Actor Conditioned Attention Maps for Video Action Detection

Oytun Ulutan[1] Swati Rallapalli[2] Mudhakar Srivatsa[2]
Carlos Torres[1] B.S. Manjunath[1]

[1]University of California, Santa Barbara [2]IBM T. J. Watson Research Centre

Abstract

While observing complex events with multiple actors, humans do not assess each actor separately, but infer from the context. The surrounding context provides essential information for understanding actions. To this end, we propose to replace region of interest(RoI) pooling with an attention module, which ranks each spatio-temporal region's relevance to a detected actor instead of cropping. We refer to these as Actor-Conditioned Attention Maps (ACAM), which amplify/dampen the features extracted from the entire scene. The resulting actor-conditioned features focus the model on regions that are relevant to the conditioned actor. For actor localization, we leverage pre-trained object detectors, which transfer better. The proposed model is efficient and our action detection pipeline achieves near real-time performance. Experimental results on AVA 2.1 and JHMDB demonstrate the effectiveness of attention maps, with improvements of 7 *mAP on AVA and* 4 *mAP on JHMDB.*

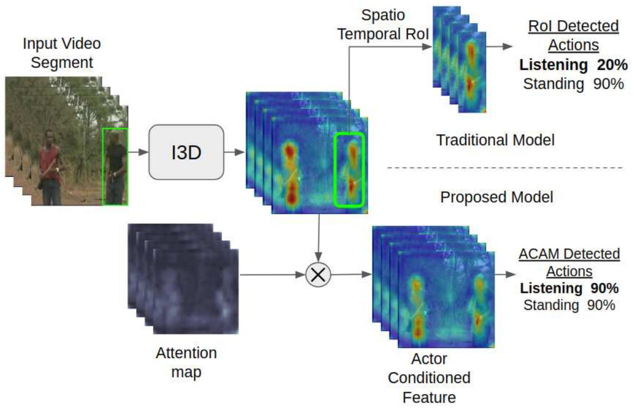

Figure 1. Comparing RoI pooling with the proposed ACAM method for video action detection. ACAM explicitly models the surrounding context and generates features from the complete scene by conditioning them on detected actors. For example, presence of a talking person next to the actor is evidence for the "listening" action, which is captured by attention maps.

1. Introduction

Motivation: Human action detection is a promising field, which can improve applications such as surveillance, robotics and autonomous driving. While many datasets (e.g., HMDB-51[21], Kinetics[19], UCF-101[36]) are very useful for video search and classification, a recent AVA[9] dataset focuses on atomic actions within short video segments. Atomic actions have the potential to transfer to different contexts, become building blocks for more complex actions and improve the general understanding of human actions/interactions in videos. For these reasons, in this work we focus on the task of atomic action detection from videos. We propose to model actor actions by using information from the surrounding context and evaluate our model on AVA[9] and JHMDB[15] datasets. We demonstrate the efficiency and transferability of our approach by implementing an action detection pipeline and qualitatively testing it on videos from various sources.

Challenges: While observing actions/activities, humans infer from the entire context and our perception depends on the surrounding objects, actors, and scene. This is a concept that has been widely studied in neuroscience and psychology [4, 11, 28, 38]. The idea of explicitly leveraging context is directly relevant to our action detection task as surroundings of actors provide valuable information.

Studies in action detection task have followed the ideas from the R-CNN architectures and extended it to videos[9, 27, 30, 35]. However, in action detection, the bounding box locates the *actor* rather than the *action* itself and datasets do not include explicit interaction labels for the actions, which makes it challenging to model context. In such setting, RoIPooling becomes insufficient for modeling actions and including the contextual information such as actors, objects and scene. In order to address this, we propose attention maps as a replacement for RoIPooling for action detection. The proposed methodology learns context in a weakly supervised manner as demonstrated in Fig. 1.

Approach: Inspired by the context [42], attention [41] and relation [37] ideas, the proposed model generates attention maps *conditioned on each actor from contextual and actor features*. Replacing the traditional way of cropping an actor RoI from the context feature maps, generated attention maps multiply and scale the context feature maps according to each actor. The scaling operation amplifies relevant regions and dampens the irrelevant regions to the conditioned actor. This allows the model to learn the complex interactions for the current actor with the scene which may include surrounding actors, objects and background.

Technical Contributions

- **Generation of ACAMs:** We propose an attention mechanism for person action detection that models the surrounding context of actors. These maps replace the RoIPooling with attention maps, which amplify and dampen regions conditioned on the actor.
- **Object detectors as transferable and modular region proposal networks (RPN):** Instead of retraining an RPN on the dataset, we use a pre-trained object detector to obtain accurate actor locations and demonstrate that it is more transferable to unseen data.
- **End-to-end pipeline for real-time video action detection on videos:** We implement a pipeline and qualitatively show that our approach transfers well on videos from various types of unseen sources.

Code will be made available at Github. A real-time demo of the pipeline from Section 4.5 is also available at Github.

2. Related Work

State of the art models on the earlier action recognition datasets [21, 33, 36] use models such as Two-Stream networks [34] combining RGB with Optical Flow, 2D Convolutions with LSTMs [45] and 3D Convolutions [12]. The release of the large-scale, high quality datasets like Sports 1M [18], Kinetics [19], allowed deeper 3D CNN models such as C3D [39], Inception 3D (I3D) [2] to be trained and achieve high performance.

In this work, we are focusing on the Atomic Visual Actions (AVA v2.1) [9] dataset. This dataset exhaustively annotates the atomic actions and spatial locations of all the actors. Initial methods on the AVA dataset extended the Faster-RCNN [29] architectures to 3D convolutions, where initial layers generate actor proposals and each proposal is analyzed by subsequent layers [9]. The recent Actor Centric Relation Network (ACRN) [37] model generates features by combining actor and scenes to represent actor's interactions with surrounding context. Our proposed model leverages this relation idea to generate attention maps.

An attention function for relating different positions of a sentence was used in [41]. The relation module in [32] combines questions with vision to generate answers. [13] uses object relations to effectively detect them. An LSTM structure is used in [22] to generate an attention map to model contextual information. Inspired by these attention models which use positional relations, our proposed model generates a spatio-temporal attention map which scales different positions of the feature map depending on each actor.

Recently, [42] used a compact feature representation that compresses non-local information from contextual features with a weighted sum of pixels for action detection and achieved state-of-the-art results. This shows that contextual information is essential and detection can be improved by replacing the RoIPooling which ignores context.

Context has also been studied on image action detection. V-COCO [10] and HICO-Det [3] datasets have exhaustive annotations on persons, objects and their interactions. Unlike our task where the interactions are weakly supervised, these annotations enable models to learn interactions efficiently. Interaction modeling from [8] achieved good performance with a multi-stream network where each stream focused on people, objects and interactions separately.

Our proposed model builds on top of the relation idea from ACRN [37] where the relation between the actor and the surrounding context is generated. Unlike ACRN where the relation features are used for classification, our model leverages the relation function to generate attention maps. This is similar to [41] where attention maps localize relevant parts in a feature map. However, in our task, the attention maps condition the features to each actor in the scene individually. Since the feature maps are conditioned individually and include context, ACAM models actor-context information more effectively than RoIPooling.

3. Proposed Method

This section describes our proposed model for action detection. From each input video segment, the objective is to detect bounding boxes for each actor and classify their actions. Each actor can have multiple action labels (e.g., "sitting" and "talking" simultaneously).

3.1. Context for Atomic Actions

Compared to object detection tasks, action boundaries are ill defined and can include interactions with the surrounding context (objects, actors and scene). Different actions require different sizes of visual areas to be considered from the input video. For example, the "walking" action requires the model to consider only the pixels on the actor and close surrounding context, whereas the "listening" action requires the model to look for at a larger context area (e.g., a talking person) around the actor in addition to the actor itself. With such variety in action classes, using traditional object detection methods such as RoIPooling can potentially lose the contextual information around the actors. Even though features cropped using RoIPooling include information from a larger receptive field, this technique compresses the information into a smaller feature map and does not explicitly model interactions (with other actors and context). Additionally, large-scale video datasets do not provide explicit interaction labels (person 1 is listening to person 2) but weak labels (person 1 - listening, person 2 - talk-

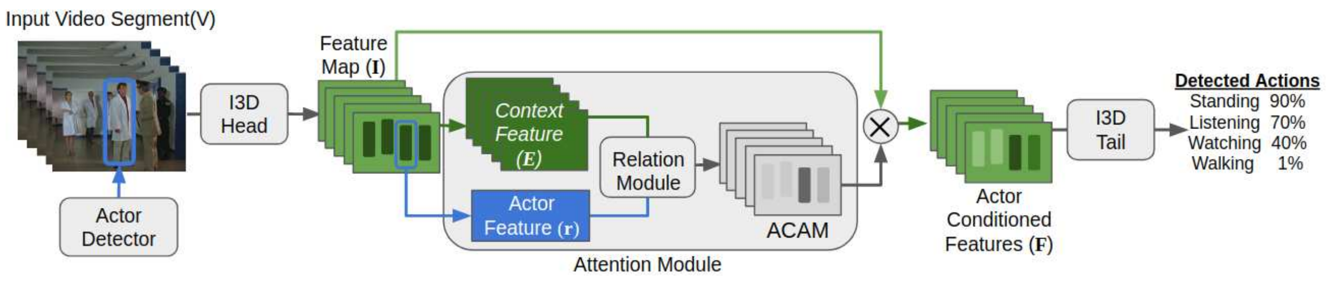

Figure 2. ACAM architecture. The input video segments are processed by the I3D back-bone. Feature vectors for each detected actor are generated from their locations on the feature map. A set of weights is generated for every spatio-temporal region in the scene by combining the actor features and contextual features extracted from the entire scene. These weights, i.e., attention maps, are multiplied by the feature map and the result represents the actor conditioned features. Four detected actors are represented by four vertical bars in I. One focused actor (boxed) is listening to a close-by actor. This action is captured by larger weights in the attention map shown as a darker vertical bar.

ing). These interactions need to be learned via weak supervision. In order to address these challenges, the proposed method generates attention maps for each detected actor to model the importance of each spatio-temporal region in the feature map by conditioning on detected actors. The proposed model architecture is shown in Fig. 2.

3.2. Actor Conditioned Attention Maps

The proposed method generates a set of weights (ACAMs) that represent the attention of different parts of the spatio-temporal input. Our action detection problem contains multiple actors performing concurrent actions that can be either related or disparate. This generates an attention task, where different actors relate differently to spatio-temporal locations in the scene. The proposed model addresses the attention problem by generating ACAMs, which capture relations between actors and context. Instead of extending RoIPooling [29] to action detection as in [9], the essence of ACAMs is to condition the features extracted from the entire scene on each actor and action dynamics.

Spatio-temporal features are extracted from the input video V with a 3D convolutional back-bone (e.g.,: I3D [2]) up-to some layer (Mixed_4f). Let $\mathbf{I}$ represent the extracted feature tensor of size $(T \times H \times W \times C)$ with temporal resolution T and spatial resolution $H \times W$ indexed by t, h, and w and feature channel dimension C. i.e., $\mathbf{I} = conv3d(V)$.

The actor feature vector $\mathbf{r}_a$ of size N (set to $C/4$) is extracted for actor a using RoI pooling extended time via:

$$\mathbf{r}_a = \phi(\mathbf{w}_\rho RoI(\mathbf{I}, a) + \mathbf{b}_\rho), \tag{1}$$

where $\phi(x) = ReLU(x) = max(0, x)$, $\mathbf{w}_\rho$ are the weights, and $\mathbf{b}_\rho$ are the biases. Similar to Faster RCNN [29], $\mathbf{r}_a$ can be used for classifying the actions of actor a. Instead of using $\mathbf{r}_a$ directly, we propose to leverage its descriptive potential to generate relations between the actor and the context.

The conditioned feature vector $\mathbf{F}_{t,h,w|a}$ is computed for each actor a in the scene and spatio-temporal indices (t, h, w). This is generated by a conditioning function of actor feature $\mathbf{r}_a$ and contextual features $\mathbf{I}_{t,h,w}$ via:

$$\mathbf{F}_{t,h,w|a} = Condition(\mathbf{I}_{t,h,w}|\mathbf{r}_a,), \; \forall \, (t, h, w) \tag{2}$$

Following steps explain the $Condition$ function. Activations in $\mathbf{I}$ are sparse; however, it is compressed by an additional layer to obtain a denser representation ($\mathbf{E}$) via:

$$\mathbf{E}_{t,h,w} = \phi(\mathbf{w}_\eta \mathbf{I}_{t,h,w} + \mathbf{b}_\eta), \tag{3}$$

where $\mathbf{w}_\eta$ and $\mathbf{b}_\eta$ are the weights and biases. The new tensor $\mathbf{E}$ has shape $(T \times H \times W \times M)$ with $M < C$ (set $M = C/4$). This approach reduces the dimensionality of $\mathbf{I}$ and captures higher level information similar to [42, 2, 40].

The relation tensor for actor a (i.e., $\mathbf{R}_a$) is inspired by the "relation" idea from [32] and it is modified to capture the relations between actor a and every location t, h, w in the context as:

$$\mathbf{R}_{a,t,h,w} = \mathbf{w}_\Omega \mathbf{r}_a + \mathbf{w}_\gamma \mathbf{E}_{t,h,w} + \mathbf{b}_\beta, \tag{4}$$

where $\mathbf{w}_\Omega$ and $\mathbf{w}_\gamma$ are the weights for actor and context features, respectively; and $\mathbf{b}_\beta$ are the biases. $\mathbf{R}_{a,t,h,w}$ describe the relation of actor features and contextual locations. We set up the shapes of $\mathbf{w}_\Omega, \mathbf{w}_\gamma, \mathbf{b}_\beta$ such that $\mathbf{R_a}$ has the same shape as $\mathbf{I}$. Note that ReLU (ϕ) is not used in Eq. 4.

Instead of using relation features for classification directly, we leverage the I3D back-bone and its pre-trained weights by conditioning $\mathbf{I}$ on the actor a for an increased performance (Section 4.3). Inspired by the "forget" gates of LSTMs, attention module generates the actor conditioned attention maps for a (i.e., $\mathbf{ACAM}_a$) by:

$$\mathbf{ACAM}_{a,t,h,w} = \sigma(\mathbf{R}_{a,t,h,w}), \tag{5}$$

and conditioned features $\mathbf{F}$ as follows:

$$\mathbf{F}_{t,h,w|a} = \mathbf{I}_{t,h,w} \odot \mathbf{ACAM}_{a,t,h,w} \tag{6}$$

where σ is the sigmoid function which scales the attention maps in $[0, 1]$ interval and $\odot$ is the elementwise multiplication of vectors. Attention maps multiplied by $\mathbf{I}_{t,h,w}$ weights

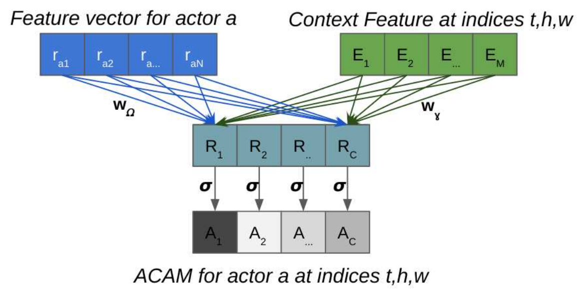
Feature vector for actor a
Context Feature at indices t,h,w
r_{a1} r_{a2} $r_{a...}$ r_{aN}
E_1 E_2 $E_{...}$ E_M
w_Ω
w_γ
R_1 R_2 $R_{..}$ R_C
σ σ σ σ
A_1 A_2 $A_{...}$ A_C
ACAM for actor a at indices t,h,w

Figure 3. Attention module for actor a at a single index t, h, w. Attention weights in ACAM at index t, h, w are generated from the actor feature $\mathbf{r}_a$ and context features at the same index $\mathbf{E}_{t,h,w}$.

the different regions on the context. This process amplifies regions relevant to actor a, while damping the irrelevant regions. The generation of ACAMs is shown in Fig. 3 for actor a at a single spatio-temporal index t, h, w.

These operations are efficiently computed using $1 \times 1 \times 1$ convolutions. For instance, Eq. 3 are fully connected layers repeated for every t, h, w index, which is equivalent to $1 \times 1 \times 1$ convolutions. In Eq. 4, $\mathbf{r}_a$ is constant for all indices t, h, w as shown in Fig. 4. This equation is computed by repeating $\mathbf{r}_a$ of shape $1 \times 1 \times 1 \times N$ to match the spatio-temporal shape of $\mathbf{E}$. The repeated actor feature has shape $T \times H \times W \times N$ and is concatenated with $\mathbf{E}$ to produce a tensor with shape $T \times H \times W \times (N + M)$. Applying the $1 \times 1 \times 1$ convolutions to the concatenated tensor is equivalent to Eq. 4 and produces $\mathbf{R}$. The sigmoid operation (Eq. 5) on $\mathbf{R}$ generates the attention maps (**ACAM**). Element-wise multiplication from Eq. 6 generates $\mathbf{F}$, which is then classified by remaining layers of the CNN back-bone.

3.3. Person Detectors as RPN

We experiment with pre-trained(COCO) frozen and fine-tuned(AVA) person detectors for actor localization. Our approach has the following three advantages over RPNs:

1. Transferability: Object detectors see large object variations (MS-COCO [23]). This allows models trained on object detection datasets to transfer to videos from different sources. Action datasets, however, usually come from similar sources such as AVA (Movies), JHMDB (Youtube), which reduces the diversity in actor views and limits transferability of fine-tuned solutions for actor localization.

2. Efficiency: ACAM requires fewer actor proposals than RoI pooling to enable its complex computations. These detections are obtained from pre-trained person detectors.

3. Modularity: The action model is trained using a slow and highly accurate actor detector. The modularity of the proposed methodology enables replacing detectors based on performance and application requirements. For example, a faster detector used for testing achieves near real-time performance, which is demonstrated in Section 4.5.

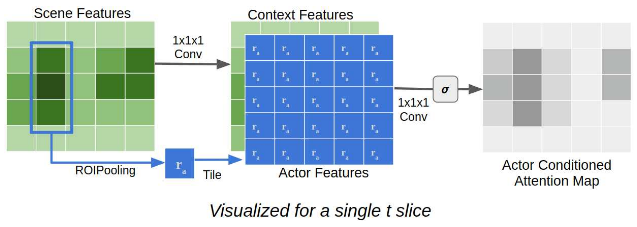

Figure 4. Calculation of attention maps with convolutions. Actor feature from the RoI is tiled and concatenated to features extracted from the context at every spatio-temporal index. Convolutions on the combined feature calculate the relations from Eq 4 efficiently.

4. Experiments and Evaluations

4.1. Datasets and Implementation Details

Datasets: The proposed ACAM model is tested on the AVA v2.1[9] and JHMDB[15] datasets.

AVA contains 2-second video segments of multiple actors with 211k training and 57k validation samples. Actor bounding boxes are annotated for the center frames only. Weak action labels are provided for the complete segment without temporal localization or explicit interactions. Actors can have multiple action labels in each segment. We follow the AVA v2.1 evaluation process and calculate the mean Average Precision (mAP) across 60 classes. There are three action super classes Person Poses (13 classes), Object Interactions (32 classes), Person Interactions (15 classes).

JHMDB contains 1-second video segments of 21 action classes across 928 video clips with single actor-action pairs.

3D CNNs: We use I3D [2] as the 3D CNN back-bone for all of our model candidates. The input video segment of RGB frames is processed by the initial I3D layers until the "Mixed_4f" layer to obtain the feature tensor $\mathbf{I}$ of size $8 \times 25 \times 25 \times 832$. The actor conditioned features $\mathbf{F}$ are computed using ACAM, where $\mathbf{F}$ is a weighted version of the original feature map ($\mathbf{I}$). The remaining I3D layers are used and initialized with pre-trained weights. We use the remaining layers up to final "Mixed_5c" for classification on $\mathbf{F}$ and call this operation "I3D Tail". A global average pooling across spatio-temporal dimensions is applied to the final feature map to compute class probabilities. Each 2-second video is uniformly subsampled down to 32 frames.

Actor Detection: Detectors process all the videos and store the detected actors locations. We use the Faster R-CNN [29] with NAS [47] detector pre-trained on MS-COCO [23] dataset. Additionally, we fine-tune the detector for actors and compare their performances on AVA and transferability on other datasets. This object detector is further analyzed and available in Tensorflow Object Detection API [14].

Data Augmentations: In addition to cropping and flipping the video sequences, we augment the actor box coordinates from the detector. This generates a slight difference in extracted $\mathbf{r}_a$ at each training step and reduces overfitting.

Model Architecture	AVA v2.1 Validation mAP
Single Frame[9]	14.20
I3D [9]	15.10
ACRN [37]	17.40
ACAM	23.29
ACAM - tuned	**24.38**

Table 1. Validation mAP results compared to published state of the art results. Proposed ACAM model with fine-tuned actor detector achieves the highest performance on the AVA v2.1 Validation set.

Model Architecture	AVA v2.1 Validation mAP
YH Technologies[44]	19.40
Megvii/Tsinghua[16]	20.01
Deep Mind[7]	21.90
ACAM	23.29
ACAM - tuned	**24.38**

Table 2. ACAM mAP results compared with models from the ActivityNet CVPR-2018 AVA challenge. We excluded the ensemble/fusion methods to evaluate the benefits of the proposed layer.

Training: We initialize our models with I3D weights trained on Kinetics-400 dataset [2] and train our models with Adam optimizer [20] and cosine learning rate [25] between max (0.02) and min (0.0001) for 70 epochs. We use a batch size of 2 per GPU and 4 Nvidia 1080Ti (total batch size of 8). Batch-norm updates are disabled. All models are implemented in Tensorflow [1].

4.2. Comparisons with the State of the Art

Table 1 shows ACAM with fine-tuned object detector outperforming the recent ACRN [37] on AVA validation set by 7mAP. We compare our model with validation results of the models from "ActivityNet 2018 AVA challenge"[6]. Table 2 shows that ACAM outperforms in validation. The table excludes results from ensemble models and focuses on comparing their highest performing single RGB model.

Additionally, we compare the effects of fine-tuning the actor detector on AVA dataset. Comparing ACAM with **ACAM-tuned** demonstrates the improvement gained by fine-tuning the actor detector on the AVA dataset. However, this comes with trade-offs as analyzed in Section 4.6.

4.3. Comparisons of Individual Modules

In this section, we demonstrate the purpose of each module and experiment with alternative models to ACAM for representing contextual interactions. These experiments use actor detectors that are pre-trained, frozen on COCO dataset and are not fine-tuned for AVA. AVA mAP results, inference speeds and number of parameters for these implementations are shown in Table 3 to analyze the trade-offs.

I3D Head + RoIPool (Base Model): The base in-house implementation follows the model from [9] and achieves 18.01 mAP. The input video goes through I3D convolutions upto the layer "Mixed_4f" and call it I3D Head. Using RoI pooling the actor feature vector $\mathbf{r}_a$ is obtained and used with fully connected layers for classification.

Model Architecture	AVA mAP	Test Speed	# Params
I3D Head + RoIPool (Base)	18.01	8.3 samples/s	7,573,244
I3D Head + RoIPool + Tail	19.83	8.1 samples/s	12,341,484
I3D Head + ACRN + Tail	20.59	7.1 samples/s	13,034,956
I3D Head + NL-RoI + Tail	20.82	7.5 samples/s	13,035,164
I3D Head + ACAM + Tail	**23.29**	6.9 samples/s	13,034,956

Table 3. mAP results of our different variants. We compare ACAM with alternate attention modules and base models. Inference speed is measured on a single 1080Ti GPU. Number of parameters include I3D Head and Tail parameters for applicable models.

Model Architecture	Pose	Objects	Interaction
I3D Head + RoIPool	36.88	9.87	19.02
I3D Head + RoIPool + Tail	38.45	12.11	20.16
I3D Head + ACRN + Tail	38.38	12.52	22.37
I3D Head + NL-RoI + Tail	40.50	12.06	22.45
I3D Head + ACAM + Tail	**42.13**	**15.02**	**24.22**

Table 4. mAP comparisons of different RoI variants on AVA action super classes. Pose: Person Pose actions (ex: walking, standing), Objects: Object Manipulation (ex: drink, pull), Interaction: Person Interaction (ex: talk to a person, watch a person).

I3D Head + RoIPool + I3D Tail: This model uses RoI pooling to extract actor features. Instead of vectorizing the RoI and using fully connected layers, we use the remaining I3D layers from "Mixed_4f" to "Mixed_5c" similar to the model from [7]. This method achieves 19.83 mAP and demonstrates that using I3D Tail improves the performance.

I3D Head + NL-RoI + I3D Tail: Non-Local Neural Networks [42] uses a compact representation to model interactions between different spatio-temporal regions in a video segment. We modify this model to generate non-local features between the detected actor features $\mathbf{r}_a$ and scene context features $\mathbf{I}$. This model achieves 20.82 mAP.

I3D Head + ACRN + I3D Tail: Similar to the ACRN [37], this implementation classifies on the relation features ($\mathbf{R}$). To improve performance, we use "I3D Tail" instead of the added 3×3 convolutions. This model achieves 20.59 mAP. Comparing ACAM to this model demonstrates that attention based context is better than relation features alone.

I3D Head + ACAM + I3D Tail: This model uses the proposed ACAMs to condition context features on actors and classifies the actions using I3D Tail. This model achieves 23.29 mAP, which is the highest performance when compared to the alternative relation models.

The breakdown of performance per action super class is demonstrated in Table 4. the AP values in the table are averaged across super classes. This experiment demonstrates leveraging contextual information with ACAM improves the performance for every super class.

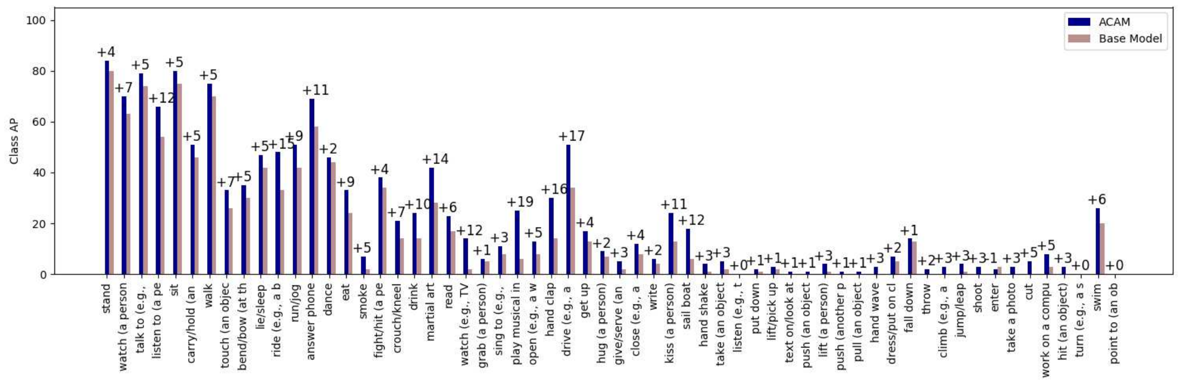

Figure 5. Per class AP results for the proposed ACAM model and the base model I3D Head + RoIPool on the AVA dataset. The classes are sorted by the number of training samples available in the dataset. Improvements achieved by ACAM are visualized on the bars.

The per class performance (AP) comparison on the proposed ACAM model and the base model is shown in Fig. 5. We observe significant (above 10 AP) improvements in passive actions such as "listen a person" and "watch TV" as in those classes context is active. Scene context improves the detection of classes such as "drive" and "play instrument".

4.4. Results on JHMDB

In addition to the AVA dataset, we evaluate our models on the JHMDB [15] dataset. We follow the evaluation protocol and cross-validate and report the video and frame mAP results on three splits. We edit and use the evaluation script from [27]. Our models are fine-tuned from Kinetics-400 weights and are trained for 10 epochs. Table 5 shows the video mAP scores of three of our models on JHMDB and demonstrates that the proposed ACAM model achieves the best performance. Proposed ACAM model achieves the best performance across all implementations which is consistent with the AVA dataset results.

Table 6 compares video and frame mAP of our ACAM model with state-of-the-art models. ACAM outperforms the by 3.80 video mAP and 1 frame mAP without optical flow.

JHMDB - Models	Split1	Split2	Split3	AVG
I3D Head + RoIPool	77.57	73.91	75.64	75.71
I3D Head + RoIPool + Tail	80.53	81.44	80.77	80.91
I3D Head + ACAM + Tail	**84.68**	**83.78**	**83.30**	**83.92**

Table 5. Video mAP results on 3 splits of JHMDB and the average.

JHMDB - Models	Frame mAP	Video mAP
Action-RCNN[27]	58.5	73.1
ACT-Tubelet[17]	65.7	73.7
I3D-RoI[9]	73.3	78.6
ACRN[37]	77.9	80.1
ACAM	**78.9**	**83.92**

Table 6. mAP values averaged across 3 splits of JHMDB dataset.

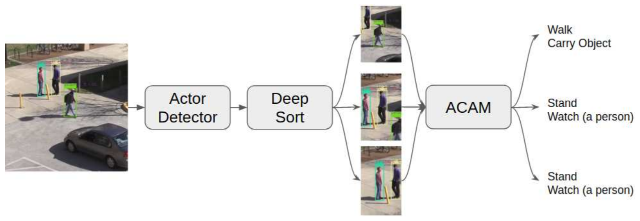

Figure 6. ACAM action detection framework running on a surveillance video from VIRAT [26]. Actors are detected by object detectors and tracked over frames by Deep Sort [43]. The generated tubes for each person is analyzed by the ACAM action detector.

4.5. Real-Time Framework for Action Detection

We evaluate the transferability and performance of the proposed model on different datasets qualitatively. We implement an end-to-end framework for detecting and tracking actors and analyzing their actions. The action model in this section is trained on AVA dataset and is not fine-tuned on the other datasets which demonstrates transferability.

We combine the person detector with the Deep Sort [43] tracker. Deep Sort is a simple tracking/re-identifying model that uses a deep association metric for matching detected person bounding boxes. This allows us to track the detections over time and generates person tubelets.

Since the proposed model explicitly models the surrounding context, a larger area than the person's tubelet is essential to model interactions. Due to the large view of surveillance videos, it is not feasible to process the entire scene. For this reason, square regions centered on the person's location and twice the size of the person's area are cropped and fed to the action detection framework.

The overall pipeline is shown in Fig. 6. First, we extract the actor tubes with a larger context area from the video using the detector and the tracker. Then, each detected tube is analyzed by the ACAM module for actions. Input frame and cropped tubes for each actor are visualized from the VIRAT [26] surveillance dataset. Notice that in interaction cases such as "watching a person" the model benefits from

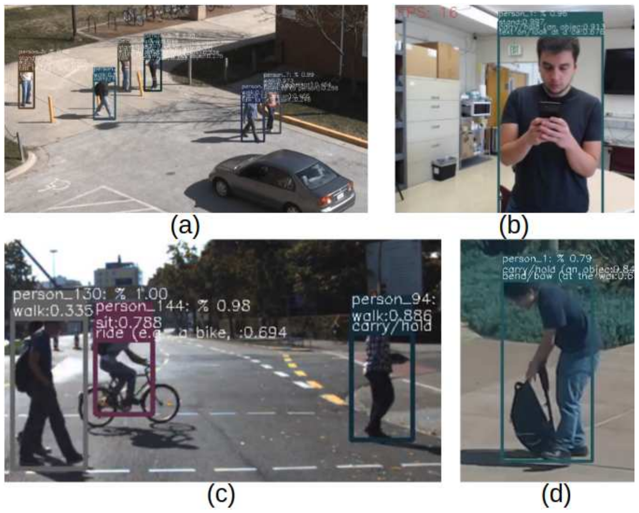

Figure 7. Qualitative results of ACAM video action detection framework visualized on different sources. a) VIRAT surveillance dataset [26], b) Webcam inputs at 16 fps, c) KITTI [5] autonomous driving dataset, d) Campus Surveillance videos.

having a person in the surrounding context.

We provide additional qualitative results on Fig. 7 for the autonomous driving dataset KITTI [5], surveillance dataset VIRAT, webcam videos and campus surveillance. These videos sources are unseen for the action model which was trained on AVA (movies) dataset. Accurate qualitative results demonstrate the transferability of our framework. Videos are available in the supplementary material.

A real-time version of this pipeline is open-sourced and available at Demo Github. It achieves 16 frames per second through a webcam on a single Nvidia GTX 1080Ti GPU using a fast SSD [24]-Mobilenet2 [31] object detector. This further demonstrates the advantage of modularity as the object detector can easily be changed for faster performance.

4.6. Ablation Analysis

Actor Detection Performance: To test the performance of the actor detector, we calculate the detection AP of every actor for every class on validation set. Table 7 shows the detection frame AP scores for the AVA v2.1 validation set.

Object Detector	AVA Actor Detection AP	Speed(ms/frame)
F RCNN-NAS	97.10	1833
F RCNN-Resnet101	95.97	106
SSD - MobileNetV2	66.16	31

Table 7. AP results for actor detection rate for different detectors and their detection speed. This demonstrates that detectors work well without fine tuning and shows the speed trade-off.

Transferable Actor Detection: The main reason of using a pre-trained frozen person detector instead of training an RPN on the action dataset is transferability. During training, object detectors see large variations in objects from large datasets such as MS-COCO [23] compared to action datasets. This makes object detectors more transferable to other datasets compared to retrained RPNs. To test this hypothesis, we compare the actor detection rates of same model architecture with and without fine-tuning. AVA model is fine-tuned on top of the COCO weights whereas COCO model is frozen and is not fine-tuned. Table 8 shows their comparisons on different datasets. Even though fine-tuned model achieve slightly better actor detection rate on the AVA dataset, the performance degradation is significant on datasets such as VIRAT [26] and KITTI [5].

Actor Detection	F RCNN AVA	F RCNN COCO	Δ
AVA	98.60	95.97	$+2.63$
VIRAT	9.94	30.44	-20.50
KITTI	27.04	54.57	-27.53

Table 8. AP results for actor detection rate of the same object detector trained on AVA and COCO and tested on different datasets. Actor detectors lose transferability to different domains when fine-tuned on action datasets (AVA in this case), which is shown by the difference (Δ: F RCNN AVA - F RCNN COCO).

Visualization of Attention Maps: In ACAM, an attention map for each feature channel is generated. This allows us to model different types of interactions efficiently. Since the feature maps are sparse, to visualize them, we average the attention map values across the feature dimension where they have non-zero values in their respective feature map. This generates a representation where each actor's relation with the scene is visible. Fig. 8 shows this visualization on different examples. Note that a higher attention value is obtained on objects and actor faces/hands.

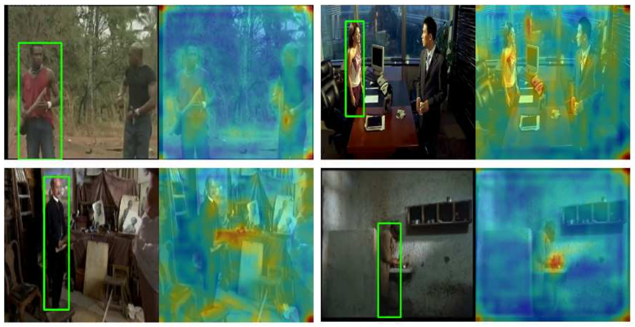

Figure 8. Generated Actor Conditioned Attention Maps. Higher attention values are usually observed around objects (paper, chairs, teapot, phones), on faces and hands of the actors.

Class Activation Maps: Using the global pooling layer at the last layer, we can generate class activation maps for each class (similar to [46]). We demonstrate activation maps for several different cases. Fig. 9 shows activation maps for different categories of actions. Activation maps are shown for actors annotated in green bounding boxes. Maximum

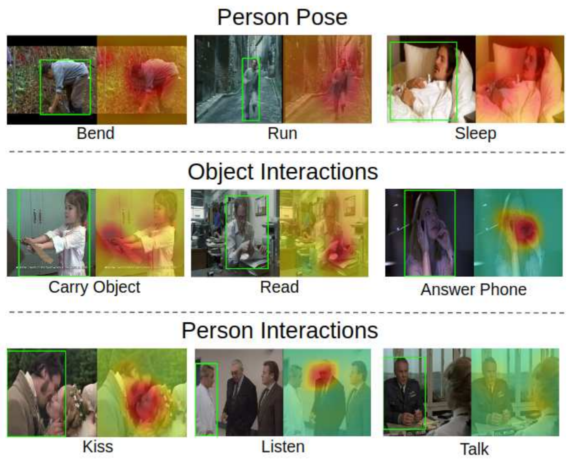

Figure 9. Class activation maps for detected actors in validation set. Each image represents the activation maps for the actor annotated by the green box and the given class. Red regions on the activation maps represent larger values.

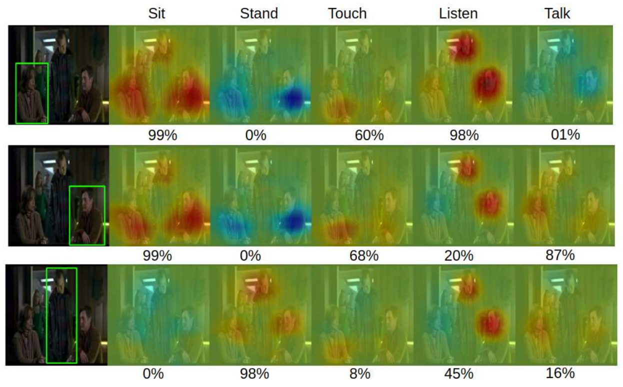

Figure 10. Class activation maps for detected actors. Each row represents the activation maps for the actor annotated by the green bounding box. Person on the right is talking in the video. Red regions on the activation maps represent the higher values.

activations across timesteps are visualized in the figures as these activations are also time sequences.

We observe that pose actions such as run/bend get activated around the actor while object interaction actions such as carry object/read are activated around the relevant objects and the actors. Person interactions also show some interesting results. The passive actions such as “watch a person”, “listen to a person” gets activated where there is another person in the scene that is “talking” or relevant.

Fig. 10 shows a scene with three people and their conditioned activation maps for specific actions. Each row represents the activation maps that are conditioned on the person in the green bounding box. We observe that complementary actions such as “talking” and “listening” gets activated on the person with the opposite action. This is due to our model architecture. As we initially extract the actor feature vector $\mathbf{r}_a$ from the actor’s location, this feature vector contains the information that the current actor a is “listening”. Therefore the attention map generated from actor’s vector $\mathbf{r}_a$ and context $\mathbf{E}$ looks for a person that is “talking” and focuses the attention on those locations.

5. Discussion

5.1. Comparisons of Attention Mechanisms

We compare ACAM with similar attention studies including: Actor-Centric Relation Network (ACRN) [37], Attentive Contexts for Object Detection (ACOD) [22] and Non-Local Neural Networks (NL) [42].

ACRN uses the Relation module (Eq. 4) without the sigmoid function to generate relation features for actors and context. The relation features are directly used for action classification with convolutional layers which are trained from scratch. In contrast, ACAM leverages the relation features to generate attention maps, which amplify/dampen the I3D feature map $\mathbf{I}$. The actor conditioned features $\mathbf{F}$ are scaled versions of $\mathbf{I}$ and the model is able to effectively leverage the pre-trained I3D-Tail on $\mathbf{F}$.

ACOD uses an LSTM branch to generate an attention map. It leverages context to generate a single global feature for every region proposal. This “attention branch” omits the individual differences of region proposals. ACAM generates attention maps by conditioning the context on each detected region (actor) individually to represent individual interactions which is better suited for action detection tasks.

NL, similar to the self-attention [41], uses matrix multiplication to find relations among pixel pairs in a spatio-temporal feature tensor. The relations are normalized through a softmax function to emulate an attention map. In an action detection setting where the actions are focused on actors, instead of finding relations among all pixel pairs, it is more effective to find relations between all context pixels and condition on individual actor features as in ACAM.

5.2. Summary

We presented a novel action detection model that explicitly captures the contextual information of actor surroundings. The proposed ACAM uses attention maps as a set of weights to highlight the spatio-temporal regions that are relevant to the actor, while damping irrelevant ones. This method is presented as a replacement to RoIPooling. ACAM is more suited for preserving interactions with surrounding context such as objects, other actors and scene. We demonstrated through thorough experimentation that ACAM improves the performance on multiple datasets and outperforms the state-of-the-art. We implemented and open-sourced a real-time atomic action detection pipeline to demonstrate the feasibility and modularity of ACAM.

References

[1] M. Abadi, A. Agarwal, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 5

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 4724–4733. IEEE, 2017. 2, 3, 4, 5

[3] Y. Chao, Y. Liu, X. Liu, H. Zeng, and J. Deng. Learning to detect human-object interactions. *arXiv preprint*. 2

[4] M. P. Eckstein, S. C. Mack, D. B. Liston, L. Bogush, R. Menzel, and R. J. Krauzlis. Rethinking human visual attention: Spatial cueing effects and optimality of decisions by honeybees, monkeys and humans. *Vision research*, 85:5–19, 2013. 1

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 7

[6] B. Ghanem, J. C. Niebles, C. Snoek, F. C. Heilbron, H. Alwassel, V. Escorcia, R. Khrisna, S. Buch, and C. D. Dao. The activitynet large-scale activity recognition challenge 2018 summary. *arXiv preprint arXiv:1808.03766*, 2018. 5

[7] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman. A better baseline for ava. *arXiv preprint arXiv:1807.10066*, 2018. 5

[8] G. Gkioxari, R. Girshick, P. Dollár, and K. He. Detecting and recognizing human-object interactions. *arXiv preprint arXiv:1704.07333*, 2017. 2

[9] C. Gu, C. Sun, D. A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2018. 1, 2, 3, 4, 5, 6

[10] S. Gupta and J. Malik. Visual semantic role labeling. *arXiv preprint arXiv:1505.04474*, 2015. 2

[11] J. M. Henderson, C. L. Larson, and D. C. Zhu. Full scenes produce more activation than close-up scenes and scene-diagnostic objects in parahippocampal and retrosplenial cortex: an fmri study. *Brain and cognition*, 66(1):40–49, 2008. 1

[12] R. Hou, C. Chen, and M. Shah. Tube convolutional neural network (t-cnn) for action detection in videos. In *IEEE international conference on computer vision*, 2017. 2

[13] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei. Relation networks for object detection. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2018. 2

[14] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, volume 4, 2017. 4

[15] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *International Conf. on Computer Vision (ICCV)*, pages 3192–3199, Dec. 2013. 1, 4, 6

[16] J. Jiang, Y. Cao, L. Song, S. Z. Y. Li, Z. Xu, Q. Wu, C. Gan, C. Zhang, and G. Yu. Human centric spatio-temporal action localization. 5

[17] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. Action tubelet detector for spatio-temporal action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4405–4413, 2017. 6

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

[19] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 1, 2

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[21] 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. 1, 2

[22] J. Li, Y. Wei, X. Liang, J. Dong, T. Xu, J. Feng, and S. Yan. Attentive contexts for object detection. *IEEE Transactions on Multimedia*, 19(5):944–954, 2017. 2, 8

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 4, 7

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 7

[25] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 5

[26] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. Aggarwal, H. Lee, L. Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *Computer vision and pattern recognition (CVPR), 2011 IEEE conference on*, pages 3153–3160. IEEE, 2011. 6, 7

[27] X. Peng and C. Schmid. Multi-region two-stream r-cnn for action detection. In *European Conference on Computer Vision*, pages 744–759. Springer, 2016. 1, 6

[28] T. J. Preston, F. Guo, K. Das, B. Giesbrecht, and M. P. Eckstein. Neural representations of contextual guidance in visual search of real-world scenes. *Journal of Neuroscience*, 33(18):7846–7855, 2013. 1

[29] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2, 3, 4

[30] S. Saha, G. Singh, M. Sapienza, P. H. Torr, and F. Cuzzolin. Deep learning for detecting multiple space-time action tubes in videos. *arXiv preprint arXiv:1608.01529*, 2016. 1

[31] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 7

[32] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976, 2017. 2, 3

[33] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016. 2

[34] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 2

[35] G. Singh, S. Saha, M. Sapienza, P. H. Torr, and F. Cuzzolin. Online real-time multiple spatiotemporal action localisation and prediction. In *ICCV*, pages 3657–3666, 2017. 1

[36] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 1, 2

[37] C. Sun, A. Shrivastava, C. Vondrick, K. Murphy, R. Sukthankar, and C. Schmid. Actor-centric relation network. *arXiv preprint arXiv:1807.10982*, 2018. 1, 2, 5, 6, 8

[38] A. Torralba, A. Oliva, M. S. Castelhano, and J. M. Henderson. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological review*, 113(4):766, 2006. 1

[39] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 2

[40] O. Ulutan, B. S. Riggan, N. M. Nasrabadi, and B. Manjunath. An order preserving bilinear model for person detection in multi-modal data. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1160–1169. IEEE, 2018. 3

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 1, 2, 8

[42] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 5, 8

[43] N. Wojke, A. Bewley, and D. Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649. IEEE, 2017. 6

[44] T. Yao and X. Li. Yh technologies at activitynet challenge 2018. *arXiv preprint arXiv:1807.00686*, 2018. 5

[45] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702, 2015. 2

[46] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016. 7

[47] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. 4