Learning Video Object Segmentation with Visual Memory

Pavel Tokmakov, Karteek Alahari, Cordelia Schmid

Learning Video Object Segmentation with Visual Memory

Pavel Tokmakov Karteek Alahari Cordelia Schmid

Inria*

Abstract

This paper addresses the task of segmenting moving objects in unconstrained videos. We introduce a novel two-stream neural network with an explicit memory module to achieve this. The two streams of the network encode spatial and temporal features in a video sequence respectively, while the memory module captures the evolution of objects over time. The module to build a "visual memory" in video, i.e., a joint representation of all the video frames, is realized with a convolutional recurrent unit learned from a small number of training video sequences. Given a video frame as input, our approach assigns each pixel an object or background label based on the learned spatio-temporal features as well as the "visual memory" specific to the video, acquired automatically without any manually-annotated frames. The visual memory is implemented with convolutional gated recurrent units, which allows to propagate spatial information over time. We evaluate our method extensively on two benchmarks, DAVIS and Freiburg-Berkeley motion segmentation datasets, and show state-of-the-art results. For example, our approach outperforms the top method on the DAVIS dataset by nearly 6%. We also provide an extensive ablative analysis to investigate the influence of each component in the proposed framework.

1. Introduction

Video object segmentation is the task of extracting spatio-temporal regions that correspond to object(s) moving in at least one frame in the video sequence. The top-performing methods for this problem [11, 31] continue to rely on hand-crafted features and do not leverage a learned video representation, despite the impressive results achieved by convolutional neural networks (CNN) for other vision tasks, e.g., image segmentation [35], object detection [36]. Very recently, there have been attempts to build CNNs for video object segmentation [6, 23, 43]. They are indeed the first to use deep learning methods for video segmentation, but suffer from various drawbacks. For example, [6, 23] rely on a manually-segmented subset of frames (typically the first frame of the video sequence) to guide the segmentation pipeline. Our previous work [43] relies solely on optical flow between pairs of frames to segment independently moving objects in a video, making it susceptible to errors in flow estimation. It also can not extract objects if they stop moving. Furthermore, none of these methods has a mechanism to *memorize* relevant features of objects in a scene. In this paper, we propose a novel framework to address these issues; see sample results in Figure 1.

Figure 1. Sample results on the DAVIS dataset. Segmentations produced by MP-Net [43] (left) and our approach (right), overlaid on the video frame.

We present a two-stream network with an explicit memory module for video object segmentation (see Figure 2). The memory module is a convolutional gated recurrent unit (GRU) that encodes the spatio-temporal evolution of object(s) in the input video sequence. This spatio-temporal representation used in the memory module is extracted from two streams—the appearance stream which describes static features of objects in the video, and the temporal stream which captures motion cues.

The appearance stream is the DeepLab network [7] pretrained on the PASCAL VOC segmentation dataset and operates on individual video frames. The temporal one is a motion prediction network [43] pretrained on the synthetic FlyingThings3D dataset and takes optical flow computed from pairs of frames as input, as shown in Figure 2. The two streams provide complementary cues for object segmenta-

*Thoth team, Inria, Laboratoire Jean Kuntzmann, Grenoble, France.

tion. With these spatio-temporal CNN features in hand, we train the convolutional GRU component of the framework to learn a *visual memory* representation of object(s) in the scene. Given a frame t from the video sequence as input, the network extracts its spatio-temporal features and then: (i) computes the segmentation using the memory representation aggregated from all frames previously seen in the video, and (ii) updates the memory unit with features from t. The segmentation is improved further by processing the video bidirectionally in the memory unit, with our *bidirectional convolutional GRU*.

The contributions of the paper are two-fold. First, we present an approach for moving object segmentation in unconstrained videos that does not require any manually-annotated frames in the input video (see §3). Our network architecture incorporates a memory unit to capture the evolution of object(s) in the scene (see §4). To our knowledge, this is the first recurrent network based approach to accomplish the video segmentation task. It helps address challenging scenarios where the motion patterns of the object change over time; for example, when an object in motion stops to move, abruptly, and then moves again, with potentially a different motion pattern. Second, we present state-of-the-art results on two video object segmentation benchmarks, namely DAVIS [34] and Freiburg-Berkeley motion segmentation (FBMS) dataset [30] (see §5.5). Additionally, we provide an extensive experimental analysis, with ablation studies to investigate the influence of all the components of our framework (see §5.3) and visualize the internal states of our memory unit (see §5.6). We will make the source code and the models available online.

2. Related work

Video object segmentation. Several approaches have been proposed over the years to accomplish the task of segmenting objects in video. One of the more successful ones presented in [4] clusters pixels spatio-temporally based on motion features computed along individual point trajectories. Improvements to this framework include dense trajectory-level segmentation [29], an alternative clustering method [22], and detection of discontinuities in the trajectory spectral embedding [13]. These trajectory based approaches lack robustness in cases where feature matching fails.

An alternative to using trajectories is formulating the segmentation problem as a foreground-background classification task [25, 31, 44]. These methods first estimate a region [31, 44] or regions [25], which correspond(s) to the foreground object, and then use them to compute foreground and background appearance models. The final object segmentation is obtained by integrating these appearance models with other cues, e.g., saliency maps [44], shape estimates [25], pairwise constraints [31]. Variants to this framework have introduced occlusion relations to compute a layered video segmentation [41], and long-range interactions to group re-occurring regions in video [11]. Two methods from this class of segmentation approaches [11,31] show a good performance on the DAVIS benchmark. While our proposed method is similar in spirit to this class of approaches, in terms of formulating segmentation as a classification problem, we differ from previous work significantly. We propose an integrated approach to learn appearance and motion features and update them with a memory module, in contrast to estimating an initial region heuristically and then propagating it over time. Our robust model outperforms all these methods [11, 25, 31, 41, 44], as shown in Section 5.5.

Video object segmentation is also related to the task of segmenting objects in motion, irrespective of camera motion. Two recent methods to address this task use optical flow computed between pairs of frames [3, 43]. Classical methods in perspective geometry and RANSAC-based feature matching are used in [3] to estimate moving objects from optical flow. It achieved state-of-the-art performance on a subset of the Berkeley motion segmentation (BMS) dataset [4], but lacks robustness due to a heuristic initialization, as shown in the evaluation on DAVIS in Table 3. Our previous approach MP-Net [43] learns to recognize motion patterns in a flow field. This frame-based approach is the state of the art on DAVIS and is on par with [3] on the BMS subset. Despite its excellent performance, MP-Net is limited by its frame-based nature and also overlooks appearance features of objects. These issues are partially addressed in a heuristic post-processing step with objectness cues in [43]. Nevertheless, the approach fails to extract objects if they stop moving, i.e., if no motion cues are present. We use MP-Net as the temporal stream of our approach (see Figure 2). We show a principled way to integrate this stream with appearance information and a new visual memory module based on convolutional gated recurrent units (ConvGRU). As shown in Table 2, our approach outperforms MP-Net.

Very recently, two CNN-based methods for video object segmentation were proposed [6, 23]. Starting with CNNs pre-trained for image segmentation, they find objects in video by fine-tuning on the first frame in the sequence. Note that this setup, referred to as semi-supervised segmentation, is very different from the more challenging unsupervised case we address in this paper, where no manually-annotated frames are available for the test video. Furthermore, these two CNN architectures are primarily developed for images, and do not model temporal information in video. We, on the other hand, propose a recurrent network specifically for the video segmentation task.

Recurrent neural networks (RNNs). RNN [21, 37] is a popular model for tasks defined on sequential data. Its main component is an internal state that allows to accumulate in-

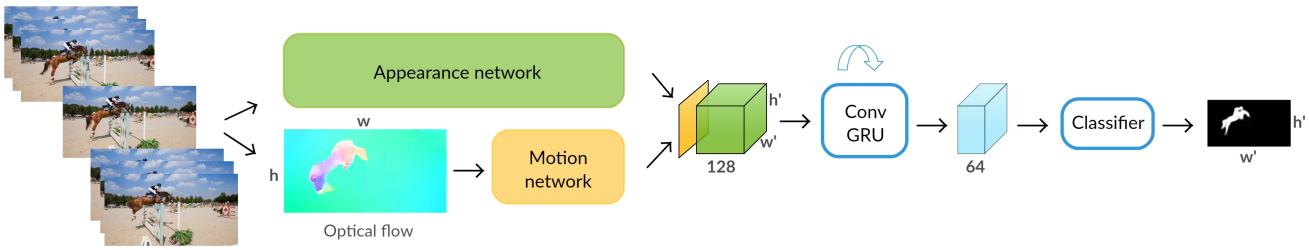


Figure 2. Overview of our segmentation approach. Each video frame is processed by the appearance (green) and the motion (yellow) networks to produce an intermediate two-stream representation. The ConvGRU module combines this with the learned visual memory to compute the final segmentation result. The width (w') and height (h') of the feature map and the output are $w/8$ and $h/8$ respectively.

formation over time. The internal state in classical RNNs is updated with a weighted combination of the input and the previous state, where the weights are learned from training data for the task at hand. Long short-term memory (LSTM) [20] and gated recurrent unit (GRU) [8] architectures are improved variants of RNN, which partially mitigate the issue of vanishing gradients [19, 32]. They introduce gates with learnable parameters, to update the internal state selectively, and can propagate gradients further through time.

Recurrent models, originally used for text and speech recognition, e.g., [17, 27], are becoming increasingly popular for visual data. Initial work on vision tasks, such as image captioning [9], future frame prediction [40] and action recognition [28], has represented the internal state of the recurrent models as a 1D vector—without encoding any spatial information. LSTM and GRU architectures have been extended to address this issue with the introduction of ConvLSTM [12, 33, 38] and ConvGRU [2] respectively. In these convolutional recurrent models the state and the gates are 3D tensors and the weight vectors are replaced by 2D convolutions. These models have only recently been applied to vision tasks, such as video frame prediction [12, 33, 38], action recognition and video captioning [2].

In this paper, we employ a visual memory module based on a convolutional GRU (ConvGRU) and show that it is an effective way to encode the spatio-temporal evolution of objects in video for segmentation. Further, to fully benefit from all the frames in a video sequence, we apply the recurrent model bidirectionally [16, 18], i.e., apply two identical model instances on the sequence in forward and backward directions, and combine the predictions for each frame.

3. Approach

Our model takes video frames together with their estimated optical flow as input, and outputs binary segmentations of moving objects, as shown in Figure 2. We target the most general form of this task, wherein objects are to be segmented in the entire video if they move in at least one frame. The proposed model is comprised of three key components: appearance and motion networks, and a visual memory module.

Appearance network. The purpose of the appearance stream is to produce a high-level encoding of a frame that will later aid the visual memory module in forming a representation of the moving object. It takes an RGB frame as input and produces a $128 \times w/8 \times h/8$ feature representation (shown in green in Figure 2). This encodes the semantic content of the scene. We use a state-of-the-art CNN for this stream, namely the largeFOV version of the DeepLab network [7]. This network relies on dilated convolutions [7], which preserve a relatively high spatial resolution of features, and also incorporate context information in each pixel's representation. It is pretrained on a semantic segmentation dataset, PASCAL VOC 2012 [10], resulting in features that can distinguish objects from background as well as from each other—a crucial aspect for the video object segmentation task. We extract features from the fc6 layer of the network, which has a feature dimension of 1024 for each pixel. This feature map is further passed through two 1×1 convolutional layers, interleaved with *tanh* nonlinearities, to reduce the dimension to 128. These layers are trained together with ConvGRU (see §5.2 for details).

Motion network. For the temporal stream we employ MP-Net [43], a CNN pretrained for the motion segmentation task. It is trained to estimate independently moving objects (i.e., irrespective of camera motion) based on optical flow computed from a pair of frames as input (shown in yellow in Figure 2). This stream produces a $w/4 \times h/4$ motion prediction output, where each value represents the likelihood of the corresponding pixel being in motion. Its output is further downsampled by a factor 2 (in w and h) to match the dimensions of the appearance stream output.

The intuition behind using two streams is to benefit from their complementarity for building a strong representation of objects that evolves over time. For example, both appearance and motion networks are equally effective when an object is moving in the scene, but as soon as it becomes stationary, the motion network can not estimate the object, unlike the appearance network. We leverage this complementary nature, as done by two-stream networks for other

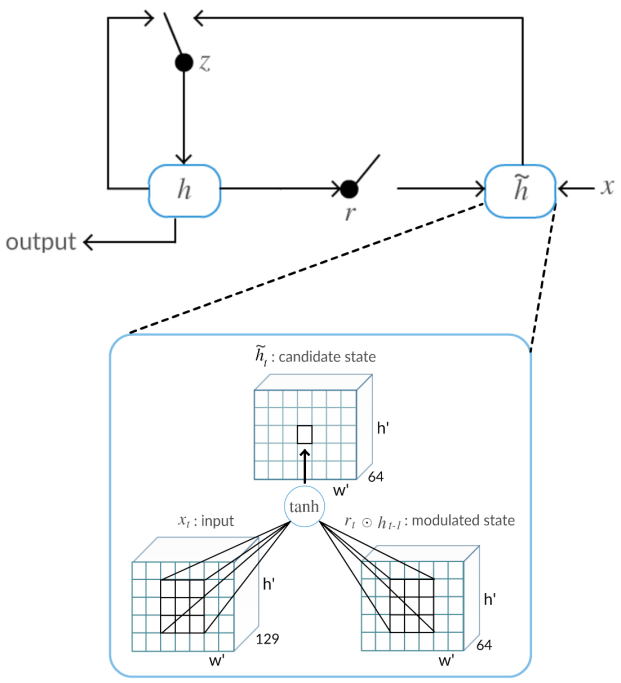


Figure 3. Illustration of ConvGRU with details for the candidate hidden state module, where $\tilde{h}_t$ is computed with two convolutional operations and a $\tanh$ nonlinearity.

vision tasks [39]. Note that our approach is not specific to the particular networks described above, but is in fact a general framework for video object segmentation. As shown is the Section 5.3, its components can easily be replaced with other networks, providing scope for future improvement.

Memory module. The third component, i.e., a visual memory module based on convolutional gated units (ConvGRU), takes the concatenation of appearance and motion stream outputs as its input. It refines the initial estimates from these two networks, and also memorizes the appearance and location of objects in motion to segment them in frames where: (i) they are static, or (ii) motion prediction fails; see the example in Figure 1. The output of this ConvGRU memory module is a $64\times\mathrm{w}/8\times\mathrm{h}/8$ feature map obtained by combining the two-stream input with the internal state of the memory module, as described in detail in Section 4. We further improve the model by processing the video bidirectionally; see Section 4.1. The output from the ConvGRU module is processed by a 1×1 convolutional layer and softmax nonlinearity to produce the final pixelwise segmentation result. These layers are trained together with ConvGRU, as detailed in Section 5.2.

4. Visual memory module

The key component of the ConvGRU module is the state matrix h, which encodes the visual memory. For frame t in the video sequence, ConvGRU uses the two-stream representation x_t and the previous state h_{t-1} to compute the new state h_t. The dynamics of this computation are guided by an update gate z_t, a forget gate r_t. The states and the gates are 3D tensors, and can characterize spatio-temporal patterns in the video, effectively memorizing which objects move, and where they move to. These components are computed with convolutional operators and nonlinearities as follows.

$$z_t = \sigma(x_t * w_{xz} + h_{t-1} * w_{hz} + b_z), \quad (1)$$
$$r_t = \sigma(x_t * w_{xr} + h_{t-1} * w_{hr} + b_r), \quad (2)$$
$$\tilde{h}_t = \tanh(x_t * w_{x\tilde{h}} + r_t \odot h_{t-1} * w_{h\tilde{h}} + b_{\tilde{h}}), \quad (3)$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t, \quad (4)$$

where $\odot$ denotes element-wise multiplication, $*$ represents a convolutional operation, σ is the sigmoid function, w's are learned transformations, and b's are bias terms.

The new state h_t in (4) is a weighted combination of the previous state h_{t-1} and the candidate memory $\tilde{h}_t$. The update gate z_t determines how much of this memory is incorporated into the new state. If z_t is close to zero, the memory represented by $\tilde{h}_t$ is ignored. The reset gate r_t controls the influence of the previous state h_{t-1} on the candidate memory $\tilde{h}_t$ in (3), i.e., how much of the previous state is let through into the candidate memory. If r_t is close to zero, the unit forgets its previously computed state h_{t-1}.

The gates and the candidate memory are computed with convolutional operations over x_t and h_{t-1} shown in equations (1-3). We illustrate the computation of the candidate memory state $\tilde{h}_t$ in Figure 3. The state at $t-1$, h_{t-1}, is first multiplied (element-wise) with the reset gate r_t. This modulated state representation and the input x_t are then convolved with learned transformations, $w_{h\tilde{h}}$ and $w_{x\tilde{h}}$ respectively, summed together with a bias term $b_{\tilde{h}}$, and passed through a $\tanh$ nonlinearity. In other words, the visual memory representation of a pixel is determined not only by the input and the previous state at that pixel, but also its local neighborhood. Increasing the size of the convolutional kernels allows the model to handle spatio-temporal patterns with larger motion.

The update and reset gates, z_t and r_t, are computed in an analogous fashion using a sigmoid function instead of $\tanh$. Our ConvGRU applies a total of six convolutional operations at each time step. All the operations detailed here are fully differentiable, and thus the parameters of the convolutions (w's and b's) can be trained in an end-to-end fashion with back propagation through time [45]. In summary, the model learns to combine appearance features of the current frame with the memorized video representation to refine motion predictions, or even fully restore them from the previous observations in case a moving object becomes stationary.

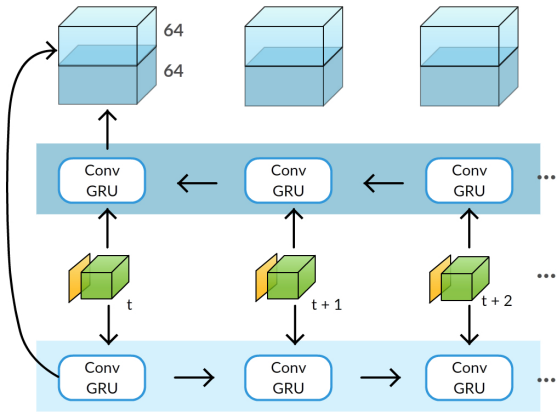


Figure 4. Illustration of the bidirectional processing with our ConvGRU module.

4.1. Bidirectional processing

Consider an example where an object is stationary at the beginning of a video sequence, and starts to move in the latter frames. Our approach described so far, which processes video frames sequentially (in the forward direction), can not segment the object in the initial frames. This is due to the lack of prior memory representation of the object in the first frame. We improve our framework with a bidirectional processing step, inspired by the application of recurrent models bidirectionally in the speech domain [16, 18].

The bidirectional variant of our ConvGRU is illustrated in Figure 4. It is composed of two ConvGRU instances with identical learned weights, which are run in parallel. The first one processes frames in the forward direction, starting with the first frame (shown at the bottom in the figure). The second instance process frames in the backward direction, starting with the last video frame (shown at the top in the figure). The activations from these two directions are concatenated at each time step, as shown in the figure, to produce a $128 \times w/8 \times h/8$ output. It is then passed through a 3×3 convolutional layer to finally produce a $64 \times w/8 \times h/8$ for each frame. Pixel-wise segmentation is then obtained with a final 1×1 convolutional layer and a softmax nonlinearity, as in the unidirectional case.

Bidirectional ConvGRU is used both in training and in testing, allowing the model to learn to aggregate information over the entire video. In addition to handling cases where objects move in the latter frames, it improves the ability of the model to correct motion prediction errors. As discussed in the experimental evaluation, bidirectional ConvGRU improves segmentation performance by nearly 3% on the DAVIS dataset (see Table 1). The influence of bidirectional processing is more prominent on the FBMS dataset, where objects can be static in the beginning of a video, with 5% improvement over the unidirectional variant.

4.2. Training

We train our visual memory module with the back propagation through time algorithm [45], which unrolls the recurrent network for n time steps and keeps all the intermediate activations to compute the gradients. Thus, our ConvGRU model, which has 6 internal convolutional layers, trained on a video sequence of length n, is equivalent to a $6n$ layer CNN for the unidirectional variant, or $12n$ for the bidirectional model at training time. This memory requirement makes it infeasible to train the whole model, including appearance and motion streams, end-to-end. We resort to using pretrained versions of the appearance and motion networks and train the ConvGRU.

We use the training split of the DAVIS dataset [34] for learning the ConvGRU weights. Objects move in all the frames in this dataset, which biases the memory module towards the presence of an uninterrupted motion stream. This results in the ConvGRU learned from this data failing, when an object stops to move in a test sequence. We augment the training data to simulate such *stop-and-go* scenarios and thus learn a more robust model for realistic videos.

We create additional training sequences, where ground truth moving object segmentation (instead of responses from the motion network) is provided for all the frames, except for the last five frames, which are duplicated, simulating a case where objects stop moving. No motion input is used for these last five frames. These artificial examples are used in place of the regular ones for a fixed fraction of iterations. Replacing motion stream predictions with ground truth segmentations for these sequences allows to decouple the task of motion mistake correction from the task of object tracking, which simplifies the learning. Given that ground truth segmentation determines the loss for training, i.e., it is used for all the frames, ConvGRU explicitly memorizes the moving object in the initial part of the sequence, and then segments it in frames where motion is missing. We do a similar training set augmentation by duplicating the first five frames in a batch, to simulates the cases where an object is static in the beginning of a video.

5. Experiments

5.1. Datasets and evaluation

We use three datasets in the experimental analysis: DAVIS for training and test, FBMS only for test, and FT3D for training a variant of our approach.

DAVIS. It contains 50 full HD videos with accurate pixel-level annotation in all the frames [34]. The annotations correspond to the task of video object segmentation. Following the 30/20 training/validation split provided with the dataset, we train on the 30 sequences, and test on the 20 validation videos. We also follow the standard protocol for evaluation from [34], and report intersection over union, F-measure for

Aspect	Variant	Mean IoU
Ours (fc6, ConvGRU, Bidir, DAVIS)		70.1
App stream	no	43.5
	RGB	58.3
	2-layer CNN	60.9
	DeepLab fc7	69.8
	DeepLab conv5	67.7
App pretrain	ImageNet only	64.1
Motion stream	no	59.6
Memory module	ConvRNN	68.7
	ConvLSTM	68.9
Bidir processing	no	67.2
Train data	FT3D GT Flow	55.3
	FT3D LDOF Flow	59.6

Table 1. Ablation study on the DAVIS validation set showing variants of appearance and motion streams and memory module. "Ours" refers to the model using fc6 appearance features together with a motion stream, and a bidirectional ConvGRU trained on DAVIS.

contour accuracy and temporal stability.

FBMS. The Freiburg-Berkeley motion segmentation dataset [30] is composed of 59 videos with ground truth annotations in a subset of the frames. In contrast to DAVIS, it has multiple moving objects in several videos with instance-level annotations. Also, objects may move only in a fraction of the frames, but they are annotated in frames where they do not exhibit independent motion. The dataset is split into training and test sets. Following the standard protocol on this dataset [22], we do not train on any of these sequences, and evaluate separately for both with precision, recall and F-measure scores. We also convert instance-level annotation to binary ones by merging all the foreground labels into a single category, as in [41].

FT3D. The FlyingThings3D dataset [26] consists of 2250 synthetic videos for training, composed of 10 frames, where objects are in motion along random trajectories in rendered scenes. Ground truth optical flow, depth, camera parameters, and instance segmentations are provided by [26], and the ground truth motion segmentation is available from [1].

5.2. Implementation details

We train our model by minimizing binary cross-entropy loss using back-propagation through time and RMSProp [42] with a learning rate of 10^{-4}. The learning rate is gradually decreased after every epoch. The weight decay is set to 0.005. Initialization of all the convolutional layers, except for those inside the ConvGRU, is done with the standard *xavier* method [14]. We clip the gradients so that they lie in a predefined range before each parameter update, to avoid numerical issues [15]. We form batches of size 14 by randomly selecting a video, and a subset of 14 consecutive frames in it. Random cropping and flipping of the sequences is also performed for data augmentation. Our full model uses 7×7 convolutions in all the ConvGRU operations. The weights of the two 1×1 convolutional (dimensionality reduction) layers in the appearance network and the final 1×1 convolutional layer following the memory module are learned jointly with the memory module. The model is trained for 30000 iterations and the proportion of batches with additional sequences (see Section 4.2) is set to 20%.

Method	Mean IoU
Ours	70.1
Ours + CRF	75.9
MP-Net	53.6
MP-Net + Obj	63.3
MP-Net + Obj + FST (MP-Net-V)	55.0
MP-Net + Obj + CRF (MP-Net-F)	70.0

Table 2. Comparison to MP-Net [43] variants on the DAVIS validation set. "Obj" refers to the objectness cues used in [43]. MP-Net-V(ideo) and MP-Net-F(rame) are the variants of MP-Net which use FST [31] and CRF respectively, in addition to objectness.

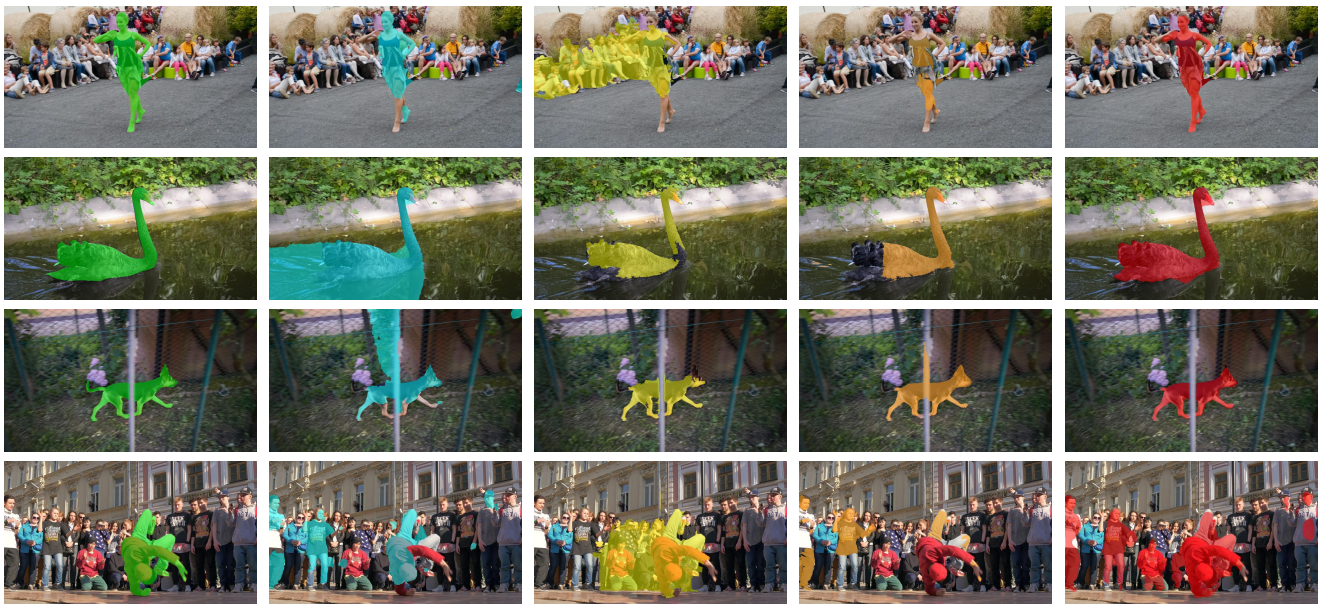
Our final model uses a fully-connected CRF [24] to refine boundaries in a post-processing step. The parameters of this CRF are taken from [43]. In the experiments where objectness is used, it is also computed according to [43]. We use LDOF [5] for optical flow estimation and convert the raw flow to flow angle field, as in [43]. We used the code and the trained models for MP-Net available at [1]. Our method is implemented in the Torch framework and will be made available online. Many sequences in FBMS are several hundred frames long and do not fit into GPU memory during evaluation. We apply our method in a sliding window fashion in such cases, with a window of 130 frames and a step size of 50.

5.3. Ablation study

Table 1 demonstrates the influence of different components of our approach on the DAVIS validation set. First, we study the role of the appearance stream. As a baseline, we remove it completely ("no" in the "App stream" in the table), i.e., the output of the motion stream is the only input to our visual memory module. In this setting, the memory module lacks sufficient information to produce accurate segmentations, which results in an 26.6% drop in performance compared to the method where the appearance stream with fc6 features is used ("Ours" in the table). We then provide raw RGB frames, concatenated with the motion prediction, as input to the ConvGRU. This simplest form of image representation leads to a 14.8% improvement, compared to the motion only model, showing the importance of the appearance features. The variant where RGB input is passed through two convolutional lay-

Measure		PCM [3]	CVOS [41]	KEY [25]	MSG [4]	NLC [11]	CUT [22]	FST [31]	MP-Net-F [43]	Ours
$\mathcal{J}$	Mean	40.1	48.2	49.8	53.3	55.1	55.2	55.8	70.0	75.9
	Recall	34.3	54.0	59.1	61.6	55.8	57.5	64.9	85.0	89.1
	Decay	15.2	10.5	14.1	2.4	12.6	2.3	0.0	1.4	0.0
$\mathcal{F}$	Mean	39.6	44.7	42.7	50.8	52.3	55.2	51.1	65.9	72.1
	Recall	15.4	52.6	37.5	60.0	51.9	61.0	51.6	79.2	83.4
	Decay	12.7	11.7	10.6	5.1	11.4	3.4	2.9	2.5	1.3
$\mathcal{T}$	Mean	51.3	24.4	25.2	29.1	41.4	26.3	34.3	56.3	25.5

Table 3. Comparison to state-of-the-art methods on DAVIS with intersection over union ($\mathcal{J}$), F-measure ($\mathcal{F}$), and temporal stability ($\mathcal{T}$).



Ground truth CUT [22] FST [31] MP-Net-Frame [43] Ours

Figure 5. Qualitative comparison with top-performing methods on DAVIS. Left to right: ground truth, results of CUT [22], FST [31], MP-Net-Frame [43], and our method.

ers, interleaved with tanh nonlinearities, that are trained jointly with the memory module ("2-layer CNN"), further improves this. This shows the potential of learning appearance representation as a part of the video segmentation pipeline. Finally, we compare features extracted from the fc7 and conv5 layers of the DeepLab model to those from fc6 used by default in our method. Features from fc7 and fc6 show comparable performance, but fc7 ones are more expensive to compute. Conv5 features perform significantly worse, perhaps due to a smaller field of view.

The importance of appearance network pretrained on the semantic segmentation task is highlighted by the "ImageNet only" variant in Table 1, where the PASCAL VOC pretrained segmentation network is replaced with a network trained on ImageNet classification. Although ImageNet pretraining provides a rich feature representation, it is less suitable for the video object segmentation task, which is confirmed by an 6% drop in performance. Discarding the motion information ("no" in "Motion stream"), although being 10.5% below our complete method, still outperforms most of the motion-based approaches on DAVIS (see Table 3). This variant learns foreground/background segmentation, which is sufficient for videos with a single dominant object, but fails in more challenging cases.

Next, we evaluate the design choices in the visual memory module. Using a simple recurrent model result in a slight decrease in performance. Such simpler architectures can be used in case of a memory vs segmentation quality trade off. The other variant using ConvLSTM is comparable to ConvRNN, possibly due to the lack of sufficient training data. Performing a unidirectional processing instead of a bidirectional one decreases the performance by nearly 3% ("no" in "Bidir processing").

Lastly, we train two variants ("FT3D GT Flow" and "FT3D LDOF Flow") on the synthetic FT3D dataset [26] instead of DAVIS. Both of them show a significantly lower performance than our method trained on DAVIS. This is due to the appearance of synthetic FT3D videos being very different from the real-world ones. The variant trained on ground truth flow (GT Flow) is inferior to that trained on LDOF flow because the motion network (MP-Net) achieves a high performance on FT3D with ground truth flow, and

Measure	Set	KEY [25]	MP-Net-F [43]	FST [31]	CVOS [41]	CUT [22]	MP-Net-V [43]	Ours
$\mathcal{P}$	Training	64.9	83.0	71.3	79.2	86.6	69.3	90.7
	Test	62.3	84.0	76.3	83.4	83.1	81.4	92.1
$\mathcal{R}$	Training	52.7	54.2	70.6	79.0	80.3	80.8	71.3
	Test	56.0	49.4	63.3	67.9	71.5	73.9	67.4
$\mathcal{F}$	Training	58.2	65.6	71.0	79.3	83.4	74.6	79.8
	Test	59.0	62.2	69.2	74.9	76.8	77.5	77.8

Table 4. Comparison to state-of-the-art methods on FBMS with precision ($\mathcal{P}$), recall ($\mathcal{R}$), and F-measure ($\mathcal{F}$).

thus our visual memory module learns to simply follow the motion stream output.

5.4. Comparison to MP-Net variants

In Table 2 we compare our method to MP-Net and its variants presented in [43] on the DAVIS validation set. Our visual memory-based approach ("Ours" in the table) outperforms the MP-Net baseline ("MP-Net"), which serves as the motion stream in our model, by 16.5%. This clearly demonstrates the value of the appearance stream and our memory unit for video segmentation. The post-processing variants in [43], using objectness cues, CRF, and video segmentation method [31], improve this baseline, but remain inferior to our result. Our full method ("Ours + CRF") is nearly 6% better than "MP-Net-Frame", which is the best performing MP-Net variant on DAVIS. Note that "MP-Net-Video" which combines MP-Net with objectness cues and the video segmentation method of [31] is also inferior to our method, as it relies strongly on the tracking capabilities of [31], which is prone to segmentation *leaking* in case of errors in the flow estimation. The example in the first row in Figure 5 shows a typical error of [31].

MP-Net-Video performs better than MP-Net-Frame on the FBMS dataset (see Table 4) since the frame-only variant does not segment objects when they stop moving. The propagation of segment(s) over time with tracking in MP-Net-Video addresses this, but is less precise due to segmentation leaks, as shown by the comparison with precision measure in the table and the qualitative results in Figure 6.

5.5. Comparison to the state-of-the-art

DAVIS. Table 3 compares our approach to the state-of-the-art methods on DAVIS. In addition to comparing our results to the top-performing unsupervised approaches reported in [34], we evaluated two more recent methods: CUT [22] and PCM [3], with the authors' implementation. Our method outperforms MP-Net-Frame, the previous state of the art, by 5.9% on the IoU measure, and is 20.1% better than the next best method [31]. We also observe a 30.8% improvement in temporal stability over MP-Net-Frame. PCM [3], which performs well on a subset of the FBMS dataset (as shown in [43]), is in fact significantly worse on DAVIS.

CUT [22] MP-Net-Video [43] Ours

Figure 6. Qualitative comparison with top-performing methods on FBMS. Left to right: results of CUT [22], MP-Net-Video [31], and our method.

Figure 5 shows qualitative results of our approach, and the next three top-performing methods on DAVIS: MP-Net-Frame [43], FST [31] and CUT [22]. In the first row, both CUT and our method segment the dancer, but our result is more accurate. FST leaks to the people in the background and MP-Net misses parts of the person due to the incorrectly estimated objectness. Our approach does not include any heuristics, which makes it robust to this type of errors. In the second row, all the methods are able to segment the swan, but only our method segments it completely and also does not leak into the background. In the next row, our approach shows very high precision, being able to correctly separate the dog and the pole occluding it. In the last row, we illustrate a failure case of our method. The people in the background move in some of the frames in this example. CUT, MP-Net and our method segment them to varying extents. FST focuses on the foreground object, but leaks to the background partially nevertheless.

FBMS. As shown in Table 4 MP-Net-Frame [43] is outperformed by most of the methods on this dataset. Our approach based on visual memory outperforms MP-Net-Frame by 15.6% on the test set and by 14.2% on the training set according to the F-measure. FST [31] based

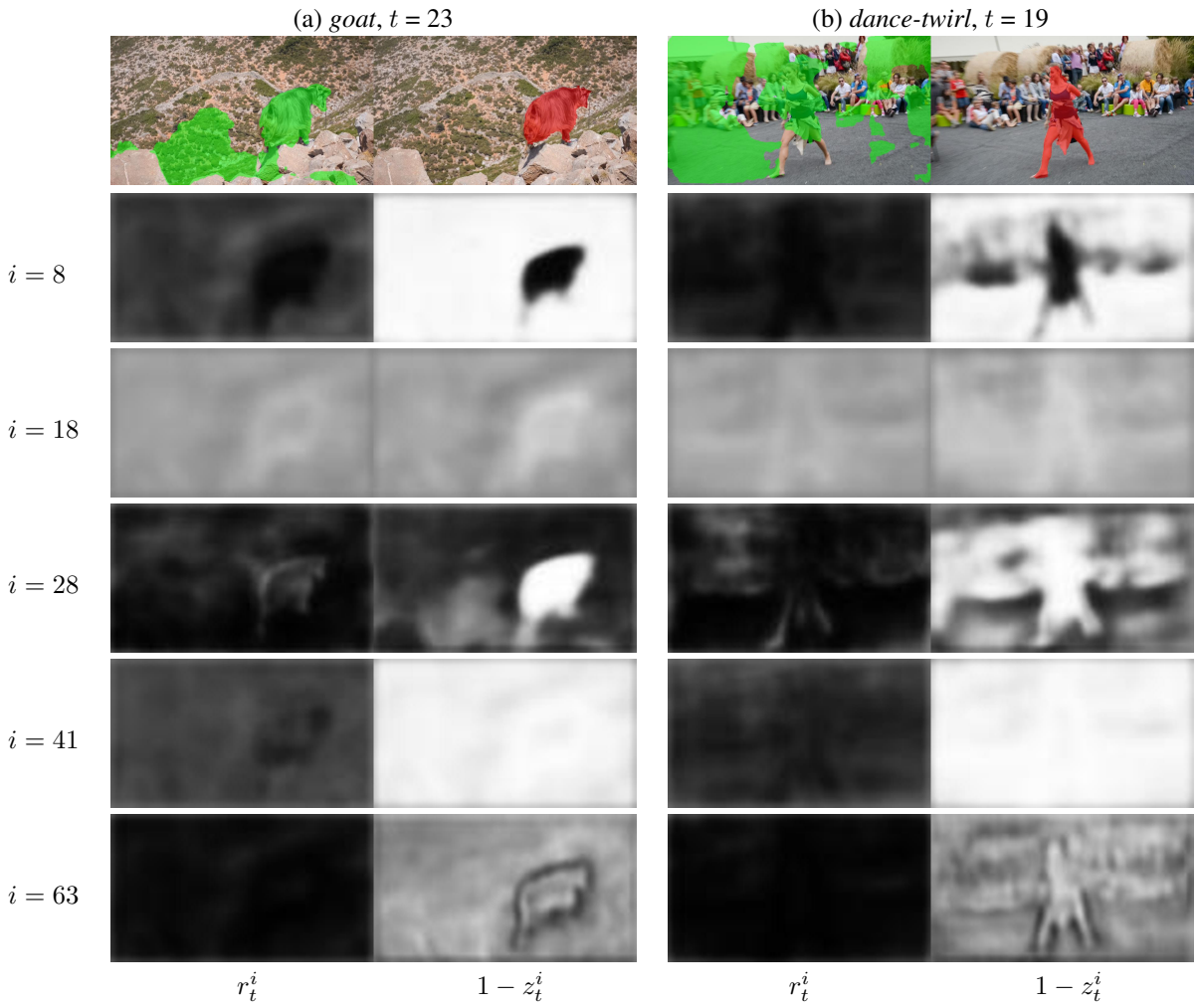


Figure 7. Visualization of the ConvGRU gate activations for two sequences from the DAVIS validation set. The first row in each example shows the motion stream output and the final segmentation result. The other rows are the reset (r_t) and the inverse of the update ($1 - z_t$) gate activations for the corresponding ith dimension. These activations are shown as grayscale heat maps, where white denotes a high activation.

post-processing ("MP-Net-V" in the table) significantly improves the results of MP-Net on FBMS, but it remains below our approach on both precision and F-measure. Overall, our method shows top results in terms of precision and F-measure but is outperformed by some methods on recall. This is due to very long static sequences present in FBMS, which our recurrent memory-based method can not handle as well as methods with explicit tracking components, such as CUT [22].

Figure 6 shows qualitative results of our method and the two next-best methods on FBMS: MP-Net-Video [43] and CUT [22]. MP-Net-Video relies highly on FST's [31] tracking capabilities, and thus demonstrates the same background leaking failure mode, as seen in all the three examples. CUT misses parts of objects and incorrectly assigns background regions to the foreground in some cases, whereas our method demonstrates very high precision.

5.6. ConvGRU visualization

We present a visualization of the gate activity in our ConvGRU unit on two videos from the DAVIS validation set. We use the unidirectional model in the following for better clarity. The reset and update gates of the ConvGRU, r_t and z_t respectively, are 3D matrices of $64 \times h/8 \times w/8$ dimension. The overall behavior of ConvGRU is determined by the interplay of these 128 components. We use a selection of the components of r_t and $(1 - z_t)$ to interpret the workings of the gates. Our analysis is shown on two frames which correspond to the middle of the *goat* and *dance-twirl* sequences in (a) and (b) of Figure 7.

The outputs of the motion stream alone (left) and the final segmentation result (right) of the two examples are shown in the top row in the figure. The five rows below correspond each to one of the 64 dimensions of r_t and $(1 - z_t)$.

These activations are shown as a grayscale heat map. High values for either of the two activations increases the influence of the previous state of a ConvGRU unit in the computation of the new state matrix. If both values are low, the state in the corresponding locations is rewritten with a new value; see equations (3) and (4).

For $i = 8$, we observe the update gate being selective based on the appearance information, i.e., it updates the state for foreground objects and duplicates it for the background. Note that motion does not play a role in this case. This can be seen in the example of stationary people (in the background) on the right, that are treated as foreground by the update gate. In the second row, showing responses for $i = 18$, both heatmaps are uniformly close to 0.5, which implies that the new features for this dimension are obtained by combining the previous state and the input at the time step t.

In the third row for $i = 28$, the update gate is driven by motion. It keeps the state for regions that are predicted as moving, and rewrites it for other regions in the frame. For the fourth row, where $i = 41$, r_t is uniformly close to 0, whereas $(1 - z_t)$ is close to 1. As a result, the input is effectively ignored and the previous state is duplicated. In the last row showing $i = 63$, a more complex behavior can be observed, where the gates rewrite the memory for regions in object boundaries, and use both the previous state and the current input for other regions in the frame.

6. Conclusion

This paper introduces a novel approach for video object segmentation. Our method combines two complementary sources of information: appearance and motion, with a visual memory module, realized as a bidirectional convolutional gated recurrent unit. The ConvGRU module encodes spatio-temporal evolution of objects in a video and uses this encoding to improve motion segmentation. The effectiveness of our approach is validated on the DAVIS and FBMS datasets, where it shows top performance. Instance-level video object segmentation is a promising direction for future work.

Acknowledgments. This work was supported in part by the ERC advanced grant ALLEGRO, the MSR-Inria joint project, a Google research award and a Facebook gift. We gratefully acknowledge the support of NVIDIA with the donation of GPUs used for this research.

References

[1] Learning motion patterns in videos. http://thoth.inrialpes.fr/research/mpnet. 6

[2] N. Ballas, L. Yao, C. Pal, and A. Courville. Delving deeper into convolutional networks for learning video representations. *ICLR*, 2016. 3

[3] P. Bideau and E. G. Learned-Miller. It's moving! A probabilistic model for causal motion segmentation in moving camera videos. In *ECCV*, 2016. 2, 7, 8

[4] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010. 2, 7

[5] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *PAMI*, 2011. 6

[6] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video segmentation. In *CVPR*, 2017. 1, 2

[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015. 1, 3

[8] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, 2014. 3

[9] 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 *CVPR*, 2015. 3

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 3

[11] A. Faktor and M. Irani. Video segmentation by non-local consensus voting. In *BMVC*, 2014. 1, 2, 7

[12] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *NIPS*, 2016. 3

[13] K. Fragkiadaki, G. Zhang, and J. Shi. Video segmentation by tracing discontinuities in a trajectory embedding. In *CVPR*, 2012. 2

[14] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010. 6

[15] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013. 6

[16] A. Graves, N. Jaitly, and A. Mohamed. Hybrid speech recognition with deep bidirectional LSTM. In *Workshop on Automatic Speech Recognition and Understanding*, 2013. 3, 5

[17] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013. 3

[18] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. 3, 5

[19] S. Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 1998. 3

[20] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[21] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. National Academy of Sciences*, 79(8):2554–2558, 1982. 2

[22] M. Keuper, B. Andres, and T. Brox. Motion trajectory segmentation via minimum cost multicuts. In *ICCV*, 2015. 2, 6, 7, 8, 9

[23] A. Khoreva. Learning video object segmentation from static images. *arXiv preprint arXiv:abs/1612.02646*, 2016. 1, 2

[24] P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *NIPS*, 2011. 6

[25] Y. J. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In *ICCV*, 2011. 2, 7, 8

[26] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 6, 7

[27] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. In *Interspeech*, 2010. 3

[28] J. Y. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015. 3

[29] P. Ochs and T. Brox. Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *ICCV*, 2011. 2

[30] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 36(6):1187–1200, 2014. 2, 6

[31] A. Papazoglou and V. Ferrari. Fast object segmentation in unconstrained video. In *ICCV*, 2013. 1, 2, 6, 7, 8, 9

[32] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. *ICML*, 2013. 3

[33] V. Patraucean, A. Handa, and R. Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *ICLR Workshop track*, 2016. 3

[34] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 2, 5, 8

[35] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. *ECCV*, 2016. 1

[36] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1

[37] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 1986. 2

[38] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. Wong, and W. Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015. 3

[39] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 4

[40] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using LSTMs. In *ICML*, 2015. 3

[41] B. Taylor, V. Karasev, and S. Soatto. Causal video object segmentation from persistence of occlusions. In *CVPR*, 2015. 2, 6, 7, 8

[42] T. Tieleman and G. Hinton. RMSProp. COURSERA: Lecture 6.5 - Neural Networks for Machine Learning, 2012. 6

[43] P. Tokmakov, K. Alahari, and C. Schmid. Learning motion patterns in videos. In *CVPR*, 2017. 1, 2, 3, 6, 7, 8, 9

[44] W. Wang, J. Shen, and F. Porikli. Saliency-aware geodesic video object segmentation. In *CVPR*, 2015. 2

[45] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proc. IEEE*, 78(10):1550–1560, 1990. 4, 5