KORSAL: Key-point Detection based Online Real-Time Spatio-Temporal Action Localization

Kalana Abeywardena, Shechem Sumanthiran, Sakuna Jayasundara, Sachira Karunasena, Ranga Rodrigo, and Peshala Jayasekara

***Abstract*—Real-time and online action localization in a video is a critical yet highly challenging problem. Accurate action localization requires utilization of both temporal and spatial information. Recent attempts achieve this by using computationally intensive 3D CNN architectures or highly redundant two-stream architectures with optical flow, making them both unsuitable for real-time, online applications. To accomplish activity localization under highly challenging real-time constraints, we propose utilizing fast and efficient key-point based bounding box prediction to spatially localize actions. We then introduce a tube-linking algorithm that maintains the continuity of action tubes temporally in the presence of occlusions. Further, we eliminate the need for a two-stream architecture by combining temporal and spatial information into a cascaded input to a single network, allowing the network to learn from both types of information. Temporal information is efficiently extracted using a structural similarity index map as opposed to computationally intensive optical flow. Despite the simplicity of our approach, our lightweight end-to-end architecture achieves state-of-the-art frame-mAP of *74.7%* on the challenging UCF101-24 dataset, demonstrating a performance gain of *6.4%* over the previous best online methods. We also achieve state-of-the-art video-mAP results compared to both online and offline methods. Moreover, our model achieves a frame rate of *41.8* FPS, which is a *10.7%* improvement over contemporary real-time methods.**

***Index Terms*—Action Localization, Spatio-Temporal, Online, Real-time**

I. Introduction

Spatio-temporal (ST) action localization is the task of classifying an action being performed in a series of video frames while localizing it in both space and time. Action localization is highly challenging when performed online in real-time such as in autonomous driving [1], [2]. Actions must be localized without having access to future information to be used in online settings, while each video frame should be processed individually without using a buffer of frames in order to achieve real-time performance.

Action localization has been performed mainly through traditional methods (e.g., dense trajectories) [3], [4] and deep learning-based methods using Convolutional Neural Networks (CNNs) [1], [2], [5], [6]. Due to the robust feature learning nature—as opposed to hand-crafted features in traditional methods—, CNN-based deep-learning methods for ST action localization have surpassed the traditional methods both in terms of accuracy and efficiency.

CNN-based deep learning methods for ST action localization use two approaches: using 3D CNN frameworks that process a video as a 3D block of pixels [7]; or using popular object detectors in two-stream 2D CNN frameworks with temporal linking of frame-wise detections [8]. While superior results may be obtained by processing the entire video at once, it is impossible to use these systems for online applications. Contemporary work that focuses on online and real-time deployment [9] temporally link detections using an algorithm that is unable to maintain the continuity of action tubes online. They also use 2-stream architectures and require expensive computation of optical flow (OF), which only marginally improves performance, but at the cost of real-time inference speed.

In this paper, we propose an online, real-time ST action localization network by utilizing efficient key-point detection [10] for spatial action localization, doing away with traditional anchor-box based detection architectures that require manual hyper-parameter tuning and extensive post-processing [11], [12]. We do not rely on computationally intensive OF to extract explicit temporal information, rather we introduce an efficient scheme to obtain sufficient temporal information by using the structural similarity (SSIM) index map between two consecutive frames. We also demonstrate that the two-stream architecture is highly redundant, and superior results can be obtained by using a single network allowing it to learn only the *required* temporal and spatial features for action localization.

Further, we introduce an improved tube-linking algorithm that leverages only past information. It extrapolates tubes for a short period in the absence of suitable detections from the object detector. Using [9] as a benchmark, our proposed architecture achieves state-of-the-art frame-mean average precision (f-mAP) and video-mAP (v-mAP) for real-time, online ST action localization on UCF101-24 and J-HMDB-21 while retaining real-time speeds. In summary, we make the following contributions: noitemsep

- We utilize key-point based detection architecture for the first time for the task of ST action localization, which reduces model complexity and inference time over traditional anchor-box based approaches.
- We demonstrate that the explicit computation of OF is unnecessary, and that the SSIM index map obtains

All authors are with the Department of Electronic and Telecommunication Engineering, University of Moratuwa, Sri Lanka (email: kalanag@uom.lk, shechems@uom.lk, sakuna@ieee.org, sachirarkarunasena@gmail.com, ranga@uom.lk, peshala@uom.lk).

This work is supported by Accelerating Higher Education Expansion and Development (AHEAD) project.

sufficient inter-frame temporal information.

- We show that the highly redundant two-stream architecture is unnecessary by providing a single network with both spatial and temporal information, and allowing it to extract necessary information through discriminative learning.
- We introduce an efficient tube-linking algorithm that extrapolates the tubes for a short period of time using past detections for real-time deployment.

II. Related Works

A. Spatio-temporal action localization

There are two approaches to ST action localization: 3D video processing, which processes either a sequence of frames or the entire video at once, and frame-based linking techniques, which attempt to spatially localize actions within a frame, and then link those actions in the temporal domain.

Traditional 3D video processing approaches include 3D sub-volume methods such as ST template matching [4], a 3D boosting cascade [13], and ST deformable 3D parts [14]. Recently, these have been outperformed by the 3D CNNs [7] that process the videos as clips in an offline fashion and localize the action in time and space. While 3D methods are able to produce good results, they inherently suffer from being highly computationally expensive, making them unsuitable for real-time applications.

Alternatively, ST action localization can be achieved by maximizing a temporal classification path of 2D boxes detected on static frames [5], [6], [15], [16], or by searching for the optimal classification result with a branch and bound scheme [17]. Recent works use existing 2D CNN object detection architectures to localize actions spatially and linking them temporally. To capture temporal dependencies when producing the bounding boxes, two-stream architectures have been introduced to process both the video frame and the OF concurrently, offline [1], [2], [5]. This method requires running two independent CNNs in parallel, leading to a two-fold increase in resource consumption. Further, it underestimates the ability of a neural network to combine spatial and temporal information to extract required semantic information through discriminative learning. Each network is restricted to spatial or temporal information only, not allowing for productive combination of both types of representations. We demonstrate that such a separation is redundant, and better performance can be achieved by using a single network that integrates both types of information.

Almost all mentioned 2D approaches rely on computationally expensive OF to extract temporal information between frames [1], [5], [8], [18]. While OF provides very accurate motion estimation [19], there is no reason to believe that such fine-grain temporal information is necessary or even useful for a CNN to learn required temporal representations from a scene. We introduce a simple, efficient alternative to OF to obtain inter-frame temporal information – the SSIM index map (SS-map) – leading to increased speed, reduced complexity and improved performance, as demonstrated by our results.

Among offline action localization methods, there is limited work that focuses on real-time action detection and classification [8], [18]. However, these approaches employ large, inefficient object detection algorithms and slow OF techniques [18]. Further, they utilize future information and cannot be used for online applications [8]. Work on online real-time ST action localization is very limited [9]. A two-stream architecture that takes video frames and traditional OF as inputs to a standard CNN object detection algorithm and fuses detections prior to real-time tube generation is utilized by [9]. However, the employed linking algorithm interpolates between disjoint sets of detections that are close in time, assuming that the object detector has missed the detections in between. Therefore, it is unable to maintain tube continuity in real-time. We address this shortcoming by extrapolating forward for short periods of time, to improve tube continuity for online applications.

B. Key-point detection for localization

Although single-stage object detectors are common in ST action detection, key-point detection based action localization has not been used. Anchor-based single-stage object detectors [11], [20] have a high computational complexity due to the large number of anchor box proposals. Further, the heuristic design of these proposals introduces many design choices, which is undesirable. Recently, key-point based object detection has proven to produce competitive results [10], [21], [22]. Detection and encoding of the top-right and bottom-left corners, followed by a matching algorithm was introduced in [21]. Some works detect more key points per object: both corners and the center point are localized in [22]. However, the performance of such algorithms suffer from having to match sets of key-points. Single key-point detection offers the best performance with reduced complexity by detecting the centers of the object and regressing the size of the bounding box [10]. Our real-time algorithm requires a simple and fast approach to localize actions in space. Therefore, to the best of our knowledge, we are the first to leverage the benefits of key-point based detection for the task of spatial action localization.

III. Proposed Architecture

We propose a novel end-to-end action localization scheme by leveraging innovative temporal information incorporation, state-of-the-art key-point based object detection, and a simple, intuitive linking algorithm. A summary of our scheme is as follows. To localize the actions occurring in a given frame of a video sequence, we utilize only that frame and the frame immediately preceding it. We shift the current frame by one pixel in all directions and use the SSIM index to determine the shifted frame which is most similar to the previous frame. We then concatenate the current frame and the full multi-channel SS-map between the selected shifted frame and the previous frame along the channel-axis. This is used as the input for feature extraction, for which we use the DLA-34 [23] backbone followed by CenterNet object detection head [10]

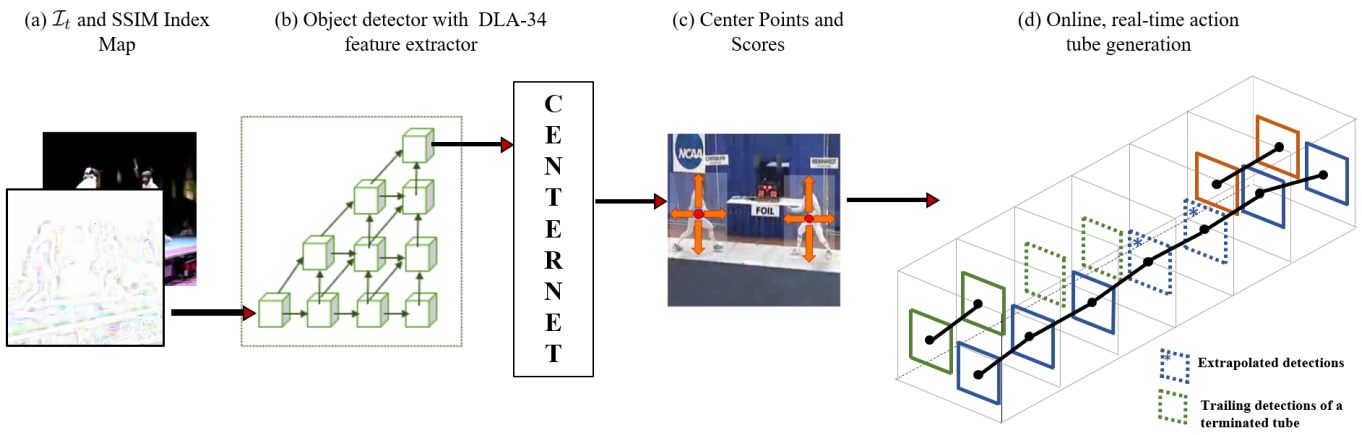


Fig. 1: a) Input is $\mathcal{I}_t$ and the SS-map between $\mathcal{I}_{t-1}$ and $\mathcal{I}_t^*$ (b) DLA-34 used for feature extraction and CenterNet used to create heatmaps (§ III-B). (c) Key-points are used to build bounding boxes. (d) Acion tubes built using detections and extrapolated when required (§ III-C).

to localize activities in the given frame using detected key-points. The bounding boxes for each action are then passed into the linking algorithm which incrementally links them to previously existing actions to form action tubes. The overall architecture is presented in Fig. 1.

A. Temporal information representation

We obtain a representation of temporal information between consecutive frames using the following 2-step procedure. Fig. 2 demonstrates the extraction method.

Small motion candidate selection: Let the current frame be denoted by $\mathcal{I}_t$, and the previous frame be denoted by $\mathcal{I}_{t-1}$. To compensate for any camera motion, we shift $\mathcal{I}_t$ by one pixel in all 8 possible directions to obtain $\{\mathcal{I}_t^1, \ldots, \mathcal{I}_t^8\}$. Then, from the candidates $\{\mathcal{I}_t, \mathcal{I}_t^1, \ldots, \mathcal{I}_t^8\}$, the candidate that is most similar to $\mathcal{I}_{t-1}$ in the SSIM sense is denoted as $\mathcal{I}_t^*$.

With a sufficient frame rate, the camera motion between two consecutive frames is assumed to be small enough such that a single pixel shift provides a simple and efficient way to warp the current image to the previous image.

SS-map extraction: Obtaining accurate OF is computationally prohibitive for real-time applications. To capture the temporal information, we replace OF with the SS-map. The SSIM index between two images, a and b, is defined as [24]

$$\mathcal{S}(a,b) = \left(\frac{2\mu_a\mu_b + C_1}{\mu_a^2 + \mu_b^2 + C_1}\right)\left(\frac{2\sigma_{ab} + C_2}{\sigma_a^2 + \sigma_b^2 + C_2}\right) \quad (1)$$

where μ and σ refer to the sample mean and sample variance. C_1 and C_2 are small constants used to ensure stability. In order to account for local variations of structure, following [25] we apply Eq. (1) over local image patches of size 7×7. This produces the SS-map.

The SS-map between $\mathcal{I}_t^*$ and $\mathcal{I}_{t-1}$ is computed using Eq. (1) which enables us to extract relevant temporal information of the objects of interest in the scene. The input to the feature extractor is the concatenated SS-map and $\mathcal{I}_t$, allowing the network access to both spatial and short term temporal information.

B. Spatial localization of actions using key-points

We use a key-point based single-stage 2D CNN object detection network to localize actions within a frame. The network is end-to-end trainable and follows the architecture presented in [10]. For a $W \times H \times C$ input, the detection network outputs a $\frac{W}{R} \times \frac{H}{R} \times N$ heatmap, where N is the number of action classes, and R is the down-sampling ratio. The heatmaps indicate the likelihood of the occurrence of center points of actions at each location. Additionally, the network also outputs a $\frac{W}{R} \times \frac{H}{R} \times 2$ map to indicate the width and height of actions located at each point, and a $\frac{W}{R} \times \frac{H}{R} \times 2$ map which indicates an offset from the down-sampled map to the actual center of the action bounding box. This leads to a greatly simplified approach over traditional anchor box-based detection architectures with reduced model complexity and inference time, which are critical for real-time applications.

C. Online action-tube generation

We present a tube-linking algorithm that matches frame-level detections obtained at time t, $\mathcal{D}^t = \{D_1^t, D_2^t \ldots\}$, to existing action tubes generated based on detections upto time $t-1$, $\mathcal{T}^{t-1} = \{T_1^t, T_2^t, \ldots\}$. A detection D_i^t has a bounding box $b_{D_i}^t$ and action class scores $\boldsymbol{s}_{D_i}^t \in \mathbb{R}^{C\times 1}$. D_i^t can be assigned to a pre-existing tube T_j^{t-1} of class c_{T_j} and score $s_{T_j}^{t-1}$ given that it has been assigned to no other tube, and $b_{D_i}^t$ has a minimum spatial overlap λ with the most recent bounding box $b_{T_j}^{t-1}$ in the tube. From the set of possible matches, similar to [9], the linking algorithm greedily selects the best match for an action tube. Our algorithm allows unassigned detections to spawn new action tubes, and it extrapolates unassigned tubes for a maximum of k frames. This ensures that tubes may begin at any point, and are not terminated due to occasional false-negative detections.

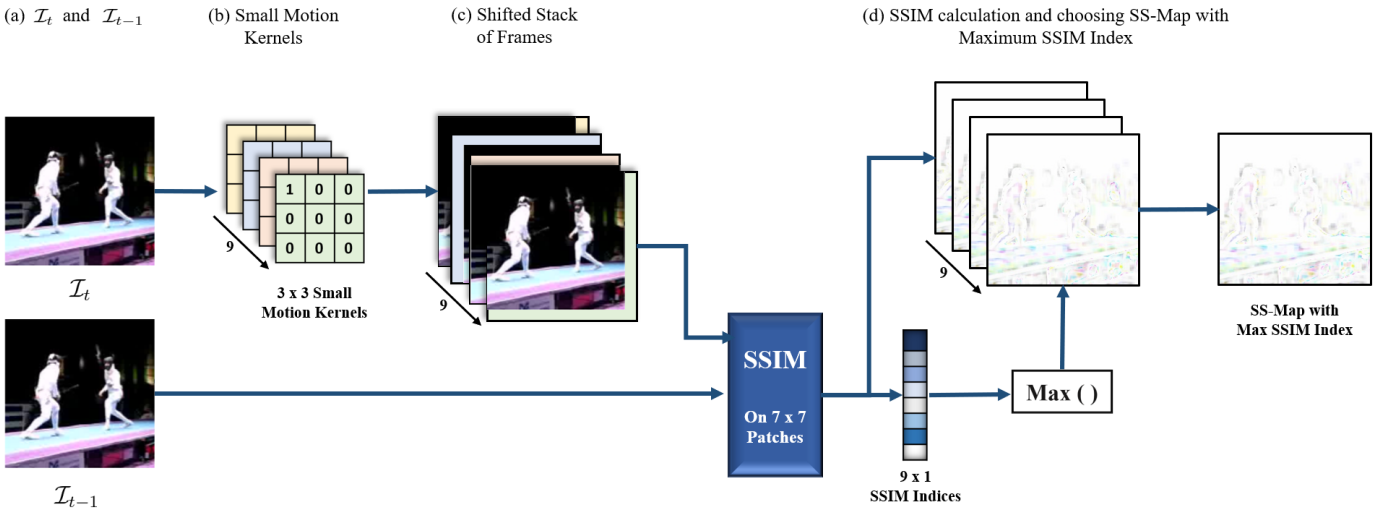


Fig. 2: Obtaining temporal information using SS-map

Tube generating algorithm: At time $t = 1$, we select the top n frame-level detections for a class c by using non-maximum suppression (NMS), and each selected detection initializes an action tube of class c. At every time t thereafter, having obtained the current frame-level detections $\mathcal{D}^t$, the tubes are processed in descending order of their mean score. For each tube T_j^{t-1}, we match the detection with the highest score for the tube's class c_{T_j} that has a minimum intersection-over-union (IoU) ratio of λ. The tube's class is updated based on the energy maximization optimization used in [9]. Once this detection has been assigned, it can no longer be assigned to any other tubes.

Bounding box extrapolation: The presence of the partial occlusions and jitter in the video frames at time t can cause missed detections. Since the tube generation is performed incrementally, this may result in a discontinuity in action tubes. Therefore, if no suitable matches are found for an action tube, we maintain it for a maximum of k time steps by assigning the same class confidence score as the most recent detection.

We also investigate the prediction of the extrapolated bounding box location $b_{T_j}^t$ for the tube at time t without an assigned detection using a simple motion prediction scheme. Although this is a natural extension to extrapolation, our results do not indicate any significant performance increase, possibly due to noisy initial detections and the simplistic approach that we use.

Algorithm 1: Online tube generation

Input: $\mathcal{T}^{t-1}$, $\mathcal{D}^t$, c, λ, k
Output: $\mathcal{T}^t$
for $T_j^{t-1} \in \mathcal{T}^{t-1}$ **do**
  $s \leftarrow 0$; $m \leftarrow 0$;
  for $D_i^t \in \mathcal{D}^t$ **do**
    if $\texttt{IoU}(b_{D_i}^t, b_{T_j}^{t-1}) \geq \lambda$ **and** $s < \boldsymbol{s}_{D_i}^t(c)$ **then**
      $b_{T_j}^t \leftarrow b_{D_i}^t$; $\tau \leftarrow 0$ $s \leftarrow \boldsymbol{s}_{D_i}^t$; $m \leftarrow i$;
  end
  if $m = 0$ **and** $\tau < k$ **then**
    if $box\_pred = True$ **then** $b_{T_j}^t \leftarrow$ $\texttt{predict\_bbox}(b_{T_j}^{t-1}, b_{T_j}^{t-2})$;
    else $b_{T_j}^t \leftarrow b_{T_j}^{t-1}$;
    $\tau \leftarrow \tau + 1$;
  end
  $s_{T_j}^t, c_{T_j} \leftarrow \texttt{update\_label}(s_{T_j}^{t-1}, \boldsymbol{s}_{D_m}^t)$;
end

IV. Experimental Results

We describe our experimental results and compare them with state-of-the-art offline and online methods that use either RGB or both RGB and OF inputs (§ IV-A). Further, for comparison we present results on action localization using only the appearance (A) information extracted by a single frame. The results of our proposed method presented in Table I and Table II demonstrate that we are able to achieve state-of-the-art performance.

Datasets: We evaluate our framework on two datasets, UCF101-24 and J-HMDB-21. **UCF101-24** is a subset of UCF101 [26] with ST labels, having 3207 untrimmed videos with 24 action classes, that may contain multiple instances for the same action class. **J-HMDB-21** is a subset of the HMDB-51 dataset [27] having 928 temporally trimmed videos with 21 actions, each containing a single action instance.

Implementation: Our keypoint-based detector is pretrained on MSCOCO object detection dataset [28]. We use an input image size of 256×256 for increased inference speed, and train the model for 150K iterations with a batch size of 8 on a single NVIDIA RTX 2080 Ti GPU. For the feature extractor to be robust to slight variations, instead of selecting the candidate with the highest SSIM during training, we randomly select one of the top 3 candidates.

Evaluation Metrics: We evaluate our model on spatial action localization by using f-mAP with an IoU threshold of 0.5. We also report the v-mAP at several IoU thresholds to compare the performance of our model on ST action localization with the

(a) Dense OF (b) Fast OF (c) SS-map using the selected candidate

Fig. 3: Comparison between motion information extraction methods.

TABLE I: ST action localization results (v-mAP) on UCF-101-24 dataset. Last two columns compare the f-mAP and FPS.

Method	v-mAP				f-mAP	FPS
	0.2	0.5	0.75	0.5:0.95	@0.5	
Saha *et al.* [1]$^{\diamond}$	66.6	36.4	7.9	14.4	-	4
Peng(w/ MR) *et al.* [2]$^{\diamond}$	72.9	-	-	-	65.7	-
Zhang *et al.* [8]$^{\diamond\star}$	74.8	46.6	16.7	21.9	67.7	37.8
ROAD (w/ AF) [9]‡	73.5	46.3	15.0	20.4	-	7
ROAD (w/ RTF) [9]‡*	70.2	43.0	14.5	19.2	-	28
Our (A+AF)‡	72.9	**46.7**	**16.2**	**20.9**	**70.8**	7.7
Our (A+RTF)‡*	69.6	42.1	15.5	19.3	69.6	**37.9**
ROAD (A) [9]†*	69.8	40.9	15.5	18.7	-	40
Ours (A)†*	**70.2**	**44.3**	**16.6**	**20.6**	71.8	**52.9**
Ours†*	**72.7**	**43.1**	**16.8**	**20.2**	**74.7**	**41.8**

$^{\diamond}$ Offline * Real-time † Online with no OF ‡ Online with OF

TABLE II: ST action localization results (v-mAP) on J-HMDB-21 dataset. Last two columns compare the f-mAP and FPS.

Method	v-mAP				f-mAP	FPS
	0.2	0.5	0.75	0.5:0.95	@0.5	
Saha *et al.* [1]$^{\diamond}$	72.6	71.5	43.3	40.0	-	4
Peng(w/ MR) *et al.* [2]$^{\diamond}$	74.3	73.1	-	-	58.5	-
Zhang *et al.* [8]$^{\diamond\star}$	-	-	-	-	37.4	37.8
ROAD (w/ AF) [9]‡	70.8	70.1	43.7	39.7	-	7
ROAD (w/ RTF) [9]‡*	66.0	63.9	35.1	34.4	-	28
Our (A+AF)‡	68.8	67.6	**49.9**	**43.7**	46.9	7.7
ROAD (A) [9]†*	60.8	59.7	37.5	33.9	-	40
Ours (A)†*	59.3	59.2	**48.2**	**41.2**	**51.2**	**52.9**
Ours†*	**58.9**	**58.4**	**49.5**	**40.6**	**50.5**	**41.8**

$^{\diamond}$ Offline * Real-time † Online with no OF ‡ Online with OF

state-of-the-art results. Both metrics are computed as defined in [6].

A. Comparison with the state-of-the-art

Fig. 3 compares the SS-map against the optical flow obtained by [29] and [30] on a sample of the UCF101-24 [26] dataset. The visualizations illustrates the SS-map captures not only motion information but also variations due to the other factors such as lighting and structural differences. Thus, the SS-map is rich with information about the relative differences between two consecutive frames compared to optical flow while being efficient.

Results on UCF101-24 are reported in Table I. Our method surpasses the benchmark [9] with a marked improvement at high IoU thresholds even with the fusion of RGB and OF for the online and real-time localization of the actions while maintaining an inference speed of 41.8 FPS. Our method achieved an f-mAP score of 74.7%, which is an improvement of 6.4% over any other work in Table I.

On J-HMDB-21, our proposed architecture outperforms all the other online, real-time models, [9] and offline models [1], [2] by a large margin at high IoU thresholds. However, the proposed method was not able to match the results produced by our own (A) only method at the low thresholds. As J-HMDB-21 is a smaller dataset with only 40 frames per video, the model is unable to generalize to extract motion features from cascaded frame inputs, which might be solved by pretraining on a large action detection dataset. This contrasts with the larger UCF101-24, for which the proposed method outperformed all others. Both our methods obtained competitive accuracy with the fastest detection speed of 41.8 FPS. Additionally, our method achieves state-of-the-art results at higher IoU threshold values when compared to other offline competitors.

Unlike [7], [8] that combine future information with causal information by processing through costly 3D CNN architectures or by using optical flow, our model is resource efficient while achieving state-of-the-art results for action detection using only causal information in a real-time manner. The inclusion of temporal information using SS-map as the candidate to the cascaded input has improved the performance for the UCF101-24 dataset both in terms of v-mAP as well as f-mAP surpassing the current state-of-the-art [9] with Real-Time OF. Due to the challenge of extracting combined features from the RGB frame and SS-map, our method tends to perform well with large datasets and struggles to improve beyond the appearance-only model of ours but still achieves vert competitive mAP scores.

V. Conclusion and Future Works

In this paper, we proposed a method to solve the challenging problem of *online* and *real-time* spatio-temporal action localization by utilizing simple and efficient key-point based detec-

tion architectures. We further improved upon existing linking algorithms to maintain temporal continuity by extrapolating the future positions of action tubes to compensate for missed detections online. We showed that the pre-computation of OF to capture motion information affects real-time performance, and we integrated temporal and appearance feature extraction into a single network. We demonstrated that our approach is able to run faster and achieve better performance than state-of-the-art methods on the UCF101-24 and J-HMDB-21 datasets.

Further extensions of this work can explore an integrated tube-linking algorithm and faster feature extraction backbones. Moreover, temporally aware feature extraction can also be investigated.

VI. Ablation Studies

We carry out various experiments to determine the impact of each of the introduced components. We analyze the impact that the different sections of the algorithm have on the overall inference time. We investigate the effects of changing the temporal information representation method, introducing extrapolation and bounding box prediction to the linking algorithm, and using an increased frame gap between cascaded inputs.

TABLE III: Inference timing analysis

Framework Module	**Ours**	**A + DSIM**	**A + $\mathcal{I}_{t-1}$**	**A**	**RTF**	**A + AF**
Temporal INFO EXT (ms)	5.0	5.0	-	-	7.0	110.0
Detection network (ms)	16.4	16.4	16.4	16.4	16.4	16.4
Tube generation time (ms)	2.5	2.5	2.5	2.5	3.0	3.0
Overall (ms)	23.9	23.9	18.9	18.9	26.4	129.4

Inference time: We analyze the inference times for different variations of our pipeline based on the different modules in the framework and the overall inference time in Table III. Evidently, any preprocessing will have an impact on the inference time. Thus, the SS-map achieves a balance between the run-time and the accuracy over the other variations in the framework.

Temporal information representation methods: We investigate different representations of temporal information for our proposed model in Table IV and Table V. Apart from SS-map, we evaluate the structural dissimilarity (DSIM) index map [31] using $\mathcal{I}_t^*$ and using $\mathcal{I}_{t-1}$ without any preprocessing as the input along with $\mathcal{I}_t$. Overall, the SS-map outperforms other methods on J-HMDB21. Although the DSIM method yields the best v-mAP on UCF101-24, SS-map provides the best f-mAP results. We propose that using $\mathcal{I}_{t-1}$ achieves lower results as the SS-map provides convenient cues to the network as to which areas it should pay attention to, which is not provided when the raw previous frame is used as the second input.

Analysis on Linking Algorithm Variations: We analyzed the proposed improvements to the linking algorithm in terms of how they affect the overall v-mAP for the two datasets in Table VI and Table VII. EXPLT denotes extrapolation, and BOXP denotes bounding box location prediction. The results indicate that extrapolating detections for a short time improves results by compensating for missed detections. The intuitive idea of bounding box prediction during the extrapolation does not improve the results of the experiments. We therefore maintain detection locations when a tube is extrapolated. This simple scheme proves sufficient to improve performance.

TABLE IV: Variations of temporal information representation (UCF-101-24)

Method	**v-mAP**				**f-mAP**
	0.2	0.5	0.75	0.5:0.95	@0.5
$\mathcal{I}_{t-1}$	71.6	44.1	17.0	20.7	74.4
SS-map	72.4	43.0	16.6	20.2	74.7
DSIM index map	73.4	44.9	16.4	20.7	74.5

TABLE V: Variations of temporal information representation (J-HMDB-21)

Method	**v-mAP**				**f-mAP**
	0.2	0.5	0.75	0.5:0.95	@0.5
$\mathcal{I}_{t-1}$	57.2	55.9	48.1	39.9	47.9
SS-map	58.9	58.4	49.4	40.5	50.5
DSIM index map	56.4	55.9	49.2	39.9	49.9

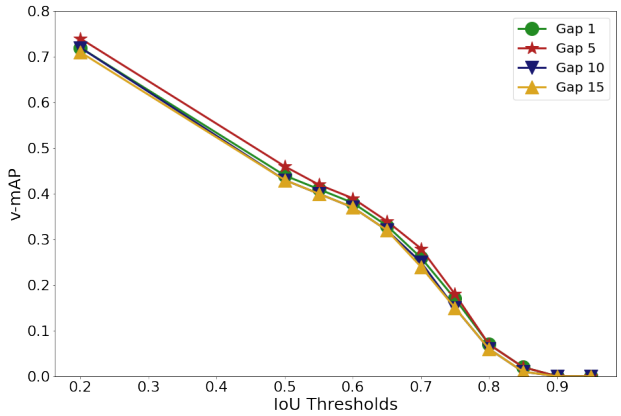
TABLE VI: Linking algorithm variations (UCF-101-24)

Linking Algorithm	**Improvement**		**UCF-101-24**			
	EXPLT	BOXP	**v-mAP**			
			0.2	0.5	0.75	0.5:0.95
Original			72.6	43.4	16.8	20.3
Ours	✓		72.7	43.1	16.8	20.2
Ours	✓	✓	72.4	43.0	16.6	20.2

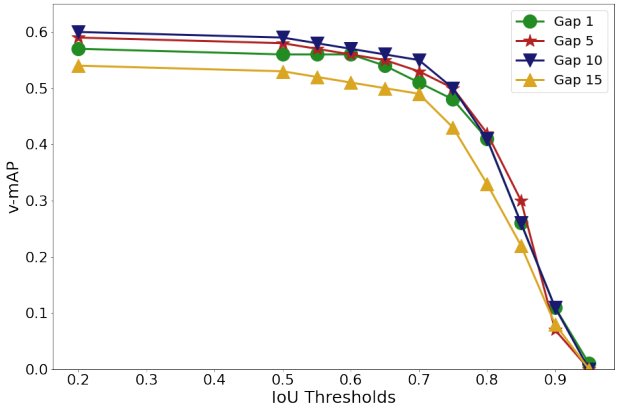
TABLE VII: Linking algorithm variations (J-HMDB-21)

Linking Algorithm	**Improvement**		**J-HMDB-21**			
	EXPLT	BOXP	**v-mAP**			
			0.2	0.5	0.75	0.5:0.95
Original			58.8	58.3	49.4	40.5
Ours	✓		58.9	58.4	49.4	40.6
Ours	✓	✓	58.9	58.4	49.4	40.5

Effect of Frame Gap on Motion: Due to high video frame rates, the difference between two consecutive frames may be negligible, thus containing little temporal information. We analyzed how action localization is impacted when varying the frame gap between the current image and the past image we use to compute the SS-map. For this we used the test setting where the input is $\mathcal{I}_t$ and $\mathcal{I}_{t-k}$, where k is the frame gap we utilize. Based on Fig. 4, obtaining temporal information using consecutive frames is difficult. There is a stronger information between frames which are further separated in time: for UCF24 the best results are obtained at frame gap of 5 and for J-HMDB21 at 10. This indicates that the optimal frame-gap is *data dependent*. However, for both the cases the frame gap of 5 between the current and the past frame



(a) UCF101-24



(b) J-HMDB21

Fig. 4: Analysis of frame gap between the current frame and the past frame utilized.

provides better results than using consecutive frames. We leave the exploration of the frame gap optimization to future work.

REFERENCES

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

[2] X. Peng and C. Schmid, "Multi-region two-stream r-cnn for action detection," in *ECCV*. Springer, 2016, pp. 744–759.

[3] L. R. Villegas, D. Colombet, P. Guiraud, D. Legendre, S. Cazin, and A. Cockx, "Image processing for the experimental investigation of dense dispersed flows: Application to bubbly flows," *Int. J. Multiph. Flow*, vol. 111, pp. 16–30, 2019.

[4] M. D. Rodriguez, J. Ahmed, and M. Shah, "Action mach a spatio-temporal maximum average correlation height filter for action recognition," in *CVPR*. IEEE, 2008, pp. 1–8.

[5] G. Gkioxari and J. Malik, "Finding action tubes," in *CVPR*, 2015, pp. 759–768.

[6] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *ICCV*, 2015, pp. 3164–3172.

[7] O. Köpüklü, X. Wei, and G. Rigoll, "You only watch once: A unified cnn architecture for real-time spatiotemporal action localization," *arXiv preprint arXiv:1911.06644*, 2019.

[8] D. Zhang, L. He, Z. Tu, S. Zhang, F. Han, and B. Yang, "Learning motion representation for real-time spatio-temporal action localization," *Pattern Recognition*, vol. 103, p. 107312, 2020.

[9] G. Singh, S. Saha, M. Sapienza, P. H. Torr, and F. Cuzzolin, "Online real-time multiple spatiotemporal action localisation and prediction," in *ICCV*, 2017, pp. 3637–3646.

[10] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *ECCV*. Springer, 2016, pp. 21–37.

[12] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *CVPR*, 2015, pp. 3367–3375.

[13] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient visual event detection using volumetric features," in *ICCV*, vol. 1. IEEE, 2005, pp. 166–173.

[14] Y. Tian, R. Sukthankar, and M. Shah, "Spatiotemporal deformable part models for action detection," in *CVPR*, 2013, pp. 2642–2649.

[15] D. Tran, J. Yuan, and D. Forsyth, "Video event detection: From subvolume localization to spatiotemporal path search," *TPAMI*, vol. 36, no. 2, pp. 404–416, 2013.

[16] G. Yu and J. Yuan, "Fast action proposals for human action detection and search," in *CVPR*, 2015, pp. 1302–1311.

[17] J. Yuan, Z. Liu, and Y. Wu, "Discriminative video pattern search for efficient action detection," *TPAMI*, vol. 33, no. 9, pp. 1728–1743, 2011.

[18] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time action recognition with enhanced motion vector cnns," in *CVPR*, 2016, pp. 2718–2726.

[19] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[20] A. Farhadi and J. Redmon, "Yolov3: An incremental improvement," *Computer Vision and Pattern Recognition*, 2018.

[21] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *ECCV*, 2018, pp. 734–750.

[22] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *ICCV*, 2019, pp. 6569–6578.

[23] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *CVPR*, 2018, pp. 2403–2412.

[24] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[25] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE signal processing letters*, vol. 9, no. 3, pp. 81–84, 2002.

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

[27] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black, "Towards understanding action recognition," in *International Conf. on Computer Vision (ICCV)*, Dec. 2013, pp. 3192–3199.

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*. Springer, 2014, pp. 740–755.

[29] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *ECCV*. Springer, 2004, pp. 25–36.

[30] T. Kroeger, R. Timofte, D. Dai, and L. Van Gool, "Fast optical flow using dense inverse search," in *ECCV*. Springer, 2016, pp. 471–488.

[31] A. Loza, L. Mihaylova, N. Canagarajah, and D. Bull, "Structural similarity-based object tracking in video sequences," in *2006 9th International Conference on Information Fusion*. IEEE, 2006, pp. 1–6.