Learning Discriminative Motion Features Through Detection

Gedas Bertasius[1,2], Christoph Feichtenhofer[1], Du Tran[1], Jianbo Shi[2], Lorenzo Torresani[1,3]
[1]Facebook Research, [2]University of Pennsylvania, [3]Dartmouth College

Abstract

Despite huge success in the image domain, modern detection models such as Faster R-CNN have not been used nearly as much for video analysis. This is arguably due to the fact that detection models are designed to operate on single frames and as a result do not have a mechanism for learning motion representations directly from video. We propose a learning procedure that allows detection models such as Faster R-CNN to learn motion features directly from the RGB video data while being optimized with respect to a pose estimation task. Given a pair of video frames—Frame A and Frame B—we force our model to predict human pose in Frame A using the features from Frame B. We do so by leveraging deformable convolutions across space and time. Our network learns to spatially sample features from Frame B in order to maximize pose detection accuracy in Frame A. This naturally encourages our network to learn motion offsets encoding the spatial correspondences between the two frames. We refer to these motion offsets as DiMoFs *(*Di*scriminative* Mo*tion* F*eatures*).*

In our experiments we show that our training scheme helps learn effective motion cues, which can be used to estimate and localize salient human motion. Furthermore, we demonstrate that as a byproduct, our model also learns features that lead to improved pose detection in still-images, and better keypoint tracking. Finally, we show how to leverage our learned model for the tasks of spatiotemporal action localization and fine-grained action recognition.

1. Introduction

Modern CNN based detection models have been highly successful on various image understanding tasks such as edge detection, object detection, semantic segmentation, and pose detection [23, 22, 32, 39, 18, 19, 21, 33, 10, 37, 38, 3, 55, 6, 49, 1, 52]. However, such models lack the means to learn motion cues from video data, and thus, they have not been used as widely for video understanding tasks where motion information plays a critical role.

Prior work [35, 40, 14] has attempted to address this issue by adopting two-stream architectures, where one stream learns appearance features, while the other stream aims to learn motion cues extracted from optical flow inputs. However, these models are expensive since they require processing optical flow with an additional CNN stream.

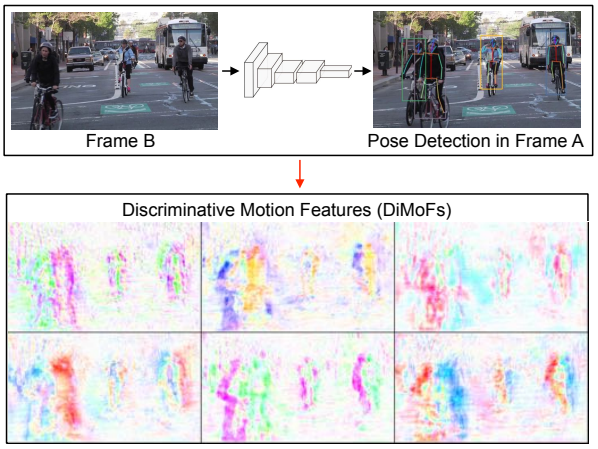

Figure 1: We extend modern detection models (e.g., Faster R-CNN) with the ability to learn discriminative motion features (DiMoFs) from RGB video data. We do so by training the detector on pairs of time-separated frames, with the objective of predicting pose in one frame using features from the other frame. This task forces the network to learn motion "offsets" relating the two frames but that are discriminatively optimized for detection. Our learned DiMoFs can be used on a variety of applications: salient motion localization, human motion estimation, improved pose detection and keypoint tracking, spatiotemporal action localization, and fine-grained action recognition.

Recently, there have been many attempts to learn effective video representations with 3D convolutional networks [44, 8]. In principle, such models are capable of capturing salient temporal and motion features, directly optimized for the end task. However, due to their high computational cost, 3D CNNs typically operate on very small spatial resolution inputs, and as a result are not very good at detecting fine grained visual cues such as the pose of a person. Furthermore, it has been recently shown that 3D CNNs optimized from RGB data for action recognition tend to be surprisingly insensitive to temporal ordering of frames [25]. This suggests that in practice they capture little motion in-

formation. Thus, it is typically necessary to apply them to optical flow inputs in order to effectively leverage motion.

In this work, we propose a learning scheme that allows modern detection models (e.g., Faster R-CNN) to learn motion cues directly from the RGB video data while being optimized for a discriminative video pose estimation task. Given a pair of annotated frames from the same video—Frame A and Frame B—we train our model to detect pose in Frame A, using the features from Frame B. To achieve this goal, our model leverages deformable convolutions [11] across space and time. Through this mechanism, our model learns to sample features from Frame B that maximize pose detection accuracy in Frame A. As a byproduct of our optimization, our model also learns "offsets" capturing the motion relating Frame A to Frame B. We refer to these offsets as DIMOFS (**Di**scriminative **Mo**tion **F**eature**s**) to emphasize that they are discriminatively optimized for detection. In our experiments, we show that our DIMOFS model pretrained on pose estimation can subsequently be used for salient motion localization, human motion estimation, improved pose detection, keypoint tracking, spatiotemporal action localization, and fine-grained action recognition.

2. Related Work

Detection in Images. Modern object detectors [23, 22, 32, 39, 18, 19, 21, 33, 10, 37, 38] are built using deep CNNs [28, 41, 24]. One of the earlier of such object detection systems was R-CNN [19], which operated in a two-stage pipeline, first extracting object proposals, and then classifying each of them using a CNN. To reduce the computational cost, RoI pooling operation was introduced in [23, 18]. A few years ago, Faster R-CNN [39] replaced region proposal methods by another network, thus eliminating a two stage pipeline. Several methods [37, 38] extended Faster R-CNN into a system that runs in real time with little loss in performance. The recent Mask R-CNN [22] introduced an extra branch that predicts a mask for each region of interest, Finally, Deformable CNNs [11] leveraged deformable convolution to model deformations of objects more robustly. While these prior detection methods work well on images, they are not designed to exploit motion cues in a video– a shortcoming we aim to address.

Detection in Videos. Several recent methods proposed architectures capable of aligning features temporally for improved object detection in video [57, 2, 50]. The method in [50] proposes a spatial-temporal memory mechanism, whereas [2] leverages spatiotemporal sampling for feature alignment. Furthermore, the work in [57] uses an optical flow CNN to align the features across time.

While the mechanisms in [2, 50] are useful for improved detection, it is not clear how to use them for motion cue extraction, which is our primary objective. Furthermore, models like [57] are redundant since they compute flow for every single pixel, which is rarely necessary for higher level video understanding tasks. Using optical flow CNN also adds $40M$ extra parameters to the model, which is costly.

Two-Stream CNNs. Recently, two-stream CNN architectures have been a popular choice for incorporating motion cues into modern CNNs [35, 40, 5, 4, 17, 14, 12, 13]. In these types of models, one stream learns appearance features from RGB data, whereas the other stream learns motion representation from the manually extracted optical flow inputs. The work in [5, 4] leverages two-stream architectures for learning more effective spatiotemporal representations. Recent methods in [46, 34, 12] explored the use of different backbone networks for action recognition tasks. Furthermore, various techniques have explored how to fuse the information from two streams [14, 13, 17]. However, these two-stream CNNs are costly and consume lots of memory. Learning discriminative motion cues from the RGB video data directly instead of relying on manually extracted optical flow inputs could alleviate this issue.

3D CNNs. Currently the most common approach for learning features from raw RGB videos is via 3D convolutional networks [44]. Whereas the method in [44] proposes a 3D network architecture for end-to-end feature learning, the recent I3D method [8] inflates all 2D filters to 3D, which allows re-using the features learned in the image domain. Additionally, there have been many recent attempts at making 3D CNNs more effective by replacing 3D convolution with separable 2D and 1D convolutions [51, 45, 9, 36].

While modern 3D CNNs are effective on popular action recognition datasets [8, 42, 29], 3D CNNs are not designed for tasks that require detection of fine-grained visual cues since they operate on small spatial resolution to accommodate long video clips. Furthermore, it has been shown that 3D CNNs do not actually learn motion cues [25], and that they still need to rely on two stream architectures.

Relational Reasoning. There have been several methods modeling temporal relations in videos [48, 56]. The work in [48] learns weights for modeling "cause and effect" type of relationship. Furthermore, a similar method to ours [56] proposes a temporal relational module for reasoning about temporal dependencies between video frames. However, the proposed relational module does not actually learn motion cues, and works effectively only when videos have strong temporal order [56], which many datasets do not [8, 42, 29]. In contrast, we aim to learn discriminative motion cues, which encode more general information and should be useful even if videos are not temporally ordered.

3. Learning Discriminative Motion Features

Our goal is to define a training procedure to learn discriminative motion features (DIMOFS) from RGB videos using a detection model (e.g., Faster R-CNN). Consider the example in Figure 2, which shows a man closing the car

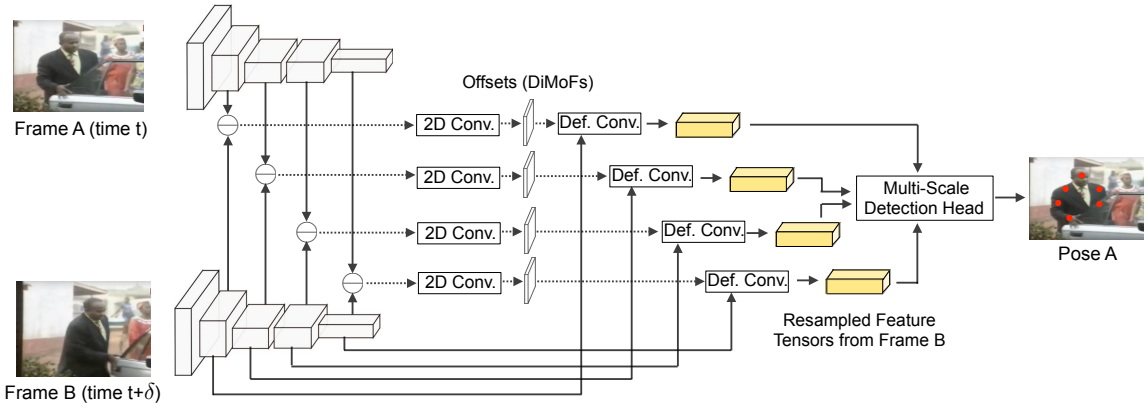

Figure 2: An illustration of our Discriminative Motion Feature (DIMOFS) training procedure. Given Frame A and Frame B, which are separated by δ steps in time, our goal is to detect pose in Frame A using the features from Frame B. First, we extract multi-scale features from both frames via a backbone CNN with shared parameters. Then, at each scale, we compute the difference between feature tensors A and B. From these tensor differences, offsets Δp_n are predicted for each pixel location p_n. The predicted offsets are used to re-sample feature tensor B. As a last step, the resampled feature tensors from each scale are fed into a multi-scale detection head, which is used to predict the pose in Frame A. Our scheme optimizes end-to-end the DIMOFS network so that the feature tensors re-sampled from Frame B maximize the pose detection accuracy in Frame A.

door. In order to recognize this action (i.e., closing the door), we do not need to know the motion of every single pixel in the image. Just knowing how the hand of the person moves relative to the car gives us enough information about the action. The key question is: how can we learn such motion information discriminatively? To do this, we formulate the following task. Given two video frames—Frame A and Frame B—our model is allowed to compare Frame A to Frame B but it must predict Pose A (i.e., the pose in Frame A) using the features from Frame B.

At first glance, this task may look overly challenging: how can we predict Pose A by merely using features from Frame B? However, suppose that we had body joint correspondences between Frame A and Frame B. In such a scenario, this task would become trivial, as we would simply need to spatially "warp" the feature maps computed from frame B according to the set of correspondences relating frame B to frame A. Based on this intuition, we design a learning procedure that achieves two goals: 1) By comparing Frame A and Frame B, our model must be able to extract motion offsets relating these two frames. 2) Using these motion offsets our model must be able to rewarp the feature maps extracted from Frame B in order to optimize the accuracy of pose detection in Frame A.

To achieve these goals, we first feed both frames through a backbone CNN with shared parameters. The aim of the backbone is to extract high-level discriminative features facilitating the computation of pose correspondences from the two frames. Then, the backbone feature maps from both frames are used to determine which feature locations from Frame B should be sampled for detection in Frame A. Finally, the resampled feature tensor from Frame B is used as input to the pose detection subnetwork which is optimized to maximize accuracy of Pose A.

Backbone Architecture. As our backbone network we use a Feature Pyramid Network [31] based on a ResNet-101 [24] design. We note, however, that our system is not constrained to this specific architecture design, and that it can easily integrate other backbone architectures as well.

Learning to Sample Features. Initially, we feed Frame A and Frame B through our backbone CNN, which outputs feature tensors $f_A^{(s)}$ and $f_B^{(s)}$ at four scales $s \in \{1, 2, 3, 4\}$. Then, we compute the difference $d_{A,B}^{(s)} = f_A^{(s)} - f_B^{(s)}$, at each of the four scales. The resulting feature tensor $d_{A,B}^{(s)}$ is provided as input to a 2D convolutional layer, which predicts offsets Δp_n (DIMOFS) at all locations p_n. The offsets are used to spatially rewarp (sample) the feature tensor $f_B^{(s)}$.

We implement the sampling mechanism via a deformable convolutional layer [11], which takes 1) the predicted offsets Δp_n, and 2) the feature tensor $f_B^{(s)}$ as its inputs, and then outputs a newly sampled feature tensor $g_{A,B}^{(s)}$. Intuitively, the newly resampled feature tensor $g_{A,B}^{(s)}$ should encode all the relevant information needed for accurate pose detection in Frame A. We use subscript (A, B) for $g_{A,B}^{(s)}$ to indicate that even though $g_{A,B}^{(s)}$ was resampled from tensor $f_B^{(s)}$, the offsets for resampling were computed using $d_{A,B}^{(s)}$, which contains information about both feature tensors. An illustration of our architecture is presented in Figure 2.

Multi-Scale Detection Head. We employ a multi-scale detection head as is done in the Feature Pyramid Network [31]. The outputs of RoI Align [22] applied on the resampled feature tensor $g_{A,B}^{(s)}$ are then fed into three branches optimized for the following 3 tasks: 1) binary classification (person or not person), 2) bounding box re-

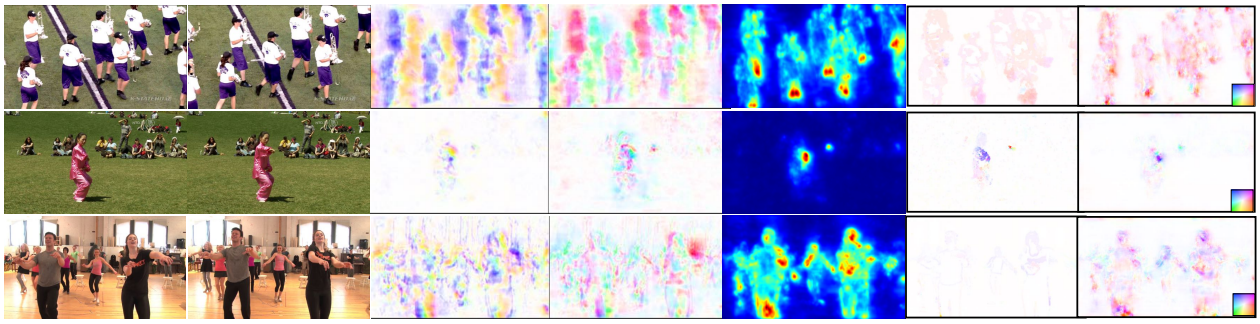

Figure 3: A figure with our DIMOFS visualizations. In the first two columns, we visualize a pair of video frames that are used as input by our model. The 3^{rd} and 4^{th} columns depict 2 (out of 256) randomly selected DIMOFS channels visualized as a motion field. It can be noticed that the DIMOFS capture primarily human motion, as they have been optimized for pose detection. Different channels appear to capture the motion of different body parts, thus performing a sort of motion decomposition of discriminative regions in the video. In the 5^{th} column, we display the magnitudes of summed DIMOFS channels, which highlight salient human motion. Finally, the last two columns illustrate the standard Farneback flow, and the human motion predicted from our DIMOFS. To predict human motion we train a **linear** classifier to regress the (x, y) displacement of each joint from the offset maps. The color wheel, at the bottom right corner encodes motion direction.

gression (predicting region bounds) and 3) pose estimation (outputting a probability heatmap per joint). Our classification and bounding box regression branches are based on the original Faster R-CNN design [39], whereas our pose heatmap branch is designed according to the architecture of the ICCV17 PoseTrack challenge winner [16].

4. Interpreting DiMoFs

Before discussing how DiMoFs can be employed in discriminative tasks (Section 5), we would like to first understand how the motion cues are encoded in DIMOFS. It turns out that interpreting what the DIMOFS have learned, is nearly as difficult as analyzing any other CNN features [54, 53]. The main challenge comes from the high dimensionality of DIMOFS: we are predicting 256 (x, y) displacements for every pixel at each of the four scales.

In Columns $3, 4$ of Figure 3, we visualize two randomly selected offset channels as a motion field. The DIMOFS visualized here were obtained by training our model for pose estimation on the PoseTrack dataset, as further discussed in the experiments section. Based on this figure, it is clear that DIMOFS focus on people in the videos. However, it is quite difficult to tell what kind of motion cues each of these channels is encoding. Specifically, different DIMOFS maps seem to encode different motions rather than all predicting the same solution (say, the optical flow between the two frames). This makes sense, as our DIMOFS are discriminatively trained. The network may decide to ignore motions of uninformative regions, while different DIMOFS maps may capture the motion of different human body parts (say, a hand as opposed to the head). Thus, our DIMOFS can be interpreted as producing a motion decomposition in different channels of the most discriminative cues in the video.

4.1. Human Motion Localization and Estimation

First, we observe that the magnitudes of our learned DIMOFS encode salient human motion, which we visualize in Column 5 of Figure 3. Then, to verify that our learned DIMOFS encode human motion relating the two frames we propose a simple visualization. For each point p_n denoting a body joint, we extract 2048-dimensional (512 channels concatenated across 4 scales) DIMOFS feature vector f_n at that specific point. Then, a *linear* classifier is trained to regress the ground truth (x, y) motion displacement of this body joint from the feature vector f_n. During inference, we apply our trained classifier on every pixel of the image in a fully convolutional manner via a 1×1 convolution.

In Column 7 of Figure 3, we visualize our predicted motion outputs. We show Farneback's optical flow in Column 6 of the same Figure. Note that our predicted human motion matches quite well Farneback optical flow, especially in regions containing people. The fact that we can recover flow with our very simple linear prediction scheme suggests that our learned DIMOFS capture cues related to human motion. Furthermore, we point out that compared to the standard Farneback optical flow, our motion fields look less noisy.

5. Detection and Recognition with DiMoFs

In this section, we discuss how DIMOFS benefit pose estimation and tracking. We also show how to adapt the DiMoFs architecture for the tasks of spatiotemporal action localization and fine-grained action recognition.

Pose Detection. During *training* for a video pose detection task (see Section 3), we select Frame B by sampling a frame δ time-steps from Frame A, where δ is randomly picked for each training frame from the set $\{-10, \ldots, 10\}$.

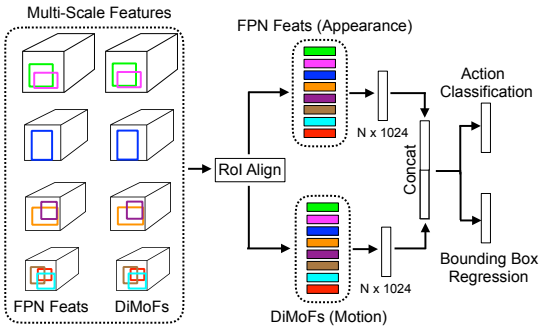

Figure 4: We extend our DIMOFS architecture for spatiotemporal action localization. Given a pair of video frames–Frame A and Frame B—we output for each person detected in Frame A, a bounding box and an action class. Up until the RoI Align, our model operates in the same way as for the pose detection task. Then, RoI Align is applied on 1) the multi-scale FPN feature tensors from Frame A (appearance), and 2) the multi-scale DIMOFS (motion). These RoI features are then fed into separate MLPs, and the resulting 1024-dimensional features are used to predict a bounding box and an action class for each RoI in Frame A.

During *inference*, we similarly feed a pair of frames into our model, and output pose predictions for Frame A by using the features from Frame B. Note that this also includes a special case when $\delta = 0$, meaning that Frame B is chosen to be the same as Frame A. In this special case, we can simply initialize the feature difference tensors $d^{(s)}_{A,B}$ to zeros, and then proceed as before. This allows us to train our pose detector on videos, and then later test it on still-images.

Keypoint Tracking. We consider the problem of tracking human body keypoints in video. We approach this task using the same ICCV17 PoseTrack winning model [16] as we did for pose detection. The tracking system in [16] consists of two stages: 1) keypoint estimation in frames, and 2) bipartite graph matching to link the tracks across adjacent frames. We simply replace the baseline model in stage 1) with our DIMOFS learned model. Note that both networks have the same network architecture, but differ in their training procedures. As will be shown later, our DIMOFS training scheme leads to better keypoint tracking results.

5.1. Spatiotemporal Action Localization

We introduce a few simple modifications to incorporate our learned DIMOFS into the Faster R-CNN architecture for spatiotemporal action localization (see Figure 4). Just as before, the model takes a pair of video frames as input. However, in this case the model outputs bounding boxes for Frame A, and an action class for each bounding box. We conjecture that a good spatiotemporal action localization model needs 1) strong visual appearance features associated with Frame A, and 2) motion features that encode how the person is moving between Frames A and B. While

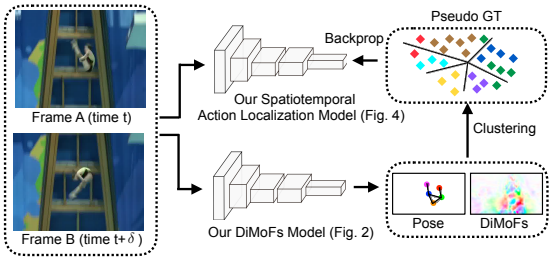

Figure 5: A high-level overview of our approach for learning fine-grained action recognition features. Initially, we use our DIMOFS model to extract pose and DIMOFS features from a given pair of video frames. We then accumulate these pose and DIMOFS features across the entire Diving48 training set, and cluster them using a k-means algorithm. The resulting cluster assignment IDs are then used as pseudo ground truth labels to optimize our action recognition features, as illustrated in Figure 4.

the appearance information is provided by the Frame A features computed through the backbone network, the motion cues are captured by our learned DIMOFS.

We keep the operations in the backbone the same as they were for the pose detection task. To adapt our pose estimation architecture to the problem of action localization, we remove all deformable convolutional layers as we do not need to resample features from B into Frame A anymore. Afterwards, we predict regions of interest (RoIs) in the same manner as we did for the pose detection task and then apply RoI Align separately on the visual features, and on the DIMOFS. Afterwards, separate 2-layer MLP head is applied to each type of features, and the resulting feature tensors are concatenated together to predict the bounding box, as well as its associated action class (See Figure 4).

5.2. Fine-Grained Action Recognition

Our model operates on frame pairs and it is not designed to process long video sequences, unlike 3D CNNs for action recognition [44, 8, 51, 45, 9, 36]. However, our DIMOFS model is explicitly designed to detect fine-grained cues such as human pose and subtle movements of various body parts. This suggests that DIMOFS can potentially be leveraged for fine-grained action recognition. We achieve this goal through a procedure inspired by the work of Caron et al. [7] who propose a two-stage deep clustering method for unsupervised learning of still-image features. Initially, we use our DIMOFS model (see Fig. 2) to extract the coordinates of each body part of every person in a video. We also extract DIMOFS features associated with each body joint and reduce them to 15 dimensions using PCA. We then concatenate these features and cluster them via K-means using $K = 100$ clusters. Intuitively, such a clustering procedure should yield clusters sharing similar pose and body motion patterns. We illustrate this procedure in Figure 5.

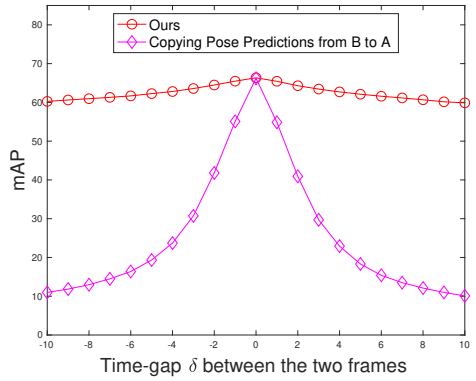

Figure 6: Pose detection results on the PoseTrack dataset. Our model predicts pose in Frame A using features from a Frame B that is δ time-steps away. During **training**, our model is optimized on frame pairs with random time-gaps $\delta \in \{-10, \ldots, 10\}$. Here, we present the results obtained by using different values of δ **at inference** time. As δ becomes large the motion between frames may become substantial. Yet, the accuracy of our model drops gracefully as δ deviates from 0. This confirms that our model estimates the human motion between the two frames reliably.

Afterwards, we train our spatiotemporal action localization network (see Figure 4) to predict such cluster IDs. Thus, instead of predicting action classes, we use cluster IDs as pseudo-ground truth labels. The rationale behind this choice is that directly predicting action classes from pairs of frames is a nearly-impossible task, especially in the case of fine-grained classification where the same poses occur in many different categories. On the other hand, predicting pose cluster IDs is a better posed task. At the same time, the aggregation of pose cluster IDs over the entire video (e.g. in the form of a simple histogram) can provide a strong descriptor for action recognition, disambiguating the recognition problem we had in a single pair of frames. Inspired by this intuition, we initialize our spatiotemporal action localization with weights learned during our DIMOFS training procedure and optimize it for pose cluster ID prediction. Afterwards, we evaluate the model on all pairs of frames in the video (sampled with time-gap $\delta = 5$) and obtain the 100 dimensional pose cluster ID prediction for each pair. The final classification is performed by training a shallow 2-layer MLP on the 100-dimensional vector obtained by summing up the pose ID activations for all pairs in the same video.

6. Experiments

In this section, we present results showing the benefit of our DIMOFS on the tasks discussed in the previous section: 1) pose detection, 2) keypoint tracking, 3) spatiotemporal action localization, and 4) fine-grained action recognition.

Method	Head	Shou	Elbo	Wri	Hip	Knee	Ank	Mean
Girdhar et al. [16]	72.8	75.6	65.3	54.3	63.5	60.9	51.8	64.1
[16] + DiMoFs	**75.2**	**78.5**	**66.8**	**56.0**	**66.7**	**62.7**	**54.0**	**66.3**

Table 1: Pose detection results on the PoseTrack dataset measured in mAP. The first row reports the results of a single-frame baseline [16], which won the ICCV17 PoseTrack challenge. In the second row, we present the results of our method, which is based on the same exact model as [16] but it is *trained* on pairs of frames with our DIMOFS training scheme (the *testing* is still done on individual frames). These results indicate consistent performance gains for each body joint, and a mean mAP improvement of 2.2%.

6.1. Pose Detection and Tracking on PoseTrack

In this section, we train and test our method on the PoseTrack [26] dataset, which contains 514 videos, 300 for training, 50 for validation and 208 for testing. The dataset includes 23,000 frames with annotated poses. Our evaluations are performed on the validation set. We demonstrate the effectiveness of our DIMOFS procedure by showing improved results for both pose estimation and keypoint tracking with respect to an analogous model trained on individual frames and thus unable to use motion information.

The Figure 6 illustrates our pose detection results according to the mAP metric for different values of the time-gap δ used at *testing* time. The δ values on the x-axis of Figure 6 captures how far apart Frames A and B are from each other. As expected, the more δ deviates from 0, the lower the accuracy becomes because the motion between two frames becomes more severe. We discuss below two key findings from the results we obtain.

Results when $\delta \neq 0$. We first observe that the accuracy of our model drops quite gracefully as the time-gap δ deviates from 0. By comparison, note the poor performance of the magenta curve which depicts a naïve baseline that simply copies pose detections from B to A (i.e., without using Frame A at all). For $\delta = \pm 10$, the accuracy of such baseline drops by 56.5% mAP with respect to the single-frame baseline [16]. In contrast, our model drops by only 6.1% mAP for the same $\delta = \pm 10$. The ability to predict poses from far-away frames suggests that our model reliably estimates the motion between the two frames and warps accurately the features of frame B for detection in frame A.

Results when $\delta = 0$. In Table 1, we compare our model with the single frame baseline of Girdhar et al. [16], which won the ICCV17 PoseTrack challenge. We note that our method is based on the same CNN architecture as [16], but is instead *trained* on pairs of frames using our DIMOFS training procedure. The *testing* when $\delta = 0$, however, is done on individual frames for both methods.

We observe that DIMOFS outperforms this strong baseline for all joints and by 2.2% on average. What are the reasons behind these gains? This happens primarily because

Method	Head	Shou	Elbo	Wri	Hip	Knee	Ank	Mean
Girdhar et al. [16]	61.7	65.5	57.3	45.7	54.3	53.1	45.7	55.2
[16] + DiMoFs	**62.9**	**67.2**	**57.4**	**45.8**	**55.5**	**53.5**	**46.5**	**56.0**

Table 2: Keypoint tracking results on the PoseTrack dataset using Multi-Object Tracking Accuracy (MOTA) metric. As our baseline, we use the ICCV17 PoseTrack challenge winning method [16]. Our model is trained using the same architecture as [16], but using our DIMOFS training scheme.

our DIMOFS training procedure on pairs of frames enables a form of data augmentation. In a single-frame training setting, the model learns a pattern connecting Frame A to Pose A. However, in our DIMOFS training, for each Frame A the model is learning patterns connecting many different Frames B (as δ is varied) to Label A. Thus, our model leverages many more (frame, label) pairs, which is beneficial.

Keypoint Tracking. In Table 2, we also demonstrate that our DIMOFS training procedure is helpful for the keypoint tracking task on the same PoseTrack dataset. The results are measured using the standard Multi-Object Tracking Accuracy (MOTA). We observe that DIMOFS yields gains in keypoint tracking across all body joints although not as substantial as in the case of pose estimation. We note that both methods in Table 2 use the same CNN architecture, but our model is trained using DIMOFS procedure.

6.2. Results of Action Localization on JHMDB

Next, we evaluate the effectiveness of our spatiotemporal action localization model (described in subsection 5.1). To do this, we use the JHMDB [27] dataset, which contains 928 temporally-trimmed clips representing 21 action classes, with bounding box annotations. We report our results on split 1 using the standard frame-mAP metric with an intersection-over-union (IOU) threshold set to 0.5.

To show that our new model can leverage the learned DIMOFS for spatiotemporal action localization, we test our model by feeding it pairs of frames, where Frame B is sampled by choosing $\delta = \{0, 2, 4, 6, \ldots, 38, 40\}$. We illustrate these results in Figure 7, which displays spatiotemporal action localization mAP versus the time-gap δ between the two frames. From this figure, we observe that the accuracy is at its lowest when $\delta = 0$, which makes sense because the model cannot leverage any motion cues. However, once we increase δ, the accuracy starts climbing steeply, which suggests that the model is leveraging the learned DIMOFS features for better spatiotemporal action localization performance. Based on the figure, we observe that the model reaches its peak performance at $\delta = 12$, which corresponds to approximately 0.5s in the original video. After that, the accuracy starts going down, which is expected because 1) the frames that are further away are less informative, and 2) it becomes harder to estimate long-range motion.

Furthermore, in Table 3, we ablate how different factors affect spatiotemporal action localization performance. First, we note that inflating a standard Faster R-CNN with the same backbone as our model to 3D [8] produces poor results (see column 1 of Table 3). Next, in columns 2 and 3, we report the accuracy of the model that uses exactly the same model architecture but where DIMOFS have not been trained for detection. From the table, we observe that such a model performs substantially worse (44.3% and 44.7% for $\delta = 0$ and 10 respectively) than our DIMOFS model (49.6% and 54.4% for $\delta = 0$ and 10 respectively). Furthermore, we point out that a model without DIMOFS training is not able to exploit motion cues effectively, which is indicated by a marginal 0.4% improvement between $\delta = 0$ and $\delta = 10$ settings. In contrast, our model exhibits a substantial 4.8% performance boost when increasing the δ from 0 to 10, which suggests the importance of learning DIMOFS discriminatively through a detection task. It is also worth noting, that pose information learned via DIMOFS training is important, i.e., even in a setting where $\delta = 0$, our DIMOFS model outperforms an equivalent Faster R-CNN by 5.3% (see columns 2 and 4 in Table 3). Lastly, we point out that learning the appearance features as before, but replacing implicit motion cues encoded by DIMOFS with explicit optical flow yields 50.6% mAP at $\delta = 10$, which is substantially worse than our 54.4%.

We also note that while models pretrained using a Kinetics dataset yield better performance [20, 43, 15], those results are not directly comparable to our evaluation since Kinetics is larger and contains many examples from the same classes as JHMDB. In this section, we aim to show that our model successfully leverages DIMOFS to improve action localization without resorting to additional action labels.

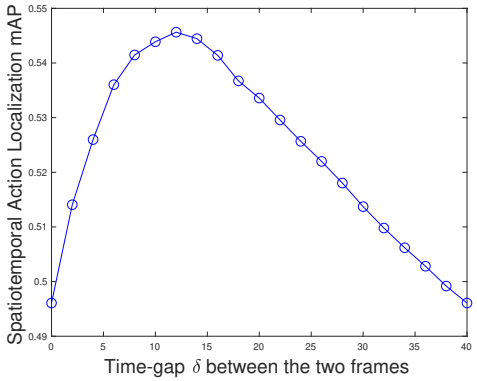

Figure 7: Our spatiotemporal action localization results on JHMDB dataset as we vary the time-gap δ separating Frame B from Frame A during inference. Based on these results, we observe that the performance is lowest at $\delta = 0$, which makes sense as there are no motion cues to leverage. However, once δ gets larger, the accuracy increases sharply indicating that the learned DIMOFS contain useful motion cues for spatiotemporal action localization task.

	1	2	3	4	5	6
DIMOFS Training?				✓		✓
Replacing DIMOFS with Optical Flow?					✓	
Faster R-CNN Inflated to 3D?	✓					
$\delta = 0$ at Inference?		✓		✓		
$\delta = 10$ at Inference?			✓		✓	✓
Mean Frame Average Precision (mAP)	43.8	44.3	44.7	49.6	50.6	**54.4**

Table 3: We examine how various factors affect spatiotemporal action localization performance on JHMDB. We observe that learning DIMOFS discriminatively through detection outperforms the model with the same architecture but that was not trained with DIMOFS scheme by 5.3% and 9.7% mAP for $\delta = 0$, and $\delta = 10$ respectively (compare column 2 to 4, and 3 to 6). We also note that replacing DIMOFS offsets with optical flow yields 50.6% mAP, which is 3.8% worse than using DIMOFS (column 5 vs 6).

6.3. Fine-Grained Action Recognition on Diving48

Finally, we evaluate our DIMOFS model on a fine-grained action recognition task (see subsection 5.2 for model description). For this experiment, we use the newly released Diving48 dataset [30], which contains 15,943 training and 2096 testing videos of professional divers performing 48 types of dives. We choose this dataset because unlike datasets such as Kinetics or UCF101, Diving48 is designed to minimize the bias towards particular scenes or objects. To do well on Diving48 dataset, it is necessary to model the subtle differences in body motion cues, and use them to predict the action class (i.e., the type of dive).

In Table 4, we present our quantitative results. In the top half of the table, we examine performance of our model in comparison to recent 2D CNNs: TSN [47] and TRN [56]. We also include an interesting baseline that uses directly the histogram of cluster ID from pose detection (bottom branch in Fig. 5) as features for action recognition (without training the top-branch in Fig. 5). This yields a pretty solid accuracy of 17.2%. By including DIMOFS (in addition to pose) for clustering, the accuracy jumps to 21.4%. Note, that if we replace implicit motion cues encoded by DIMOFS with explicit optical flow the accuracy drops to 18.8%, which is substantially worse than our 21.4%. We also point out that training our spatiotemporal action recognition model to predict pose cluster IDs further improves the performance, and allows our model to achieve 24.1%. These results suggest 1) the usefulness of our learned pose and DIMOFS features, and also 2) highlight the benefits of optimizing the network to pseudo ground-truth pose cluster IDs.

In the lower half of the table, we compare our method with popular 3D models [44, 45] . Our model does not perform as well as the pre-trained long-range 3D CNNs. However, this comparison is not exactly fair as these 3D models have been pre-trained on larger-scale action recognition dataset such as Kinetics [8]. In contrast, our model is pre-trained on a much smaller PoseTrack dataset, which requires significantly less computational resources. Furthermore, our pre-training is done on a more general pose estimation task, which allows our learned DIMOFS to generalize to a variety of different tasks as shown in the paper. We note that without Kinetics pre-training, our model outperforms R(2+1)D [45] baseline by a solid 2.7% margin. Finally, we also show that combining our model with the pre-trained R(2+1)D [45] allows us to further improve the performance, and achieve state-of-the-art results, which suggests that the two models learn complementary cues.

2D Models	Pre-training Data	Accuracy
TSN (RGB) [47]	ImageNet (objects)	16.8
TSN (RGB + Flow) [47]	ImageNet (objects)	20.3
TRN [56]	ImageNet (objects)	22.8
Ours-init (Pose)	PoseTrack (poses)	17.2
Ours-init (Pose+Optical Flow)	PoseTrack (poses)	18.8
Ours-init (Pose+DIMOFS)	PoseTrack (poses)	21.4
Ours-final (Pose+DIMOFS)	PoseTrack (poses)	**24.1**

3D Models	Pre-training Data	Accuracy
R(2+1)D [45]	None	21.4
C3D [44]	Sports1M (actions)	27.6
R(2+1)D [45]	Kinetics (actions)	28.9
Ours + R(2+1)D [45]	Kinetics + PoseTrack	**31.4**

Table 4: Our results on Diving48 dataset. In the top half of the table, we show that our system achieves 24.1% accuracy and outperforms 2D methods TSN [47] and TRN [56]. We also observe that even using the initial cluster ID histograms as features produces solid results (i.e. 17.2% and 21.4%). Note that replacing DIMOFS with optical flow reduces the accuracy by 2.6% (compared to our 21.4%). Our model does not perform as well as the pre-trained R(2+1)D [45] baseline. However, our pre-training is done on a much smaller dataset, and on a more general pose detection task. Without Kinetics pre-training, we outperform R(2+1)D [45] by a solid 2.7% margin. Finally, we show that combining DIMOFS with pre-trained R(2+1)D [45] improves the results, suggesting complementarity of the two methods.

7. Conclusions

In this work, we introduced DIMOFS, discriminatively trained offsets that encode human motion cues. We showed that our DIMOFS can be used to localize and estimate salient human motion. Furthermore, we show that as a byproduct of our DIMOFS learning scheme, our model also learns features that lead to improved pose detection, and better keypoint tracking. Finally, we showed how to leverage our learned DIMOFS features for spatiotemporal action localization and fine-grained action recognition tasks. Our future work involves designing unified CNN architectures that can leverage the strengths of modern 3D CNNs, and detection-based systems. We will release our source code and our trained models upon publication of the paper.

References

[1] G. Bertasius, J. Shi, and L. Torresani. Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1

[2] G. Bertasius, L. Torresani, and J. Shi. Object detection in video with spatiotemporal sampling networks. In *ECCV (12)*, volume 11216 of *Lecture Notes in Computer Science*, pages 342–357. Springer, 2018. 2

[3] G. Bertasius, L. Torresani, S. X. Yu, and J. Shi. Convolutional random walk networks for semantic image segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1

[4] H. Bilen, B. Fernando, E. Gavves, and A. Vedaldi. Action recognition with dynamic image networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2799–2813, 2017. 2

[5] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould. Dynamic image networks for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2

[6] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. 1

[7] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, pages 139–156, 2018. 5

[8] J. Carreira and A. Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4724–4733, 2017. 1, 2, 5, 7, 8

[9] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, 2017. 2, 5

[10] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems 29*, pages 379–387. Curran Associates, Inc., 2016. 1, 2

[11] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, volume 00, pages 764–773, Oct. 2017. 2, 3

[12] C. Feichtenhofer, A. Pinz, and R. Wildes. Spatiotemporal residual networks for video action recognition. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3468–3476, 2016. 2

[13] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal multiplier networks for video action recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[14] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1, 2

[15] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman. A better baseline for AVA. *CoRR*, abs/1807.10066, 2018. 7

[16] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, and D. Tran. Detect-and-Track: Efficient Pose Estimation in Videos. In *CVPR*, 2018. 4, 5, 6, 7

[17] R. Girdhar and D. Ramanan. Attentional pooling for action recognition. In *NIPS*, 2017. 2

[18] R. Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 1, 2

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1, 2

[20] C. Gu, C. Sun, S. Vijayanarasimhan, C. Pantofaru, D. A. Ross, G. Toderici, Y. Li, S. Ricco, R. Sukthankar, C. Schmid, and J. Malik. AVA: A video dataset of spatio-temporally localized atomic visual actions. *arXiv preprint arXiv:1705.08421*, 2017. 7

[21] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*, 2014. 1, 2

[22] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 1, 2, 3

[23] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *Computer Vision – ECCV 2014*, 2014. 1, 2

[24] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2, 3

[25] D.-A. Huang, V. Ramanathan, D. Mahajan, L. Torresani, M. Paluri, L. Fei-Fei, and J. C. Niebles. What makes a video a video : Analyzing temporal information in video understanding models and datasets. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, USA, 2018*, 2018. 1, 2

[26] U. Iqbal, A. Milan, and J. Gall. Posetrack: Joint multi-person pose estimation and tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6

[27] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3192–3199, Sydney, Australia, Dec. 2013. IEEE. 7

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

[29] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011. 2

[30] Y. Li, Y. Li, and N. Vasconcelos. Resound: Towards action recognition without representation bias. In *The European Conference on Computer Vision (ECCV)*, September 2018. 8

[31] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 3

[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal Loss for Dense Object Detection. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 1, 2

[33] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1, 2

[34] J. Y.-H. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, pages 4694–4702. IEEE Computer Society, 2016. 2

[35] X. Peng and C. Schmid. Multi-region two-stream R-CNN for action detection. In *ECCV - European Conference on Computer Vision*, volume 9908 of *Lecture Notes in Computer Science*, pages 744–759, Amsterdam, Netherlands, Oct. 2016. Springer. 1, 2

[36] Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *ICCV*, 2017. 2, 5

[37] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 779–788, 2016. 1, 2

[38] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6517–6525, 2017. 1, 2

[39] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015. 1, 2, 4

[40] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 568–576. Curran Associates, Inc., 2014. 1, 2

[41] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2

[42] K. Soomro, A. R. Zamir, M. Shah, K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, 2012. 2

[43] C. Sun, A. Shrivastava, C. Vondrick, K. Murphy, R. Sukthankar, and C. Schmid. Actor-centric relation network. In *ECCV (11)*, volume 11215 of *Lecture Notes in Computer Science*, pages 335–351. Springer, 2018. 7

[44] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 4489–4497, Washington, DC, USA, 2015. IEEE Computer Society. 1, 2, 5, 8

[45] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, USA, 2018*, 2018. 2, 5, 8

[46] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV (8)*, volume 9912 of *Lecture Notes in Computer Science*, pages 20–36. Springer, 2016. 2

[47] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 8

[48] X. Wang, A. Farhadi, and A. Gupta. Actions ~ transformations. In *CVPR*, 2016. 2

[49] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016. 1

[50] F. Xiao and Y. J. Lee. Video object detection with an aligned spatial-temporal memory. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[51] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XV*, pages 318–335, 2018. 2, 5

[52] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, 2015. 1

[53] J. Yosinski, J. Clune, A. M. Nguyen, T. J. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579, 2015. 4

[54] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. 4

[55] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*, 2015. 1

[56] B. Zhou, A. Andonian, A. Oliva, and A. Torralba. Temporal relational reasoning in videos. *European Conference on Computer Vision*, 2018. 2, 8

[57] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei. Flow-guided feature aggregation for video object detection. In *International Conference on Computer Vision (ICCV)*, 2017. 2