A*: Atrous Spatial Temporal Action Recognition for Real Time Applications

Myeongjun Kim Federica Spinola Philipp Benz Tae-hoon Kim
Deeping Source Inc., Seoul, Republic of Korea
{myeongjun.kim, federica.spinola, philipp.benz, pete.kim}@deepingsource.io

Abstract

Deep learning has become a popular tool across various fields and is increasingly being integrated into real-world applications such as autonomous driving cars and surveillance cameras. One area of active research is recognizing human actions, including identifying unsafe or abnormal behaviors. Temporal information is crucial for action recognition tasks. Global context, as well as the target person, are also important for judging human behaviors. However, larger networks that can capture all of these features face difficulties operating in real-time. To address these issues, we propose A: Atrous Spatial Temporal Action Recognition for Real Time Applications. A* includes four modules aimed at improving action detection networks. First, we introduce a Low-Level Feature Aggregation module. Second, we propose the Atrous Spatio-Temporal Pyramid Pooling module. Third, we suggest to fuse all extracted image and video features in an Image-Video Feature Fusion module. Finally, we integrate the Proxy Anchor Loss for action features into the loss function. We evaluate A* on three common action detection benchmarks, and achieve state-of-the-art performance on JHMDB and UCF101-24, while staying competitive on AVA. Furthermore, we demonstrate that A* can achieve real-time inference speeds of 33 FPS, making it suitable for real-world applications.*

1. Introduction

Deep learning has undergone a significant transformation in recent years with the shift towards handling video data that more closely simulates real-world scenarios, in contrast to its previous focus on image-based problems [6, 7]. With the proliferation of surveillance cameras, there is an increasing need for analyzing human behavior. For example, detecting unsafe behavior such as falling and fainting can be life changing in elderly homes. Recognizing theft and other illegal acts can improve safety and reduce losses if detected on time. Additionally, the increasing interest in autonomous vehicles has raised the need for recognizing human behaviors to help predict future actions of pedestrians and drivers. These applications require real-time and instantaneous action recognition. Therefore, there is a critical need for network architectures that can accurately and promptly detect real-world actions.

The early works on action-related tasks proposed a two-stream approach [19, 41] that combines human joint prediction and optical flow estimation. Although this approach is computationally advantageous, it cannot capture long-range motion interactions, leading to suboptimal results. By considering how humans recognize actions, we observe that both spatial and temporal, local and global context plays a significant role in distinguishing actions. To address these limitations, 3D convolutions [21, 49, 51] are designed to capture not only spatial information but also temporal changes in motion. Recent networks [13, 50] have proposed to leverage 3D convolutions and have demonstrated significant performance improvements in detecting actions. However, the use of convolutions restricts the networks to consider information at a single scale, and multi-scale spatial and temporal cues are essential to better describe actions. Moreover, these networks are usually computationally more expensive and cannot be used for real-time applications. Current works [25] focus on further improving action detection results while aiming for real-time inference. However, faster networks still tend to confuse similar actions such as *sitting* and *standing*. To address these limitations, we propose A*: Atrous Spatial Temporal Action Recognition for Real Time Applications. A* focuses on improving action recognition in real-world scenarios, where predictions need to be made in real-time and without knowledge of future frames. To achieve this, A*'s network is designed to include space and time information at different scales.

A* is an end-to-end network that simultaneously localizes humans and predicts their actions. For the person detection task, we use the fast YOLO-based [2, 38] framework, and for the action recognition task, we use a common 3D convolution-based network. Within this baseline architecture, A* introduces four aspects aimed at boosting performance. First, it considers the surrounding global context by aggregating low-level features in a low-level feature aggregation (LLFA) module. Second, it proposes Atrous Spatio-Temporal Pyramid Pooling (ASTPP) to consider multi-range temporal information. Third, it suggests

to extract frame-specific features and combine all features in an Image-Video Feature Fusion (IVFF) module. Finally, it uses Proxy Anchor Loss (PAL) [23] for action features to distinguish between similar behaviors. A* operates in real-time at 33 Frames Per Second (FPS) on a RTX 3090 GPU, achieving state-of-the-art (SOTA) performance in the real-time setting on the JHMDB [20], UCF101-24 [43] and Atomic Visual Actions (AVA) v2.1 [15] datasets. In summary, our contributions include the following:

- We propose four additions that are necessary to obtain richer features for the action detection task: LLFA, ASTPP, IVFF and action-PAL.
- We propose a single-stage network architecture based on YOLO and demonstrate a processing speed of 33 FPS, making it suitable for real-time applications.
- We achieve outstanding performance on the J-HMDB-21, UCF101-24 and AVA v2.1 datasets against other real-time action recognition methods.

2. Related works

Video Understanding The main purpose of backbone networks is to extract domain-specific features. The correct choice of backbone network, whether it is ResNet [18], EfficientNet [47], ViT [8], or Swin Transformer [32], plays an important role in enhancing image-based downstream tasks such as object detection and semantic segmentation. For video understanding tasks, the choice of backbone is equally as important. Various methods propose to expand the above-mentioned networks to 3D, to simultaneously account for the spatial (2D) and temporal (1D) dimensions. The Inflated-3D convolution (I3D) network [3] is constructed by extending the 2D pre-trained weights of the Inception network [46] to 3D. A 3D version of the widely used ResNet model can easily be constructed by swapping the 2D convolution for 3D convolution. The R(2+1)D [52] network, is designed to consider the computational cost of 3D convolution. It speeds up the original 3D convolution computations by processing convolutions for the spatial axis first, and then for the temporal axis. The SlowFast [13] network tries to learn the task of optical flow by considering semantic information in a slow path (low frame rate) and by capturing frame-to-frame motion in a fast path (high frame rate). The X3D [12] authors propose a family of networks that expand a 2D classifier in various dimensions based on video data factors (video data length, sampling rate, etc.) and model-related factors (channel, number of layers, etc.). Given its success in image understanding tasks, several approaches using the Transformer architecture for video understanding [1, 10, 33] and video action classification [27, 59] are being introduced.

Action Detection Networks Previous action detection works can be classified into multi-stage or single-stage approaches. Multi-stage methods first detect bounding boxes, then perform the action recognition task. Single-stage action detection methods classify actions and regress bounding boxes simultaneously in an end-to-end manner. The multi-stage methods can be further subdivided into two categories.

For the first set of networks (detector-based networks) [10, 11, 13, 48], a Faster-RCNN [39] network trained on the COCO [30] dataset is typically used to predict bounding boxes. Then, features are extracted for each target person using 3D RoIAlign [17]. Finally, action classification is performed on each feature. The second set of multi-stage networks (two-stage action detection networks) [12, 14, 35, 56] first extracts rough bounding boxes through a Region Proposal Network (RPN). The subsequent process is similar to the detector-based methods described above. However, bounding box regression and action classification are learned together. Detector-based methods show higher performance because bounding box regression and action classification are performed separately, making it easier compared to learning both simultaneously. However, because the detector and classification network are separated, there is latency during inference time. Moreover, global context information can be lost due to the use of the 3D RoIAlign features. This is because the RoIAligned features are extracted from the predicted bounding box areas. Thus, the features consider only the area within the target bounding box, and not the global context.

The single-stage methods [5, 25, 31, 57, 60] are based on the YOLO framework [2, 38], which is effective for both training and inference. Unlike the 2D YOLO network, these approaches use video data. They perform action detection on a target frame (keyframe) by constructing a light 2D convolution path that extracts the keyframe's features, and a 3D convolution path that captures the video's entire context. Compared to the multi-stage methods, the YOLO-based approach can operate at speeds that are considered real-time.

Our work aims to operate in real-world applications, where not only accuracy but also speed are of utmost importance. Therefore, we adopt the single-stage YOLO-based action detection network, YOWO [25], as our baseline. We address three shortcomings of the YOWO network. Firstly, we improve the low-level feature aggregation. Secondly, we propose the ASTPP module to obtain multi-range temporal information and spatial long-range information. Thirdly, we modify the loss function and include the PAL [23] for action features to distinguish between similar actions effectively. As a result, our network achieves SOTA performance while operating in real time.

3. Methods

This Section dives deeper into our proposed approach to action detection, A*, illustrated in Figure 1. Our method tackles four main aspects that should be considered when detecting actions. Firstly, understanding the surrounding

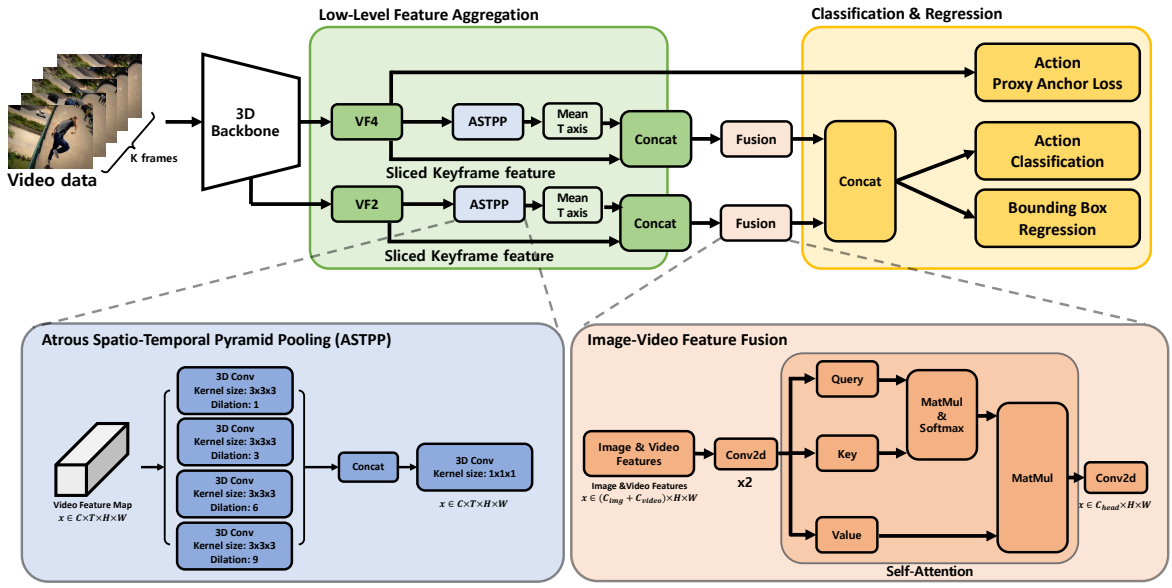

Figure 1. The Overall Network Architecture of A*. A video clip is fed into a 3D backbone to extract low-level and high-level video features. The features are passed through the ASTPP module to gather multi-range spatial and temporal contexts. All features are then fused and used to regress bounding-box coordinates and to classify actions. Best viewed in color.

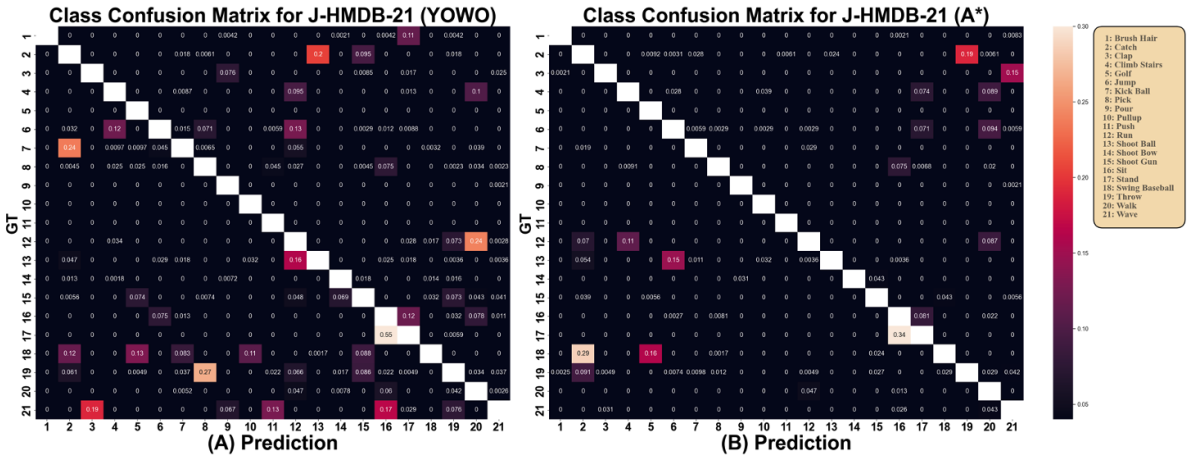

Figure 2. Confusion Matrix Between Predicted and Ground-Truth Actions for the JHMDB-21 Dataset. Predictions from YOWO are shown in matrix A, and predictions from A* are shown in matrix B. The confusion score between wrong predictions and ground-truth actions is illustrated: the higher the score, the higher the confusion. Confusion scores of correct predictions are ignored (white diagonal in the figures) for visualization purposes. Best viewed in color.

global context is an important factor in understanding human behavior. For example, the *surfing* action can more easily be detected if one recognizes that the surrounding environment is the sea. Therefore, we propose to use high-level along with low-level features to capture the global surrounding context of the image. This is done through our LLFA module detailed in Subsection 3.1 and shown in green in Figure 1. Secondly, unlike image tasks, summarizing temporal information is crucial for video-understanding problems. Therefore, we propose an ASTPP module that can consider multi-range temporal information explained in Subsection 3.2 and illustrated in blue in Figure 1. Thirdly, when detecting actions at a specific time-instance, temporal and spatial noise from other time-instances might hinder the prediction. Therefore, we propose to extract keyframe-specific features in the LLFA and fuse all our features in an IVFF module, described in Subsection 3.3 and shown in orange in Figure 1. Finally, human behaviors can have very

similar features that models have difficulty differentiating. For example, *sitting* and *standing* have the same behavior if the temporal axis is reversed. Therefore, we propose to use the action-PAL to better discriminate between action features (shown as a yellow sub-box in Figure 1).

Our method is designed as a single-stage network for real-time inference. It is based on the YOLO framework [2, 38], using the YOLO training methods, but applied to video tasks. Our model takes K video frames of size $H \times W \times C$ as input x (x has size $(C \times K \times H \times W)$[1]). The input frames are fed into a 3D video backbone network to extract video features (VF). VFs at different levels are fed into the LLFA module. Within the LLFA, VFs pass through our ASTPP module. Then, the outputs from the LLFA module go through the IVFF module, outputting features that now include all the necessary spatio-temporal signals for action detection. Finally, the fused features are concatenated and used for the final network predictions: action classification and bounding box regression. During the training phase, additional features from the LLFA module are used to compute the action PAL.

3.1. Low-level Feature Aggregation

To correctly detect actions, important aspects to consider are the surrounding global context, the human-object interactions, and the human-human interactions. However, as the convolution layers get deeper, only the high-level features remain. This results in losing information about the surroundings (global context). Among former methods, HRNet [53] combines low-level and high-level features to preserve the semantic information of the spatial axis. With a similar motivation, we adopt a feature aggregation approach illustrated in green in Figure 1. The inputs to our low-level feature aggregation module are VFs extracted from different stages our 3D video backbone network. We extract both the usual high-level features from our backbone's fourth (last) stage and low-level features from our backbone's second stage. Empirically, this feature combination yielded the best results (refer to ablation Table 5). The low-level features, $VF2$, have dimensions: $(C_{VF2} \times T \times H \times W)$. The high-level features, $VF4$, have dimensions: $(C_{VF4} \times T \times H \times W)$. T refers to the dimension of the temporal axis, which is 8 in our case. Then, $VF2$ and $VF4$ go through the ASTPP module explained in Subsection 3.2 to obtain spatially and temporally rich low-level and high-level features for the entire video sequence. Concurrently, features from $VF2$ and $VF4$ corresponding to the target frame are extracted, by drawing out features only at the temporal dimension corresponding to this keyframe. Assuming temporal consistency along the temporal axis, this is done by slicing the tensor at the center of the temporal axis[2]. Therefore if $T = 8$, we extract the feature located at $t = 4, t \in [1, T]$. This allows to focus the action detection predictions on the desired keyframe, and further motivation behind this is detailed in Subsection 3.3. These image-specific features, $ImageVFi, i = 2, 4$ with size $(C^{VFi}_{img} \times H \times W, i = 2, 4)$, are concatenated with the outputs from the ASTPP module, $VideoVFi, i = 2, 4$. To do so, these ASTPP outputs are first averaged about the temporal axis to match the size of the image-specific features. Then, the concatenation is done along the channel dimension and the final output has size $((C^{VFi}_{img} + C^{VFi}_{video}) \times H \times W)$ where $C^{VFi}_{img} = C^{VFi}_{video} = C_{VFi}, i = 2, 4$. Finally, the LLFA module outputs descriptive high-level features whilst also conserving low-level features, both of which are crucial for accurate action classification and bounding-box regression.

3.2. Atrous Spatio-Temporal Pyramid Pooling

Considering multi-range temporal features and long-range spatial information is very important in action recognition/detection tasks. 2D and 3D convolution operations inherently capture some local spatial and temporal cues. A known way to further increase the network's receptive field is to use dilated (atrous) convolutions. In [4], the authors introduce the Atrous Spatial Pyramid Pooling (ASPP) scheme to capture objects and contextual information at multiple spatial scales. The ASPP is computed by employing multiple parallel convolution filters with different dilation rates. In our work, we extend the ASPP to additionally capture temporal information and propose the ASTPP module, shown in blue in Figure 1.

For a simple 1D signal $x(a)$, the 1D dilated convolution $D(a)$ with a filter ω can be written as follows [4, 62]:

$$D(a) = \sum_{m=1}^{M} x[a + r * m] * \omega_m. \tag{1}$$

Where M represents the kernel length, and r indicates the input's sampling stride (i.e. the dilation size of the convolutions). Note that a rate $r = 1$ results in the standard 1D convolution formulation. To capture both the spatial and temporal information of video data, we extend the 1D dilated convolution from Equation 1 to a 3D dilated convolution formulation. For a 3D input $x(a, b, c)$, we express the 3D dilated convolution $D(a, b, c)$ as follows:

$$D(a, b, c) = \sum_{m,n,o}^{M,N,O} x[a + r_a * m, b + r_b * n, c + r_c * o] * \omega_{m,n,o} \tag{2}$$

Where the kernel ω is now of size (M, N, O), and the rates r_a, r_b, and r_c represent the dilation strides along all three

[1] We purposely omit the batch-size dimension throughout the paper to ease readability.

[2] This is the case when past and future frames are used during training. If only past frames are used, as in the real-time setting, the tensor will be sliced at the last index of the temporal axis.

dimensions respectively. In practice, we use cuboid kernels such that $M = N = O$. Moreover, for a given pyramid level j in our ASTPP, we utilize equal dilation rates such that $r_a^j = r_b^j = r_c^j = r^j$. The 3D dilated convolution from Equation 2 is the building block of our ASTPP module. In our ASTPP, the convolution operations are performed on the $VF_i, i = 2, 4$ with four different dilation rates ($r^1 = 1, r^2 = 3, r^3 = 6, r^4 = 9$), and the channels after each convolution decrease from $C_{in} = C_{VFi}, i = 2, 4$ to $C_{out} = C_{in}/4$. In contrast, the kernel size remains the same for all convolutions. Finally, all four 3D convolution outputs are concatenated along the channel dimension and fed into a last $1 \times 1 \times 1$ convolution layer to better aggregate the atrous convolution feature maps. The output of our ASTPP ($VideoVFi, i = 2, 4$) is of size ($C_{video}^{VFi} \times T \times H \times W$) where $C_{video}^{VFi} = C_{VFi}, i = 2, 4$, which is the same as the input size to the module. Our ASTPP differs from the ASPP [4] in two ways. Firstly, our ASTPP employs 3D instead of 2D convolutions, to simultaneously capture the surrounding context and facilitate long-range interactions. Secondly, there is no Global Average Pooling (GAP) path. Empirically, the addition of the GAP results in a decrease in performance (refer to ablation Table 6 (column 3)). This is because the information for classification and regression is lost in the upsampling process occurring after the GAP-Convolution stage. With our ASTPP, we can extract richer features compared to the conventional convolution layer by considering spatial and temporal features with long-range and multi-range interactions.

3.3. Image-Video Feature Fusion

The input video data for action detection tasks often includes unintended frames such as scene transitions. This induces noise when classifying the keyframe's action due to information unrelated to this keyframe. Moreover, the position of people changes in successive frames of the input video, introducing a large amount of noise in the bounding box regression task. As action recognition is performed on the keyframe, this extra noise causes a decrease in accuracy. Some works [25, 60] suggest to feed the keyframe into a 2D backbone and the video sequence into a 3D backbone, later aggregating the features maps. Instead, in our work we propose to extract the keyframe information directly from the VFs by the slicing process explained in Subsection 3.1. Adding this keyframe-specific feature helps to focus the action classification and bounding-box regression on the target frame, and is empirically confirmed by our ablation results in Table 6 (column 4). Feature fusion is not a new concept. Our novelty comes from the features used as inputs to our IVFF module: the $ImageVFi, i = 2, 4$ and $VideoVFi, i = 2, 4$ from the LLFA module. The fusion of these features is the scope of our IVFF module described in orange in Figure 1.

The input to the IVFF is a set of 2D feature maps (temporal axis is averaged in the LLFA module), thus 2D convolutions are now used instead. Our IVFF is mainly composed of a typical self-attention block [54], which aims to embed information about the frame-to-predict into the full video sequence. Moreover, the self-attention block helps emphasize and fuse the global context from the video features and the local information from the image features, crucial for action recognition. Our IVFF outputs C_{head} feature maps of size ($H \times W$), where C_{head} is smaller than the input channel size to reduce computational cost.

3.4. Loss function

In real-world situations, various types of human behaviors can look the same action when viewed in reverse time order and models often confuse them. Figure 2 A shows the correlation matrix between ground-truth (GT) actions (vertical axis) and actions predicted by the baseline network YOWO [25] (horizontal axis). We notice that several action pairs having a temporal ambiguity are erroneously predicted by YOWO. For example, *standing* (17) is predicted as *sitting* (16), and *running* (12) is confused with *walking* (20). To solve this problem, we adapt the PAL [23] from metric-learning and use it for action detection as an additional loss to better discriminate between temporally ambiguous actions. This allows the model to learn different features even for similar actions. The formulation for the action-PAL is as follows:

$$L_{pal}(X) = \frac{1}{|P^+|} \sum_{p \in P^+} \left(1 + \sum_{x \in X_p^+} e^{-\alpha(s(x,p)-\delta} \right) + \frac{1}{|P|} \sum_{p \in P} \left(1 + \sum_{x \in X_p^-} e^{\alpha(s(x,p)+\delta} \right) \tag{3}$$

Where δ and α are positive numbers representing a margin and a scaling factor respectively. $s(.,.)$ refers to the cosine similarity. P is the set of all proxies and P^+ is the subset of positive proxies in a given batch. For each proxy $p \in P$, we partition the batch's action embedding vectors X into two subsets: X_p^+ contains the positive action embeddings associated with p, while X_p^- consists of the remaining vectors in X (i.e., $X_p^- = X - X_p^+$).

Other than the action-PAL, our approach builds upon YOLO's [2, 38] typical loss functions to train our network. Specifically, for the confidence score (L_{obj}) we utilize the Binary Cross Entropy loss (BCE), for classification (L_{cls}) we employ the Focal loss [29], and for bounding box regression (L_{box}) we adopt the GIoU [40] loss. In the case of the AVA dataset [15], requiring multi-label classification, we replace the Focal loss with a BCE Loss for each class. Each loss term is weighted by a corresponding coefficient λ and the total loss is calculated as follows:

$$L = \lambda_{obj} L_{obj} + \lambda_{cls} L_{cls} + \lambda_{box} L_{box} + \lambda_{pal} L_{pal} \tag{4}$$

4. Experiments

To verify the effectiveness of the proposed A*, we perform experiments on four benchmarks, detailed in Subsection 4.1. Then we explain our experimental setup in Subsection 4.2. Finally, we report our results and ablations in Subsections 4.3 and 4.4.

4.1. Datasets

J-HMDB-21 is a subset of the HMDB-51 [26] dataset, specific for action recognition. It consists of 21 action classes, each having up to 55 video clips, for a total of $31,838$ annotated frames (320×240). Each video clip is trimmed to capture a single action. To ensure comparability with other methods, we present the frame mean average precision (mAP) results for split-1 of the dataset. The frame mAP is evaluated at an Intersection over Union (IoU) threshold of 0.5, which is consistent with the evaluation protocol used by other methods.

UCF101-24 is a subset of the UCF101 [43] dataset, specifically designed for spatio-temporal action detection. It comprises 24 action classes, mainly related to sports activities, and consists of $3,207$ untrimmed videos, each annotated with human bounding boxes at the frame level. We adopt the same testing protocol as for the J-HMDB-21 dataset.

AVA [15] is a realistic and multi-class dataset consisting of 80 action classes and a total of 430 videos. The videos are split into 235 for training, 64 for validation, and 131 for testing. The 1.62M annotations are assigned at 1 FPS and for each annotation. As in previous work [15], we evaluate our performance on 60 classes and use at least 25 elements for validation. The AVA benchmark uses frame-level Average Precision (Frame-AP) for evaluation, with an IoU threshold of 0.5. We benchmark on AVA v2.1 and v2.2.

4.2. Training and Testing

During the training stage, we create small clips of 32 frames in two ways: only past and current frames for the real-time setting (RTS) and frames on either side of the keyframe for the offline setting (OS). Each frame undergoes various augmentations (random horizontal flip, random crop, distortion) and is resized to 224^2 for the RTS and to 256^2 for the OS. We use the AdamW [34] optimizer and a multi-step learning rate decay scheduler. SlowFast-R50 [13] is employed as the backbone network for JHMDB and UCF101-24, while SlowFast-R101 [13] is used for the AVA experiments. We start from SlowFast checkpoints [9] pretrained on the Kinetics dataset [22]. For the JHMDB and UCF101-24 datasets, we train our network for 5 epochs with a learning rate (lr) warm-up, an initial lr of $1e-4$ and a 0.5 decay at every epoch. For AVA v2.1 and v2.2, we perform a 0.5 decay the 3rd, 4th, 5th, and 6th epoch, using an initial lr of $1e-4$. During the testing stage, the video clips are created as for training, depending on the RTS or OS. No additional augmentation techniques are employed.

Table 1. Comparison of our Model's Performance Against Previous Methods on the JHMDB and the UCF101-24 datasets. Underlined values are the best of each category of methods. **Bold** values are best overall results. f: number of frames. ♣: models use additional flow information as input. †: our re-implementation

Model	Input	Backbone	JHMDB	UCF101-24
Multi-Stage Offline Methods				
TacNet [42]	16f	-	65.5	72.1
MOC [28]	1f	-	70.8	78.0
PCSC♣ [44]	-	-	74.8	79.2
HISAN♣ [37]	15f	-	76.7	73.7
ACRN [45]	20f	-	77.9	-
HIT [11]	32f	SFR50	83.8	84.8
End-to-End Offline Methods				
AVA♣ [15]	10f	-	73.3	78.8
WOO [5]	8f×8	-	80.5	-
TubeR [64]	-	CSN-152	82.3	83.2
STMixer [57]	32f×2	SFR101-NL	86.7	83.7
A* (Ours OS)	32f	SFR50	**90.4**	**89.2**
End-to-End Real-Time Methods				
YOWO [25]†	16f	X3D M	67.9	-
YOWO [25]	16f	RNext101	74.4	80.4
YOWO [25]†	32f	SFR50	78.2	
A* (Ours RTS)	16f	X3D M	71.4	83.21
A* (Ours RTS)	16f	ResNext101	75.9	83.82
A* (Ours RTS)	32f	SFR50	81.5	85.62

4.3. Comparison to Previous Works

Our quantitative results are shown in Table 1 for the J-HMDB-21 and UCF101-24 datasets, and in Table 2 for the AVA dataset. In our analysis, we categorize methods into *real-time methods* that care about inference speed and use only past frames for prediction, and *offline methods* (*multi-stage* and *end-to-end*) that use large backbones and use future frames for prediction.

Real-time methods: Our primary focus is to perform action recognition in real-world setting, so we compare A* (ours with RTS) to YOWO [25], the main method tackling the RTS. For JHMDB experiments, we train A* and YOWO with multiple small and large backbones. A* consistently outperforms YOWO by large margins: from **+1.5 frame-mAP** to **+3.5 frame-mAP** depending on the backbone. Meanwhile, both A* and YOWO with SFR50 backbone achieve real-time inference speeds on an RTX 3090 GPU, respectively 33 and 57 FPS. A more comprehensive speed analysis in provided in the Appendix. Similarly, on UFC101-24, our A* surpasses YOWO by **3.4 frame-mAP** with the ResNext101 backbone. On AVA v2.1, a similar trend is observed, with A* surpassing YOWO by **3.3 frame-AP**. A*'s inference speed decreases to 20 FPS (on RTX 3090) due the larger backbone used for AVA, but remains in the real-time realm. Our results confirm A* as the method-of-choice for real-world applications where instantaneous predictions are needed, such as in autonomous driving.

Offline methods: Most previous works perform action recognition in an OS, as intuitively using future frames

Table 2. Comparison of our Model's Performance Against Previous Methods on the AVA v2.1 and v2.2 datasets. Underlined values are the best of each category of methods. **Bold** values are best overall results. Abbreviations are the following. f: nb. frames, R-I3D: ResNet-I3D, RNext101: ResNext-101 [16], I3D: Inflated 3D Convolutions [3], S3D(+G): Separable 3D Convolutions (with gating) [58], Tx: Transformer [36], GCN: Graph Convolutional Network [24], SFR: SlowFast [13], NL: Non-Local [54], K: Kinetics [22]. ♠: methods randomly scale the short side of images within a range during training, and the reported number is the minimum value of the range.

Model	Input	Input Size	Backbone	Frame-AP (0.5)	
				v2.1	v2.2
Multi-Stage Offline Methods					
ACRN [45]	RGB (-), Flow (-)	-	S3D-G	17.4	-
SMAD [63]	RGB (36f)	256^2	I3D, GCN	22.2	-
MeMViT [55]	RGB (16f × 4)	224♠	MViTv2-16 (K400)	-	28.5
MeMViT [55]	RGB (32f × 3)	224♠	MViTv2-24 (K600)	-	31.5
HIT [11]	RGB (32f)	256♠	SFR101 (K700)	-	**32.6**
End-to-End Offline Methods					
AVA Baseline [15]	RGB (40f), Flow (40)	320 × 400	I3D	15.6	-
STEP [61]	RGB (12f)	400^2	I3D	18.6	-
I3D Baseline [3]	RGB (64f)	256^2	I3D	20.78	-
R-I3D Baseline	RGB (32f)	256^2	R-I3D	21.2	-
VTr [14]	RGB (64f)	400^2	I3D, Tx	24.9	-
WOO [5]	RGB (8f × 8)	320♠	SFR50 (K400)	25.2	25.4
WOO [5]	RGB (8f × 8)	320♠	SFR101 (K600)	28.0	28.3
STMixer [57]	RGB (32f × 2)	256♠	SFR50 (K400)	27.2	27.8
STMixer [57]	RGB (32f × 2)	256♠	SFR101-NL (K600)	29.8	30.1
TubeR [64]	RGB (32f × 2)	288♠	SFR101 (K400+K700)	**31.6**	-
A* (Ours OS)	RGB (32f × 2)	256^2	SFR101-NL (K600)	27.7	27.7
End-to-End Real-Time Methods					
YOWO [25]	RGB (32f)	224^2	RNext101 (K400)	18.3	19.1
A* (Ours RTS)	RGB (32f)	224^2	RNext101 (K400)	21.64	-
A* (Ours RTS)	RGB (32f)	224^2	SFR101 (K400)	23.5	-

will boost performance. Therefore, we also compare A* (ours with OS) to these previous methods. On JHMDB and UCF101-24, A* outperforms all past multi-stage and end-to-end methods by large margins. Specifically, it beats the SOTA method STMixer [57] by **3.7 frame-mAP** on JHMDB and the SOTA method HIT [11] by **+4.4 frame-mAP** on UCF101-24. On AVA, out of the previous methods, the only comparable work that outperforms A* is STMixer [57]. TubeR [64] and HIT [11] use more powerful pretrained checkpoints, whilst WOO [5] and MeMViT [55] use very different video-clip sampling rates, image resizing strategies and backbone (for MeMViT). We speculate that this performance gap is partly due to the difference in image resolution used by STMixer and A*. STMixer keeps the image's aspect ratio, but A* uses square images to easily utilize anchors in the bounding-box regression task. Therefore, the resolution of images in STMixer is always higher than for A*. Image resolution does not seem to greatly affect results on JHMDB and UCF101-24 that are single person, single action datasets. However, AVA's multi-label, multi-person task requires to capture more fine-grained details that can be lost by using square images. Therefore, for AVA, a close attention to image resolution is needed.

Qualitative results: Qualitatively, A* competes with the SOTA as illustrated in Figure 3. It accurately predicts bounding-box locations and the multiple action classes in different crowded and low-lighting settings. Further qualitative results are reported in the Appendix.

Overall, our SOTA and competitive results on multiple datasets exhibit the generalization capabilities of A* in multiple action recognition scenarios, and under RTS and OS.

4.4. Ablation Studies

Confusion Matrix Analysis Figure 2 illustrates the confusion matrix between GT action labels and model predictions on JHMDB. Sub-figure 2 A shows the confusion results achieved with YOWO and Sub-figure 2 B shows the results from our A* method. Confusion matrix 2 A illustrates that the temporally ambiguous actions such as *sitting* and *standing* are highly confused (confusion value of 0.6). In contrast, A* is capable of better distinguishing between similar actions, reducing the confusion value between *sitting* and *standing* to 0.3. The closer the elements are to 0 (excluding diagonal elements), the higher the prediction accuracy. We can then calculate a confusion score by summing all the non-diagonal values. Consequently, YOWO yields a confu-

Figure 3. Qualitative results on the AVA v2.2 dataset comparing A* (our OS) in column A, and STMixer in column B. Green action predictions are correct predictions and red action predictions are wrong predictions. Dashed-lines represent predicted bounding boxes and full-lines are GT bounding boxes. Best viewed in color.

sion score of 6.0, whereas A* achieves a considerably lower confusion score of 3.0. These results demonstrate the benefits of integrating the action-PAL in our method to better distinguish between similar actions.

Module Contribution The aggregated impact of our modules is measured and reported in Table 3. All module additions show significant performance improvement. Each of our four contributions leads to an increase in frame-mAP of at least 2% points on JHMDB.

ASTPP Design We investigate the use of atrous convolutions along various axes for our ASTPP design. Ablation results in Table 4 demonstrate that considering multi-range interactions along both spatial and temporal axes yields the best performance. Moreover, we experiment whether using the GAP from ASPP in our ASTPP is beneficial. Results from Table 6 (column 3) suggest that omitting the GAP from our ASTPP yields better performance. We suppose that this is because information for classification and regression is lost in the upsampling process needed after the GAP. Finally, we investigate tuning the atrous convolution dilation size. Our method uses dilation sizes of [1, 3, 6, 9]. Through the ablation results in Table 6 (column 2), we show that with further hyperparameter tuning, we can achieve even better action recognition results (+0.7 frame-mAP). In fact, more suitable dilation sizes are [1, 3, 5, 7]. The reason for this is that our temporal axis has dimension 8, so when using dilations of [1, 3, 6, 9], we ineffectively consider the temporal interactions due to excessive zero padding on the temporal dimension at dilation sizes 6 and 9.

Low-level Feature Aggregation We experiment which combination of features is best to aggregate in our LLFA module. We extract features from different levels of our backbone and report results in Table 5. Notably, using VF2 and VF4 features together produces optimal results. The remaining experiments show that using too many features can introduce noise, whilst using features that are too low-level can result in information loss.

Table 3. Comparison of module effects on J-HMDB-21. RNext101: ResNext-101 [16], SFR: SlowFast [13]

Method	Frame mAP (0.5)
baseline	74.4
+ LLFA (with IVFF)	76.38
+ Convert backbone (RNext101 to SFR50)	79.1
+ Future frames (offline setting)	85.67
+ ASTPP	87.85
+ Action-PAL	**90.44**

Table 4. Experiments on the use of different convolutional kernels for the ASTPP module on JHMDB.

Method	mAP (0.5)
Spatial Kernel	87.75
Temporal Kernel	87.32
Both	**90.44**

Table 5. Experiments on the choice of features for the LLFA module on JHMDB.

Block Number	mAP (0.5)
VF4	87.21
VF1 + VF4	86.8
VF3 + VF4	88.75
VF2 + VF4	**90.44**
VF2 + VF3 + VF4	87.25

Table 6. Experiments on the influence of ASTPP dilation size and use of image features on J-HMDB-21.

kernel size	dilation size	GAP	Image Feature	mAP (0.5)
3, 3, 3, 3	1, 3, 6, 9	✗	✓	90.44
3, 3, 3, 3	1, 3, 5, 7	✗	✓	**91.18**
3, 3, 3, 3	1, 3, 6, 9	✓	✓	87.69
3, 3, 3, 3	1, 3, 6, 9	✗	✗	88.39

Effect of Video Feature Slicing We investigate the benefits of extracting frame-specific features directly from VFs and thus the necessity for the IVFF module. Our ablation results in Table 6 (column 4) show that using image features and fusing them with the VFs greatly benefits our model's performance. This further supports our assumption about the temporal consistency of the temporal axis in the VFs.

5. Conclusion

In this paper, we proposed A*, a single-stage action detection network based on the YOLO framework that can detect actions in real-time, making it suitable for real-world applications. Our experiments show that A* achieves SOTA performance on the JHMDB, UCF101-24, and AVA v2.1 datasets for the RTS.

Moreover, our OS results demonstrate the importance of considering future frames when analyzing the current frame. In human cognition, we often predict or infer future frames based on the current one. Our future work will focus on developing networks that can "remember" past experiences and "infer" future frames from a given frame to improve action detection performance in our RTS.

In conclusion, A* represents a significant step forward in real-time action detection, and we believe that our proposed approach could lead to further advances in video understanding research.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 2

[2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 1, 2, 4, 5

[3] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 2, 7

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 4, 5

[5] Shoufa Chen, Peize Sun, Enze Xie, Chongjian Ge, Jiannan Wu, Lan Ma, Jiajun Shen, and Ping Luo. Watch only once: An end-to-end video action detection framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8178–8187, 2021. 2, 6, 7

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 1

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[9] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. Pyslowfast. https://github.com/facebookresearch/slowfast, 2020. 6

[10] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6824–6835, 2021. 2

[11] Gueter Josmy Faure, Min-Hung Chen, and Shang-Hong Lai. Holistic interaction transformer network for action detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3340–3350, 2023. 2, 6, 7

[12] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 203–213, 2020. 2

[13] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019. 1, 2, 6, 7, 8

[14] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2019. 2, 7

[15] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018. 2, 5, 6, 7

[16] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018. 7, 8

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[19] Berthold KP Horn and Brian G Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics, 1981. 1

[20] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black. Towards understanding action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3192–3199, 2013. 2

[21] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2012. 1

[22] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 6, 7

[23] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3238–3247, 2020. 2, 5

[24] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 7

[25] Okan Köpüklü, Xiangyu Wei, and Gerhard Rigoll. You only watch once: A unified cnn architecture for real-time spatiotemporal action localization. *arXiv preprint arXiv:1911.06644*, 2019. 1, 2, 5, 6, 7

[26] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563. IEEE, 2011. 6

[27] Xinyu Li, Yanyi Zhang, Chunhui Liu, Bing Shuai, Yi Zhu, Biagio Brattoli, Hao Chen, Ivan Marsic, and Joseph Tighe. Vidtr: Video transformer without convolutions. *arXiv e-prints*, pages arXiv–2104, 2021. 2

[28] Yixuan Li, Zixu Wang, Limin Wang, and Gangshan Wu. Actions as moving points. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 68–84. Springer, 2020. 6

[29] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[30] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 2

[31] Yuanzhong Liu, Zhigang Tu, Liyu Lin, Xing Xie, and Qianqing Qin. Real-time spatio-temporal action localization via learning motion representation. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 2

[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 2

[33] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022. 2

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[35] Junting Pan, Siyu Chen, Mike Zheng Shou, Yu Liu, Jing Shao, and Hongsheng Li. Actor-context-actor relation network for spatio-temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 464–474, 2021. 2

[36] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. *arXiv preprint arXiv:1802.05751*, 2018. 7

[37] Rizard Renanda Adhi Pramono, Yie-Tarng Chen, and Wen-Hsien Fang. Hierarchical self-attention network for action localization in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 61–70, 2019. 6

[38] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1, 2, 4, 5

[39] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian 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

[40] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 5

[41] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 1

[42] Lin Song, Shiwei Zhang, Gang Yu, and Hongbin Sun. Tacnet: Transition-aware context network for spatio-temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11987–11995, 2019. 6

[43] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 2, 6

[44] Rui Su, Wanli Ouyang, Luping Zhou, and Dong Xu. Improving action localization by progressive cross-stream cooperation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12016–12025, 2019. 6

[45] Chen Sun, Abhinav Shrivastava, Carl Vondrick, Kevin Murphy, Rahul Sukthankar, and Cordelia Schmid. Actor-centric relation network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–334, 2018. 6, 7

[46] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[47] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 2

[48] Jiajun Tang, Jin Xia, Xinzhi Mu, Bo Pang, and Cewu Lu. Asynchronous interaction aggregation for action detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 71–87. Springer, 2020. 2

[49] Graham W Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European conference on computer vision*, pages 140–153. Springer, 2010. 1

[50] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *arXiv preprint arXiv:2203.12602*, 2022. 1

[51] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 1

[52] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 2

[53] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020. 4

[54] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 5, 7

[55] Chao-Yuan Wu, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13587–13597, 2022. 7

[56] Jianchao Wu, Zhanghui Kuang, Limin Wang, Wayne Zhang, and Gangshan Wu. Context-aware rcnn: A baseline for action detection in videos. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 440–456. Springer, 2020. 2

[57] Tao Wu, Mengqi Cao, Ziteng Gao, Gangshan Wu, and Limin Wang. Stmixer: A one-stage sparse action detector. *arXiv preprint arXiv:2303.15879*, 2023. 2, 6, 7

[58] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv preprint arXiv:1712.04851*, 1(2):5, 2017. 7

[59] Shen Yan, Xuehan Xiong, Anurag Arnab, Zhichao Lu, Mi Zhang, Chen Sun, and Cordelia Schmid. Multiview transformers for video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3333–3343, 2022. 2

[60] Jianhua Yang and Kun Dai. Yowov2: A stronger yet efficient multi-level detection framework for real-time spatio-temporal action detection. *arXiv preprint arXiv:2302.06848*, 2023. 2, 5

[61] Xitong Yang, Xiaodong Yang, Ming-Yu Liu, Fanyi Xiao, Larry S Davis, and Jan Kautz. Step: Spatio-temporal progressive learning for video action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 264–272, 2019. 7

[62] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 4

[63] Yubo Zhang, Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. A structured model for action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9975–9984, 2019. 7

[64] Jiaojiao Zhao, Yanyi Zhang, Xinyu Li, Hao Chen, Bing Shuai, Mingze Xu, Chunhui Liu, Kaustav Kundu, Yuanjun Xiong, Davide Modolo, et al. Tuber: Tubelet transformer for video action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13598–13607, 2022. 6, 7