Motion-Augmented Self-Training for Video Recognition at Smaller Scale

Gavrilyuk, K.; Jain, M.; Karmanov, I.; Snoek, C.G.M.

DOI
10.1109/ICCV48922.2021.01026

Publication date
2021

Document Version
Author accepted manuscript

Published in
2021 IEEE/CVF International Conference on Computer Vision

Citation for published version (APA):
Motion-Augmented Self-Training for Video Recognition at Smaller Scale

Kirill Gavrilyuk\textsuperscript{1} Mihir Jain\textsuperscript{2} Ilia Karmanov\textsuperscript{2} Cees G. M. Snoek\textsuperscript{1}

\textsuperscript{1}University of Amsterdam \textsuperscript{2}Qualcomm AI Research*

\{kgavrilyuk, cgmsnoek\}@uva.nl \{mijain, ikarmano\}@qti.qualcomm.com

Abstract

The goal of this paper is to self-train a 3D convolutional neural network on an unlabeled video collection for deployment on small-scale video collections. As smaller video datasets benefit more from motion than appearance, we strive to train our network using optical flow, but avoid its computation during inference. We propose the first motion-augmented self-training regime, we call MotionFit. We start with supervised training of a motion model on a small, and labeled, video collection. With the motion model we generate pseudo-labels for a large unlabeled video collection, which enables us to transfer knowledge by learning to predict these pseudo-labels with an appearance model. Moreover, we introduce a multi-clip loss as a simple yet efficient way to improve the quality of the pseudo-labeling, even without additional auxiliary tasks. We also take into consideration the temporal granularity of videos during self-training of the appearance model, which was missed in previous works. As a result we obtain a strong motion-augmented representation model suited for video downstream tasks like action recognition and clip retrieval. On small-scale video datasets, MotionFit outperforms alternatives for knowledge transfer by 5\%-8\%, video-only self-supervision by 1\%-7\% and semi-supervised learning by 9\%-18\% using the same amount of class labels.

1. Introduction

The goal of this paper is to self-train a 3D convolutional neural network on an unlabeled video collection, such that it can be effectively fine-tuned on small scale datasets. This is of interest for applications in small-sized companies, a household, or search and rescue robotics where large amounts of labeled video are often unavailable and the deployment in compute-efficient scenarios is preferred. The common self-training approach is to transfer knowledge from pre-trained appearance models by pseudo-label predic-

\*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.
from a pre-trained 2D convolutional neural network to a 3D convolutional neural network [12, 19] or from depth to an appearance stream [17, 18]. In particular, the works of Crasto et al. [11] and Stroud et al. [50] helped shape our idea. Both these works explore the transfer of motion knowledge to an appearance model using optical flow as privileged information, together with the large-scale labeled Kinetics [9] dataset. We also transfer from a motion to an appearance representation, but different from [11, 50] class labels on a large-scale dataset are unavailable during transfer in our setting. Instead, we propose to obtain pseudo-labels by first training a motion model on a small-scale labeled dataset, like UCF101 [49] or HMDB51 [32]. We perform unsupervised K-means clustering on the extracted motion features to obtain cluster assignments as pseudo-labels. Then we train an appearance model to predict these pseudo-labels via a self-training procedure on a larger source dataset, without using any additional class labels, see Figure 1.

Our key contribution is a motion-augmented self-training procedure, we call MotionFit. It extracts motion knowledge and transfers it to the appearance model, via self-training on a large-scale unlabeled video dataset. By such motion transfer we avoid time-consuming optical flow computation during inference, similar in objective to motion knowledge distillation methods [11, 50], but without the need for labels during transfer. Our second contribution is an empirical study to discover what form video pseudo-labels should take at smaller scale, starting from the training of the pseudo-label generator to temporally mapping pseudo-labels to videos. We train the pseudo-label generator on the motion representation of a small-scale and labeled video dataset by a multi-clip loss that makes our motion model less susceptible to the background motion irrelevant to the video label. During self-training with pseudo-labels we also study different levels of temporal video granularity by exploring several partitions of whole videos, which was not taken into account in related approaches, e.g., [2, 60]. Finally, we experimentally evaluate the importance of each component of our method and compare with state-of-the-art for action classification and clip retrieval on two datasets. For clip retrieval, we improve over the state-of-the-art on UCF101 and match it on HMDB51. For action classification, our self-trained representation performs considerably better than the alternative knowledge transfer (up to +8%), self-supervised (up to +7%) and semi-supervised (up to +18%) methods.

2. Related Work

Video self-training. The deep clustering of [6] introduces an iterative approach by first assigning pseudo-labels using unsupervised K-means clustering, followed by predicting these assignments with a deep convolutional neural network. In [7], Caron et al. utilize a large non-curated dataset to train the deep clustering model. Asano et al. [4] suggest a principled learning formulation to overcome the problem of degenerate solutions of simultaneously learning and clustering features by maximizing the information between labels and input data indices. Zhan et al. [64] propose an online deep clustering approach that performs clustering and network update simultaneously, rather than alternating. Both [26] and [8] align cluster assignments for pairs of different transformations of images. A semi-supervised approach is presented in [46] where pseudo-labels are obtained directly from the predictions of a model trained on a labelled subset of the dataset. They propose a method for uncertainty-aware pseudo-label selection to iteratively select reliable unlabelled samples to be used with the labeled samples for training. Most similar to our work are Noroozi et al. [41] and Yan et al. [60], who transfer knowledge from pre-trained models with self-training by pseudo-label prediction. Differently, we transfer knowledge from the motion representation to the appearance representation, while [41] and [60] exploit the same representation during pre-training and pseudo-label prediction. We also consider pseudo-labels on different levels of temporal video granularity, in an effort to better model the dynamic nature of video.

Video self-supervision. An alternative to self-training via pseudo-labeling is self-supervision. Following the success of image-based self-supervision, early approaches for video self-supervision also explore similar pre-text tasks [28, 29, 54]. Other works pay more attention to the temporal video nature by exploiting chronological order [16, 33, 39, 59], pace [5, 10, 61], arrow of time [56] or video visual correspondence [24, 30, 45]. The temporal structure is also beneficial for predicting future video states such as prediction of raw pixel representations of future frames [13, 38] or their feature representation [14, 20, 21]. Most similar to our work are [21, 22, 36, 47, 63] who also explore optical flow. However, different from Sayed et al. [47] and Mahendran et al. [36], who align features of appearance and motion representations, we align two representations via pseudo-label self-training. Zhan et al. [63] utilize sparse motion guidance to recover full-image motion from appearance while we predict just the same pseudo-label for both representations. Different from [21] and [22] who use flow in the two-stream fashion [48], we still perform downstream tasks using only an appearance representation to be computationally efficient during inference. Differently from self-supervised methods we may make a second use of the small labeled target dataset to train the motion network for pseudo-label generation instead of only using it for fine-tuning the learned video representation model.

Multimodal self-supervision. Many methods exploit the multimodal nature of videos, such as audio [1–3, 31, 40, 42–44] or corresponding text [34] and speech [51] by aligning multiple modalities between each other. Differently, we use an optical flow representation that can be derived from the
raw RGB representation of the video. This representation helps to better model motion in the videos. Moreover, it does not introduce new information that is not contained in the original appearance representation, like audio and text.

3. MotionFit Model

Our goal is to learn a video representation suitable for downstream tasks, like video action recognition and video retrieval, on relatively small-scale target datasets $D_{target}$ such as UCF101 [49] and HMDB51 [32]. We first train a motion model, which we refer to as the motion pseudo-label generator $M_{MPLG}$, also on a small-scale dataset $D_{MPLG}$, which can be the same as $D_{target}$. Next, we use the $M_{MPLG}$ network to extract motion features and to obtain pseudo-labels on a large-scale unlabeled dataset $D_{source}$. Then we switch to the RGB representation of $D_{source}$ and self-train a new appearance network $M_{RGB}$ to predict the pseudo-labels on $D_{source}$ to obtain a motion-augmented video representation model suitable for downstream tasks on $D_{target}$. The overall approach is illustrated in Figure 2 and detailed next.

3.1. Training the motion pseudo-label generator

To train the motion pseudo-label generator on $D_{MPLG}$ we can rely on the appearance representation by optimizing a cross-entropy loss to predict ground truth labels. However, current state-of-the-art networks are too large to be efficiently trained on small-scale datasets [9, 15, 52, 58]. Instead, we pay more attention to the video nature and rely on a motion representation to train the $M_{MPLG}$. The motion representation helps the convolutional neural network to concentrate on more important local motion changes, rather than the repetitive and abundant appearance representation. Hence, it has the advantage of learning from more generalizable information, while at the same time not heavily relying on context as the appearance representation does.

To further boost the strength of the $M_{MPLG}$ representation we consider a longer temporal extent in each training sample, so it is more likely to contain the foreground part of the action of interest. This is important as unlike an appearance representation, motion cannot rely on background context for recognition. One way to cover longer temporal extents is to consider longer clips as input samples rather than the common practice of using 16-frame clips [29, 51, 59, 65]. However, pooling over the temporal dimension can attenuate the foreground motion signal when a larger part is from the background. We propose an alternative strategy where each training sample consists of multiple clips, of standard 16-frames length, from the same video. The loss for each sample is obtained by averaging the cross-entropy losses over its clips. The foreground clips are more likely to have a unimodal distribution across the final activations, while background ones are likely to be relatively uniform. This

Figure 2: MotionFit model. Our approach consists of three main steps. In the first step we train a motion pseudo-label generator, $M_{MPLG}$, on the motion representation of the dataset $D_{MPLG}$ with a multi-clip loss. Next we use the $M_{MPLG}$ to obtain pseudo-labels for the source dataset $D_{source}$. We use these pseudo-labels for motion-augmented self-training of an appearance network $M_{RGB}$ on $D_{source}$, without any additional class labels. The learned video representation model is suitable for downstream tasks on $D_{target}$ using only appearance as input. We do so by fine-tuning $M_{RGB}$ on $D_{target}$.
means that averaging after the softmax (due to the nature of the exponential function) will produce higher logits and not attenuate the gradient information as much. Thus, the proposed multi-clip loss preserves more temporal information and allows any clip from the foreground to contribute constructively to the back-propagating gradients.

\[
\mathcal{L}_{mc}(y, \hat{y}) = \frac{1}{B} \sum_{b=1}^{B} \left( \frac{1}{R} \sum_{i=1}^{R} \mathcal{L}_{class}(y_{b,i}, \hat{y}_{b,i}) \right),
\]

where \( \mathcal{L}_{class} \) is the cross-entropy loss for clip label prediction, \( y_{b,i} \) is the ground truth clip label, \( \hat{y}_{b,i} \) is the model predicted clip label, \( R \) is the number of sampled clips per video, and \( B \) is batch size. In our experiments we show the benefits of this simple approach, which we call training with multi-clips.

Xu et al. [59] show that clip order prediction can be beneficial for self-supervised learning:

\[
\mathcal{L}_{co}(p, \hat{p}) = \frac{1}{B} \sum_{b=1}^{B} \mathcal{L}_{order}(p_{b}, \hat{p}_{b}),
\]

where \( \mathcal{L}_{order} \) is the cross-entropy loss for clip order prediction, \( p_{b} \) is the correct order of the clips and \( \hat{p}_{b} \) is the model predicted clip order. In our multi-clip setting it is easy to combine it with our multi-clip loss \( \mathcal{L}_{mc} \) for more efficient training of the \( M_{MPLG} \). We do so by a weighted sum:

\[
\mathcal{L}_{batch} = \lambda \mathcal{L}_{mc} + (1 - \lambda) \mathcal{L}_{co},
\]

where \( \mathcal{L}_{batch} \) is the batch loss, \( \lambda \in [0; 1] \) is a weighting parameter. With \( \lambda=0 \) and using an appearance representation as input we have exactly the same self-supervised formulation as in [59]. However, in our experiments we will show the benefits of using a motion representation and just relying on training with multi-clips.

### 3.2. Motion-augmented self-training

Following [41] and [60] we utilize \( K \)-means clustering with Euclidean distance to obtain pseudo-labels for the source dataset \( D_{source} \). Rather than relying on appearance features as [41] and [60] do, we do so on the motion features extracted by our \( M_{MPLG} \). The cluster centers computed by \( K \)-means are considered as pseudo-labels for the previously unseen videos from the \( D_{source} \) dataset. We train another network \( M_{RGB} \), but with an appearance representation as input, to predict the pseudo-labels of \( D_{source} \). By doing so, we transfer motion knowledge from the \( M_{MPLG} \) to the appearance network \( M_{RGB} \). Noroozi et al. [41] and Yan et al. [60] show that this self-training procedure is the most efficient way to do so, compared to other model distillation and transfer techniques. However, where they rely on RGB appearance only, we consider different video representations. We train the \( M_{MPLG} \) on motion and network \( M_{RGB} \) on appearance. The motion representation allows us to train the \( M_{MPLG} \) using only a small-scale dataset, while the appearance representation of \( M_{RGB} \) is computationally efficient during inference time. Note that \( M_{MPLG} \) and \( M_{RGB} \) can also have a different architecture, allowing us to train network \( M_{RGB} \) with different backbones using the same set of pseudo-labels obtained by \( M_{MPLG} \). In the experiments we also compare with other knowledge transfers including transfer from pre-trained 2D CNNs [12, 19] and motion [11, 50], in either case showing the benefits of our approach.

**Temporal modeling for self-training.** To further enrich the representation learned by \( M_{RGB} \), we use a similar loss as Equation 3 to train \( M_{RGB} \) on \( D_{source} \). Different from [2, 60] we also consider assigning pseudo-labels to subparts of the video, taking into account the most suitable temporal scale for the pseudo-labels. To do so, we first extract features using a trained \( M_{MPLG} \) for each video clip that we densely sample from all videos \( V = \{ V_i \}_{i=1}^{N_{source}} \) of the dataset \( D_{source} \):

\[
F_i = \{ f_{i,t} \}_{t=1}^{T_i}, \quad \text{where } T_i \text{ is number of clips in } V_i; \quad N_{source} \text{ is the number of videos in } D_{source}.
\]

Video clip is a sequence of adjacent frames in the video; 16, 32 or 64 frames in our experiments. After this we consider three levels of temporal granularity: clip level, segment level and video level. These levels differ in the way they aggregate features using \( F_i \) to obtain new \( h_i = \{ h_{i,s} \}_{s=1}^{S_i} \) representation for each video \( V_i \).

For clip level we consider \( H_i = F_i \). We define a segment as a sequence of adjacent video clips and consider two methods to obtain them. For the first, we utilize the same approach as in [25]. For a given video, we find segment boundaries \( B_i \) by looking for time steps where features of adjacent clips change abruptly compared to the previous time step:

\[
B_i = \{ t : \| f_{i,t} - f_{i,t-1} \|_1 > \tau \}
\]

where \( \tau \) is set to the \( p \)-th percentile, so the number of segments in a video \( V_i \) is directly proportional to its length \( T_i \). For the second, we consider the approach where the video is equally divided into three segments similar to [55]. Following [25], to obtain the feature representation for each segment we average clip features within its boundaries. For video level we consider the whole video as one segment and represent it by also averaging features of all clips in the video similar to [60]. Independent of the level of temporal granularity the other parts of our method are kept unchanged.

### 4. Experiments

#### 4.1. Datasets

**UCF101 and HMDB51.** As instances of our \( D_{MPLG} \) and \( D_{target} \) datasets we rely on UCF101 [49] and HMDB51 [32], two well-known datasets\(^1\) for video action recognition. However, in many works [9, 15, 52, 58] it is shown that these two

\(^1\)Datasets used in this paper were downloaded and experimented on by primary author
datasets are too small for an efficient training of modern 3D CNNs on the appearance representation. Therefore these datasets are good choice for our experiments. UCF101 contains 13k videos with 101 human actions. There are 3 splits available with around 9k training and 4k testing videos per split. HMDB51 contains 7k videos with 51 human classes from movies. It also has 3 splits with around 5.5k training and 1.5k testing videos per split. We consider these datasets for both downstream tasks of video action recognition and video clip retrieval.

**Kinetics-400.** As instance of \( D_{\text{source}} \), we rely on Kinetics-400 [9]. It contains 400 human action classes with 10-second video clips. There are around 246k training and 50k validation videos. Kinetics-400 is considered as a standard choice for pre-training of modern 3D CNNs [9, 15, 52, 58]. Similar to self-supervised methods we use this dataset without human annotation to pre-train 3D CNNs. We perform ablation experiments on its validation split (Kinetics-val) and make comparison with knowledge transfer and self-supervised methods using its train split (Kinetics-train).

### 4.2. Implementation details

**Motion pseudo-label generator.** For our motion representation we utilize TV-L1 [62], which is widely used for video action recognition. Following the recent literature [3, 11, 43], we choose the R(2+1)D-18 [52] as the backbone for our \( M_{\text{MPLG}} \).

**Self-training.** For appearance model \( M_{\text{RGB}} \), we consider again the R(2+1)D-18 backbone as well as the S3D-G [58].

**Training details.** We randomly split around 10% videos from the training set to do validation both during training \( M_{\text{MPLG}} \) and self-training. The input video clips are first resized to 128×171, then randomly cropped to 112×112 during training. During validation or testing, the clip is cropped in the center. For S3D-G, we use longer clips with 64 frames and higher resolution of 224×224. Recent findings [37] show that small mini-batch sizes provide more up-to-date gradient calculations and yield more stable and reliable training. Thus we use 8 multi-clips per batch while training on smaller datasets. For self-training on Kinetics-400 we use a batch size of 16 for R(2+1)D and 32 for S3D-G. We train our model with an initial learning rate of 0.001 and decay the learning rate by a factor of 10 at steps of 40, 60, 80 epochs. This regime is followed during \( M_{\text{MPLG}} \) training and fine-tuning (only conv4, conv5 and fc layers) for each dataset. For self-training, we decay the learning rate at 20 and 40 epochs. The \( M_{\text{MPLG}} \) training and fine-tuning is done for up to 120 epochs, while self-training on the Kinetics-400 train set is done for 45 epochs. Our models are implemented with PyTorch and optimized with vanilla synchronous SGD algorithm with momentum of 0.9 and weight decay of 0.0005.

<table>
<thead>
<tr>
<th>Representation</th>
<th>Clip length</th>
<th>Multi-clips ((R))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appearance</td>
<td>32</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Motion</th>
<th>59.4</th>
<th>60.3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>58.9</td>
<td>57.0</td>
</tr>
<tr>
<td></td>
<td>59.1</td>
<td>58.4</td>
</tr>
</tbody>
</table>

Table 1: **Benefit of \( M_{\text{MPLG}} \) motion representation.** Comparison between appearance and motion representation for varying clip length (in frames) and multi-clips per video on UCF101 split 1. When using multiple clips for training we consider 16-frames clips as input. Using multi-clip training is even better than using larger input temporal extent which allows us to efficiently train \( M_{\text{MPLG}} \) without additional computational and memory cost. The motion representation is 20% better than appearance for all settings.

### 4.3. Ablation of motion pseudo label generator

**Benefit of \( M_{\text{MPLG}} \) motion representation.** We first perform an ablation to demonstrate the benefit of training the pseudo-label generator on the optical flow representation. We report action recognition results on UCF101 split 1 for this experiment. First, we train the \( M_{\text{MPLG}} \) with the loss function in Eq. 3, setting \( \lambda=1 \) to show only the importance of using multiple clips from the same video in the batch. In Table 1 we ablate appearance and motion representations for a varying number of clips \( R \) in Eq. 3. We also compare with standard single clip training, but with longer temporal length of the input clip. Independent of the clip length and number, the motion representation is 20% better than appearance, even when using a small temporal scale. Our multi-clip training is helpful for motion representation learning giving around 4% boost over standard single-clip training. It is also more than 1% better than using clips with larger temporal scale. While for appearance, which relies heavily on context, there is not much improvement when using longer temporal scale or multiple clips for training.

**Choice of \( M_{\text{MPLG}} \) parameter \( \lambda \).** Next we perform experiments varying parameter \( \lambda \) in Eq. 3. While for the appearance representation the improvement matches the results of [59] (73.7) the motion representation does not benefit from explicit clip order prediction. With \( \lambda=0 \) we achieve only 69.9 accuracy, which is even lower than for the appearance representation. For values of \( \lambda<1 \) we did not see any gain in action recognition accuracy compared to the case of \( \lambda=1 \). This supports that our multi-clip training procedure by itself provides enough temporal information when trained with a motion representation. Therefore, we use a motion representation for our \( M_{\text{MPLG}} \) multi-clip training with \( R=3 \) and \( \lambda=1 \) in the rest of the paper.

**Importance of \( M_{\text{MPLG}} \) for self-training.** First, we use the above \( M_{\text{MPLG}} \) \((R=3, \lambda=1)\) to generate pseudo-labels (128 clusters) for self-training on Kinetics-400 train set, and obtain 85.2% accuracy on UCF101. Then, we perform the
We compare the choices of the self-trained appearance model.

Table 2: Choice of temporal granularity for motion-augmented self-training of model $M_{\text{RGB}}$ on Kinetics-val. Comparison is performed on the downstream task of video action recognition on UCF101 split 1. Clip level gives 4% boost over video level and is slightly better than segments.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>76.5</td>
<td>79.0</td>
<td>77.3</td>
<td>80.3</td>
</tr>
</tbody>
</table>

Table 3: Choice of number of clusters for self-training of model $M_{\text{RGB}}$ on the Kinetics-val and Kinetics-train. Comparison is performed on the downstream task of video action recognition on UCF101 split 1. A larger number of clusters benefits self-training up to 1000 clusters.

<table>
<thead>
<tr>
<th>Number of clusters</th>
<th>Kinetics-val</th>
<th>Kinetics-train</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>79.0</td>
<td>85.2</td>
</tr>
<tr>
<td>500</td>
<td>79.0</td>
<td>85.7</td>
</tr>
<tr>
<td>1000</td>
<td>79.7</td>
<td>86.5</td>
</tr>
<tr>
<td>1600</td>
<td>74.2</td>
<td>85.6</td>
</tr>
</tbody>
</table>

Table 4: Impact on target dataset for varying clip-length, where $D_{\text{MPLG}}$ is UCF101 and $D_{\text{source}}$ is Kinetics. Average accuracy with standard deviation over three splits is reported.

<table>
<thead>
<tr>
<th>Clip-length</th>
<th>Target dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 frames</td>
<td>UCF101</td>
</tr>
<tr>
<td></td>
<td>HMDB51</td>
</tr>
<tr>
<td>32 frames</td>
<td>87.4 ± 1.19</td>
</tr>
<tr>
<td>Random initialized [59]</td>
<td>58.9</td>
</tr>
</tbody>
</table>

same experiment but with $M_{\text{MPLG}}$ trained on appearance instead. By self-training on these appearance pseudo-labels, we obtain only 74.5%, which confirms the better quality of our motion pseudo-labels. Note that the setting with appearance pseudo-labels is similar to [41] and [60], which are developed for the image domain and very large-scale datasets, respectively. Unlike them, we effectively train our model for small-scale video datasets using motion, but only relying on appearance during inference.

4.4. Ablation of motion-augmented self-training

Next we ablate the self-training choices for the network $M_{\text{RGB}}$, where we use the Kinetics-400 validation set as our source dataset $D_{\text{source}}$. We train our $M_{\text{MPLG}}$ on UCF-101 split 1 with the motion representation, and use it to extract features for densely sampled clips from each video in $D_{\text{source}}$. We compare the choices of the self-trained appearance model $M_{\text{RGB}}$ on the downstream task of video action recognition on UCF-101 split 1. Note that during self-training the model $M_{\text{RGB}}$ does not see any videos from UCF-101 neither any provided class labels of the Kinetics-400 dataset.

Choice of temporal granularity. We first analyze the importance of considering temporal scale for generating pseudo-labels, as discussed in Section 3.2. We show the results in Table 2 for three possible levels: video, segment and clip. Interestingly, a more semantic partition of videos as suggested in [25] and [55] does not improve the representation of $M_{\text{RGB}}$ compared to just a clip-level granularity. However, both segment and clip levels outperform video level [2,60] up to 2.5%. We also vary parameter $\lambda$ in Eq. 3 during self-training of $M_{\text{RGB}}$. For none of the temporal levels we have seen any considerable improvement over using $\lambda=1$. To further investigate the role of clip order we add the loss for pseudo-label prediction of the next and previous clips for each sampled clip of the video. This gives us an additional 0.4% improvement. These results again support that a self-training procedure with just a multi-clip loss helps successfully transfer motion knowledge to the appearance model, even without any additional modeling of clip order.

Choice of number of clusters. Next we compare the influence of the number of clusters used in $K$-means to obtain pseudo-labels for the source dataset $D_{\text{source}}$ in Table 3. When using Kinetics-400 validation for self-training of $M_{\text{RGB}}$ the impact of using more clusters is minimal. Even on the larger Kinetics-400 train set there is only small improvement of up to 1.3%, indicating our motion-augmented self-training is not sensitive to the choice of number of clusters. We choose $K=1000$ for all the following experiments.

Effect of common classes. UCF101 and Kinetics-400 have 55 classes in common. To evaluate their impact, we exclude them from the latter and re-train MotionFit ($K=1000$) with the remaining $\sim170$k videos. We obtain 86.3% on split1 of UCF101 vs. 86.5% with all the classes, which shows the common classes have hardly an effect on accuracy.

Impact on target dataset. Our self-training approach trains an appearance model with motion information to be effectively fine-tuned on small video datasets. First we evaluate our method for the case when $D_{\text{MPLG}}$ and $D_{\text{target}}$ are the same i.e. UCF101, and $D_{\text{source}}$ is always Kinetics. We fine-tune and evaluate on each of the three splits of UCF101 and report the results in the first row of Table 4. For self-training we use 16-frame clips, however, increasing the clip-length to 32 for fine-tuning leads to improved performance, with little additional computation. Then, we change the $D_{\text{target}}$ to HMDB51 but keep the same self-trained model. Again, there is considerable accuracy gain over random initialization [5], and longer clips help, but the main conclusion is that our approach can be easily applied to other small datasets without requiring the whole pipeline to be redone (flow extraction and then self-training on Kinetics). A considerable practical advantage. Next, we compare with the state-of-the-art.
Table 5: **Comparison with knowledge transfer** methods on video action recognition. We report top-1 accuracy of fine-tuned models averaged over all 3 splits of UCF101 and HMDB51. Our approach outperforms MERS and MARS for transferring motion knowledge to the appearance stream. MotionFit is also better than methods that also rely on ImageNet class labels.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>Frames</th>
<th>Resolution</th>
<th>Additional labels</th>
<th>UCF101</th>
<th>HMDB51</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random initialisation [59]</td>
<td>R(2+1)D-18</td>
<td>16</td>
<td>112</td>
<td>–</td>
<td>58.9</td>
</tr>
<tr>
<td>MERS [11]†</td>
<td>R(2+1)D-18</td>
<td>16</td>
<td>112</td>
<td>–</td>
<td>78.3</td>
</tr>
<tr>
<td>MARS [11]†</td>
<td>R(2+1)D-18</td>
<td>16</td>
<td>112</td>
<td>–</td>
<td>82.2</td>
</tr>
<tr>
<td>STC [12]</td>
<td>STC-ResNext</td>
<td>16</td>
<td>112</td>
<td>ImageNet</td>
<td>84.7</td>
</tr>
<tr>
<td>DistInit [19]</td>
<td>R(2+1)D-18</td>
<td>32</td>
<td>112</td>
<td>ImageNet</td>
<td>85.7</td>
</tr>
<tr>
<td>Supervised [43]</td>
<td>R(2+1)D-18</td>
<td>16</td>
<td>112</td>
<td>Kinetics-400</td>
<td>95.0</td>
</tr>
<tr>
<td><strong>MotionFit (ours)</strong></td>
<td>R(2+1)D-18</td>
<td>16</td>
<td>112</td>
<td>–</td>
<td>87.4</td>
</tr>
</tbody>
</table>

† MERS and MARS results are based on our implementation of [11].

Table 6: **Comparison with self-supervised methods on video action recognition.** We report top-1 accuracy of fine-tuned models averaged over all 3 splits of UCF101 and HMDB51. For fair comparison we list only the results obtained using the Kinetics-400 dataset and networks with similar depth. Our approach is the best when only considering the visual modality (V). On UCF101 we are even on par with methods that use an additional audio modality (V+A) during training.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>Frames</th>
<th>Resolution</th>
<th>Modality</th>
<th>UCF101</th>
<th>HMDB51</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sun et al. [51]</td>
<td>S3D</td>
<td>16</td>
<td>112</td>
<td>V + T</td>
<td>79.5</td>
</tr>
<tr>
<td>Asano et al. [3]</td>
<td>R(2+1)D-18</td>
<td>30</td>
<td>112</td>
<td>V + A</td>
<td>83.1</td>
</tr>
<tr>
<td>Alwassel et al. [2]</td>
<td>R(2+1)D-18</td>
<td>32</td>
<td>224</td>
<td>V + A</td>
<td>86.8</td>
</tr>
<tr>
<td>Xiao et al. [57]</td>
<td>SlowFast</td>
<td>64</td>
<td>224</td>
<td>V + A</td>
<td>87.0</td>
</tr>
<tr>
<td>Morgado et al. [40]</td>
<td>R(2+1)D-18</td>
<td>32</td>
<td>224</td>
<td>V + A</td>
<td>87.5</td>
</tr>
<tr>
<td>Patrick et al. [43]</td>
<td>R(2+1)D-18</td>
<td>32</td>
<td>224</td>
<td>V + A</td>
<td>89.3</td>
</tr>
<tr>
<td>Kim et al. [29]</td>
<td>R3D-18</td>
<td>16</td>
<td>112</td>
<td>V</td>
<td>65.8</td>
</tr>
<tr>
<td>Kong et al. [30]</td>
<td>R3D-18</td>
<td>8</td>
<td>112</td>
<td>V</td>
<td>69.4</td>
</tr>
<tr>
<td>Han et al. [20]</td>
<td>R-2D3D-34</td>
<td>25</td>
<td>224</td>
<td>V</td>
<td>75.7</td>
</tr>
<tr>
<td>Jing et al. [28]</td>
<td>R3D-18</td>
<td>64</td>
<td>112</td>
<td>V</td>
<td>76.6</td>
</tr>
<tr>
<td>Zhang et al. [65]</td>
<td>SlowFast</td>
<td>16</td>
<td>112</td>
<td>V</td>
<td>77.0</td>
</tr>
<tr>
<td>Han et al. [21]</td>
<td>R-2D3D-18</td>
<td>25</td>
<td>224</td>
<td>V</td>
<td>78.1</td>
</tr>
<tr>
<td>Benaim et al. [5]</td>
<td>S3D-G</td>
<td>64</td>
<td>224</td>
<td>V</td>
<td>81.1</td>
</tr>
<tr>
<td>Han et al. [22]</td>
<td>S3D</td>
<td>32</td>
<td>128</td>
<td>V</td>
<td>87.9</td>
</tr>
<tr>
<td><strong>MotionFit (ours)</strong></td>
<td>R(2+1)D-18</td>
<td>32</td>
<td>112</td>
<td>V</td>
<td>88.9</td>
</tr>
<tr>
<td><strong>MotionFit (ours)</strong></td>
<td>S3D-G</td>
<td>64</td>
<td>224</td>
<td>V</td>
<td>90.1</td>
</tr>
</tbody>
</table>

Table 6: **Comparison with self-supervised methods on video action recognition.** We report top-1 accuracy of fine-tuned models averaged over all 3 splits of UCF101 and HMDB51. For fair comparison we list only the results obtained using the Kinetics-400 dataset and networks with similar depth. Our approach is the best when only considering the visual modality (V). On UCF101 we are even on par with methods that use an additional audio modality (V+A) during training.

**4.5. Comparisons with state-of-the-art**

**Knowledge transfer.** In Table 5 we compare with knowledge transfer methods. As knowledge transfer methods we consider two main families: transfer knowledge from pre-trained 2D CNN and knowledge transfer from the motion to the appearance stream, the same as we do. For the former we choose STC [12] and DistInit [19], for the latter we choose MERS and MARS from [11]. We train MERS by matching features of the student appearance network with the teacher motion network on Kinetics-400. Then we further fine-tune the student model on the target dataset as we did for MotionFit. MARS additionally combines the feature matching with a cross-entropy loss and needs class-labels, so it is trained directly on the target dataset. For the teacher network we use the same $M_{MPLG}$ used to obtain pseudo-labels to train MotionFit. We have a good improvement (>5% on both datasets for R(2+1)D-18) over motion transfer methods that are based on feature matching. We are even better than knowledge transfer approaches from pre-trained 2D CNN, despite them also using ImageNet class labels.

**Self-supervised action recognition.** Next, we compare our approach with state-of-the-art self-supervised methods as we also do not use ground truth labels during self-training on the Kinetics-400 dataset. We first compare on video action recognition on UCF101 and HMDB51 in Table 6. Note that all reported methods fine-tune their models on these datasets and hence use the same amount of class labels as we do. These methods vary vastly for the choice of backbone networks, number of input frames and input resolutions, making fair comparison a challenging task by itself. However, thanks to the simplicity of our method, we can easily train different backbone appearance networks $M_{RGB}$ to predict the pseudo-labels generated by the same $M_{MPLG}$. We report for two backbone networks and outperform most other methods that use only the video modality with good margins on both the datasets. Using the same S3D-G backbone as Benaim *et al.* [5], we obtain an increase of 9.0% on UCF101 and 1.8%
We conclude that our motion-augmented self-trained model we compare with semi-supervised methods for video recognition. Despite Jing et al. [27] and Rizve et al. [46] using extra labels to pre-train a 2D CNN, we outperform them by 9% and 5.7% for 20% and 50% labelled data, respectively. The gain over Rizve et al. is even more, around 18.3% and 8.8% for the two labelled subsets. As the number of labelled videos decrease, we observe further advantage of MotionFit. More experimental analysis is reported in the supplementary material.

### 5. Conclusion

We propose a motion-augmented video self-training regime that transfers knowledge from motion to an appearance network. The trained model is motion-augmented and does not require costly optical flow computation during inference. Making it well suited for deployment on small-scale video datasets and video applications with constrained compute budgets. In order to improve the quality of the pseudo-labeling we introduce a simple yet effective multi-clip loss for training our pseudo-label generator. Our MotionFit provides a self-trained model that can be effectively fine-tuned on small-scale datasets for downstream tasks such as action recognition and clip retrieval. We fine-tune our model on two small-scale datasets and compare with state-of-the-art knowledge transfer, self-supervised and semi-supervised learning approaches that use the same amount of human labels for training. In all cases, our approach compares favorably to existing vision-only alternatives.

**Table 8:** Comparison with semi-supervision on video recognition at smaller scale. We report top-1 accuracy of models fine-tuned on 20% (or 50%) of UCF101 training data. Here $D_{MPLG}$ (same as $D_{target}$) is fraction of UCF101 train set and $D_{source}$ is the UCF101 train set.

<table>
<thead>
<tr>
<th></th>
<th>UCF101, split 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20% labeled</td>
</tr>
<tr>
<td>Jing et al. [27]</td>
<td>48.7</td>
</tr>
<tr>
<td>Rizve et al. [46]</td>
<td>39.4</td>
</tr>
<tr>
<td><strong>MotionFit (ours)</strong></td>
<td><strong>57.7</strong></td>
</tr>
</tbody>
</table>

$^1$ Use extra labels to pre-train a 2D CNN.
References


