DynamoNet: Dynamic Action and Motion Network

Ali Diba[1,*], Vivek Sharma[2,*], Luc Van Gool[1,3], Rainer Stiefelhagen[2]
[1]ESAT-PSI, KU Leuven, [2]CV:HCI, KIT, [3]CVL, ETH Zürich
{first.last}@esat.kuleuven.be, {first.last}@kit.edu

Abstract

In this paper, we are interested in self-supervised learning the motion cues in videos using dynamic motion filters for a better motion representation to finally boost human action recognition in particular. Thus far, the vision community has focused on spatio-temporal approaches using standard filters, rather we here propose dynamic filters that adaptively learn the video-specific internal motion representation by predicting the short-term future frames. We name this new motion representation, as dynamic motion representation (DMR) and is embedded inside of 3D convolutional network as a new layer, which captures the visual appearance and motion dynamics throughout entire video clip via end-to-end network learning. Simultaneously, we utilize these motion representation to enrich video classification. We have designed the frame prediction task as an auxiliary task to empower the classification problem.

With these overall objectives, to this end, we introduce a novel unified spatio-temporal 3D-CNN architecture (DynamoNet) that jointly optimizes the video classification and learning motion representation by predicting future frames as a multi-task learning problem. We conduct experiments on challenging human action datasets: Kinetics 400, UCF101, HMDB51. The experiments using the proposed DynamoNet show promising results on all the datasets.

1. Introduction

Human action recognition [3, 6, 7, 9, 55] in videos has attracted a huge attention over the last decade, due to the potential applications in video surveillance, understanding, analysis, retrieval tasks and more. In practice, the performance of computer vision systems still falls behind that of humans. On top of the challenges that make object class recognition hard, complicating aspects like camera motion and the continuously changing viewpoints negatively influences the vision system. Although, Convolutional Neural Networks (ConvNets) have successfully upsurged several sub-fields of vision, for action recognition particularly, they still lack to model the motion cues effectively.

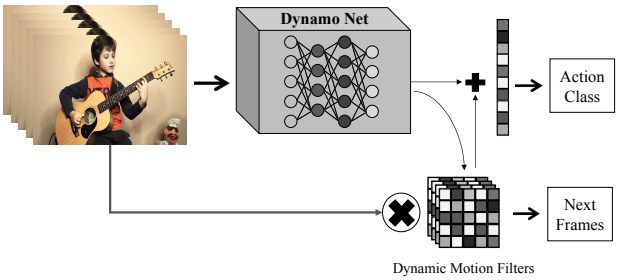

Figure 1. Overview of the proposed multi-task learning ConvNet architecture to simultaneously classify action and learn motion representation by predicting future frames in end-to-end learning manner. $\times$, $+$ denotes convolution and concatenation operations.

Neural networks for action recognition can be mainly categorized into two types, namely *architecture-driven* ConvNets [42, 49] (which utilize standard filters and pooling kernels in a video architecture to exploit long-range dynamics) and *encoding-driven* ConvNets [9, 16] (which integrate new encoding methods in addition to standard filters and pooling kernels in a video architecture to learn spatio-temporal feature representation). This paper falls in the pool of *encoding-driven* ConvNets, where we extend the standard video understanding architecture to incorporate a new layer to learn dynamic motion representation conditioned on the *action-specific* basis.

In this paper, we propose to extend the training of ConvNets-based action classification to incorporate the high-level goal to learn *action-specific* motion representation via future frame prediction. Our contribution named as DynamoNet is a method that jointly optimizes a ConvNet for video action classification and future frame prediction as a multi-task learning problem. We achieve this by adaptively learning the motion features on a *video-specific* basis via dynamic motion filters, which enables the motion prediction model to selectively utilize only those motion features that lead to improved video classification.

Since we understand the vital role of motion feature representation, we propose to use the dynamic convolutional

*Ali Diba and Vivek Sharma contributed equally to this work and listed in alphabetical order.

filters to dynamically discover and learn *video-specific* internal motion representations for improved video classification (see Fig.1 for a graphical overview). Our paper is inspired from [1, 38]. However, while Brabandere et al. [1] applies the dynamic filters to transform an angle to a filter (steerable filter) and Sharma et al. [38] emulates a range of enhancement filters to generate image enhancement methods. We used the same terminology as in [1, 38]. In contrast to these works, our work differs substantially in technical approach and the application scope. Our DynamoNet is designed to learn motion representation, which we achieve by adaptively extracting informative features by predicting the short-term future frames to improve classification. We believe predicting (or reconstructing) the future frames selectively transfers the *fundamental notion* of motion content to the filters, which in turn improve the overall effectiveness of the motion representations. The motion representation learning is jointly optimized along with the video classification as a multi-task learning problem. Using short-term future frame prediction as a proxy task is promising, and we clearly show that this works for an accurate action recognition in videos.

Precisely, our network takes the current video-clip with T frames and generates $T+1$ future frames using T dynamic motion kernels or dynamic filters. The network structure is based on 3D ConvNets. Specifically, given an input video-clip x with T frames the network generates T dynamic motion kernels (F_T) to predict the consecutive next frame given the previous one i.e. $F_t : x_t \rightarrow \hat{x}_{t+1}$, $t \in \{1, \ldots, T\}$. These filters are video dependent motion kernels and are conditioned on the input and therefore vary from one sample to another during training and testing phase, which means that the filter dynamically extracts important motion representation from a given input. Further, we utilize these T d-dimensional dynamic motion features along with STC-ResNext [6, 20] 3D-ConvNet features for video classification. Moreover, we believe the dynamic motion kernels capture the important concepts of motion representation from the temporal cue and the extracted motion features are robust and compact global temporal representation for the whole video, this makes them a perfect fit for action recognition task well. Our method is evaluated on three challenging benchmark action recognition datasets, namely UCF101, HMDB51 and Kinetics 400. We experimentally show that the 3D ConvNets when combined with our dynamic motion filters (see Sec. 4) achieve state-of-the-art performance on UCF101 (97.8%), HMDB51 (76.8%) and Kinetics 400 (77.9%).

2. Background and Related Work

Action Recognition with ConvNets. With Convolutional Neural Networks (ConvNets) the vision community has successfully made huge leaps forward in several sub-fields of vision and has outperformed hand-engineered representations by a significant margin particularly for action recognition. End-to-end ConvNets have been introduced in [13, 24, 42, 49, 57] to exploit the appearance and the temporal information. These methods operate on 2D (individual image-level) or 3D (video-clips or snippets of K frames). In the 2D setting, spatial and/or temporal information are modeled via LSTMs/RNNs to capture long-term motion cues [10, 65], or via feature pooling and encoding methods using Bilinear models [9], Vector of Locally Aggregated Descriptors (VLAD) [16], and Fisher vector encoding (FVs) [48]. While, in the 3D settings, the input to the network consists of either RGB video clips or stacked optical-flow frames to capture the long-term temporal information. The filters and pooling kernels for these architectures are 3D (x, y, time) i.e. 3D convolutions ($s \times s \times d$) [65] where d is the kernel temporal depth and s is the kernel spatial size. Simonyan et al. [42] used two-stream 2D ConvNets cohorts of RGB images and a stack of 10 optical-flow frames as input. Tran et al. [49] on the other side explored 3D ConvNets with fixed kernel size of $3 \times 3 \times 3$, where spatio-temporal feature learning for clips of 16 RGB frames was performed. In this way, they avoid to calculate the optical flow explicitly and still achieve good performance. Further, in [50, 51] Tran et al. extended the ResNet architecture with 3D convolutions. In [13] Feichtenhofer et al. propose 3D pooling. Sun et al. [47] decomposed the 3D convolutions into 2D spatial and 1D temporal convolutions. Wang et al. [57] propose to use sparsely sampled non-overlapping frames from the whole clip as input for both spatial and temporal streams and then combine their scores in a late fusion approach. Carreira et al. [3] propose converting a pre-trained 2D ConvNet [22] to 3D ConvNet by inflating the filters and pooling kernels with an additional temporal dimension d. All these architectures have fixed temporal 3D convolution kernel depths throughout the whole architecture. In T3D [7], Diba et al. propose temporal transition layer that models variable temporal convolution kernel depths over shorter and longer temporal ranges. Furthermore in [6], Diba et al. propose spatio-temporal channel correlation that models correlations between channels of a 3D ConvNets wrt. both spatial and temporal dimensions. In contrast to these prior works, our work differs substantially in scope and technical approach. We propose an architecture to learn dynamic motion filters for modeling an effective internal motion representation in order to adaptively extract informative motion features conditioned on a video-specific basis to improve action recognition.

Finally, it is worth noting the self-supervised learning works on "harvesting" training data from unlabeled sources for action recognition. Fernando *et al*. [14] and Mishra *et al*. [31] shuffle the video frames and treat them as positive/negative training data; Sharma et al. [37, 39] mines la-

bels using a distance matrix based on similarity although for video face clustering; Wei et al. [59] divides a single clip into non-overlapping 10-frame chunks, and then predict the ordering task; Ng et al. [32] estimates optical flow while recognizing actions. We compare all these methods against our unsupervised future frame prediction based ConvNet training in the experimental section.

Future Frame Prediction. Given an observed image or a sequence of frames predict the future frames is very popular these days [15, 18, 33, 40, 44, 60, 61, 66], where mostly researchers predict low-level pixels or motion, until recently image synthesis is done via neural networks: Generative adversarial networks [5, 17, 34, 53, 54], variational auto-encoders [25, 63, 64], deep regression networks [52] to anticipate the visual representation in the future. These works basically form to be a part of either deterministic prediction frameworks [41, 52] or probabilistic prediction frameworks [53, 63]. It has been shown that the probabilistic content-aware motion prediction models the motion fields or image features better in comparison to deterministic models which cannot model the uncertainty well. Successful predicting the future will demonstrate the computers understanding of objects in the scene. One straight forward approach would be discriminate training, etc to predict the next future frames, but our world is full of uncertainty. Its not the predictions are wrong, its just an intrinsic ambiguity in the prediction. In contrast to previous work, our main goal is not to predict future frame, but rather to learn motion representations by predicting future frames. Precisely, we propose a method that predicts high-level concepts such as objects and actions by learning dynamic motion filters to predict the consecutive next frame given the previous one in a self-supervised manner.

Further, in the similar spirit of frame prediction although for different tasks using action recognition datasets, it is worth noting works of Mathieu et al. [29] and Srivastava et al. [45]. We quantitatively compare to Mathieu et al. [29] in our experiments.

Filter Generating Networks. Starting with the seminal work of Jaderberg et al. [23] where the authors propose transformation filters to do translation and rotation, all these papers [1, 26, 38] utilize the same concept (deep-down) to learn a steerable filter [1], weather prediction filter [26], or an enhancement filter [38] using input-output image pairs. Different from these works, we propose to apply these filters for learning motion representation in videos with an overall goal to improve action classification. We clearly show that this works in our experiments.

Learning Motion Representation. We share the same motivation with these works [8, 11, 35, 36, 46] for learning motion representation in deep learning fashion, but without using optical flow information as the target output.

3. Proposed Method

Our aim is to learn a dynamic motion representation model with an overall goal to improve video classification. To this end, we propose two ConvNet architectures described in this section. Our first architecture is proposed to learn dynamic motion filters by predicting the short-term future frames in an end-to-end self-supervised learning fashion. The other proposed end-to-end network is designed to simultaneously classify action, in addition to learn motion representation by predicting the future frames.

We use 3D ConvNets [20, 6], STCnet/3D-ResNext architecture as the base models, and incorporated two branches to do classification and frames prediction. The input to the network is a stack of 16, 32 or 64 frames in different experimental setups, which we refer as a video clip.

3.1. Dynamic Motion Filters

The filter generating network (DynamoNet) is inspired from [1, 26, 38] and is composed of 3D filters and pooling kernels, with the last fully-connected layer (i.e., dynamic motion filter parameters). The DynamoNet is self-supervised or unsupervised. The DynamoNet maps the input to the filter. Precisely, the network takes a video-clip x with T frames and outputs filters $F_{\Theta,t}, \Theta \in \mathbb{R}^{s\times s\times t}$, where Θ is the transformation parameter that learns the mapping, s is the spatial kernel size and t is the number of filters - which is driven by the input stack of frames i.e. $t \in T$.

Given an input video-clip $x \in \mathbb{R}^{H\times W\times T}$, the network produces dynamic motion filters to predict the consecutive future frames $\hat{x} \in \mathbb{R}^{H\times W\times T}$ where H, W denotes the frame height and width. The scheme is illustrated in Fig. 2. The future frame predictor network can be formulated as:

$$\hat{x}_{t+1} = F_{\Theta,t}(x_t) \tag{1}$$

The $F_{\Theta,t}$ motion filters are convolved with the input t^{th} frame x_t to generate $\hat{x}_{t+1}$ frame. $F_{\Theta,t}$ is applied to x_t at every spatial position (H, W) to output the predicted frame $\hat{x}_{t+1} \in \mathbb{R}^{H\times W}$. Note that the filters are sample-specific and are conditioned on the input x_t. In Fig. 2, we show filter generating network to predict dynamic filters and thus predicting the future frames. The filter size determines the receptive field and is application dependent. From the literature [38], we exploited a lot of insights about kernel sizes. For motion prediction, we have tested with different filter sizes $s = \{3, 4, 5, 6, 7\}$ and for our setup we found that a filter size of 5×5 gave the best result, and others ($> 5\times5$ or $< 5 \times 5$) produced smoother images, with a drop in classification performance by approx $\sim$2-3%. Further, we found that frame prediction done at deep-level performed better in comparison to the intermediate levels.

For generating the motion filter parameters, the network is trained using Huber loss function [4] between the target

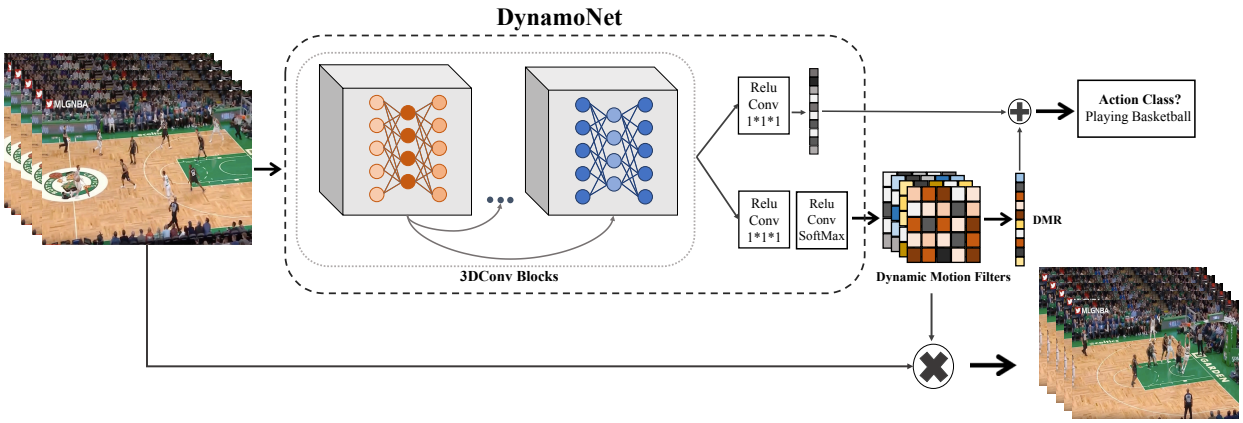

Figure 2. **DynamoNet.** Input to the network is a video with T frames, that generates T dynamic motion filters to predict the consecutive next frame given the previous one to learn motion representation. The motion filters are then concatenated together to form a global representation, along with the STC-ResNext features and then fed to classifier. The network is jointly optimized with a classification objective to adaptively extract informative motion features for improved classification. $\times$, $+$ denotes convolution and concatenation operations.

future frame x_{t+1} and the network's predicted future frame $\hat{x}_{t+1}$. The frame prediction (FP) using the Huber loss function is defined as:

$$\mathcal{L}_{FP} = \begin{cases} \frac{1}{2}||\hat{x}_{t+1} - x_{t+1}||_2^2 & \textbf{if} \quad ||\hat{x}_{t+1} - x_{t+1}||_1 < \delta \\ \delta||\hat{x}_{t+1} - x_{t+1}||_1 - \frac{1}{2}\delta^2 & otherwise \end{cases} \tag{2}$$

where the threshold δ is set to 0.01. More details on the training and the network architecture is discussed in the experimental section.

Our future frames prediction method differs from all the recent methods, as we utilize motion filters to synthesize the next frames given the previous one, $F_t : x_t \rightarrow \hat{x}_{t+1}, t \in \{1, \ldots, T\}$. The main difference of our method for generating next frames in comparison to other methods differs substantially in technical approach, we do not generate frames directly by conv-deconv layers, rather we use these layers to generate motion filters to reconstruct and predict the next frames. We believe predicting (or reconstructing) the future frames selectively transfers the *fundamental notion* of motion content to the filters, which in turn improve the overall effectiveness of the motion representations. Furthermore, in this way, our network configuration and the learning scheme helps to learn the pixel motion information in an spatio-temporal regime.

Inspired by dynamic filter networks [1], we believe for learning an effective motion representation from a video, dynamically generated filters is a robust solution to discover and capture video-specific internal motion variations. As the parameters of the filters are conditioned on the input, they vary from one sample to another which is perfect for learning internal motion representations and variations in the video. By extracting this intra-information from the clip, we believe we model motion representation from the same clip, since the generated filters are demonstrating the dynamic information of frames. This representation can be effective for video classification task, in the next section we show how joint optimization can be modeled for motion representation learning and classification - simultaneously to have a more solid action classifier.

Example frame prediction structure: Let's assume that a given input clip has 16 frames, the predicted future frames are 16 frames consisting 15 reconstructed ones (2nd to 16th frames from input) and a new future frame obtained from the 16th frame. Each predicted (reconstructed) frame is obtained by applying the dynamic motion filter to the previous frame (input frame) at every spatial position. In this way, the dynamic motion filters are learned by predicting the consecutive next frame given the previous one.

3.2. Action Recognition

We utilize 3D ConvNets over 2D ConvNets for video classification and frame prediction because of their compelling advantages of exploiting long-range temporal cues rather than merely spatial cues. Precisely, we use the recently proposed STCnet [6] or 3D-ResNet/ResNext [20] as the main building block of the DynamoNet. We chose these architectures because of their promising performance for action classification both in terms of accuracy and high computation speed. Further, the 3D temporal convolution kernels efficiently capture the visual appearance and the temporal information across frames in videos in comparison to 2D convolutions - which lacks to model the temporal dimension. That being said, 3D-STCnet/ResNet are good candidates to extract the spatio-temporal feature representations for action classification and motion analysis.

Here, we recycle the dynamic motion filters architecture from Section 3.1. Figure 2 shows the schematic layout of

the whole architecture. Our architecture has two network branches, one branch learns the dynamic motion representations $F_{\Theta,T}$ and the second branch is the standard fully-connected layer of the 3D-STCnet (AR). Both of the network branches are trained together, therefore we have the action representation (AR) and dynamic motion representation learned at the same time. As the last step to action classification, we flatten the motion filters and then followed by a fully-connected layer of size $\mathbb{R}^d$, the d-dimensional DMR is then concatenated with AR and then fed to classification layer. By this design we incorporate the motion information for action classification task.

End-to-end learning. Finally, we now extend the loss of approach 1, by adding the softmax-loss (classification) for joint optimization of motion filter learning by future frame prediction with a classification objective. The total loss of the whole pipeline is given by:

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{FP} + \beta \mathcal{L}_{Classification} \tag{3}$$

where α, β are losses weights and both of the tasks are optimized together. We show it both qualitatively and quantitatively that the trained DMR is optimized to capture an accurate motion information for sample-specific actions in this manner. Figure 3 shows qualitative results with dynamic filter based predicted frames.

3.3. Unsupervised Training

For comparison with the previous self-supervised or unsupervised representation learning methods [14, 31, 32, 59], we remove the classification branch and just keep the dynamic motion filters with the frame prediction part only, an unsupervised video learning pipeline is obtained. As we already know, the important aspect of videos are meaningful motions. Using our approach, in practice, one can easily learn motion representation in an unsupervised way by simple reconstructing and/or predicting the future frames via the self supervisory signal available in the sequence of frames.

We have investigated the method with a number of unlabeled videos and trained the network from scratch. We show that such a unsupervised pre-training can be very beneficial for a stable model weight initialization, and thus this reduces the need of large labeled video datasets for training 3D ConvNets from scratch for the action classification task. More details on the training schemes and their results are discussed in the experimental section.

4. Experiments

In this section, we first introduce the datasets, implementation details of our proposed approach, and then show the applicability of unsupervised pre-training, followed by the role of frame prediction in the training scheme. Finally, we compare our method with the state-of-the-art methods on three challenging human action and activities datasets.

4.1. Network Design

The DynamoNet consists of three parts: first is the 3D-Conv which in our experiments is STCnet or 3D-ResNet/ResNext with different depths. We have applied more layers for two branches, for the action classification part to extract an efficient representation, we added two conv-layer (64 filters each) and a fully connected layer. On the frame prediction, we use 2 conv-layers (64 filters each) with a softmax layer to yield the dynamic filters. After flattening the filters, there is a fully connected layer with size of 512 to extract the dynamic motion representation. The AR and DMR features are concatenated together and then fed to the classification loss. In the rest of this section, we use the backbone architecture: STC-ResNext101 and ResNext101 for the DynamoNet (STCnet) and DynamoNet (ResNext).

4.2. Datasets

We evaluated our proposed DynamoNet on three challenging human action and activities datasets; HMDB51 [28], UCF101 [43] and Kinetics [3]. We use the pre-defined training/testing splits and protocols provided originally. We report the mean average accuracy over the three splits for HMDB51 and UCF101 and for Kinetics, we report the performance on the validation set.

Kinetics. Kinetics is a challenging human action recognition dataset introduced by [3], which contains 400 and 600 action classes. There are two versions of this dataset: untrimmed and trimmed. The untrimmed videos contain the whole video in which the activity is included in a short period of it. However, the trimmed videos contain the activity part only. We evaluate our models on the trimmed version. We use all training videos for training our models from scratch.

UCF101. To evaluate our DynamoNet action recognition performance, we first trained it on the Kinetics dataset and then fine-tuned on UCF101. Furthermore, we also evaluate our models by training them from scratch on UCF101 using randomly initialized weights and unsupervised pre-training method to study the impact of pre-training on a huge dataset such as Kinetics, and also the unsupervised pre-training method.

HMDB51. For HMDB51, we employ the same methodology as UCF101 and we fine-tune the DynamoNet on HMDB51, which was pre-trained on Kinetics. Also, we similarly evaluate our model by training it from scratch on HMDB51 using randomly initialized weights.

4.3. Implementation Details

We have used the PyTorch framework for the implementation and all the networks are trained on 8 P100 NVIDIA

GPUs. Here, we describe the implementation details of our proposed DynamoNet, frame prediction and action classification.

Training. We train our DynamoNet with just frame prediction part as the pre-training step on 500K unlabeled video clips from YouTube8M dataset. DynamoNet operates on a stack of 16 or 32 RGB frames. We have resized the video frames to 122px when smaller and then randomly do the 5 crops (and their horizontal flips) of size 112×112 as the main network input size. For the network weight initialization, we adopt the same technique proposed in [21]. For the model training, we use SGD, Nesterov momentum of 0.9, weight decay 10^{-4} and batch size of 64. The learning rate for start is set to 0.1 and reduced by a factor of 10 manually when the validation loss is saturated. To train the total loss, we set the coefficients of the losses as: $\alpha = 0.1$ and $\beta = 1.0$. Once the unsupervised pre-training is done, the main training with both branches of action recognition and frame prediction is done on Kinetics dataset with a maximum number of 200 epochs. We also employ batch normalization for network training. In our experiments, we use different version of STCnet and 3D-ResNet/ResNext as the main convolutional section of DynamoNet, since they are state-of-the-art methods in 3D-CNN action models. We have evaluated different depth of these networks in our experiments. STCnet has similar structure of 3D-ResNet with an extra module to handle spatio-temporal channel correlations in Conv layers.

Testing. For action recognition on videos, we separate each video into non-overlapping clips of 16/32/64 frames. The DynamoNet is applied over the video clips by taking a 112×112 center-crop, and for a video-level prediction, we average the prediction scores over all clips in a video.

4.4. Unsupervised Pre-Training

Since the frame prediction part can be trained separately without the need of labeled video, we have studied the effect of unsupervised pre-training the 3D-Conv part of network which carries most of information. As mentioned before the network is trained for frame prediction on 500K unlabeled video clips from YouTube8m. While doing pre-training, the action classification part is detached. After the pre-training is done, both of the branches are activated to be trained also for action classification as well.

We fine-tune with lower learning rate the self-supervised (or unsupervised) pre-trained network on UCF101/HMDB51 directly, and in Table 1 we show that our method performs better in comparison to state-of-the-art self-supervised methods [14, 31, 32, 59] when trained from scratch on UCF101/HMDB51. It is obvious that our DynamoNet used more data to train, but the extra data is just video clips without any labels. So our method is an effective way to do pre-training without any cost on data labeling for video classification methods.

Model	UCF101	HMDB51
3D-ResNet 101	55.4	29.2
STC-ResNet 101	56.7	30.8
Shuffle and learn [31]	50.9	19.8
Odd-One-Out [14]	60.3	32.5
AVTS [27]	83.7	53.0
ActionFlowNet [32]	83.9	56.4
AOT [59]	86.5	-
DynamoNet (ResNext)	**87.3**	**58.6**
DynamoNet (STCnet)	**88.1**	**59.9**

Table 1. Comparison of self-supervised methods on UCF101 and HMDB51 split-1 with RGB input. All of the methods (except baseline networks) has been trained with a self-supervised method and then fine-tuned on UCF101 and HMDB51.

We have also evaluated the impact of the self-supervised (or unsupervised) pre-training on the demand of labeled training data for training on large datasets like Kinetics. To train a 3D-CNN like STCnet or 3D-ResNet from scratch, we need a huge amount of labeled video clips. We showed with a pre-trained 3D-ConvNet by DynamoNet pipeline, we can use a fraction of Kinetics videos but still achieve a reasonable performance. Table 2 shows how we can handle the cases with limited number of videos. The evaluation is done on the validation set of Kinetics. We can observe that DynamoNet performance when trained with half of the dataset is still comparable with others trained on full amount of data.

Model	Data size	Top1-Val (%)
3D-ResNext 101 [19]	half	53.9
3D-ResNext 101 [19]	full	65.1
STC-ResNext 101 [6]	half	55.4
STC-ResNext 101 [6]	full	66.2
DynamoNet (STCnet)	half	**63.6**
DynamoNet (STCnet)	full	**67.6**

Table 2. Evaluation of training on half and full Kinetics dataset.

Model	SSIM
Mathieu et al. [30]	0.92
Ours	**0.95**

Table 3. Frame prediction quantitative performance comparison on 378 test videos from UCF101.

4.5. Frame Prediction Learning Impact

There might be a question about the effect of prediction loss on the training pipeline. We have done experiments to generate filters and have used them as feature combined with action representation for action classification without frame prediction objective. A DynamoNet which is trained on the UCF101 achieves 50.2%, training DynamoNet without prediction loss performs 43.2%. As expected, the per-

Figure 3. **Qualitative results.** Comparison between the actual ground truth frame and the predicted future frame - obtained using our proposed approach. First column is an actual frame from video clip as Ground-Truth (GT). Second column is the DynamoNet predicted frames. Third is the difference image of the GT and the predicted frames, and in fourth column, we show the optical flow extracted from the predicted frames which presents corresponding motion regarding the last frame. Best viewed in color.

formance was poor and filters do not present any meaningful information in absence of prediction objective function.

4.6. Frame Prediction

We have compared our frame prediction performance with Mathieu et al. [30] which provided results on 378 test videos from UCF101 in Table 3. Moreover in Figure 3, we present a few examples of qualitative prediction results using our DynamoNet. The predicted frames show that the dynamic motion filters are able to capture the motion information, and thereby help to predict the future frames.

4.7. Action Recognition

In this section, we compare the DynamoNet performance with the state-of-the-art methods by first pre-training on Kinetics and then fine-tuning on target dataset, i.e. all three splits of the UCF101 and HMDB51 datasets. For UCF101 and HMDB51, we report the average accuracy over all three splits. Our experiments are best with STCnet and 3D-ResNet/Next configuration which is of depth 101.

Method	Top1-Val	Top5-Val
DenseNet3D	59.5	-
Inception3D	58.9	-
C3D [19]	55.6	-
3D ResNet101 [19]	62.8	83.9
3D ResNext101 [19]	65.1	85.7
RGB-I3D [3]	68.4	88
STC-ResNet101 (16 frames) [6]	64.1	85.2
STC-ResNext101 (16 frames) [6]	66.2	86.5
STC-ResNext101 (32 frames) [6]	68.7	88.5
S3D-G [62]	74.7	93.4
R(2+1)D [51]	74.3	91.4
NL-I3D [58]	77.7	93.3
DynamoNet (ResNext) (16 frames)	66.3	86.7
DynamoNet (ResNext) (32 frames)	68.2	88.1
DynamoNet (STCnet) (16 frames)	67.6	87.2
DynamoNet (STCnet) (32 frames)	71.4	90
DynamoNet (STCnet) (64 frames)	**77.9**	**94.2**

Table 4. Performance (%) comparison of **DynamoNet** with other state-of-the-art methods on Kinetics-400 dataset.

Table 4 presents Kinetics dataset results for DynamoNet compared with the other 3D ConvNets who have provided results on the dataset. The DynamoNet (STCnet101) with 64 frames input depth outperforms STC-ResNext101 [6] which has the input size of 32 frames, and also I3D [3] with 64 frames input.

Method	UCF101	HMDB51
DT+MVSM [2]	83.5	55.9
iDT+FV [55]	85.9	57.2
C3D [49]	82.3	56.8
C3D+iDT [49]	90.4	–
LTC+iDT [49]	92.4	67.2
Conv Fusion [13]	82.6	56.8
Two Stream [42]	88.6	–
TDD+FV [56]	90.3	63.2
RGB+Flow-TSN [57]	94.0	68.5
ST-ResNet [12]	93.5	66.4
TSN [57]	94.2	69.5
RGB-I3D [3]	95.6	74.8
Inception3D [6]	87.2	56.9
3D ResNet 101 (16 frames) [19]	88.9	61.7
3D ResNext 101 (16 frames) [19]	90.7	63.8
STC-ResNext 101 (16 frames) [6]	92.3	65.4
STC-ResNext 101 (64 frames) [6]	96.5	74.9
DynamoNet (ResNext) (16 frames)	**91.6**	**66.2**
DynamoNet (ResNext) (32 frames)	**93.1**	**68.5**
DynamoNet (STCnet) (32 frames)	**96.6**	**74.9**
DynamoNet (STCnet) (64 frames)	**97.8**	**76.8**

Table 5. Accuracy (%) performance comparison of **DynamoNet** with the state-of-the-art methods over all three splits of UCF101 and HMDB51. For a fair comparison, in this table we report the performance of methods which utilize only RGB frames as input.

In Table 5, we compare the performance of DynamoNet with current state-of-the-art methods on UCF101/HMDB51. Our DynamoNet (with STCnet model) outperforms STCnet [6], 3D-ResNet [50], RGB-I3D[3] and C3D [49] on both UCF101 and HMDB51 and achieves 97.8% and 76.8% accuracy respectively. As shown in Table 5, DynamoNet performs better than STC-ResNext101 by almost 2% on UCF101. Note that most of current methods [3, 57] utilize optical-flow maps in addition to RGB frames, such as I3D which obtains a performance of 98% on UCF101 and 80% on HMDB51, also utilize flow information. Since our DynamoNet is providing motion representation for action classification in an end-to-end fashion, we show that a nice performance can also be obtained even without incorporating flow information.

Despite of not using optical-flow information, our results show how DynamoNet can exploit spatio-temporal appearance and motion information together with 3D-Conv structure, in addition to dynamic motion representation learning. Our work encourages similar approaches to exploit motion cues for action and activity classification in a more efficient manner thus leading to improve both accuracy and computation performance.

5. Conclusion

The capability of the current video understanding architectures to effectively learn and exploit motion representation is a key issue in the field of action classification. In this work, we propose to learn an action classification driven motion representation in videos using dynamic motion filters by predicting the future frames. Furthermore, we show that the learned motion representation is effective for action classification. We demonstrate the effectiveness of our proposed method on three challenging action recognition benchmark datasets: UCF101, HMDB51 and Kinetics. In addition to yielding a better performance than the state-of-the-art methods, our dynamic motion representations are robust and compact - which retains a global motion representation in a more expressive way.

Even though, in this paper we have focused on action classification only, we believe the our motion information can be added as a complementary cue for other tasks, like video understanding, video retrieval and more. Since our motion filter learning is self-supervised, we believe abundantly available unlabeled videos are an effective resource to acquire knowledge and to learn a effective feature representation. In future, we would like to explore two-stream network paradigm for making a more efficient pipeline of frame prediction and action classification.

Acknowledgements: This work was supported by DBOF PhD scholarship, KU Leuven:CAMETRON project, and KIT:DFG-PLUMCOT project.

References

[1] Bert De Brabandere, Xu Jia, , Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *NIPS*, 2016.

[2] Zhuowei Cai, Limin Wang, Xiaojiang Peng, and Yu Qiao. Multi-view super vector for action recognition. In *CVPR*, 2014.

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

[4] Emily L Denton and Vighnesh Birodkar. Unsupervised learning of disentangled representations from video. In *NIPS*, 2017.

[5] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015.

[6] Ali Diba, Mohsen Fayyaz, Vivek Sharma, M Mahdi Arzani, Rahman Yousefzadeh, Juergen Gall, and Luc Van Gool. Spatio-temporal channel correlation networks for action classification. In *ECCV*, 2018.

[7] Ali Diba, Mohsen Fayyaz, Vivek Sharma, A Hossein Karami, M Mahdi Arzani, Rahman Yousefzadeh, and Luc Van Gool. Temporal 3d convnets using temporal transition layer. In *CVPR Workshops*, 2018.

[8] Ali Diba, Ali Mohammad Pazandeh, and Luc Van Gool. Efficient two-stream motion and appearance 3d cnns for video classification. In *ECCV Workshops*, 2016.

[9] Ali Diba, Vivek Sharma, and Luc Van Gool. Deep temporal linear encoding networks. In *CVPR*, 2017.

[10] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[11] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, pages 2758–2766, 2015.

[12] Christoph Feichtenhofer, Axel Pinz, and Richard Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, pages 3468–3476, 2016.

[13] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016.

[14] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *CVPR*, 2017.

[15] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *NIPS*, 2016.

[16] Rohit Girdhar, Deva Ramanan, Abhinav Gupta, Josef Sivic, and Bryan Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *CVPR*, 2017.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[18] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv:1502.04623*, 2015.

[19] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In *ICCV*, 2017.

[20] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet. In *CVPR*, 2018.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[23] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015.

[24] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[25] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

[26] Benjamin Klein, Lior Wolf, and Yehuda Afek. A dynamic convolutional layer for short range weather prediction. In *CVPR*, 2015.

[27] Bruno Korbar, Du Tran, and Lorenzo Torresani. Cooperative learning of audio and video models from self-supervised synchronization. In *NeuIPS*, 2018.

[28] Hilde Kuehne, Hueihan Jhuang, Rainer Stiefelhagen, and Thomas Serre. Hmdb51: A large video database for human motion recognition. In *High Performance Computing in Science and Engineering*. 2013.

[29] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv:1511.05440*, 2015.

[30] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

[31] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV*, 2016.

[32] Joe Yue-Hei Ng, Jonghyun Choi, Jan Neumann, and Larry S Davis. Actionflownet: Learning motion representation for action recognition. In *WACV*, 2018.

[33] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *NIPS*, 2015.

[34] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*, 2015.

[35] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017.

[36] Anurag Ranjan, Javier Romero, and Michael J Black. Learning human optical flow. 2018.

[37] Veith Röthlingshöfer, Vivek Sharma, and Rainer Stiefelhagen. Self-supervised face-grouping on graphs. In *ACMMM*, 2019.

[38] Vivek Sharma, Ali Diba, Davy Neven, Michael S Brown, Luc Van Gool, and Rainer Stiefelhagen. Classification-driven dynamic image enhancement. In *CVPR*, 2018.

[39] Vivek Sharma, Makarand Tapaswi, M Saquib Sarfraz, and Rainer Stiefelhagen. Self-supervised learning of face representations for video face clustering. In *International Conference on Automatic Face and Gesture Recognition.*, 2019.

[40] Yuge Shi, Basura Fernando, and Richard Hartley. Action anticipation with rbf kernelized feature mapping rnn. In *ECCV*, 2018.

[41] Jan C. van Gemert Silvia L. Pintea and Arnold W. M. Smeulders. Dejavu: Motion prediction in static images,. In *ECCV*, 2014.

[42] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

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

[44] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015.

[45] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015.

[46] Jonathan C Stroud, David A Ross, Chen Sun, Jia Deng, and Rahul Sukthankar. D3d: Distilled 3d networks for video action recognition. *arXiv:1812.08249*, 2018.

[47] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *ICCV*, 2015.

[48] Peng Tang, Xinggang Wang, Baoguang Shi, Xiang Bai, Wenyu Liu, and Zhuowen Tu. Deep fishernet for object classification. *arXiv preprint arXiv:1608.00182*, 2016.

[49] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[50] Du Tran, Jamie Ray, Zheng Shou, Shih-Fu Chang, and Manohar Paluri. Convnet architecture search for spatiotemporal feature learning. *arXiv:1708.05038*, 2017.

[51] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018.

[52] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *CVPR*, 2016.

[53] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *NIPS*, 2016.

[54] Carl Vondrick and Antonio Torralba. Generating the future with adversarial transformers. In *CVPR*, 2017.

[55] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.

[56] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015.

[57] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *ECCV*, 2016.

[58] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[59] Donglai Wei, Joseph Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *CVPR*, 2018.

[60] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *ECCV*, 2016.

[61] Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. Synthesizing dynamic textures and sounds by spatial-temporal generative convnet. *stat*, 2016.

[62] S Xie, C Sun, J Huang, Z Tu, and K Murphy. Rethinking spatiotemporal feature learning for video understanding. In *ECCV*, 2018.

[63] Tianfan Xue, Jiajun Wu, Katherine Bouman, and William Freeman. Visual dynamics: Stochastic future generation via layered cross convolutional networks. *PAMI*, 2018.

[64] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, 2016.

[65] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015.

[66] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, 2016.