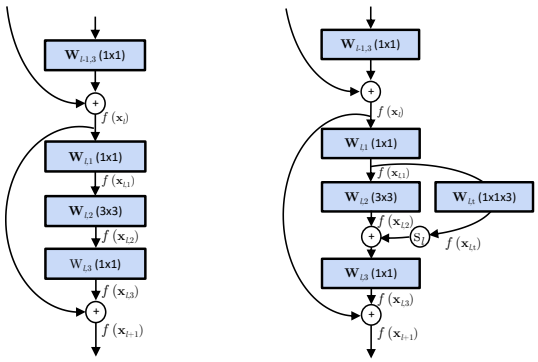

Figure 2: Comparison between the original residual units (a) and our proposed spatiotemporal residual units (b), which augment the "bottleneck structure" with an additional temporal conv block and an affine scaling layer (Sl).

as illustrated in Fig. 2a.

Our proposed spatiotemporal residual unit $\hat{\mathcal{F}}$ injects temporal information into residual blocks via 1D temporal filtering. Building on the inception idea [39], our temporal convolution block operates on the dimensionality reduced input, $\mathbf{x}_{l,1}$, with a bank of spatiotemporal filters, $\mathrm{W}_{l,t}$, and by applying biases, $b \in \mathbb{R}^C$, according to

$$\mathbf{x}_{l,t} = \mathrm{W}_{l,t}\mathbf{x}_{l,1} + b, \tag{4}$$

where biases $b \in \mathbb{R}^C$ are initialized to zero and the weights, $\mathrm{W}_{l,t}$, come as a 3-tap temporal filter bank. These filters are initialized randomly for the same feature dimensionality as the spatial 3×3 filters, $\mathbf{W}_{l,2}$, working in parallel on input $\mathbf{x}_{l,1}$. $\mathrm{W}_{l,t}$ is able to model the temporal structure of the features from the previous layer. Moreover, by stacking several such kernels through the hierarchy of the network we are able to grow the temporal receptive field. See for example Fig. 1, where the temporal receptive field for a unit at the third convolutional layer is highlighted.

Our proposed spatiotemporal residual unit $\hat{\mathcal{F}}$ is now defined as

$$\hat{\mathcal{F}} = f\Big(\mathrm{W}_{l,3}\Big(\mathrm{S}_l f(\mathbf{x}_{l,t}) + f(\mathrm{W}_{l,2} f(\mathbf{x}_{l,1}))\Big)\Big), \tag{5}$$

where S_l is a channel-wise affine scaling weight initialized with a scaling of .01 and zero biases. We found adaptive scaling of the temporal residuals to facilitate generalization performance. The final unit is illustrated in Fig. 2b.

Discussion. Our design builds on two good practices for designing image ConvNets: First, it builds on the inception concept that dimensionality reduction should be performed before spatially/temporally aggregating filters since outputs of neighbouring filters are highly correlated and therefore these activations can be reduced before aggregation [40]. Second, it exploits spatial factorization into asymmetric filters, which reduces computational cost and also has been found to ease the learning problem [38].

Scaling down residuals also has been found important for stabilizing training [38], where residuals are scaled down by a constant factor before being added to the accumulated layer activations. Activations also have been usefully rescaled before combining them over several layers of the network [3]. As an alternative to scaling has been used in [15] where the network was first pre-conditioned by training with a very low learning rate, before the training with higher learning rate proceeded.

2.2. Global pooling over spacetime

The "Network In Network" [24] architecture has shown that the fully connected layers used in previous models [20, 36] can be replaced by global average pooling after the last convolutional layer and this replacement has been adopted by recent ConvNet architectures [15, 39]. The general idea behind global pooling of activations is that at the last conv-layer the units see all pixels at the input due to the growth of the receptive field; *e.g.* for the ResNet-50 architecture that we use in this work, the last convolutional layer theoretically has a receptive field that covers 483×483 pixels at the input, even though the input is only of size 224×224. Practically, however, the utilized receptive field of a unit is postulated to be smaller [52].

Correspondingly, a temporal network can be expected to capture similar spatial features across time. Therefore, it is reasonable to develop a spatiotemporal network by global pooling over the temporal support. We found in our experiments that globally max pooling over time, *i.e.*

$$\mathbf{x}(i,j,c) = \max_{1 \le k' \le T'} \mathbf{x}(i,j,k',c), \tag{6}$$

works better ($\approx 2\%$ accuracy gain) than global averaging of temporal activations. We conjecture that this result is due to the derivative of the sum operation uniformly backpropagating gradients to the temporal inputs. Thus, the network is not able to focus on the most discriminating instance in time when employing temporal averaging. Even though max-pooling over time only backpropagates a single temporal gradient map to the input, it can guide the learning of long-term temporal features because the filter's temporal receptive field on the gradient maps grows *from the output to the input*.

Discussion. We conducted several experiments using max-pooling earlier in the network and it consistently led to reduced performance with accuracy decreasing more, the earlier pooling starts ($\approx 1-6\%$). We also experimented with a more natural decrease of frames by valid convolutions in time. Typically, ConvNet architectures zero-pad the inputs

before each spatial (*e.g.* 3×3) convolution such that the output size is unchanged. A cleaner strategy is to use valid convolutions (*i.e.* not filtering over the border pixels) together with lager sized inputs *e.g.* used in the early layers of inception-v4 [38]. We investigated if there is any gain in performance for having a temporal architecture with valid filtering operations across time. For this experiment, we increase the number of frames at the input and use temporal residual blocks that do not pad the input in time. Since the network now hierarchically downsamples the input by two frames at each temporal residual block, the final max-pooling layer now receives less frames (when keeping GPU-memory constant compared to a padded design). In our experiments this architectural change leads to an error increase of 2.4%, in comparison to the padded architecture equivalent. In conclusion, pooling as late as possible, together with padded convolutions across time, enables best accuracy for our T-ResNet in dynamic scene classification.

2.3. Implementation details

We build on the ResNet-50 model pretrained on ImageNet [15] and replace the last (prediction) layer. Next we transform every first and third residual unit at each conv stage (conv2_x to conv5_x) that hold residual units to our proposed temporal residual units. For our temporal residual blocks, we switch the order of batch normalization [17] and ReLU from post-activation to pre-activation [16]. The temporal filters are of dimension $W' \times H' \times T' \times C \times C = 1 \times 1 \times 3 \times C \times C$ and initialized randomly. We use 16 frame inputs and temporal max-pooling is performed immediately after the spatial global average pooling layer.

The training procedure follows standard ConvNet training [15, 20, 36], with some subtle differences. We set the learning rate to 10^{-2} and decrease it by an order of magnitude after the validation error saturates. We use batch normalization [17] and no dropout. To accelerate training, we train the network in a two-stage process with a batchsize of 256: First, we train the network in a purely spatial manner where we randomly sample a single frame from different videos (ResNet); second, we transform the residual units to spacetime and re-start the training process by sampling 16-frame stacks from 256/32 videos per batch (T-ResNet).

For data augmentation we obtain multiple frame-stacks by randomly selecting the position of the first frame and apply the same random crop to all samples. Instead of cropping a fixed sized 224×224 input patch, we perform multi-scale and aspect-ratio augmentation by randomly jittering its width and height by $\pm 25\%$ and resizing it to obtain a fixed sized 224×224 network input. We randomly crop translated patches at a maximum of 25% distance from the image borders (relative to the width and height). Compared to training spatial ConvNets, training spatiotemporal ConvNets is even more prone to overfitting. In response, we use temporal frame jittering: In each training iteration we sample the 16 frames from each of the training videos in a batch by randomly sampling the starting frame, and then randomly sampling the temporal stride $\in [5, 15]$. We do not apply RGB colour jittering [20].

During testing, we take a sample of 16 equally spaced frames from a video and propagate these through the net to yield a single prediction for each video. Instead of cropping the image corners, centre and their horizontal flips, we apply a faster fully convolutional testing strategy [36] on the original image and their horizontal flips and average the predictions from all locations. Thus inference can be performed in a *single* forward pass for the whole video.

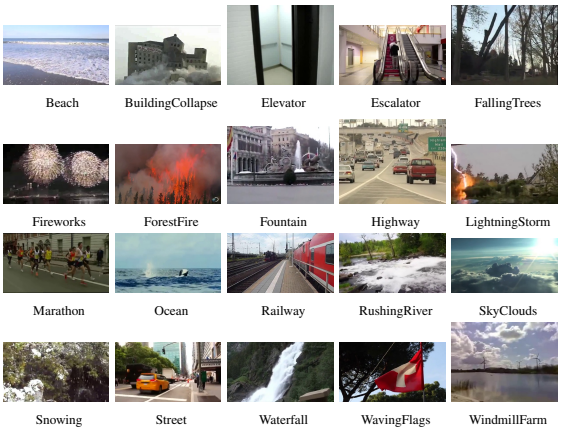

Figure 3: Thumbnail examples of the **YUP++** dataset.

3. Dynamic scenes dataset

As discussed in Sec. 1, the previously best performing algorithms on dynamic scene recognition have saturated performance on extant datasets [12, 44]. In response, this section introduces a new dynamic scenes dataset to support the current and future studies in this domain.

3.1. Specifications

The new dynamic scenes dataset samples 20 scene classes, while encompassing a wide range of conditions, including those arising from natural within scene category differences, seasonal and diurnal variations as well as viewing parameters. Thumbnail examples of each class are shown in Figs. 3 and 4; representative videos are available in the supplemental material for this paper. Details of the dataset are provided in the remainder of this section.

The new dataset builds on the earlier YUPenn dataset [5]. This dataset is taken as a point of departure rather than the Maryland dataset [34] as it includes one additional scene class and three times as many videos for each class represented. To the original dataset, six additional classes have been added for a total of twenty. The final set of classes represented in the dataset are as follows: beach, city street, elevator, forest fire, fountain, highway, lightning storm, ocean,

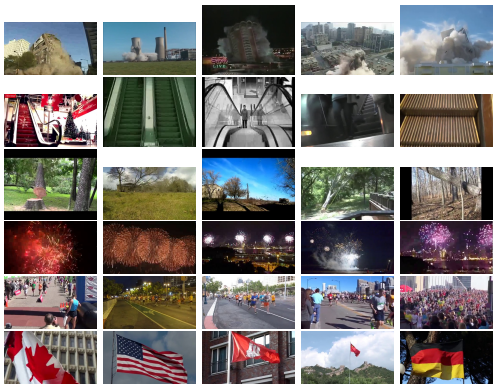

Figure 4: Variations within the new classes in the **YUP++**. Top-to-bottom: Building Collapse, Escalator, Falling Trees, Fireworks, Marathon and Waving Flags.

railway, rushing river, sky clouds, snowing, waterfall, windmill farm, building collapse, escalator, falling trees, fireworks, marathon, waving flags. The last six listed classes are in addition to those available in the earlier YUPenn. Due to its extended number of classes and addition of moving camera videos this novel dataset is termed **YUP++**.

For each scene class in the dataset, there are 60 colour videos, with no two samples for a given class taken from the same physical scene. Half of the videos within each class are acquired with a static camera and half are acquired with a moving camera, with camera motions encompassing pan, tilt, zoom and jitter. Having both static and moving camera instances for each class allows for systematic consideration of the role this variable plays in scene categorization, something that was not supported in either of the previous dynamic scenes datasets. Beyond camera motion and natural variation of individual scenes within a given class, a wide range of additional acquisition parameters are varied, including illumination (*e.g.* diurnal variations), seasonal, scale and camera viewpoint.

The videos were acquired from online video repositories (YouTube [51], BBC Motion Gallery [2] and Getty Images [14]) or a handheld camcorder. All videos have been compressed with H.264 codec using the ffmpeg video library. Duration for each video is 5 seconds, with original frame rates ranging between 24 and 30 frames per second. All have been resized to a maximum width of 480 pixels, while preserving their original aspect ratio.

Overall, the new dynamic scenes dataset more than doubles the size of previous datasets for this task. All of the videos are distinct from the earlier Maryland dataset. In comparison to the YUPenn dataset, six new scene classes have been added and all moving camera videos are new.

3.2. Experimental protocol

For the purpose of dynamic scene recognition, the dataset has been divided into training and test sets. A random split is employed to generate the two sets, by randomly choosing for each class an equal number of static camera and moving camera videos. This random split protocol is in contrast to the previously used leave-one-out protocol on the YUPenn and Maryland datasets. As documented in Sec. 4, using a random split with such a train/test ratio is better suited to providing a challenging benchmark protocol in a computationally tractable fashion. Moreover, a random split protocol is common practice in other domains, e.g. action recognition on HMDB51 [21] and UCF101 [19] as well as indoor scene classification on MIT67 [33].

4. Empirical evaluation

To establish the state-of-the-art in dynamic scene recognition, 7 representative algorithms for video-based recognition are evaluated along with T-ResNet introduced in Sec. 2. Three of the evaluated algorithms, C3D [44], BoSE [11] and SFA [42], have shown the first, second and fourth best performance in previous dynamic scenes evaluations. The third best performer, [10], is an ancestor of BoSE and is not considered here. The remaining algorithms, while not previously evaluated on dynamic scene recognition, are selected to provide a balanced coverage of contemporary strong performers on image-based classification tasks. To see how well a strong performer on single image classification can perform on video-based scene classification, very deep convolutional networks with Fisher vector encoded features are considered (S-CNN) [4]. To see how well an approach that serves as the basis for a variety of strong performing action recognition algorithms can be adapted to scene recognition, (improved) dense trajectories (IDTs) are considered [47]. Also, to test further temporal ConvNets (in addition to spatiotemporal C3D), a representative based on optical flow is considered (T-CNN) [35]. Finally, to judge the improvements that the spatiotemporal T-ResNet offers over the spatial ResNet [15], we report results for the fine-tuned ResNet.

Details of how these approaches are applied to dynamic scene recogntion are supplied in the supplemental material.

4.1. Is there a need for a new dataset?

The first question investigated is if a new dynamic scenes dataset is truly needed to challenge the available algorithms or if only the existing evaluation protocols need an improvement. To answer this question, the largest previously available dataset, **YUP** [5], was explored in the following fashion with three of the best performing algorithms to date (C3D, BoSE, and SFA): Instead of using the leave-one-video-out (LOO) protocol, as in previous evaluations [5, 11, 34, 42], fixed train/test splits are used, as it is common practice in action recognition tasks [19, 21]. Splits were generated by randomly choosing training and testing clips from each class. Three splits are employed for any given ratio; final recognition accuracy is taken as the aver-

age across the three. This experiment was performed for several train/test ratios. The results for the three considered algorithms are reported in Table 1, left. Surprisingly, performance can remain on par with the leave-one-out results on this dataset [11, 42, 44]; it is also surprising that even very low train/test ratios can still score high. It is concluded that simply changing the relative sizes of the training and testing sets does not have a significant enough impact on recognition rates to continue using this dataset in evaluations.

In comparing results for various 10/90 splits we find little difference, even for the most difficult moving camera component of the new dataset, **YUP++ moving camera**; see Table 1, right. This finding suggests that the 10/90 split supports stable algorithm performance evaluation, even while being most challenging. Moreover, since there is little variation between random splits it justifies using just a single split in evaluation, as it provides for less overhead in evaluation, especially for cases involving ConvNet training. Therefore, in all subsequent experiments a single 10/90 split is employed. In particular, we employ split #1 of Table 1.

4.2. Does adding new classes solve the problem?

The next question investigated is if adding additional classes would lead to a sufficiently more challenging dynamic scene benchmark. Table 2 (left) lists the results for including six additional classes BuildingCollapse, Escalator, FallingTrees, Fireworks, Marathon and WavingFlags to the previously existing YUPenn. Note that still all videos are taken from a static camera and thereby this subset is called the **YUP++ static camera**. While all algorithms decrease in performance compared to the original YUP, the best performers suffer only negligible deficits. It is desirable to further increase the challenge.

4.3. Does more challenging data help?

Since adding more classes has too limited an effect on performance, this section presents a way of increasing the difficulty of the data. The challenge is increased by including camera motion during acquisition of videos. The overall size of the datasets is thereby doubled, as each class contains an equal number of videos captured with and without camera motion. Details are provided in Sec. 3. The results for just the new videos are reported in Table 2 (right), with this subset referred to as **YUP++ moving camera**. Here it is seen that the challenge has increased so that even the top performing algorithm scores at 81.5% accuracy and there is spread between algorithms that allows for interesting comparisons, as discussed next.

4.4. Detailed algorithm comparisons

Consistent with previous results (*e.g.* [5]), top performers on the static (Table 2, left) and moving (Table 2, right) camera subsets as well as the entirety (Table 3, left) of YUP++ are dominated by algorithms that include both spatial and temporal measurements, *i.e.* our novel T-ResNet, C3D, IDT and BoSE. Interestingly, algorithms based on purely spatial features, S-CNN and ResNet, also show reasonable performance. Apparently, even for dynamic scenes defining features can be abstracted on a spatial basis. In contrast, basing feature abstraction on motion alone (T-CNN) apparently loses too much information.

The top performing algorithm on the static, moving and entire YUP++ is the newly proposed T-ResNet. It is particularly interesting to compare it to ResNet, as T-ResNet is initialized with ResNet and transformed from the spatial to spatiotemporal domain; see Sec. 2. Surprisingly, this transformation succeeds on the basis of a very small training set, *i.e.* a mere 10% of the dynamic scenes dataset. These results show that well initialized spatial networks can be transformed very efficiently to extract discriminating spatiotemporal information. Indeed, this discrimination tops that of a rival spatiotemporal network, C3D, as well as the best hand-crafted spatiotemporal performer IDT.

Comparing performance on the static (Table 2, left) vs. moving (Table 2, right) camera subsets, it is seen that all algorithms show a decrement in performance in the presence of camera motion. Apparently, the algorithms have difficulty disentangling image dynamics that arise from scene intrinsics vs. camera motion and this is an area where future research should be focused. As it stands, the greatest performance loss is suffered by T-CNN, which suggests that building representations purely on motion information makes extraction of scene intrinsic dynamics especially difficult in the presence of camera motion. The smallest decrease in performance is seen by C3D, which, once again, shows that combined spatial and temporal information provides the strongest basis for dynamic scene characterization, even in the presence of camera motion. Here, it is worth noting that because no previous dynamic scenes dataset contained both static and moving camera examples for each class, it was more difficult to draw such conclusions.

No single algorithm is the top performer across all scene categories (Table 3). It is particularly interesting to compare the two approaches based on hand-crafted features (BoSE and IDT), as the nature of what they are extracting is most explicitly defined. Trajectory-based IDT excels where scenes can be characterized by motion of features across time, *e.g.* the operation of an elevator or the falling of a tree. In complement, spatiotemporal orientation-based BoSE excels where scenes can be characterized by dynamic texture, *e.g.* flickering of forest fires and turbulence of waterfalls. Along similar lines, while T-ResNet is the overall better performer than IDT, it seems that T-ResNet exhibits weaker performance for scene dynamics with rather irregular or mixed defining motion patterns as snowing or fireworks, both categories being resolved quite well by IDT. It also is

Train/Test:	LOO	90/10	70/30	50/50	30/70	10/90
C3D Acc:	98.1	97.6	96.8	94.8	94.2	86.0
BoSE Acc:	96.2	95.3	95.1	94.8	94.1	82.54
SFA Acc:	85.5	84.7	83.4	81.0	80.0	70.0

#split	SFA	BoSE	T-CNN	S-CNN	IDT	C3D
1	51.1	61.9	36.3	68.1	70.4	76.3
2	49.3	60.2	38.8	72.2	69.4	77.6
3	44.8	60.0	36.5	72.8	69.3	78.3

Table 1: Left: Performance of 3 of the previously best approaches for dynamic scene recognition on on the **YUP** [5] dataset. Different train/test ratios have no significant effect on the classification accuracy, except for a very aggressive ratio of 10/90 (i.e. using 3 videos for training and 27 videos for testing per class). Right: Comparison of different algorithms on the **YUP++ moving camera** dataset using a 10/90 train test ratio. Performance levels are consistent across different random splits.

Class	SFA	BoSE	T-CNN	S-CNN	IDT	C3D	ResNet	T-ResNet
Beach	74.1	88.9	85.2	74.1	100.0	92.6	74.1	96.3
BuildingCollapse	74.1	92.6	74.1	96.3	100.0	92.6	100.0	100.0
Elevator	81.5	96.3	100.0	100.0	96.3	100.0	100.0	100.0
Escalator	40.7	66.7	22.2	81.5	51.9	70.4	81.5	88.9
FallingTrees	63.0	63.0	29.6	74.1	96.3	92.6	88.9	77.8
Fireworks	63.0	85.2	44.4	77.8	92.6	85.2	88.9	96.3
ForestFire	25.9	85.2	25.9	96.3	74.1	92.6	92.6	92.6
Fountain	14.8	55.6	22.2	44.4	74.1	33.3	77.8	92.6
Highway	66.7	63.0	55.6	63.0	85.2	70.4	81.5	88.9
LightningStorm	33.3	59.3	88.9	77.8	96.3	81.5	74.1	92.6
Marathon	48.1	85.2	92.6	96.3	88.9	100.0	96.3	100.0
Ocean	96.3	85.2	88.9	100.0	100.0	100.0	100.0	100.0
Railway	33.3	48.1	51.9	88.9	74.1	59.3	81.5	96.3
RushingRiver	66.7	92.6	44.4	96.3	74.1	100.0	100.0	85.2
SkyClouds	85.2	100.0	63.0	96.3	96.3	100.0	96.3	100.0
Snowing	44.4	77.8	63.0	66.7	85.2	51.9	37.0	77.8
Street	96.3	92.6	63.0	100.0	96.3	96.3	100.0	96.3
Waterfall	74.1	66.7	25.9	70.4	33.3	96.3	59.3	70.4
WavingFlags	48.1	81.5	55.6	88.9	100.0	96.3	100.0	96.3
WindmillFarm	92.6	85.2	81.5	96.3	100.0	96.3	100.0	100.0
Average	61.1	78.5	58.9	84.3	85.7	85.4	86.5	92.41

Class	SFA	BoSE	T-CNN	S-CNN	IDT	C3D	ResNet	T-ResNet
Beach	77.8	77.8	18.5	70.4	66.7	81.5	96.3	96.3
BuildingCollapse	44.4	33.3	0.0	40.7	44.4	44.4	40.7	51.9
Elevator	81.5	100.0	77.8	100.0	100.0	100.0	100.0	100.0
Escalator	51.9	74.1	29.6	88.9	59.3	85.2	92.6	96.3
FallingTrees	55.6	77.8	63.0	77.8	96.3	88.9	85.2	96.3
Fireworks	48.1	74.1	25.9	33.3	85.2	77.8	59.3	81.5
ForestFire	29.6	66.7	14.8	88.9	59.3	55.6	88.9	96.3
Fountain	29.6	11.1	18.5	18.5	37.0	25.9	55.6	74.1
Highway	14.8	22.2	29.6	37.0	44.4	48.1	25.9	55.6
LightningStorm	25.9	59.3	59.3	85.2	81.5	85.2	88.9	92.6
Marathon	74.1	77.8	92.6	96.3	100.0	100.0	100.0	100.0
Ocean	40.7	37.0	33.3	51.9	55.6	85.2	22.2	48.1
Railway	18.5	66.7	25.9	92.6	59.3	88.9	100.0	100.0
RushingRiver	55.6	59.3	66.7	81.5	77.8	96.	85.2	85.2
SkyClouds	63.0	70.4	63.0	77.8	55.6	96.3	92.6	92.6
Snowing	14.8	40.7	14.8	22.2	77.8	40.7	25.9	37.0
Street	70.4	85.2	3.7	77.8	85.2	96.3	77.8	92.6
Waterfall	77.8	66.7	18.5	77.8	77.8	88.9	66.7	63.0
WavingFlags	70.4	70.4	51.9	77.8	81.5	74.1	92.6	96.3
WindmillFarm	77.8	66.7	18.5	66.7	63.0	66.7	74.1	74.1
Average	51.1	61.9	36.3	68.1	70.4	76.3	73.5	81.5

Table 2: Performance of different algorithms on the **YUP++ static camera** (left) and **YUP++ moving camera** (right) subsets.

Class	SFA	BoSE	T-CNN	S-CNN	IDT	C3D	ResNet	T-ResNet
Beach	92.6	83.3	72.2	75.9	87.0	83.3	90.7	74.1
BuildingCollapse	66.7	66.7	37.0	81.5	87.0	83.3	83.3	94.4
Elevator	85.2	98.1	79.6	100.0	100.0	98.1	100.0	100.0
Escalator	48.1	74.1	37.0	90.7	66.7	87.0	88.9	92.6
FallingTrees	42.6	79.6	53.7	88.9	98.1	88.9	92.6	88.9
Fireworks	51.9	83.3	38.9	66.7	98.1	81.5	87.0	96.3
ForestFire	29.6	77.8	9.3	92.6	72.2	79.6	96.3	100.0
Fountain	18.5	44.4	11.1	38.9	57.4	35.2	83.3	75.9
Highway	55.6	50.0	50.0	63.0	68.5	64.8	74.1	79.6
LightningStorm	42.6	79.6	77.8	81.5	94.4	87.0	90.7	90.7
Marathon	66.7	88.9	92.6	96.3	98.1	100.0	100.0	100.0
Ocean	64.8	70.4	51.9	83.3	74.1	96.3	66.7	85.2
Railway	29.6	83.3	53.7	96.3	88.9	88.9	100.0	100.0
RushingRiver	55.6	81.5	72.2	87.0	87.0	100.0	88.9	85.2
SkyClouds	83.3	94.4	74.1	90.7	88.9	98.1	96.3	96.3
Snowing	14.8	57.4	33.3	51.9	90.7	46.3	33.3	53.7
Street	79.6	90.7	44.4	92.6	96.3	98.1	100.0	98.1
Waterfall	77.8	85.2	13.0	88.9	66.7	90.7	57.4	75.9
WavingFlags	53.7	81.5	61.1	87.0	98.1	88.9	96.3	98.1
WindmillFarm	79.6	70.4	50.0	87.0	92.6	83.3	94.4	94.4
Average	56.9	77.0	50.6	82.0	85.6	84.0	85.9	89.0

Table 3: Performance comparison of different algorithms on the entire **YUP++ dataset (static and moving camera)**.

interesting to note that the most challenging classes for the spatially-based approaches, S-CNN and ResNet, are those where motion is particularly important, *e.g.* with snowing being the most or second most difficult for each. More generally, classes that are most strongly defined by motion tend to be the most difficult for most algorithms considered, suggesting that capturing differences between scenes based on their dynamics remains an area for additional research.

4.5. Impact of the new dataset

The new **YUP++** dataset has allowed for empirical study of the state-of-the-art in visual recognition approaches applied to dynamic scenes in ways not previously possible. First, by providing increased overall difficulty in comparison to previous datasets, it has allowed for clear performance distinctions to be drawn across a range of algorithms. Second, it has documented that even the strongest extant approaches suffer non-negligible performance decrements when operating in the presence of camera motion in comparison to a stabilized camera scenario. For example, the top overall performer, T-ResNet, has an overall decrement of over 10% in moving from static to moving camera scenarios. Third, the dataset has been shown to have adequate diversity to support ConvNet training on as little as 10% of its total, *e.g.* with T-ResNet transformed from ResNet for great performance improvements on that basis. Fourth, the dataset has provided insight into how different scene characteristics can impact algorithm performance, *e.g.* the relative impact of regular vs. irregular motion patterns.

Moving forward the dataset can continue to support advances in dynamic scene research. First, algorithmic ad-

	UCF101		HMDB51	
State of the art	92.4% [49] 92.5% [13] 93.4% [9]		62.0% [49] 65.4% [13] 66.4% [9]	
Ours	ResNet	T-ResNet	ResNet	T-ResNet
	50 layer model (ResNet-50 [15])			
Appearance	82.3%	85.4%	48.9%	51.3%
Flow	87.0%	89.1%	55.8%	62.0%
Fusion	91.7%	**93.9%**	61.2%	**67.2%**
+ IDT [48]		94.6%		70.6%

Table 4: Classification accuracy for our two-stream ConvNets on UCF101 and HMDB51 without (ResNet) and with our temporal residual (T-ResNet) architecture.

vances focused on discounting camera motion can be developed relative to a dataset that controls exactly for this variable. For example, the impact of image stabilization preprocessing can be studied. Similarly, the development of feature representations that aim for invariance with respect to camera motion can be supported. Second, from a learning perspective the impact of training on stabilized and testing on moving camera scenarios (and vice versa) can be studied. Third, and more generally, given that top performance of the evaluated algorithms exhibits less than 90% accuracy on the entire dataset and less than 82% on the moving camera subset, there is ample room for further benchmarking of improved algorithms using **YUP++**.

4.6. Does T-ResNet generalize to other tasks?

In Table 4 we report the accuracy for our T-ResNet architecture when it is evaluated for action recognition on the popular UCF101 [19] and HMDB51 [21] datasets. In contrast to dynamic scene classification, optical flow is a particularly discriminative input modality for action recognition [22, 35, 47]; so, we apply our temporal residual units to the appearance and flow networks of a two-stream architecture [35]. Our T-ResNet is trained as outlined in Sec. 2.3. For comparison to previous work, testing is done as in [35], by using 25 uniformly sampled input frames/optical flow stacks and their horizontal flips. For T-ResNet a max-pooling layer, Sec. 2.2, pools over the temporal dimension and the network produces a single prediction for the input (instead of averaging the 25 frame predictions as in [35]). Fusion of the two streams is implemented by averaging their prediction layer outputs (without applying softmax as in [35]). For HMDB51 we weight the temporal network scores by a factor of three before averaging scores [35].

In Table 4 we observe that ResNet on its own does not perform much better than a VGG-16 two-stream network (which produces 91.4% and 58.5% on UCF101 and HMDB51 [49], resp.), but enjoys the advantage of less parameters and being around twice as fast (ResNet-50 has 3.8 billion FLOPs *vs*. 19.6 in VGG-16).

A more interesting comparison comes with our proposed T-ResNet architecture, which provides a healthy performance boost to both appearance and flow streams. The large gains of 91.7% *vs*. 93.9% on UCF101 and 61.2% *vs*. 67.2% on HMDB51 can be explained by the temporal residual units operating over long temporal extents on the input, which the two-stream baseline is not capable of providing. More generally, T-ResNet clearly outperforms state-of-the-art approaches [13, 49] and benefits further from simple addition of SVM scores from IDT-FV [48] trajectory features.

The most competitive approach, ST-ResNet [9] uses temporal convolution kernels that are initialized by replicating pre-trained spatial 1×1 kernels over time; *i.e*. the averaging of feature maps over time. We conjecture that this sum-initialization is suboptimal for capturing characteristic temporal patterns. Our approach, however, uses temporal filters that are trained from scratch and therefore are able to receive more informative gradients. Another conceptual advantage of our proposed spatiotemporal residual unit over [9] is that it can effectively ignore or enhance temporal information by an extra non-linearity packed into the temporal residual path.

The performance boost provided by T-ResNet comes at a very low cost, since each of our temporal filters is 1×1 in the spatial dimensions and only spans 3 instances in time. In fact, our T-ResNet is much faster in evaluation than the two-stream baseline (which averages the predictions for the 25 frames), as T-ResNet is fully convolutional in space-time. During testing we propagate the 25 inputs through T-ResNet, globally pool in time after the last conv-layer and end up with a single prediction for the whole video in less than a second.

5. Summary

We have presented a general spatiotemporal ConvNet, T-ResNet, based on transforming a purely spatial network to one that can encompass spacetime via hierarchical injection of temporal residuals. In comparison to a representative set of strong performing alternative approaches to video-based recognition, our approach has produced the best overall performance on a novel dynamic scene recognition dataset. We have also shown that T-ResNet can perform competitively on the complementary task of action recognition, opening potential applications to other areas.

Our new database extends previous dynamic scenes evaluation sets in both diversity and size: It adds new scene classes and provides balanced samples with and without camera motion. Significantly, all algorithms show a decrement in performance when confronted with camera motion, suggesting that a promising research direction for future studies is the development of approaches that are robust to this variable.

Acknowledgments. This work was partly supported by the Austrian Science Fund (FWF P27076) and NSERC.

References

[1] N. Ballas, L. Yao, C. Pal, and A. C. Courville. Delving deeper into convolutional networks for learning video representations. In *Proc. ICLR*, 2016. 2

[2] BBC Motion Gallery. http://www.bbcmotiongallery.com. 5

[3] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proc. CVPR*, 2016. 3

[4] M. Cimpoi, S. Maji, and A. Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proc. CVPR*, 2015. 5

[5] K. Derpanis, M. Lecce, K. Daniilidis, and R. P. Wildes. Dynamic scene understanding: The role of orientation features in space and time in scene classification. In *Proc. CVPR*, 2012. 1, 2, 4, 5, 6, 7

[6] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proc. CVPR*, 2015. 2

[7] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *IJCV*, 51(2):91–109, 2003. 2

[8] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *Proc. CVPR*, 2005. 1

[9] C. Feichtenhofer, A. Pinz, and R. Wildes. Spatiotemporal residual networks for video action recognition. In *Proc. NIPS*, 2016. 2, 8

[10] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spacetime forests with complementary features for dynamic scene recognition. In *Proc. BMVC*, 2013. 2, 5

[11] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Bags of spacetime energies for dynamic scene recognition. In *Proc. CVPR*, 2014. 1, 2, 5, 6

[12] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Dynamic scene recognition with complementary spatiotemporal features. *PAMI*, 38(12):2389–2401, 2016. 2, 4

[13] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proc. CVPR*, 2016. 2, 8

[14] Getty Images. http://www.gettyimages.com. 5

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 2, 3, 4, 5, 8

[16] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *Proc. ECCV*, 2016. 2, 4

[17] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. ICML*, 2015. 4

[18] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proc. CVPR*, 2014. 2

[19] A. R. Z. Khurram Soomro and M. Shah. Ucf101: A dataset of 101 human actions calsses from videos in the wild. Technical Report CRCV-TR-12-01, UCF Center for Research in Computer Vision, 2012. 5, 8

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*, 2012. 2, 3, 4

[21] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proc. ICCV*, 2011. 5, 8

[22] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proc. CVPR*, 2008. 8

[23] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, 2006. 1

[24] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 3

[25] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *Proc. CVPR*, 2009. 1, 2

[26] J. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proc. CVPR*, 2015. 1

[27] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proc. CVPR*, 2015. 1, 2

[28] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42:145–175, 2001. 1

[29] P. Over, G. Awad, M. Michael, J. Fiscus, G. Sanders, W. Kraaij, A. Smeaton, and Q. Quenot. Trecvid 2013 - an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proc. TRECVID*, 2013. 1

[30] D. Park, C. L. Zitnick, D. Ramanan, and P. Dollár. Exploring weak stabilization for motion feature extraction. In *Proc. CVPR*, 2013. 1

[31] Y. Poleg, A. Ephrat, S. Peleg, and C. Arora. Compact CNN for indexing egocentric videos. In *Proc. CVPR*, 2015. 1, 2

[32] M. Potter. Recognition and memory for briefly presented scenes. *Frontiers in Psychology*, 3(32):1–9, 2012. 1

[33] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *Proc. CVPR*, 2009. 5

[34] N. Shroff, P. Turaga, and R. Chellappa. Moving vistas: Exploiting motion for describing scenes. In *Proc. CVPR*, 2010. 1, 2, 4, 5

[35] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proc. NIPS*, 2014. 1, 2, 5, 8

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2014. 2, 3, 4

[37] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using LSTMs. In *Proc. ICML*, 2015. 2

[38] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016. 3, 4

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. CVPR*, 2015. 3

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567*, 2015. 3

[41] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *Proc. ECCV*. 2010. 2

[42] C. Theriault, N. Thome, and M. Cord. Dynamic scene clas-

sification: Learning motion descriptors with slow features analysis. In *Proc. CVPR*, 2013. 2, 5, 6

[43] A. Torralba, K. Murphy, and W. Freeman. Using the forest to see the trees: Object recognition in context. *Commun. ACM*, 53:107–114, 2010. 1

[44] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. In *Proc. ICCV*, 2015. 1, 2, 4, 5, 6

[45] A. Vailaya, M. Figueiredo, A. Jain, and H. Zhang. Image classification for content-based indexing. *TIP*, 10:117–130, 2001. 1

[46] A. Vedaldi and K. Lenc. MatConvNet – convolutional neural networks for MATLAB. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015. 2

[47] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, pages 1–20, 2013. 5, 8

[48] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proc. ICCV*, 2013. 2, 8

[49] X. Wang, A. Farhadi, and A. Gupta. Actions ˜ transformations. In *Proc. CVPR*, 2016. 8

[50] Z. Xu, Y. Yang, and A. G. Hauptmann. A discriminative CNN video representation for event detection. 2015. 1

[51] YouTube. http://www.youtube.com. 5

[52] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. In *Proc. ICLR*, 2014. 3