J. Vis. Commun. Image R. 95 (2023) 103879

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Full length article

Accumulated micro-motion representations for lightweight online action detection in real-time☆

Yu Liu *, Fan Yang, Dominique Ginhac

ImViA EA7535, University of Burgundy, Dijon 21078, France

ARTICLE INFO

Dataset link: https://github.com/alphadadajuju/AMMA

MSC:
41A05
41A10
65D05
65D17

Keywords:
Motion representation
Spatiotemporal action localization
Online action detection
Real-time computing
Embedded system

ABSTRACT

In the last decade, the explosive growth of vision sensors and video content has driven numerous application demands for automating human action detection in space and time. Aside from reliable precision, vast real-world scenarios also mandate continuous and instantaneous processing of actions under limited computational budgets. However, existing studies often rely on heavy operations such as 3D convolution and fine-grained optical flow, therefore are hindered in practical deployment. Aiming strictly at a better mixture of detection accuracy, speed, and complexity for online detection, we customize a cost-effective 2D-CNN-based tubelet detection framework coined Accumulated Micro-Motion Action detector (AMMA). It sparsely extracts and fuses visual-dynamic cues of actions spanning a longer temporal window. To lift reliance on expensive optical flow estimation, AMMA efficiently encodes actions' short-term dynamics as accumulated micro-motion from RGB frames on-the-fly. On top of AMMA's motion-aware 2D backbone, we adopt an anchor-free detector to cooperatively model action instances as moving points in the time span. The proposed action detector achieves highly competitive accuracy as state-of-the-arts while substantially reducing model size, computational cost, and processing time (6 million parameters, 1 GMACs, and 100 FPS respectively), making it much more appealing under stringent speed and computational constraints. Codes are available on https://github.com/alphadadajuju/AMMA.

1. Introduction

In recent years, spatiotemporal action detection/localization has been an active area of research driven by numerous application demands such as unmanned surveillance, driver-assistance systems, and interactive robot services, etc. [1]. When compared to the task of video action recognition, detecting actions in space and time poses more challenges, as it aims to predict the spatial positions, temporal boundaries, and action categories of individual action instances in the video (rather than inferring a global action label). On top of the complex nature of the problem, action detection becomes more difficult when it needs to fulfill online settings, *i.e.*, continuously observing ongoing actions (from streaming videos) and updating detection results in an efficient and real-time fashion. These criteria are crucial in many of the above application scenarios but often overlooked by ongoing research, which solely address high-precision detection while disregarding computational budgets.

Modern solutions for action detection mostly rely on adopting CNN (Convolutional Neural Network) detectors to localize action instances. To extract the temporal cues from actions, one leading approach is to employ the two-stream CNN architecture pioneered by Simonyan and Zisserman [2]. This architecture decouples spatiotemporal reasoning into separate learning of frame-wise RGB and optical flow features, followed by designated fusion strategies such as those proposed by Peng and Schmid [3],Singh et al. [4]. Other approaches further model short-term appearance variations by extending frame-wise detection to the clip-level [5–8]. These methods input a sequence of consecutive frames from which they directly infer action tubelets (*i.e.*, a sequence of bounding boxes). When integrated with the two-stream architecture, action tubelet detectors achieve state-of-the-art performance with 2D CNN backbones. Inspired by leading techniques for action recognition, the latest action detectors also leverage 3D CNN to augment frame-wise prediction with additional spatiotemporal context [7,9–12]. Equipped with a hierarchy of 3D convolutional filters to simultaneously model spatial and temporal variations, 3D CNN-based detectors are capable of learning high-dimensional video representations from consecutive RGB frames alone. Fusing optical flow cues proves to further enhance the temporal modeling capability and detection accuracy in the above methods [13–15].

☆ This paper has been recommended for acceptance by Zicheng Liu.

* Corresponding author.
E-mail addresses: yu_liu@etu.u-bourgogne.fr (Y. Liu), fanyang@u-bourgogne.fr (F. Yang), dominique.ginhac@ubfc.fr (D. Ginhac).

https://doi.org/10.1016/j.jvcir.2023.103879
Received 5 September 2022; Received in revised form 28 April 2023; Accepted 10 June 2023
Available online 16 June 2023

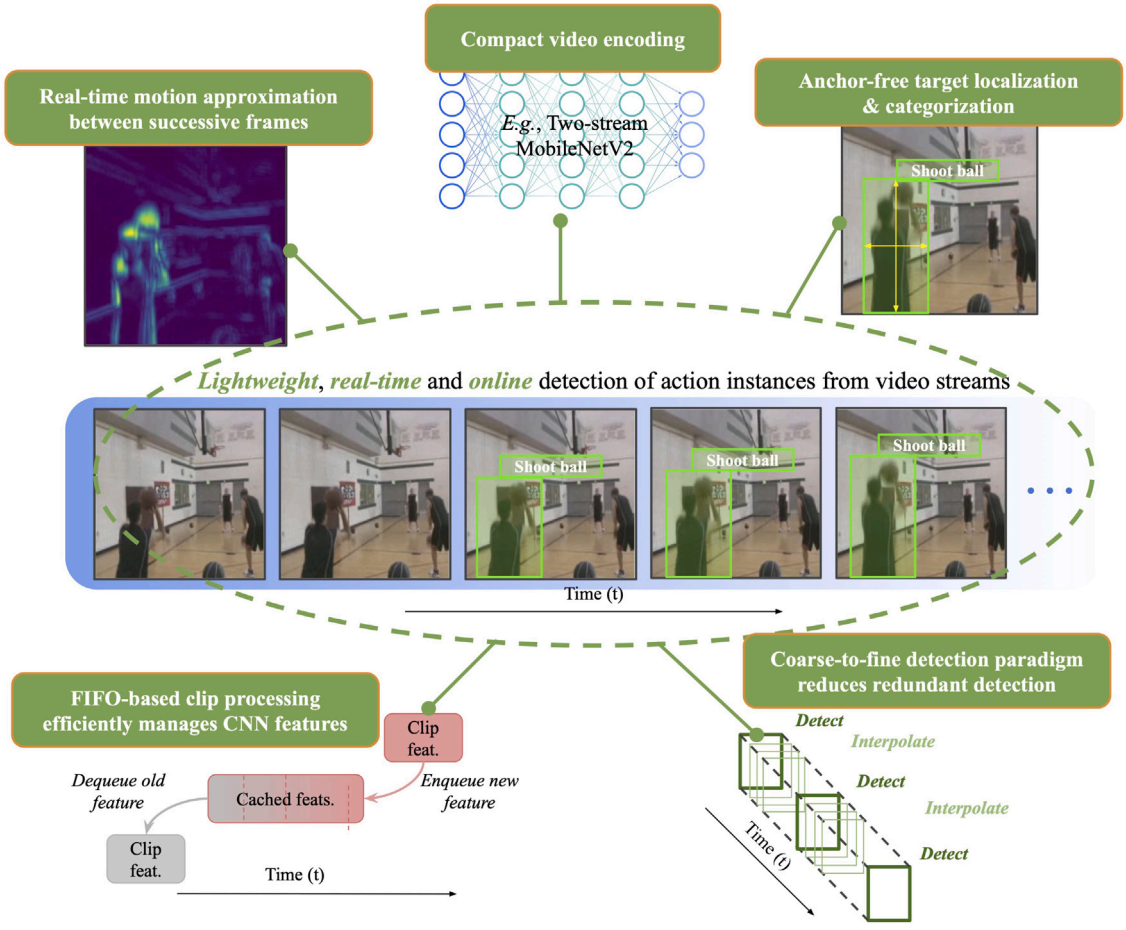

Fig. 1. Overview of AMMA. Given continuous video frames, AMMA incrementally detects underlying action instances by their bounding boxes and categories. Different from existing accuracy-dominating detectors, the design of AMMA integrates a number of highly efficient modules to achieve low-cost, real-time, and online action detection for practical deployment.

In spite of the aforementioned advancements, we argue that trending action detection pipelines have been tailored to solely obtain superior detection scores in public benchmarks. In turn, they are sub-optimal in terms of efficiency for practical deployment. Firstly, short-term dynamics in the form of optical flow are often exploited to model temporal structures of actions. Under this setup, however, optical flow inevitably needs to be prepared in advance as it is expensive and time-consuming to acquire on-site, not only incurring wasteful data storage, but also prohibiting online detection. Meanwhile, even though 3D CNNs can extract spatiotemporal cues directly from RGB frames, they introduce significantly higher computational cost and training difficulty by an order of magnitude than their 2D counterpart. Due to the above drawbacks, existing detectors hardly meet the requirements of vast real-world applications, which seek many other qualities beyond just accuracy, for instance high-speed and continuous workflow. Lately, moving the computation closer to the sensor has become even more critical in order to manage enormous data flow (*i.e.*, video streams). Such migration in the sensing paradigm shifts the dependence from powerful workstation GPUs to resource-limited embedded/edge devices, further demanding drastic reduction in the computational cost of deployed methods.

In this paper, we propose an action detection solution more pertinent to the stringent criteria of practical detection scenarios, termed **A**ccumulated **M**icro-**M**otion **A**ction detector (AMMA, as summarized in Fig. 1). AMMA is a real-time tubelet detector operating on lightweight 2D CNN backbones and raw video clips. It adopts the tubelet detection scheme, acquiring actions' spatiotemporal context by combining successively sampled RGB visuals and complementary dynamic cues. To encode short-term action dynamics in an efficient manner, we devise a simple yet effective motion representation loosely inspired by optical flow by accumulating learnable motion boundaries captured at each video clip (referred to as "micro-motion"). In AMMA's 2D-CNN backbones, micro-motion is computed on-the-fly from RGB frames, whose abstract features can then be adaptively fused with the appearance ones at multiple convolutional scales to produce temporal-aware features.

On top of its spatiotemporal backbone, AMMA aggregates multiple temporal-aware features from successive clips at its detector head, permitting longer-range action modeling. Precisely, it adopts an anchor-free detector head popularized by Zhou et al. [16], which is computationally more efficient than mainstream detectors such as SSD [17] and YOLO [18] who rely on pre-defined anchor boxes densely distributed across the entire scene. Inspired by the recent work of Li et al. [8], AMMA's detector head consists of three cooperative branches for coarsely recognizing and localizing action instances' centers, modeling their movement over time, and regressing their sizes. Furthermore, due to the smoothness nature of continuous actions, our detector can efficiently infer temporally coarse action tubelets across an extended temporal window while interpolating intra-frame detection. When handling online video streams in real-time, AMMA incrementally detects coarse tubelets and links them over time to yield long-range action tubes for spatiotemporal action localization.

To the best of our knowledge, AMMA is one of the few works primarily focusing on highly efficient action detection solutions for realistic deployment on low-end devices. The main contributions of our work can be summarized as follows:

- We propose a compact micro-motion representation to encode short-term action dynamics. Compensating for the low efficiency of traditional optical flow methods, our motion representation can be generated on-the-fly from video streams in real-time.
- We devise a lightweight action tubelet detector integrating 2D CNN backbones, micro-motion generation & fusion, and cooperative detection branches. It adopts a coarse-to-fine detection paradigm to efficiently infer actions in online settings.
- We tailor the proposed detection pipeline with three ultra-lightweight CNN backbones and validate their superior mixture of performances in precision, speed, and complexity on spatiotemporal action benchmarks.

2. Related work

Different from action recognition or temporal action detection, spatiotemporal action detection not only requires localizing when actions occur along the time span, but also spatial localization down to the frame level. Hence, leading methods in the field typically leverage varied temporal modeling techniques on top of CNN-based object detectors. In this section, we briefly review recent progresses in object detection and spatiotemporal action detection.

2.1. CNN-based object detection

Existing object detectors mostly deduce objects' categories and locations from pre-defined proposals of bounding boxes (*i.e.*, "anchor" boxes) densely placed over the input image. Under the anchor-based framework, two detection pipelines are widely explored. R-CNN and follow-up research [19–21] adopt a two-stage approach. In the first stage, class-agnostic region-of-interest (RoIs) are regressed from a set of anchors via the Region Proposal Network (RPN). Next, features pooled from each RoI are further categorized, and that RoI's spatial extent is refined to form the final bounding box. Such a two-stage workflow imposes a bottleneck upon real-time inference speed. To accelerate detection, single-stage detectors such as YOLO by Redmon and Farhadi [18] and SSD by Liu et al. [17] remove the intermediate region proposal step. In a single forward-pass, they directly perform bounding box regression and classification on anchors across every grid of the image feature. With this pipeline, one-stage detectors can operate in real-time while retaining competitive accuracy as the two-stage variants.

Integrating anchor boxes has become the mainstream design choice in modern detectors. Nevertheless, utilizing anchors introduces excessively more design parameters associated with anchor sizes, aspect ratios, and number of pre-defined boxes, etc. These hyperparameters largely impact the final detection performance and require heuristic tuning for different datasets. Further, anchors incur complicated intersection-over-union (IoU) computation when matched with groundtruth boxes. In contrast, some newly proposed detectors demonstrate comparable accuracy by directly regressing objects' shapes and locations without pre-defined anchors (such as the works by Zhou et al. [16],Law and Deng [22],Liu et al. [23],Tian et al. [24],Xie et al. [25]). For instance, Zhou et al. [16]'s CenterNet represents an object by its bounding box's center, converting the detection task to a keypoint estimation problem. After acquiring image features, the network predicts objects' center points in the form of a multi-channel heatmap. Peaks within the heatmap are regarded as the center locations of detected objects, and each channel is associated with a class. Objects' bounding boxes can then be regressed from image features whose locations match those of the deduced centers. In a similar spirit, Law and Deng [22]'s CornerNet detects objects as pairs of keypoints by predicting two separate heatmaps, each encoding the top-left and bottom-right corners of objects' bounding boxes. Detected corners are considered belonging to the same objects and then grouped together based on their similarity embedding.

2.2. Spatiotemporal action detection

Many efforts have been made to extend image-based object detectors to video action detection. One popular approach to embed temporal information for action reasoning is to adapt the two-stream CNN architecture pioneered by Simonyan and Zisserman [2]. In this architecture, appearance and motion features are independently extracted from RGB and optical flow inputs using two feed-forward networks. Fusing the results from both modalities first demonstrates beneficial for action recognition accuracy, inspiring many subsequent works in related fields [26–28]. Under the hood, Sevilla-Lara et al. [29] found that fusing optical flow with RGB modalities not only provides additional temporal information, but also helps to capture appearance-invariant structures (*i.e.*, moving targets' "boundaries"). This facilitates model generalization against vast inter-class appearance variations across action videos. Building upon the two-stream framework, Peng and Schmid [3],Saha et al. [30] employ two parallel R-CNNs (*i.e.*, for RGB and optical flow) and combine action proposals from both RPNs to augment detection. Frame-wise results of the entire video is then linked over time into action tubes by solving two energy maximization problems, ensuring optimal temporal coherence and smoothness. Alternatively, Singh et al. [4] employ two SSD, a reduced optical flow estimator, and an incremental linking algorithm to enable online action detection in real-time. Under a similar setup, [31] integrates an optical flow sub-network within the detection framework which allows joint optimization of optical flow generation tailored to the task of action detection.

To further encode the intra-frame temporal relationships between action regions spanning continuous video frames, Kalogeiton et al. [5],Yang et al. [7] take a short clip of consecutive RGB frames as input, regressing 3D anchor cuboids to obtain tubelet detection. Saha et al. [32] also formulate a similar clip-based learning scheme utilizing two successive frames (but not necessarily consecutive), enabling learning from action sequences without dense per-frame annotation. Inspired by recent advancements in various 3D CNN architectures which can model highly abstracted video representations with proper pretraining [33], 3D CNN has been widely exploited in the latest studies of action detection [10–15,34]. More recently, the adoption of self-attention and transformer architecture by Vaswani et al. [35] for modeling actions over long sequences also receives rising attention [12,36].

Extracting relevant spatiotemporal cues from actions is a challenging problem that often relies on convoluted strategies at high computational cost (*e.g.*, 3D CNN or optical flow generation). Alternatively, several research seek more efficient architectures or workflows to cope with action detection toward practical configurations. Instead of employing parallel CNNs, [6] leverage an innovative two-stream fusion scheme, only extracting shallow optical flow context and using it to directly modulate low-level RGB features for efficiency gains. As computing optical flow itself imposes bottleneck upon online and real-time inference, Zhang et al. [37] propose gathering motion boundaries by finding the temporal offsets between shallow-CNN features, lifting reliance on traditional optical flow. To avoid the heuristic anchor design adopted by most tubelet detector, Li et al. [8] extends CenterNet by treating each actor as a point, and further models actions over time via the trajectory of moving points. Considering that dense per-frame detection is redundant for efficient action inference, Li et al. [15] further introduces a progressive paradigm, which initially estimates coarse spatiotemporal action tubes spanning the entire video and then selectively refines these tubes at sampled timestamps. Even though the above works do not specifically address spatiotemporal action detection in online settings, they share our incentives of approaching action reasoning jointly from the point-of-view of high accuracy and efficiency.

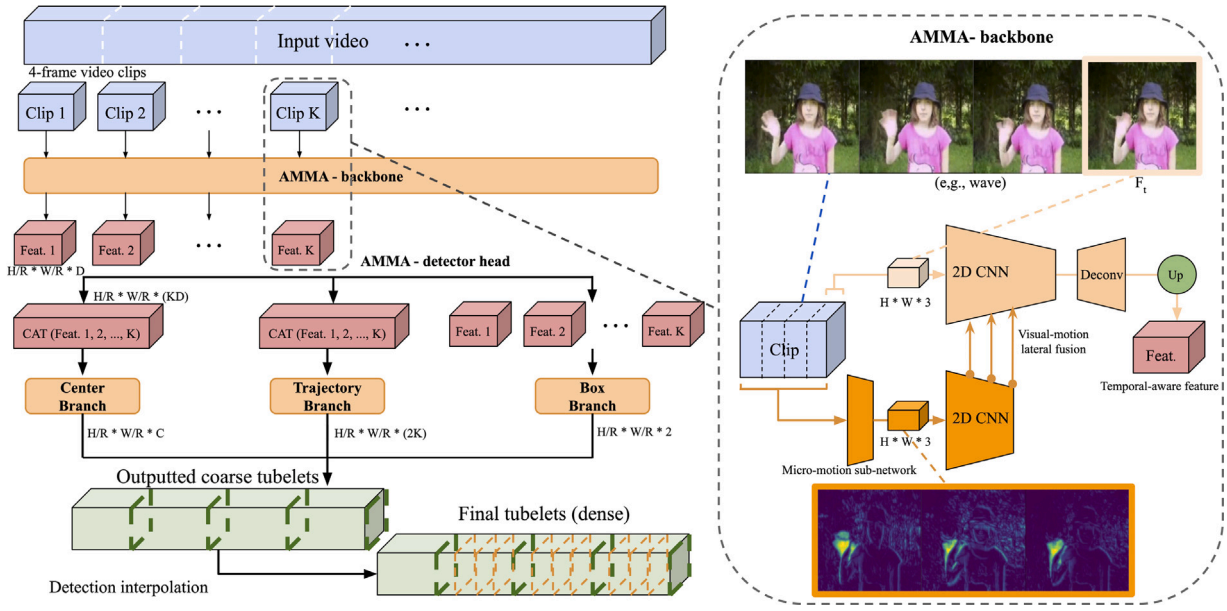

Fig. 2. Overview of AMMA. AMMA's backbone takes a video clip of t frames as input at a time ($t = 4$ in this study), encodes short-term action dynamics as accumulated micro-motion, and outputs a motion-aware feature tensor by merging appearance (from F_t) and complementary motion information via lateral fusion. Beyond a single clip, AMMA enables long-range spatiotemporal modeling by aggregating multiple clip-level features at its detector head consisting of three cooperative branches. After merging results of the detector branches, the predicted tubelets are coarse in time. From the coarse tubelets, dense frame-wise detection can be interpolated between any two clips in a later stage. In this figure (and throughout the manuscript), K denotes the number of clips, H and W denote the height and width of the RGB/motion frame, R denotes the spatial downsampling ratio, D denotes the number of channels, and C denotes the number of action categories. CAT is short for concatenation.

3. Methodology

3.1. Overview

Our proposed detection framework, termed Accumulated Micro-Motion Action detector (AMMA), is an end-to-end 2D-CNN-based tubelet detector. As summarized in Fig. 2, AMMA takes multiple short video clips as input and produces temporally coarse action tubelets spanning the input sequence. Each video clip comprises t consecutive frames. From each video clip, appearance information is extracted from the latest frame (F_t) by the 2D CNN backbone. Alongside appearance feature extraction, each clip is fed to our micro-motion sub-network which generates and accumulates short-term dynamics of actions. From the micro-motion, temporal cues are further extracted and fused with the appearance features from F_t via multiple lateral connections to encode short-term spatiotemporal context for the clip.

To model longer spatiotemporal structures across multiple video clips, AMMA aggregates their respective temporal-aware features by stacking them in the channel dimension at its detector head. In essence, the aggregated features are fed to three branches to recognize and spatially localize action instances' centers, model the trajectories of action centers over time, and regress their spatial extent (*e.g.*, height and width). Cooperative modeling of the three branches produces action tubelets that are temporally coarse, where detection takes place only at F_t of each clip. Action tubelets can be incrementally detected and linked over time following the designated matching strategy, forming long-range action tubes for spatiotemporal localization. Finally, dense frame-wise detection is acquired by intra-frame interpolation between any two detection. The following sections describe each working module of AMMA in detail.

3.2. AMMA-backbone

Clip-level appearance information. We define an input video clip V_{cp} to contain t consecutive RGB frames, where $V_{cp} = [F_1, F_2, \dots, F_t]$. The dimension of each frame is $H \times W \times 3$. Since neighboring frames share highly resembling visual cues, we only extract appearance information from F_t via a 2D CNN. Formally, we adopt a reduced variant of the encoder–decoder architecture used by Zhou et al. [16] as the 2D backbone. Originally, three deconvolution layers have been added at the end of ResNet's [38] final convolution layer as the decoder. This serves to adaptively project highly abstracted features onto a spatially larger feature map to facilitate dense detection of small/overlapped objects. Different from object detection, it can be reasonably assumed that the likelihood of actors emerging densely in a scene is low. With this insight, AMMA's backbone decoder is implemented with only one deconvolution layer followed by bilinear upsampling. The resulted appearance feature is a tensor with dimension $\frac{H}{R} \times \frac{W}{R} \times D$, where R and D correspond to the downsampling ratio and channel dimension of the feature, respectively. In practice, R and D are 8 and 256, respectively.

Accumulated micro-motion as clip-level action dynamics. A short sequence of t frames still potentially embeds crucial dynamic information which F_t alone does not carry. Such a motion cue, often encoded in optical flow, consistently grants two-stream CNN networks better discriminating capacity to recognize actions. Alternative to optical flow which is commonly prepared in advance due to its high computational cost, we devise a simpler, adaptive motion representation which highlights the small displacements of motion boundaries.

We uncover motion information of a clip by simply accumulating the appearance variation between F_t and its precedent frames in the shallow-CNN feature space. Specifically, shallow-CNN features tend to reflect local patterns (*e.g.*, edges or textures) with low receptive fields. The difference map between two such low-level features within close temporal proximity inherently encapsulates the temporal evolution of various general patterns. This simulates the "motion boundaries" described by Sevilla-Lara et al. [29] as introduced in Section 2.2. We refer to our implicit motion representation as accumulated micro-motion.

Formally, we define the shallow convolutional block, $Conv_{5\times5}$, as eight 5×5 convolutions with strides of 1 and paddings of 3. The input to the convolutional block is any clip V_{cp} where all its frames are first downsampled by two via a max pooling layer. The downsampling operation comes from our observation that the difference map between two shallow features within close temporal proximity retains values

near 0 in most areas, *i.e.*, it only contains high responses in motion salient regions. As the difference map exhibits high sparsity, it is more efficient to process it in a low-resolution space without much loss of information. Concretely, the above steps are described as follows:

$$[f_1, f_2, \dots, f_t] = Conv_{5\times5}(MaxPool([F_1, F_2, \dots, F_t])) \tag{1}$$

$$MM_i^d(x,y) = f_t^d(x,y) - f_i^d(x,y), \quad for\ i = 1:t-1, \tag{2}$$

where in Eq. (1), $f_1, f_2, \dots, f_t$ represent shallow-CNN features of frames in V_{cp}, each with a dimension of $\frac{H}{2} \times \frac{W}{2} \times 8$. In Eq. (2), $f^d(x,y)$ denotes the intensity of a feature at its dth channel and pixel location (x,y). As expressed in this equation, each micro-motion MM_i corresponds to the feature-level difference between the respective frame F_i and F_t. Note that our design of shallow layers is intentional, as deep-CNN features with large receptive fields have been overly abstracted and lost essential spatial information associated with the boundaries of moving targets.

To efficiently encode motion variation across different feature spaces and time steps, all MM^d are first accumulated into one channel to manifest the motion magnitude, as shown in Eq. (3):

$$AMM_i(x,y) = \sqrt{\sum_{d=1}^{8} (MM_i^d(x,y))^2}, \quad for\ i = 1:t-1. \tag{3}$$

Unlike optical flow fields which typically encode horizontal and vertical motion vectors, each AMM cue is an appearance-invariant saliency map reflecting small displacement of motion boundaries. To further incorporate temporal structures of the clip, we concatenate all $t-1$ AMM_i in the channel dimension, followed by a bilinear upsampling operation so that the final clip-level micro-motion matches the spatial dimension of the original frames (as we first downsampled the frames via max pooling in Eq. (1)). In practice, clip length t is defined as 4 in this study; the accumulated micro-motion of a clip thus has the same dimension as an RGB frame (*i.e.*, $H \times W \times 3$). This design allows us to further extract temporal features from the accumulated micro-motion while seamlessly leveraging a pre-trained CNN model (which typically takes a color image with three channels as input).

Multi-scale spatiotemporal fusion. In AMMA's backbone, multi-scale lateral fusion is utilized to incorporate micro-motion features into the appearance ones. We construct a separate 2D CNN similar to the one used by the RGB stream, and feed the accumulated micro-motion to this new network. Essentially in the above setup, each CNN backbone dedicates to extracting clip-level spatial or temporal information. Lateral connections are established between designated layers of the two CNNs, where weighted summation is carried out to fuse their respective features. Specifically, we devise uni-lateral fusion connections; only the spatial CNN is aware of the complementary motion context when a clip is inputted.

In AMMA, the weights to sum spatial and temporal information are learnable scalars (ranging from 0 to 1) that add up to 1. In addition, the number of lateral connections dictates the extent of fusion between actions' visual and dynamic cues, which will be studied more thoroughly in Section 4.2. Since fusion is conducted by summation at the spatial CNN, the dimension of the short-term motion-aware features remains $\frac{H}{R} \times \frac{W}{R} \times D$.

3.3. AMMA-detector head

Once consecutive clip-level features are extracted from their respective clips, AMMA's detector head aggregates them for richer spatiotemporal context and action tubelet inference. The detector head is composed of three branches: Center branch, Trajectory branch, and Box branch. The function of each branch is summarized in Fig. 3.

Center branch. Given a sequence of K clip-level features, Center branch aggregates these features and locates action instances by their centers at the end of the sequence. In other words, it finds centers of actions taking place at the final frame of the Kth clip (*i.e.*, F_t^K). To aggregate clip-wise context, all K features are first stacked together in the channel dimension to form video feature representation $f_{stack} \in \mathbb{R}^{\frac{H}{R}\times\frac{W}{R}\times KD}$. Afterwards, f_{stack} is fed to a standard 3×3 and 1×1 convolutional layer interleaved with ReLU non-linearity, outputting action heatmap $\hat{L} \in \mathbb{R}^{\frac{H}{R}\times\frac{W}{R}\times C}$ for F_t^K, where C corresponds to the number of action classes. Each grid of $\hat{L}_{x,y,c}$ reflects the probability of detecting action instance of class c at location (x,y) of the heatmap.

To train Center branch, the groundtruth heatmap $L \in \mathbb{R}^{\frac{H}{R}\times\frac{W}{R}\times C}$ associated with a K-clip sequence is first derived from the groundtruth center location (x_{c_i}, y_{c_i}) of F_t^K, where c_i corresponds to the true class of action instance i. We set heatmap $L_{x,y,c} = 0$ for all classes except for the true class. When $c = c_i$, a Gaussian kernel is applied to generate soft heatmap $L_{x,y,c_i} = exp(-\frac{(x-x_{c_i})^2+(y-y_{c_i})^2}{2\sigma^2})$, where the salient region surrounds (x_{c_i}, y_{c_i}), and its dimension is determined by σ^2 derived from the groundtruth instance's size. The training objective for Center branch follows the same focal loss used by [16] as shown below:

$$l_{Center} = -\frac{1}{n}\sum_{x,y,c}\begin{cases}(1-\hat{L}_{xyc})^{\alpha}log(\hat{L}_{xyc}), & \text{if } L_{xyc}=1\\ (1-L_{xyc})^{\beta}(\hat{L}_{xyc})^{\alpha}log(1-\hat{L}_{xyc}), & \text{otherwise,}\end{cases} \tag{4}$$

where n is the number of groundtruth instances; α and β are hyperparameters of the focal loss.

At the inference stage, the resulted heatmap is further filtered independently for each class to only keep local peaks that are greater than their 8-connected neighbors. Finally, the top N peaks across all classes are considered candidate action centers. In this study, we follow the work of Zhou et al. [16] and set α, β, and N to 2, 4, and 100 respectively.

Trajectory branch complements Center branch by modeling action instances' center movement between frames $F_t^1, F_t^2, \dots, F_t^{K-1}$ and F_t^K. Similar to Center branch, Trajectory branch first aggregates K clip-level features by concatenation across the channel dimension, followed by a standard 3×3 and 1×1 convolution interleaved with ReLU. The output of the branch is movement map $\hat{m}^{F_t^K} \in \mathbb{R}^{\frac{H}{R}\times\frac{W}{R}\times 2K}$, where $2K$ denotes the center offsets (in X and Y directions) sequentially for $F_t^1, F_t^2, \dots, F_t^K$ with respect to action centers at F_t^K.

For training, groundtruth action centers at $F_t^1, F_t^2, \dots, F_t^K$ are first computed the same way as for Center branch. Then, the groundtruth movement ($m^{F_t^K}$) of any action instance with respect to F_t^K is simply the offset between its center at F_t^K and those at other frames. Finally, movement map $\hat{m}_i^{F_t^K}$ is optimized based on L1 loss as follows:

$$l_{Trajectory} = \frac{1}{n}\sum_{i=1}^{n} |\hat{m}_i^{F_t^K} - m_i^{F_t^K}|, \tag{5}$$

where i indicates the ith out of n action instances.

During inference, Center Branch obtains action centers at the end of the input sequence as references, while Trajectory branch adjusts all action centers at the end of each clip according to the predicted offsets with respect to the reference centers. Note that the predicted center offset at F_t^K from itself is expected to be zero; as a result, we do not adjust action centers at F_t^K.

Box branch. Box branch serves to regress the spatial extent of action instances at $[F_t^1, F_t^2, \dots, F_t^K]$, whose locations have been previously deduced by Center and Trajectory branch. Unlike these first two branches, incorporating temporal information from multiple frames contributes less to frame-wise class-agnostic bounding box regression. Hence, our Box branch regresses actions' width and height for each clip independently. It comprises a 3×3 and 1×1 convolutional layer in sequence (interleaved with ReLU) as the other branches, and generates spatial prediction map $\hat{s} \in \mathbb{R}^{\frac{H}{R}\times\frac{W}{R}\times 2}$, where 2 corresponds to the height and width prediction. As Box branch is shared by all K clip-level features, it outputs K spatial maps, each one being associated

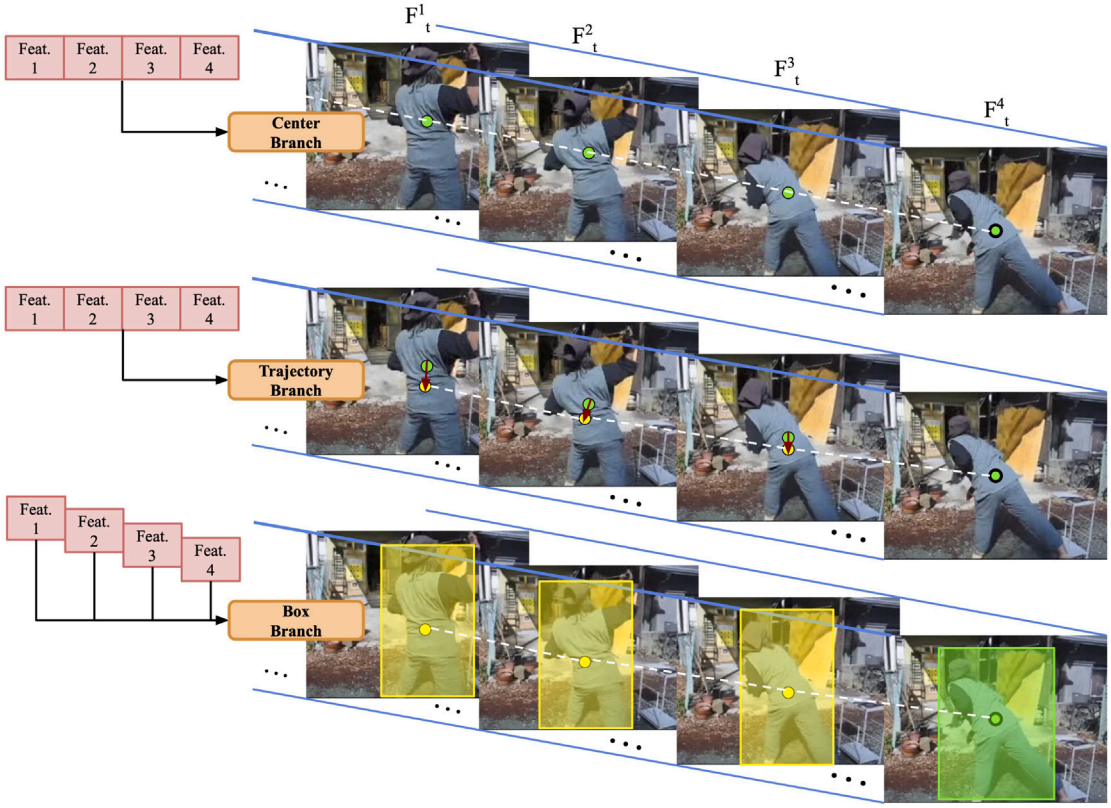

Fig. 3. Overview of AMMA's detector head. Given an input sequence of K clips ($K = 4$ in this figure), Center branch (TOP) detects action centers at F_t^K. Trajectory branch (MIDDLE) infers center offsets with respect to Center branch's prediction, and adjusts action centers for $F_t^1, F_t^2, \ldots, F_t^{K-1}$ accordingly. Finally, Box branch (BOTTOM) regresses action instances' spatial extent (*i.e.*, height and width) at action centers deduced by the other two branches.

with the size prediction at $F_t^1, F_t^2, \ldots, F_t^K$. We optimize this branch by summing the L1 loss at all clips as follows:

$$l_{Box} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{K}|\hat{s}_i^j - s_i^j|, \tag{6}$$

where s_i^j corresponds to the groundtruth height and width of the ith action instance (out of n instances) belonging to the jth clip.

The overall training objective of AMMA is shown in Eq. (7), where hyperparameter a, b, and c are set to 1, 1, and 0.1 respectively in accordance with [16].

$$l_{AMMA} = a l_{Center} + b l_{Trajectory} + c l_{Box} \tag{7}$$

3.4. Online/incremental detection via feature-caching-dequeuing

Our proposed action detector requires only RGB frames as input. As it generates motion representations on-the-fly, AMMA can be applied directly to real-time video streams. To efficiently and continuously handle incoming video frames, we employ a simple feature-caching mechanism that allows AMMA to focus on extracting relevant features only from the current clip while still being able to exploit clip-level features from the past for longer-range spatiotemporal reasoning. Fig. 4 illustrates such an online detection workflow.

In detail, given that K clips are needed for action inference, AMMA's backbone initially obtains K clip-level features from which action tubelets are regressed at the detector head. Meanwhile, the K clip-level features are also cached in AMMA's buffer. Once enough incoming frames are gathered as a valid new clip (*i.e.*, reaching t frames), our detector only extracts the Kth clip-level feature from this new clip. The past $K - 1$ features can be efficiently retrieved from the buffer and combined with the current one, from which the detector head predicts new action tubelets. Clip-level feature-caching and dequeuing enable AMMA to incrementally infer action tubelets covering past and incoming new frames while processing only the newly arrived clip. During video streaming, AMMA's buffer will be updated accordingly to only keep the most recent K features.

3.5. From coarse tubelets to dense action tubes

Given an incoming video stream, AMMA detects tubelets on top of the latest K clips. Notably, the lastly detected tubelets have a temporal overlap with the previous ones by a duration of $(K - 1)$ clips (as illustrated in the bottom-right corner of Fig. 4). When tubelet results within the temporal overlaps are consistent, AMMA can incrementally link local tubelets over time into action tubes, yielding long-range space–time proposals for localizing actions in trimmed/untrimmed videos.

We adopt an online linking algorithm similar to the one used by Kalogeiton et al. [5]. Given a video stream with sufficient frames as input, AMMA detects N initial tubelets. Among them, the top ten tubelets with the highest confidence scores are kept as "active" action links for subsequent tubelet linking. As the video continues to be streamed, we incrementally extend active links with new tubelet candidates if their detections at corresponding temporal positions match (*i.e.*, the average IoU exceeds threshold $\tau = 0.5$). It is noteworthy that each candidate tubelet can only be assigned to an active link. On the other hand, an active link stops extending and is terminated ("inactive") either when there no longer exists temporal overlaps with the newly detected tubelets, or the video stops being streamed.

The final action tubes are constructed from all the inactive action links, where each tube's confidence score is calculated as the average score of all its enclosed tubelets. The temporal extent of any action tube is determined by the starting frame of the initialized tubelet and the end frame of the last tubelet. Lastly, we discard any final action tube having either a low confidence score or a short temporal duration. To acquire

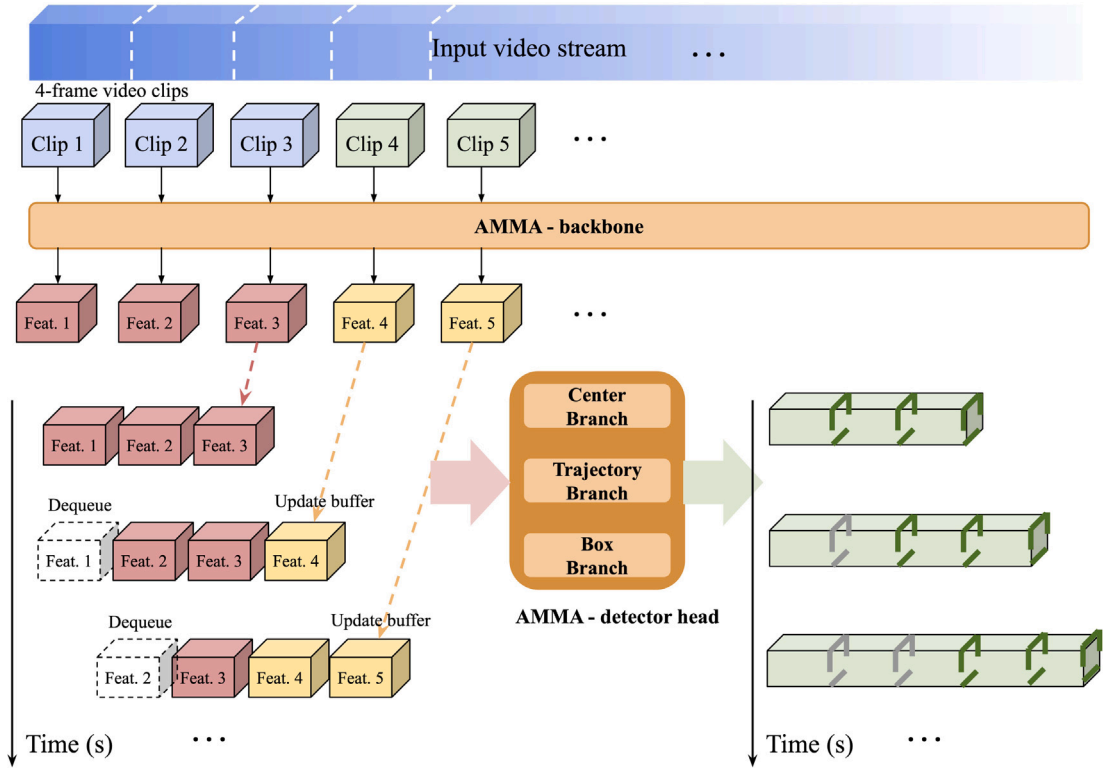

Fig. 4. AMMA's incremental feature-caching-dequeuing mechanism for online action detection on streaming videos.

temporally dense (*i.e.*, frame-wise) detection, we apply coordinate-wise linear interpolation between bounding boxes located at separate clips to infer detection for intermediate frames. This design form is reasonable as transitions of actions across consecutive frames are typically smooth and continuous.

4. Experiments

4.1. Experimental setup

Dataset. Our proposed action detector is evaluated on two popular action datasets: UCF-24 [39] and JHMDB-21 [40]. The former one is composed of 3207 temporally trimmed/untrimmed sports videos of 24 classes. The number of action instances varies in this dataset. The latter consists of 928 short videos (maximum of 40 frames) divided into three splits, with 21 action categories in daily life such as *sit*, *stand*, and *walk*, etc. Each video is temporally trimmed and has a single action instance. For JHMDB-21, the experimental results are reported over the average of its three splits. Unlike datasets for action recognition or temporal action detection, both of the above datasets provide actions' temporal extent and frame-level bounding-box annotations which AMMA strictly seeks for model training.

Metrics. Following previous studies in spatiotemporal action detection, we evaluate the accuracy of our proposed detector using frame-mAP and video-mAP (mean Average Precision). The former metric validates the IoU between the detected and groundtruth boxes at each frame and is independent of the online linking strategy. For frame-mAP, the IoU threshold is fixed at 0.5 throughout all experiments. On the other hand, video-mAP inspects spatiotemporal overlaps between linked action tubes and groundtruth tubes at multiple IoU thresholds. Furthermore, to evaluate the efficiency of AMMA, we also report its model size (number of trainable parameters), MACs (number of multiply-accumulate operations), and speed (frame-per-second, or FPS).

Implementation details. Aiming to conduct highly accelerated and efficient detection, we first employ ResNet-18 as AMMA's main 2D CNN backbone. All RGB frames inputted to our model are resized to 288×288. AMMA's backbone includes an encoder–decoder feature extractor followed by a bilinear upsampling layer, transforming video clips to clip-level representations of dimension $36 \times 36 \times 256$. Prior to AMMA's detector head, clip-level features are first fed to another 1×1 convolutional layer to reduce their channel dimension by 4 in order to gain efficiency at Center and Trajectory branches (who operate on channel-wise stacked features).

Within AMMA's backbone, the fusion of spatial and temporal information is realized by lateral fusion. In the case of ResNet-18, we establish uni-lateral connections at the "stage" level. To investigate the influence of combining micro-motion and RGB features at different scales, we vary the extent of fusion by progressively deepening the network dedicated to micro-motion abstraction while adding a lateral connection at the output of each stage (up to five connections for ResNet-18). To verify our detection framework on ultra-lightweight architectures for resource-constrained devices, we also evaluate its integration with MobileNet-V2 [41] and ShuffleNet-V2 [42]. The weights of all 2D CNN backbones are initialized with COCO pretrain (except for ShuffleNet-V2 which uses ImageNet pretrain).

During training, we apply common practices of data augmentation such as photometric transformation, scale jittering, random cropping/expansion, and location jittering, etc. To train AMMA on K-clip sequences, each action tubelet is expected to last $K \times t$ frames. For any action video having a shorter duration, we pad the beginning of its K-clip sequence by the first frame of the video until the minimum length requirement is met, simulating an action without movement at the beginning. At AMMA's detector head, only the groundtruth associated with the final frame of a clip (*i.e.*, F_t^K) is used to train Center branch. On the other hand, Trajectory and Box branch learn to regress movement and spatial dimension of action instances over all the clips, thus requiring groundtruth labels of $F_t^1, F_t^2, \ldots, F_t^K$.

We use the Adam optimizer to train our models. AMMA is trained in an end-to-end manner where all its components (the backbone, micro-motion estimator and detector head) are jointly optimized. An

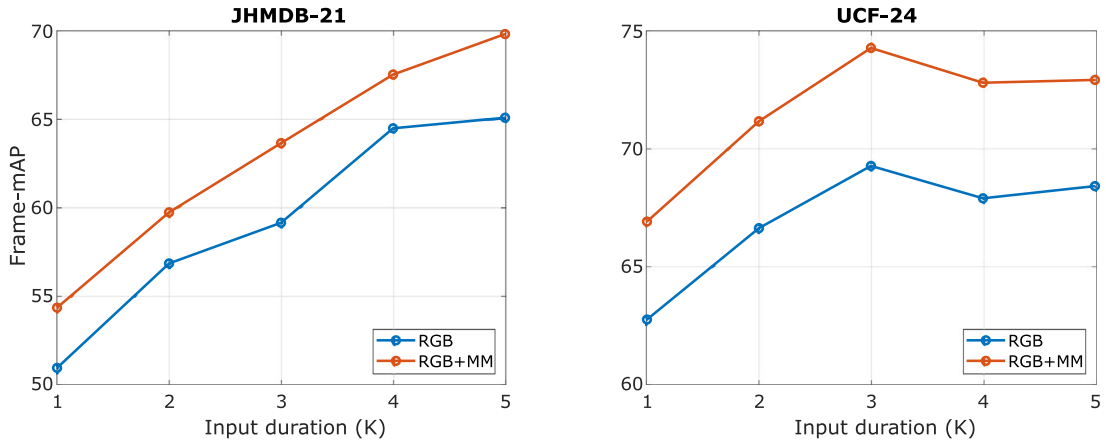

Fig. 5. Frame-mAP performance under varied input duration (*i.e.*, number of clips). "MM" denotes micro-motion.

initial learning rate of $5e^{-4}$, $2.5e^{-4}$, and $2.5e^{-4}$ is applied when employing ResNet-18, MobileNet-V2 and ShffleNet-V2 as AMMA's backbone, respectively. For JHMDB-21, we train AMMA for 10 epochs while reducing the learning rate by a factor of 10 at the 6th and 8th epoch. Likewise, UCF-24 is trained for 10 epochs, but with the learning rate reduced by half at every epoch after the second one. In our experiments, all the training is conducted on an NVIDIA Titan V GPU while the mini-batch size is fixed to 16.

4.2. Ablation study

In this section, we investigate various architectural configurations of AMMA. For efficient exploration, the following studies are conducted using ResNet-18 unless specified otherwise.

Effect of input duration and micro-motion fusion. The core of AMMA lies in detecting action tubelets across successive video clips. Intuitively, combining more clips as input encapsulates richer spatiotemporal context. However, longer sequences could potentially introduce irrelevant background cues, as well as raising difficulty to track tubelets' trajectories. To investigate how the input duration affects the proposed detector, we conduct experiments on both JHMDB-21 and UCF-24 by varying the number of input clips (denoted as "RGB"). To jointly examine the influence of incorporating dynamic features under varied input length, we replicate the above experiment while introducing micro-motion fusion ("RGB + MM"). In these experiments, the extent of motion fusion is fixed to three lateral connections (at the output of the first three stages in ResNet-18). The corresponding frame-mAP results are depicted in Fig. 5.

From the above experiments, we first observe that AMMA generally produces more accurate tubelets the longer video sequences it sees. This result matches our hypothesis that reasoning from longer video clips enriches spatiotemporal feature learning. Notably, AMMA's accuracy continues to benefit on JHMDB-21 as K increases. We observe that longer input sequences improve accuracy mainly by reducing false-positive detection in videos where ambiguous visual cues are present. Fig. 6 displays several examples where AMMA manages to detect correctly when enlarging its temporal receptive field across longer video sequences. We stop increasing the number of clips at 5 (equivalent to 20 frames), as that nearly takes up half of the frames in most videos of this dataset. We adopt a similar setup when evaluating UCF-24. Interestingly, although input duration and accuracy remain positively correlated, AMMA performs best when $K = 3$. We deduce that as UCF-24 consists of temporally untrimmed videos, AMMA is prone to produce more temporal false-positive detection (on frames having no groundtruth action) when assigning a unified action label to a longer sequence.

Alongside varied input duration, all the configurations with micro-motion feature fusion consistently outperform those using only appearance cues, confirming the efficacy of modeling short-term dynamic motion to help differentiate actions. For instance, the baseline configuration $K = 1$ ("RGB") is the least accurate due to a complete lack of temporal modeling (neither incorporating short-term dynamic nor successive appearance variation cues). To better understand predicted tubelets' accuracy with and without motion fusion, we examine five mutually exclusive factors that result in the final frame-mAP. Following common practices adopted by Kalogeiton et al. [5], we define five sources of errors, namely localization error (E_L), classification error (E_C), time error (E_T), other error (E_O), and missing-detection error (E_M). In brevity, E_L is associated with detection that contains the correct class, but does not precisely localize the target. In contrast, E_C is raised when a detection properly bounds the underlying target, but the predicted class is incorrect. E_T refers to having a detection in an untrimmed video for the correct class, but the groundtruth temporal extent of the action does not cover this frame. E_O corresponds to detection that is falsely detected in terms of both localization and classification. Finally, E_M refers to false-negative detection. Note that the sum of these errors subtracted by 100 results in the obtained frame-mAP of a particular model.

Fig. 7 reports the error distribution comparison before and after fusing micro-motion features. On JHMDB-21, one can perceive that most of AMMAs' false detection comes from E_C while the rest of errors are small and scattered ($E_T = 0$ as JHMDB-21 is a temporally trimmed dataset). Specifically, the "RGB+MM" model significantly outperforms the "RGB" one by improving E_C for nearly 5% (27.4 vs. 32.21), which gives rise to the discrepancy between their frame-mAP. This demonstrates that integrating micro-motion features plays an essential role to induce learning action-specific dynamics and reduce false-positive classification. Our results are aligned with many previous studies which find motion cues particularly useful for JHMDB-21. Different from JHMDB-21, the highest frame-mAP loss for UCF-24 originates from time error E_T while the remaining loss is scattered across all other types of errors. One can observe that fusing the proposed motion features raises AMMA's overall frame-mAP by reducing false-positive and false-negative detection throughout all error categories.

More on micro-motion generation, fusion, and complexity. The previous experiments demonstrate AMMA's extensible temporal modeling capacity along with varied input duration, as well as the benefit of incorporating micro-motion features. Here, we further investigate different forms of micro-motion generation and fusion, along with associated computational cost.

Table 1 summarizes AMMA's detection accuracy and computation on JHMDB-21 in accordance with varied input forms. Building upon the input-duration experiment, we adopt the 5-clip input and three

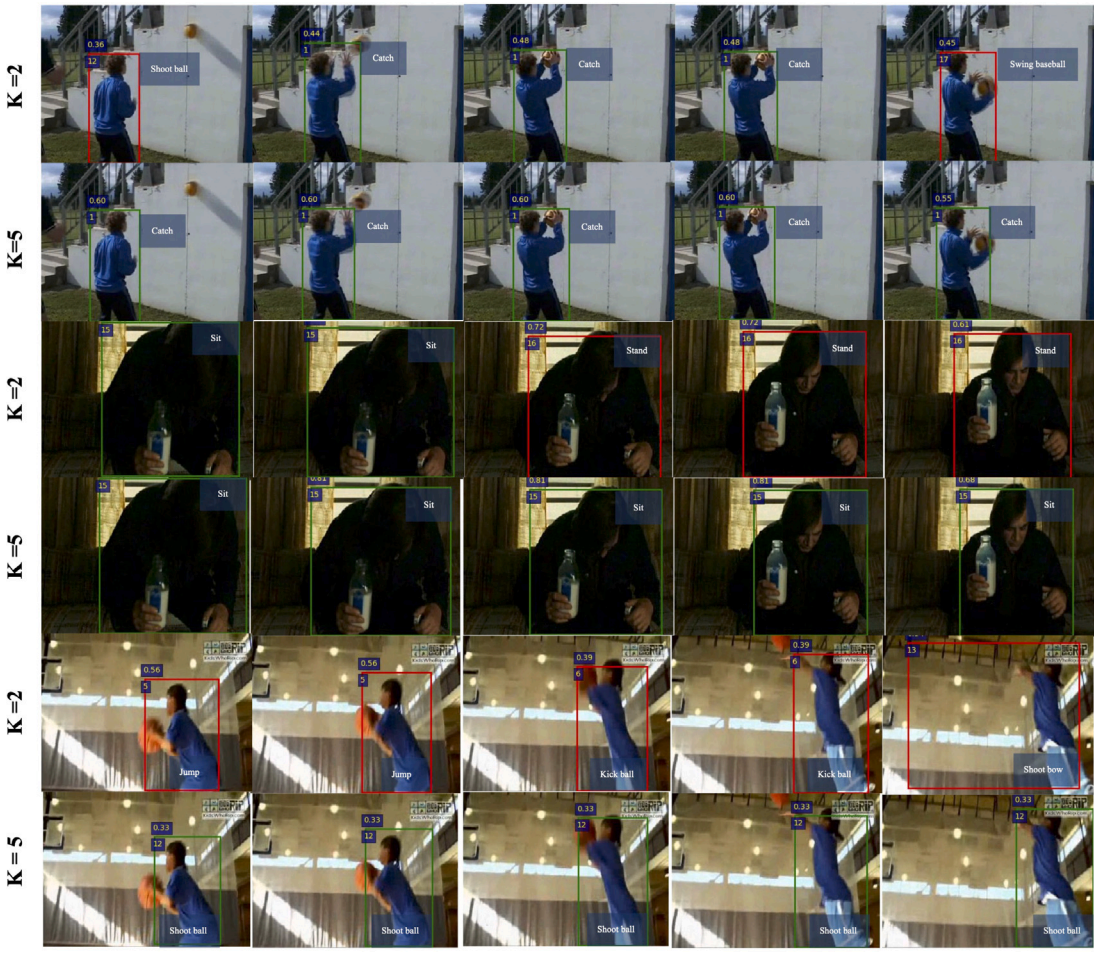

Fig. 6. Examples of short-tubelet ($K = 2$) and long-tubelet ($K = 5$) detection on JHMDB-21. The groundtrue actions (from top to bottom) are *catch*, *sit*, and *shoot_ball*. The green and red boxes correspond to correct and incorrect detection, respectively. Each colored box also displays the detected class and associated confidence score. Longer input sequences help to reduce false-positive detection which are prone to occur in the presence of ambiguous visual cues (*e.g.*, confusion between *sit* & *stand*, or *catch* & *shoot_ball*.).

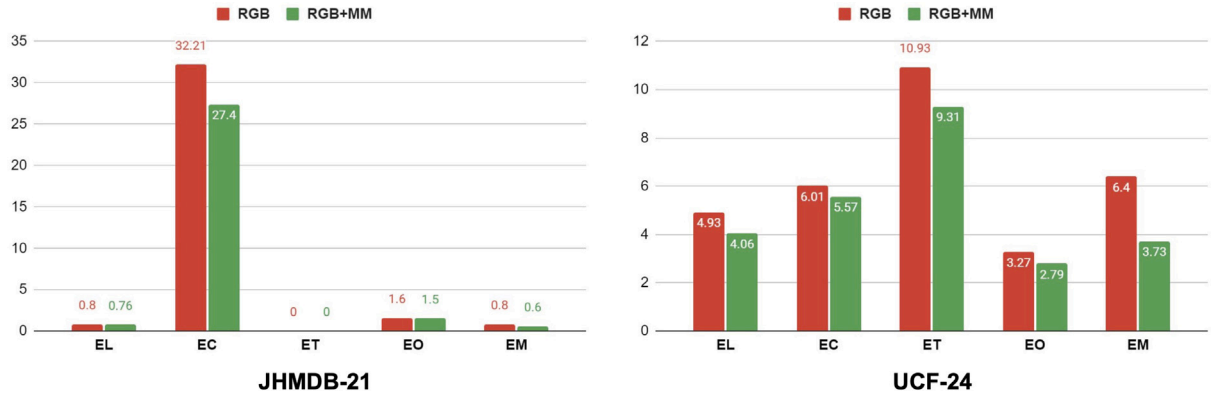

Fig. 7. Frame-mAP error distribution before and after micro-motion ("MM") feature fusion (JHMDB-21 and UCF-24). Each bar corresponds to a specific type of error (out of five error categories). A model is more accurate when incurring lower error rates.

stages of lateral connections (when fusion is applied). To approximate our model's complexity under the streaming-video setting, we report the MACs needed for tubelet inference over K clips and then divide it by K. The model size in terms of number of parameters is also recorded. In addition, to verify the necessity of our micro-motion sub-network operating on shallow patterns (expressed as MM_{Conv_Diff}), we implement a simpler micro-motion variant (coined MM_{Diff}) which directly accumulates RGB difference maps to encode motion.

Our experiments show that at minor increase in the model size, fusing MM_{Conv_Diff} features largely enhances AMMA's accuracy from that of only using RGB frames. Relative to the model size, which rises due to adding the micro-motion sub-network and duplicating three early stages of ResNet-18, the elevation is more prominent in the required MACs. We found that the additional operations associated with fusing MM_{Conv_Diff} all take place toward early layers of AMMA's backbone where target tensors still retain large spatial dimensions, resulting in a

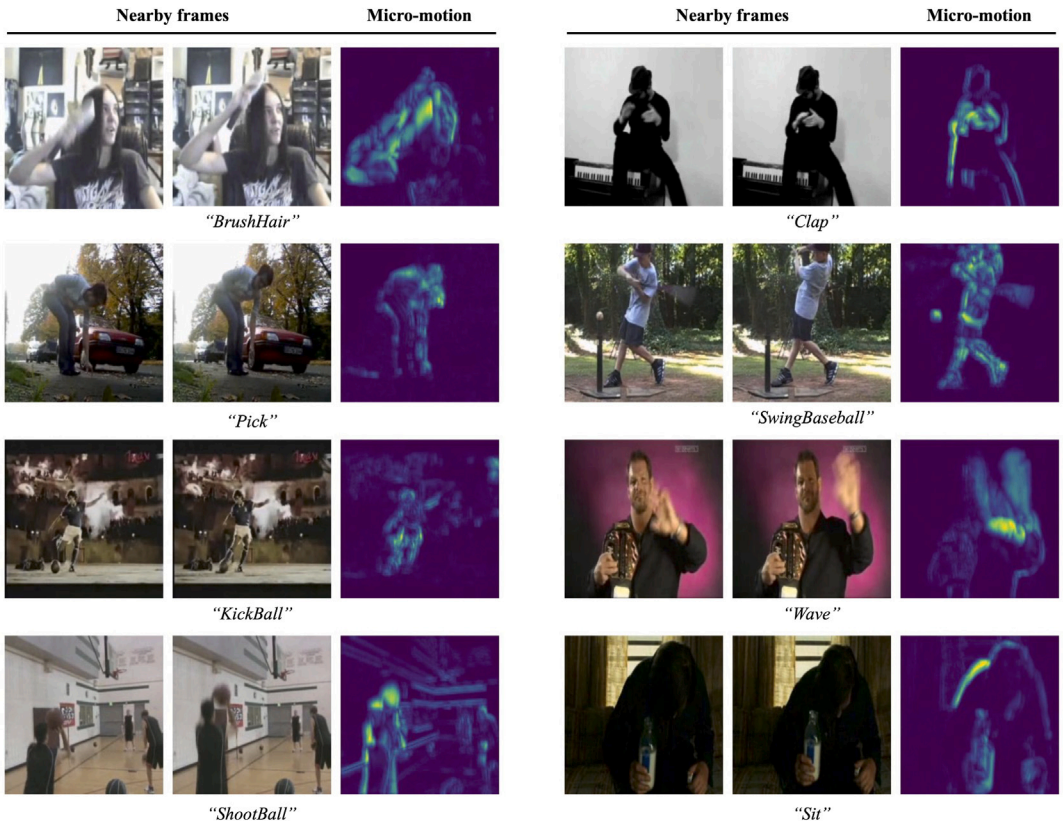

Fig. 8. Visualization of micro-motion cues between pairs of action frames.

Table 1
Performance summary of different forms of micro-motion on JHMDB-21. Input duration is fixed to 5 clips, and three lateral connections are attached to the output of the first three stages in ResNet-18.

	Frame-mAP	GMACs	# params (M)
RGB only	65.08	3.49	15.07
RGB + MM_{Diff}	67.69	5.14	15.75
RGB + MM_{Conv_Diff}	69.74	5.40	15.75

Table 2
Performance summary of varied extents of fusion between appearance and micro-motion features on JHMDB-21. Input duration is fixed to 5 clips.

	Frame-mAP	GMACs	# params (M)
–	65.08	3.49	15.07
Stage 1	66.58	3.95	15.08
Stage 1–2	68.74	4.72	15.23
Stage 1–3	69.74	5.40	15.75
Stage 1–4	70.22	6.08	17.85
Stage 1–5	72.48	6.76	26.24

more noticeable raise in multiply-accumulate operations than in model size. On the other hand, fusing MM_{Conv_Diff} obtains higher accuracy than MM_{Diff} by more than 2 frame-mAP at a negligible increase in GMACs and model size. This suggests that the temporal evolution of general patterns better encodes dynamic information than raw RGB differences, which are more likely to carry local noises. In Fig. 8, we show some examples of our micro-motion representation which successfully captures motion boundaries near moving subjects.

Next, We explore different extents of fusion between appearance and micro-motion information by incrementally increasing the number of lateral connections. As shown in Table 2, the more stages lateral fusion takes place, the more accurate AMMA becomes, indicating that motion boundary features, from shallow to abstracted forms, facilitate detection accuracy. Multi-scale lateral fusion ensures AMMA to simultaneously learn complementary spatiotemporal information throughout the backbone. In exchange for enhanced accuracy, more lateral fusion inevitably raises the model size and computation associated with extracting micro-motion features at deeper layers. To conclude, AMMA's capacity to jointly model actions' visual and dynamic information can be improved when adopting deep fusion. In spite of that, with the aim of keeping an overall efficient detection architecture, we continue leveraging 3 stages of lateral fusion throughout the rest of the experiments.

From lightweight to ultra-lightweight architectures. Ultimately aiming at deploying the detector onto resource-constrained devices, we examine AMMA's generalization ability on ultra-lightweight mobile architectures: MobileNet-V2 and ShuffleNet-V2. Both detection accuracy (*e.g.*, frame-mAP and video-mAP) and model efficiency (*e.g.*, inference speed, model complexity, and size) are assessed. In particular, speed is recorded based on the per-frame processing time of the entire action detection pipeline, *i.e.*, the total runtime of generating action proposals for all videos divided by the total number of their frames.

Integration of micro-motion and lateral fusion in these mobile-friendly architectures closely follows our design in ResNet-18. With MobileNet-V2 as the backbone, we append three lateral connections at the output of the 1st, 3rd, and 6th bottleneck residual block (MobileNet-V2 consists of 17 of these building blocks). For ShuffleNet-V2, three lateral connections are established at the output of "Conv1", "Stage2", and "Stage3" (naming of these layers/blocks follow the same conventions in Ma et al. [42]). The resulting models are represented by AMMA_{18} (ResNet-18), AMMA_{M} (MobileNet-V2), and AMMA_{S} (ShuffleNet-V2) for simplicity. Note that we apply different clip-lengths on the two datasets following their most accurate configurations found in Fig. 5.

Results of the three AMMA variants are reported in Table 3. We observe that AMMA_{18} consistently obtains higher accuracy than the other

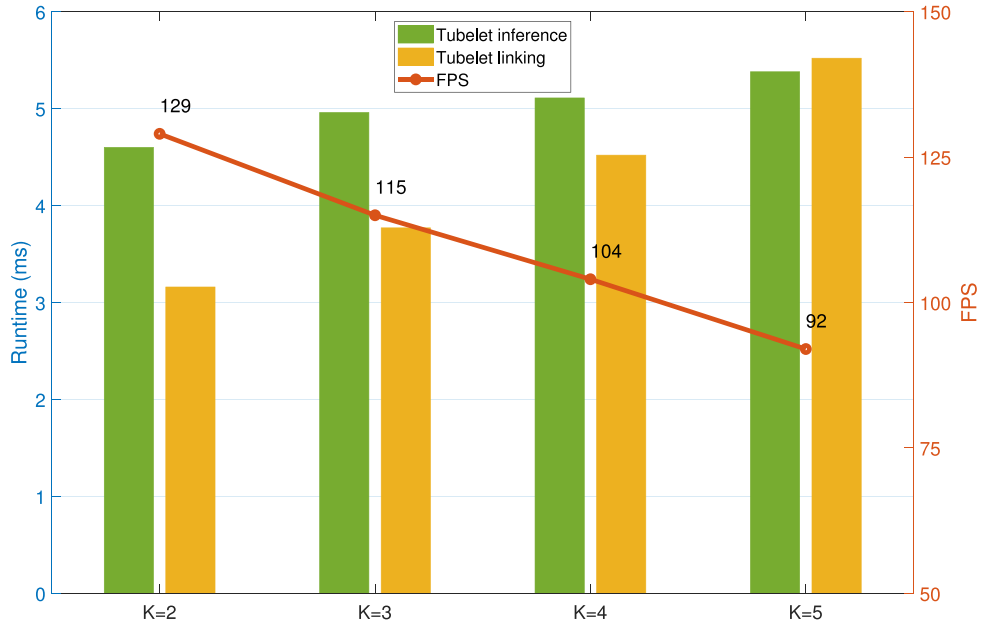

Fig. 9. $AMMA_{18}$'s runtime breakdown over varied sequence length (K) evaluated on UCF-24. The bar graph (referencing Y-axis on the left) captures the decomposition of detection runtime. The red plot (referencing Y-axis on the right) describes the average FPS at each configured K.

Table 3
Performance summary of integrating different 2D CNN backbones.

	JHMDB-21 ($K = 5$)							
	Frame-mAP	Video-mAP @0.2	0.5	0.75	0.5:0.95	GMACs	Param. (M)	FPS
$AMMA_{18}$	69.7	73.7	72.7	60.1	50.3	5.2	15.8	80
$AMMA_{M}$	66.1	70.0	69.0	53.7	45.3	1.3	6.8	77
$AMMA_{S}$	67.7	72.3	70.9	47.9	43.0	1.0	6.0	75
	UCF-24 ($K = 3$)							
	Frame-mAP	Video-mAP @0.2	0.5	0.75	0.5:0.95	GMACs	Param. (M)	FPS
$AMMA_{18}$	74.6	81.1	53.5	24.6	26.3	5.2	15.8	115
$AMMA_{M}$	71.8	78.0	49.7	22.0	23.5	1.3	6.8	110
$AMMA_{S}$	71.3	78.7	47.4	20.9	22.5	1.0	6.0	100

two (especially reflected in video-mAP at high detection thresholds). This is expected as ResNet has higher capacity to extract richer visual context in general than the mobile architectures who prioritize efficiency. Indeed, both datasets consist of actions embedding prominent appearance cues such as *shoot_bow* and *pole_vault* that could benefit from a more powerful feature extractor. In terms of efficiency, the average GMACs of $AMMA_{M}$ and $AMMA_{S}$ are approximately $1/4$ and $1/5$ of that of $AMMA_{18}$ due to their highly optimized architectural design. Similarly, the model size of $AMMA_{M}$ and $AMMA_{S}$ are also significantly smaller.

Countering the above observations, the two ultra-lightweight variants have slightly slower runtime than $AMMA_{18}$ even though their computational cost is substantially lower. On the one hand, this phenomenon has been addressed by Orsic et al. [43], which points out that the implementation of depth-wise separable convolution is not optimized in the cuDNN library (therefore, MobileNet-V2 tends to be slower than ResNet-18 in standard experimental setups). Moreover, computational complexity does not necessarily guarantee faster runtime as GMAC does not take into account factors such as memory access cost and platform characteristics [42]. Further, even though all AMMA models are equipped with three lateral connections, their extents of spatial–temporal fusion still differ according to the architectural designs of their backbone CNNs. For example, ShuffleNet-V2 has more convolutional layers in "Stage3" than those in ResNet-18 to process motion cues. All of our models still exceed real-time performance by a large margin.

One may have noticed AMMA's conspicuous difference in speed between JHMDB-21 and UCF-24, as shown in the last column of Table 3. Such discrepancy is mainly associated with the choice of input sequence length. In Fig. 9, we demonstrate the influence of sequence length ($K = \{2, 3, 4, 5\}$) on $AMMA_{18}$'s runtime based on UCF-24. The average runtime (millisecond, or ms) of a detection cycle can be decomposed into tubelet inference (including feature extraction) and tubelet linking (including intra-frame interpolation); the average FPS is plotted in red. Note that the tubelet inference time is nearly invariant to K, as our detector exploits an efficient feature-caching and retrieval workflow on clip-level features (see Fig. 4). In fact at a larger K, the minor increase in runtime is related to filling AMMA's buffer with K clip features during initialization, as well as a slight increase of computation at the detector head. On the other hand, the runtime associated with tubelet linking prominently rises along sequence length. As tubelet linking depends on calculating the mean IoUs of detection across $K - 1$ overlapping frames, determining whether two tubelets match becomes more computationally demanding when longer tubelets are considered. These results pin-point the importance of a carefully chosen sequence length for balancing AMMA's accuracy and speed performance.

4.3. Comparison with state-of-the-arts

In this section, we evaluate AMMA against several state-of-the-art methods on JHMDB-21 and UCF-24. We emphasize that as our tubelet

Table 4
Comparison with the state-of-the-art methods. Under column "Input", "+OF" indicates applying optical flow as the additional input modality (alongside RGB input).

Method	Input	JHMDB-21					UCF-24				
		F-mAP	Video-mAP				F-mAP	Video-mAP			
			0.2	0.5	0.75	0.5:0.95		0.2	0.5	0.75	0.5:0.95
2D backbone											
Saha et al. [30]	+OF	–	72.6	71.5	43.3	40.0	–	66.7	35.9	7.90	14.4
Peng and Schmid [3]	+OF	58.5	74.3	73.1	–	–	–	73.5	32.1	2.70	7.30
Saha et al. [32]	+OF	–	73.5	72.8	59.7	48.1	–	78.5	49.7	22.2	24.0
Kalogeiton et al. [5]	+OF	65.7	74.2	73.7	52.1	44.8	69.5	76.5	49.2	19.7	23.4
Singh et al. [4]	+OF	–	73.8	72.0	44.5	41.6	–	73.5	46.3	15.0	20.4
Yang et al. [7]	+OF	–	–	–	–	–	75	76.6	–	–	–
Zhao and Snoek [6]	+OF	–	–	74.7	53.3	45.0	–	78.5	50.3	22.2	24.5
Song et al. [44]	+OF	65.5	74.1	73.4	52.5	44.8	72.1	77.5	52.9	21.8	24.1
Zhang et al. [31]	+OF	37.8	–	–	–	–	67.7	74.8	46.6	16.7	21.9
Li et al. [8]	+OF	68.0	76.2	75.4	68.5	54.0	76.9	81.3	54.4	29.5	28.4
Liu et al. [45]	–	64.7	67.9	67.4	53.7	44.7	70.8	74.6	50.4	21.8	25.0
AMMA$_{18}$	–	69.7	73.7	72.7	60.1	50.3	74.6	81.1	53.5	24.6	26.3
AMMA$_{M}$	–	66.1	70.0	69.0	53.7	45.3	71.8	78.0	49.7	22.0	23.5
AMMA$_{S}$	–	67.7	72.3	70.9	47.9	43.0	71.3	78.7	47.4	20.9	22.5
3D backbone											
Hou et al. [10]	–	61.3	78.4	76.9	–	–	67.3	73.1	–	–	–
Gu et al. [11]	–	73.2	–	–	–	–	77.0	–	–	–	–
Qiu et al. [34]	–	–	77.3	74.2	–	–	–	69.3	–	–	–
Li et al. [15]	–	–	84.8	83.7	62.4	51.8	69.7	79.4	62.7	–	25.5
Zhao et al. [12]	–	–	–	79.5	–	58.0	–	–	52.0	–	25.2

detector concurrently seeks competitive accuracy, low complexity, and real-time runtime for practical deployment, only state-of-the-arts with loosely comparable architectures as AMMA are listed in Table 4. Recent top-performing approaches/models that employ much heavier configurations (such as those relying on both 3D CNN and optical flow) are excluded for fair comparison.

It can be observed from Table 4 that AMMA$_{18}$ achieves competitive accuracy on both datasets. Notably, our proposed model utilizes the most lightweight feature backbone than all other methods on the list, such as two-stream VGG16, two-stream DLA-34, C3D, I3D, and S3D, etc. Furthermore, leveraging only RGB frames as input, AMMA$_{18}$ still outperforms most of the other two-stream methods relying on fine-grained optical flow (especially reflected in its frame-mAP and video-mAP at high detection thresholds). Further, AMMA$_{18}$ scores competitively against several 3D CNN-based methods even though ResNet-18 has far less capacity to reason spatiotemporal information, indicating the effectiveness of fusing coarse-scale visuals and complementary dynamic cues at limited computational budgets. Finally for AMMA$_{M}$ and AMMA$_{S}$, due to their CNN backbones being less capable of abstracting visual patterns in exchange for substantially lower computational cost, there remains a perceivable margin from the accuracy of other top-performing detectors.

Beyond competitive accuracy, the evident strength of AMMA lies in its cost-effective architecture and workflow tailored for real-world scenarios and deployment. Specifically, the vast improvement in AMMA's processing efficiency is attributed to its coarse-detection paradigm as well as being free of expensive optical flow extraction. The former not only bypasses redundancy associated with dense per-frame detection, but also facilitates capturing actions' prominent appearance variation over time. Adopting on-the-fly motion cues instead of pre-computed optical flow, AMMA supports detecting actions in an online manner from video streams when exploiting feature-caching and interpolation from coarse-level detection. As shown in Fig. 10, while retaining competitive video-mAP on UCF-24, our models considerably outperform other action detectors reporting real-time or near-real-time performance.

Note that speed performances can be impacted by other elements such as hardware devices and manners of measurement. For example, the delay in generating optical flow was not considered in works such as Kalogeiton et al. [5],Zhao and Snoek [6], and Li et al. [8]. On the other hand, efficiency measures in terms of MACs and model size are independent of the above factors. We refer our readers to Table 3 for such information. To summarize, our most lightweight model (AMMA$_{S}$) incurs 1 GMACs in the online detection setting while requiring 6M parameters. To put these values in perspective, the standard SSD which is widely used in the domain of spatiotemporal action detection such as Singh et al. [4],Kalogeiton et al. [5],Saha et al. [32], and Zhao and Snoek [6], has around 27M trainable parameters (54M in the two-stream setup). Similarly, the two-stream SSD architecture incurs approximately 32 GMACs (with input image size of 300 × 300), which is nearly 32 times more computationally expensive than our lightest model. Methods leveraging 3D CNN [11,13] such as I3D or S3D as the feature backbone, are estimated to exceed 45 and 32 GMACs, respectively.

5. Conclusion and future works

In this paper, we present a lightweight, online action tubelet detector based on 2D CNN (termed AMMA). It makes use of a coarse detection paradigm to efficiently model actions from underlying appearance and variation cues over video sequences. Specifically, AMMA incorporates complementary motion dynamics by accumulating adaptive micro-motion representations generated on-the-fly, facilitating learning appearance-motion correspondences. Our integrated solution conforms to stringent design constraints sought after in many practical application scenarios. As demonstrated in two challenging action benchmarks, AMMA achieves competitive precision (frame and video-mAP) while utilizing significantly more compact backbones and executing at an inference speed far beyond real-time (up to 100 FPS).

In the future, we will evaluate AMMA on more challenging public benchmarks, *e.g.*, **AVA**, which contains more complex scenes and sophisticated action categories. Note that our coarse detection pipeline is designed to smoothly adapt to the sparse training annotations of this dataset. We also attempt to extend AMMA's 2D backbones to ultra-lightweight 3D CNN for enhancing its spatiotemporal modeling capacity. Further, aiming at a fully resource-efficient vision system for deployment, we will also precisely customize AMMA for embedding onto different edge devices such as NVIDIA Jetson TX2 or Xavier GPUs.

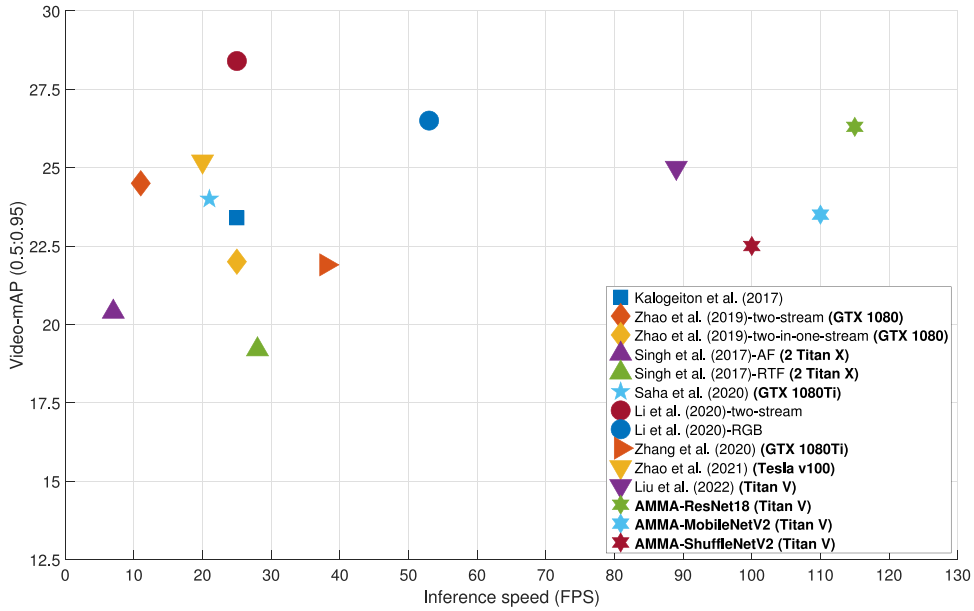

Fig. 10. Comparisons of runtime-accuracy trade-off between AMMA and state-of-the-arts on UCF-24 (video-mAP). "AF" and "RTF" denote accurate flow and real-time flow, respectively. It is note-worthy that methods that depend on externally calculated optical flow typically omit this part of the computation in their runtime measurement.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Codes are available at https://github.com/alphadadajuju/AMMA.

Acknowledgments

This work was supported by the H2020 ITN project ACHIEVE (H2020-MSCA-ITN-2017: agreement no. 765866).

References

[1] X. Hu, J. Dai, M. Li, C. Peng, Y. Li, S. Du, Online human action detection and anticipation in videos: A survey, Neurocomputing 491 (2022) 395–413.
[2] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: NIPS, 2014, pp. 568–576.
[3] X. Peng, C. Schmid, Multi-region two-stream R-CNN for action detection, in: ECCV, Springer, 2016, pp. 744–759.
[4] G. Singh, S. Saha, M. Sapienza, P.H. Torr, F. Cuzzolin, Online real-time multiple spatiotemporal action localisation and prediction, in: IEEE ICCV, 2017, pp. 3637–3646.
[5] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, C. Schmid, Action tubelet detector for spatio-temporal action localization, in: IEEE ICCV, 2017, pp. 4405–4413.
[6] J. Zhao, C.G. Snoek, Dance with flow: Two-in-one stream action detection, in: IEEE CVPR, 2019, pp. 9935–9944.
[7] X. Yang, X. Yang, M.Y. Liu, F. Xiao, L.S. Davis, J. Kautz, Step: Spatio-temporal progressive learning for video action detection, in: IEEE CVPR, 2019, pp. 264–272.
[8] Y. Li, Z. Wang, L. Wang, G. Wu, Actions as moving points, in: European Conference on Computer Vision, Springer, 2020, pp. 68–84.
[9] O. Köpüklü, X. Wei, G. Rigoll, You only watch once: A unified CNN architecture for real-time spatiotemporal action localization, 2019, arXiv preprint arXiv: 1911.06644.
[10] R. Hou, C. Chen, M. Shah, Tube convolutional neural network (t-cnn) for action detection in videos, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5822–5831.
[11] 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: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6047–6056.
[12] J. Zhao, X. Li, C. Liu, S. Bing, H. Chen, C.G. Snoek, J. Tighe, TubeR: Tube-transformer for action detection, 2021, arXiv preprint arXiv:2104.00969.
[13] C. Sun, A. Shrivastava, C. Vondrick, K. Murphy, R. Sukthankar, C. Schmid, Actor-centric relation network, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 318–334.
[14] R. Su, W. Ouyang, L. Zhou, D. Xu, Improving action localization by progressive cross-stream cooperation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 12016–12025.
[15] Y. Li, W. Lin, J. See, N. Xu, S. Xu, K. Yan, C. Yang, CFAD: Coarse-to-fine action detector for spatiotemporal action localization, in: European Conference on Computer Vision, Springer, 2020, pp. 510–527.
[16] X. Zhou, D. Wang, P. Krähenbühl, Objects as points, 2019, arXiv Preprint arXiv:1904.07850.
[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, Ssd: Single shot multibox detector, in: ECCV, Springer, 2016, pp. 21–37.
[18] J. Redmon, A. Farhadi, YOLO9000: better, faster, stronger, in: IEEE CVPR, 2017, pp. 7263–7271.
[19] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: NIPS, 2015, pp. 91–99.
[20] J. Dai, Y. Li, K. He, J. Sun, R-fcn: Object detection via region-based fully convolutional networks, in: NIPS, 2016, pp. 379–387.
[21] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2017.
[22] H. Law, J. Deng, Cornernet: Detecting objects as paired keypoints, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 734–750.
[23] Z. Liu, T. Zheng, G. Xu, Z. Yang, H. Liu, D. Cai, Training-time-friendly network for real-time object detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Volume 34, Number 07, 2020, pp. 11685–11692.
[24] Z. Tian, C. Shen, H. Chen, T. He, Fcos: A simple and strong anchor-free object detector, IEEE Trans. Pattern Anal. Mach. Intell. (2020).
[25] Y. Xie, J. Zheng, X. Hou, Y. Xi, F. Tian, Dynamic dual-peak network: A real-time human detection network in crowded scenes, J. Vis. Commun. Image Represent. 79 (2021) 103195.
[26] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, L. Wang, Tea: Temporal excitation and aggregation for action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 909–918.
[27] J. Yang, Y. Huang, Z. Shao, C. Liu, Learning discriminative motion feature for enhancing multi-modal action recognition, J. Vis. Commun. Image Represent. 79 (2021) 103263.
[28] M. Suneetha, M. Prasad, P. Kishore, Multi-view motion modelled deep attention networks (M2DA-Net) for video based sign language recognition, J. Vis. Commun. Image Represent. 78 (2021) 103161.
[29] L. Sevilla-Lara, Y. Liao, F. Güney, V. Jampani, A. Geiger, M.J. Black, On the integration of optical flow and action recognition, in: German Conference on Pattern Recognition, Springer, 2018, pp. 281–297.
[30] S. Saha, G. Singh, M. Sapienza, P.H.S. Torr, F. Cuzzolin, Deep learning for detecting multiple space-time action tubes in videos. CoRR abs/1608.01529, 2016,

[31] D. Zhang, L. He, Z. Tu, S. Zhang, F. Han, B. Yang, Learning motion representation for real-time spatio-temporal action localization, Pattern Recognit. 103 (2020) 107312.
[32] S. Saha, G. Singh, F. Cuzzolin, Two-stream amtnet for action detection, CoRR abs/2004.01494, 2020.
[33] J. Carreira, A. Zisserman, Quo vadis, action recognition? a new model and the kinetics dataset, in: IEEE CVPR, 2017, pp. 6299–6308.
[34] Z. Qiu, T. Yao, C.-W. Ngo, X. Tian, T. Mei, Learning spatio-temporal representation with local and global diffusion, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 12056–12065.
[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[36] R. Girdhar, J. Carreira, C. Doersch, A. Zisserman, Video action transformer network, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 244–253.
[37] C. Zhang, Y. Zou, G. Chen, L. Gan, PAN: Persistent appearance network with an efficient motion cue for fast action recognition, in: Proceedings of the 27th ACM International Conference on Multimedia, 2019, pp. 500–509.
[38] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[39] K. Soomro, A.R. Zamir, M. Shah, UCF101: A dataset of 101 human actions classes from videos in the wild, 2012, arXiv preprint arXiv:1212.0402.
[40] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, M.J. Black, Towards understanding action recognition, in: IEEE ICCV, 2013, pp. 3192–3199.
[41] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.
[42] N. Ma, X. Zhang, H.-T. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 116–131.
[43] M. Orsic, I. Kreso, P. Bevandic, S. Segvic, In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 12607–12616.
[44] L. Song, S. Zhang, G. Yu, H. Sun, Tacnet: Transition-aware context network for spatio-temporal action detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11987–11995.
[45] Y. Liu, F. Yang, D. Ginhac, TEDdet: Temporal feature exchange and difference network for online real-time action detection, IEEE Access 10 (2022) 37870–37881.