You Only Watch Once: A Unified CNN Architecture for Real-Time Spatiotemporal Action Localization

Okan Köpüklü[1,*], Xiangyu Wei[1], Gerhard Rigoll

Institute for Human-Machine Communication, Technical Univ. of Munich, Germany

Abstract

Spatiotemporal action localization requires the incorporation of two sources of information into the designed architecture: (1) temporal information from the previous frames and (2) spatial information from the key frame. Current state-of-the-art approaches usually extract these information with separate networks and use an extra mechanism for fusion to get detections. In this work, we present YOWO, a unified CNN architecture for real-time spatiotemporal action localization in video streams. YOWO is a single-stage architecture with two branches to extract temporal and spatial information concurrently and predict bounding boxes and action probabilities directly from video clips in one evaluation. Since the whole architecture is unified, it can be optimized end-to-end. The YOWO architecture is fast providing 34 frames-per-second on 16-frames input clips and 62 frames-per-second on 8-frames input clips, which is currently the fastest state-of-the-art architecture on spatiotemporal action localization task. Remarkably, YOWO outperforms the previous state-of-the art results on J-HMDB-21 and UCF101-24 with an impressive improvement of ∼3% and ∼12%, respectively. We make our code and pretrained models publicly available[2].

Keywords: spatiotemporal action localization, attention based channel fusion, real-time performance, modular architecture design

*Corresponding author
Email address: okan.kopuklu@tum.de (Okan Köpüklü)
[1]The authors contribute equally to this work.
[2]https://github.com/wei-tim/YOWO

1. Introduction

The topic of spatiotemporal human action localization has been spotlighted in recent years, which aims to not only recognize the occurrence of an action but also localize it in both time and space. In such a task, comparing with object detection in static images, temporal information plays an essential role. Finding an efficient strategy to aggregate spatial as well as temporal features makes the problem even more challenging. On the other hand, real-time human action detection is becoming increasingly crucial in numerous vision applications, such as human-computer interaction (HCI) systems, unmanned aerial vehicle (UAV) monitoring, autonomous driving, and urban security systems. Therefore, it is desirable and worthwhile to explore a more efficient framework to tackle this problem.

Inspired by the remarkable object detection architecture Faster R-CNN [1], most state-of-the-art works [2, 3] extend the classic two-stage network architecture to action detection, where a number of proposals are produced in the first stage, then classification and localization refinement are performed in the second stage. However, these two-stage pipelines have three main shortcomings in the spatiotemporal action localization task. Firstly, the generation of action tubes which consist of bounding boxes across frames is much more complicated and time-consuming than 2D case. The classification performance is extremely dependent on these proposals, where the detected bounding boxes might be sub-optimal for the following classification task. Secondly, the action proposals focus only on the features of humans in the video, neglecting the relationship between humans and some attributes in the background, which yet is able to provide considerably crucial context information for action prediction. The third problem of a two-stage architecture is that training the region proposal network and the classification network separately does not guarantee to find the global optimum. Instead, only local optimum from the combination of two stages can be found. The training cost is also higher than single-stage networks, hence it takes longer time and needs more memory.

Figure 1: Standing or sitting? Although the person can be successfully detected, correct classification of the action cannot be made by looking only at the key frame. Temporal information from previous frames needs to be incorporated in order to understand if the person is sitting (left) or standing (right). Examples are from J-HMDB-21 dataset.

In this paper, we propose a novel single-stage framework, YOWO (You Only Watch Once), for spatiotemporal action localization in videos. YOWO prevents all of the three shortcomings mentioned above with a single-stage architecture. The intuitive idea of YOWO arises from human's visual cognitive system. For example, when we are absorbed into the story of a soap opera in front of the TV, each time our eyes capture a single frame. In order to understand which action each artist is performing, we have to relate current frame information (2D features from key frame) to the obtained knowledge from previous frames saved in our memory (3D features from clip). Afterwards, these two kinds of features are fused together to provide us with a reasonable conclusion. The example in Fig. 1 illustrates our inspiration.

YOWO architecture is a single-stage network with two branches. One branch extracts the spatial features of the key frame (i.e. current frame) via a 2D-CNN while the other branch models the spatiotemporal features of the clip consisting of previous frames via a 3D-CNN. In order to aggregate these features smoothly, a channel fusion and attention mechanism is used, where we get the utmost out of inter-channel dependencies. Finally, we produce frame-level detections using the fused features, and provide a linking algorithm to generate action tubes.

In order to maintain real-time capability, we have operated YOWO on RGB

modality. However, it must be noted that YOWO architecture is not restricted to operate only on RGB modality. Different branches can be inserted into YOWO for different modalities such as optical flow, depth etc. Moreover, in its 2D-CNN and 3D-CNN branches, any CNN architecture can be used according to the desired run-time performance, which is critical for real-world applications.

YOWO operates with maximum 16 frames input since short clip lengths are necessary to achieve faster runtime for spatiotemporal action localization task. However, such small clip size is a limiting factor for the accumulation of temporal information. Therefore, we have made use of the long-term feature bank [4] by extracting features with 3D-CNN for non-overlapping 8-frame clips for the whole videos using the trained 3D-CNN. Training of YOWO performed normally, but at inference time, we have averaged the 3D features centering the key-frame. This brought a considerable 6.9% and 1.3% frame-mAP increase on the final performance of the network.

Contributions of this paper are summarized as follows:

(i) We propose a real-time single-stage framework for spatiotemporal action localization in video streams, named YOWO, which can be trained end-to-end with high efficiency. To the best of our knowledge, this is the first work which achieves bounding box regression on features extracted by a 2D-CNN and 3D-CNN, concurrently. These two kinds of features have a complementary effect to each other for the final bounding box regression and action classification. Moreover, we use a channel attention mechanism to aggregate the features smoothly from two branches above. We experimentally prove that channel-wise attention mechanism models the inter-channel relationship within the concatenated feature maps and boosts the performance significantly by fusing features more reasonably.

(ii) We perform a detailed ablation study on the YOWO architecture. We examined the effect of 3D-CNN, 2D-CNN, their aggregation and the fusion mechanism. Moreover, we have experimented different 3D-CNN architectures and dif-ferent clip lengths to explore a further trade-off between the precision

and speed.

(iii) We evaluate YOWO on J-HMDB-21 and UCF101-24 benchmarks and establish new state-of-the-art results with an impressive 3.3% and 12.2% improvements on frame-mAP, respectively. Moreover, YOWO runs with 34 fps for 16-frames input clips and 62 fps for 8-frames input clips, which is the fastest state-of-the-art architecture available for spatiotemporal action localization task.

2. Related Work

Action recognition with deep learning. Since deep learning brings significant improvements in image recognition, numerous recent research efforts have been devoted to extend it for action recognition in videos. For action recognition, however, besides spatial features extracted from each individual image, temporal context across these frames also needs to be taken into account. Two-stream CNN is one effective strategy to extract spatial and temporal features separately and aggregate them together [5] [6] [7]. Most of these works are based on optical flow, which requires significant computational power to extract, resulting in a time-consuming process. An alternative option to integrate CNN features over time is the implementation of recurrent networks, whose performance, however, is not so satisfying as recent CNN-based methods [8]. 3D-CNNs have been increasingly explored in video analysis tasks recently, which learns the features from both spatial and temporal dimensions simultaneously. 3D-CNN is first exploited to extract spatiotemporal features in [9] and some effective network architectures like C3D [10] and I3D [11] are explored. Inspired by the 2D-CNN residual networks [12], skip connections over layers are also applied to 3D-CNNs to overcome the problem of vanishing gradients [13]. However, 3D-CNN architectures have much more parameters compared to 2D-CNNs, making them computationally expensive. In [14], 3D versions of some famous resource efficient CNN architectures are investigated. For resource efficiency, some other works focus on learning 2D features from single images

with a 2D-CNN and then fusing them together to learn temporal features with a 3D-CNN [15].

Spatiotemporal action localization. For object detection in images, R-CNN series extract region proposals using selective search [16] or RPN [1] in the first stage and classify the objects in these potential regions in the second stage. Although Faster R-CNN [1] achieves state-of-the-art results in object detection, it is hard to implement it for real-time tasks due to its time-consuming two-stage architecture. Meanwhile, YOLO [17] and SSD [18] aim to simplify this process to one stage and have outstanding real-time performance. For action localization in videos, due to the success of R-CNN series most of the research approaches propose first detecting the humans in each frame and then linking these bounding boxes reasonably as action tubes [19, 3, 2]. Two-stream detectors introduce an additional stream on the base of the original classifier for optical flow modality [3] [20] [21]. Some other works produce clip tube proposals with 3D-CNNs and achieve regression as well as classification on the corresponding 3D features [2] [20], thus region proposal is necessary for them. In a recent work [22], authors propose a 3D capsule network for video action detection which can jointly perform pixel-wise action segmentation along with action classification. However, it is too expensive in terms of computational complexity and number of parameters since it is a U-Net [23] based 3D-CNN architecture.

Attention modules. Attention is an effective mechanism to capture long-range dependencies and has been attempted to be used in CNNs to boost the performance in image classification [24] [25] [26] and scene segmentation [27]. Attention mechanism is implemented spatial-wise and channel-wise in these works, in which spatial attention addresses the inter-spatial relationship among features while channel attention enhances the most meaningful channels and weakens the others. As a channel-wise attention block, Squeeze-and-Excitation module [28] is beneficial to increase CNN's performance with little computational cost. On the other hand, for video classification tasks, non-local block [29] takes spatio-temporal information into account to learn the dependencies of features across frames, which can be viewed as a self-attention strategy.

Different from previous works, we have proposed a novel, unified framework called YOWO for the task of spatio-temporal action localization. We name it as YOWO as we make use of a clip only once and detect the corresponding actions in the key frame. However, to avoid the complex optical flow computation, we use 2D features of the key frame and 3D features of the clip together. Afterwards, these two kinds of features are fused together carefully with the application of attention mechanism such that rich contextual relationships are well taken into account.

3. Methodology

In this section, we first present YOWO's architecture in detail, which extracts 2D features from the key frame as well as 3D features from the input clip concurrently and aggregates them together. Afterwards the implementation of channel fusion and attention mechanism is discussed, which provides the essential performance boost. Finally we describe the details of the training process for the YOWO architecture and the improved bounding box linking strategy for generation of action tubes in untrimmed videos.

3.1. YOWO architecture

The YOWO architecture is illustrated in Fig. 2, which can be divided into four major parts: 3D-CNN branch, 2D-CNN branch, CFAM and bounding box regression parts.

3D-CNN Branch Since contextual information is crucial for human action understanding, we utilize 3D-CNN to extract spatiotemporal features. 3D-CNNs are able to capture motion information by applying convolution operation not only in space dimension but also in time dimension. The basic 3D-CNN architecture in our framework is 3D-ResNext-101 due to its high performance in Kinetics dataset [13]. In addition to 3D-ResNext-101, we have also experimented with different 3D-CNN models in our ablation study. For all 3D-CNN architectures, all of the layers after the last conv layer are discarded. The input to the

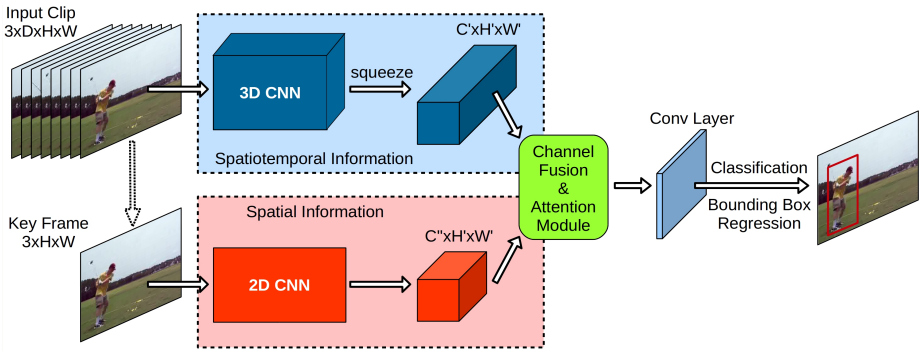

Figure 2: The YOWO architecture. An input clip and corresponding key frame is fed to a 3D-CNN and 2D-CNN to produce output feature volumes of $[C'' \times H' \times W']$ and $[C' \times H' \times W']$, respectively. These output volumes are fed to channel fusion and attention mechanism (CFAM) for a smooth feature aggregation. Finally, one last conv layer is used to adjust the channel number for final bounding box predictions.

3D network is a clip of a video, which is composed of a sequence of successive frames in time order, and has a shape of $[C \times D \times H \times W]$, while the last conv layer of 3D ResNext-101 outputs a feature map of shape $[C' \times D' \times H' \times W']$ where $C = 3$, D is the number of input frames, H and W are height and width of input images, C' is the number of output channels, $D' = 1$, $H' = \frac{H}{32}$ and $W' = \frac{W}{32}$. The depth dimension of the output feature map is reduced to 1 such that output volume is squeezed to $[C' \times H' \times W']$ in order to match the output feature map of 2D-CNN.

2D-CNN Branch In the meantime, to address the spatial localization problem, 2D features of the key frame are also extracted in parallel. We employ Darknet-19 [30] as the basic architecture in our 2D CNN branch due to its good balance between accuracy and efficiency. The key frame with the shape $[C \times H \times W]$ is the most recent frame of the input clip, thus there is no need for an additional data loader. The output feature map of Darknet-19 has a shape of $[C'' \times H' \times W']$ where $C = 3$, C'' is the number of output channels, $H' = \frac{H}{32}$ and $W' = \frac{W}{32}$ similar to the 3D-CNN case.

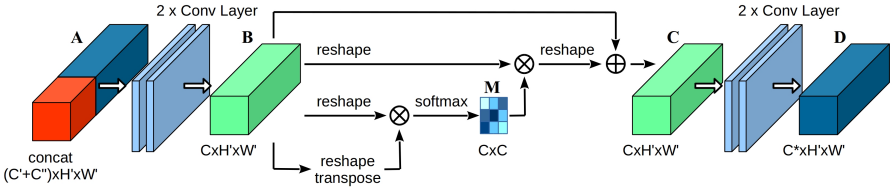

Figure 3: Channel fusion and attention mechanism for aggregating output feature maps coming from 2D-CNN and 3D-CNN branches.

Another important characteristic of YOWO is that architectures in 2D CNN and 3D CNN branches can be replaced by arbitrary CNN architectures, which makes it more flexible. YOWO is designed to be simple and effort-saving to switch models. It must be noted that although YOWO has two branches, it is a unified architecture and can be trained end-to-end.

Feature aggregation: Channel Fusion and Attention Mechanism (CFAM) We make the outputs of both 3D and 2D networks are of the same shape in the last two dimensions such that these two feature maps can be fused easily. We fuse the two feature maps using concatenation which simply stacks the features along channels. As a result, the fused feature map encodes both motion and appearance information which we pass as input to the CFAM module, which is based on Gram matrix to map inter-channel dependencies. Although Gram matrix based attention mechanism is originally used for style transfer [31] and recently in segmentation task [27], such an attention mechanism is beneficial for fusing features coming from different sources reasonably, which improves the overall performance significantly.

Fig. 3 illustrates the used CFAM module. The concatenated feature map $\mathbf{A} \in \mathbb{R}^{(C'+C'')\times H\times W}$ can be regarded as an abrupt combination of 2D and 3D information, which neglects interrelationship between them. Therefore, we first feed A into two convolutional layers to generate a new feature map $\mathbf{B} \in \mathbb{R}^{C\times H'\times W'}$. Afterwards, several operations are performed on the feature map $\mathbf{B}$.

Assume $\mathbf{F} \in \mathbb{R}^{C\times N}$ is the reshaped tensor from feature map $\mathbf{B}$, where

$N = H \times W$, which means that features in every single channel is vectorized to one dimension:

$$\mathbf{B} \in \mathbb{R}^{C \times H \times W} \xrightarrow{vectorization} \mathbf{F} \in \mathbb{R}^{C \times N} \tag{1}$$

Then a matrix product between $\mathbf{F} \in \mathbb{R}^{C \times N}$ and its transpose $\mathbf{F}^{\mathrm{T}} \in \mathbb{R}^{N \times C}$ is performed to produce Gram matrix $\mathbf{G} \in \mathbb{R}^{C \times C}$, which indicates the feature correlations across channels [31]:

$$\mathbf{G} \;=\; \mathbf{F} \cdot \mathbf{F}^{\mathrm{T}} \quad with \quad G_{ij} \;=\; \sum_{k=1}^{N} F_{ik} \cdot F_{jk} \tag{2}$$

where each element G_{ij} in the Gram matrix $\mathbf{G}$ represents the inner product between the vectorized feature maps i and j. After computing the Gram matrix, a softmax layer is applied to generate channel attention map $\mathbf{M} \in \mathbb{R}^{C \times C}$:

$$M_{ij} \;=\; \frac{exp(G_{ij})}{\sum_{j=1}^{C} exp(G_{ij})} \tag{3}$$

where M_{ij} is a score measuring the j^{th} channel's impact on the i^{th} channel. Therefore M summaries the inter-channel dependency of features given a feature map. To perform the impact of attention map to original features, a further matrix multiplication between $\mathbf{M}$ and $\mathbf{F}$ is carried out and the result is reshaped back to 3-dimensional space $\mathbb{R}^{C \times H \times W}$, which has the same shape as the input tensor:

$$\mathbf{F}' \;=\; \mathbf{M} \cdot \mathbf{F} \tag{4}$$

$$\mathbf{F}' \in \mathbb{R}^{C \times N} \xrightarrow{reshape} \mathbf{F}'' \in \mathbb{R}^{C \times H \times W} \tag{5}$$

The output of channel attention module $\mathbf{C} \in \mathbb{R}^{C \times H \times W}$ combines this result with the original input feature map $\mathbf{B}$ with a trainable scalar parameter α using an element-wise sum operation, and α gradually learns a weight from 0:

$$\mathbf{C} \;=\; \alpha \cdot \mathbf{F}'' + \mathbf{B} \tag{6}$$

The Eq. (6) shows that the final feature of each channel is a weighted sum of the features of all channels and original features, which models the long-range semantic dependencies between feature maps. Finally, the feature map

$\mathbf{C} \in \mathbb{R}^{C \times H' \times W'}$ is fed into two more convolutional layers to generate the output feature map $\mathbf{D} \in \mathbb{R}^{C^* \times H' \times W'}$ of the CFAM module. Two convolutional layers at the beginning and the end of CFAM modules contain utmost importance since they help to mix the features coming from different backbones and having possibly different distributions. Without these convolutional layers, CFAM marginally improves the performance.

Such an architecture promotes the feature representativeness in terms of inter-dependencies among channels and thus the features from different branches can be aggregated reasonably and smoothly. Besides, Gram matrix takes the whole feature map into consideration, where the dot product of each two flattened feature vectors presents the information about the relation between them. A larger product indicates that the features in these two channels are more correlated while a smaller product suggests that they are different from each other. For a given channel, we allocate more weights to the other channels which are much correlated and have more impact to it. By means of this mechanism, contextual relationship is emphasized and feature discriminability is enhanced.

Bounding box regression We follow the same guidelines of YOLO [30] for bounding box regression. A final convolutional layer with 1×1 kernels is applied to generate desired number of output channels. For each grid cell in $H' \times W'$, 5 prior anchors are selected by k-means technique on corresponding datasets with *NumCls* class conditional action scores, 4 coordinates and confidence score making the final output size of YOWO $[(5 \times (NumCls + 5)) \times H' \times W']$. The regression of bounding boxes are then refined based on these anchors.

We have used multi-scale training while the resolution of each frame is set to 224×224 at test time. We select the mini-batch stochastic gradient decent algorithm with momentum and weight decay strategy to optimize the loss function, which is defined similar to the original YOLO network [30] except that we apply smooth L_1 loss for localization as in [32] since it is less sensitive to outliers than the L_2 loss and focal loss [33] for classification loss.

3.2. Implementation details

We initialize the 3D and 2D network parameters separately: 3D part with pretrained models on Kinetics [11] and 2D part with pretrained models on PASCAL VOC [34]. Although our architecture consists of 2D-CNN and 3D-CNN branches, the parameters are able to be updated jointly. The learning rate is initialized as 0.0001 and reduced with a factor of 0.5 after 30k, 40k, 50k and 60k iterations. For the dataset UCF101-24, the training process is completed after 5 epochs while for J-HMDB-21 after 10 epochs. The complete architecture is implemented and trained end-to-end in PyTorch using a single Nvidia Titan XP GPU.

In the trainings, because of the small number of samples in J-HMDB-21, we freeze all the 3D conv network parameters thus the convergence is faster and over-fitting risk can be reduced. In addition, for both UCF101-24 as well as J-HMDB-21, we deploy several data augmentation techniques such as flipping images horizontally in the clip, random scaling and random spatial cropping. During testing, only detected bounding boxes with confidence score larger than threshold 0.25 are selected and then post-processed with non-maximum suppression with a threshold of 0.4.

3.3. Linking Strategy

As we have already obtained frame-level action detections, next step is to link these detected bounding boxes to construct action tubes in the whole video. We make use of the linking algorithm described in [19, 3] to find the optimal video-level action detections.

Assume R_t and R_{t+1} are two regions from consecutive frames t and $t+1$, the linking score for an action class c is defined as

$$\begin{aligned} s_c(R_t, R_{t+1}) \;=\; & \psi(ov) \cdot [s_c(R_t) + s_c(R_{t+1}) \\ & + \alpha \cdot s_c(R_t) \cdot s_c(R_{t+1}) \\ & + \beta \cdot ov(R_t, R_{t+1})] \end{aligned} \tag{7}$$

where $s_c(R_t)$, $s_c(R_{t+1})$ are class specific scores of regions R_t and R_{t+1}, ov is the intersection-over-union of these two regions, α and β are scalars. $\psi(ov)$ is a constraint which is equal to 1 if an overlap exists ($ov > 0$), otherwise $\psi(ov)$ is equal to 0. We extend the linking score definition in [3] with an extra element $\alpha \cdot s_c(R_t) \cdot s_c(R_{t+1})$, which takes the dramatic change of scores between two successive frames into account and is able to improve the performance of video detection in experiments. After all the linking scores are computed, Viterbi algorithm is deployed to find the optimal path to generate action tubes.

3.4. Long-Term Feature Bank

Although YOWO's inference is online and causal with small clip size, 16-frame input limits the temporal information required for action understanding. Therefore, we make use of a long-term feature bank (LFB) similar to [4], which contains features coming from 3D backbone at different timestamps. At inference time, 3D features centering the key-frame are averaged and the resulting feature map is used as input to the CFAM block. LFB features are extracted for non-overlapping 8-frame clips using the pretrained 3D ResNeXt-101 backbone. We have used 8 features (if available) centering the key-frame. So, total number of 64 frames are utilized at inference time. Utilization of LFB increases action classification performance similar to difference between clip accuracy and video accuracy in video datasets. However, introduction of LFB makes the resulting architecture non-causal since future 3D features are used at inference time.

4. Experiments

To evaluate YOWO's performance, two popular and challenging action detection datasets, UCF101-24 [35] and J-HMDB-21 [36] are selected. We follow the official evaluation metrics strictly to report the results and compare the performance of our method with the state of the art.

4.1. Datasets and evaluation metrics

UCF101-24 is a subset of UCF101 [35], which is originally an action recognition dataset of realistic action videos. UCF101-24 contains 24 action classes and 3207 videos, for which the corresponding spatiotemporal annotations are provided. In addition, there might be multiple action instances in each video, which have the same class label but different spatial and temporal boundaries. Such a property makes video-level action detection much more challenging. As in previous works, we perform all the experiments on the first split.

J-HMDB-21 is a subset of the HMDB-51 dataset [37] and consists of 928 short videos with 21 action categories in daily life. Each video is well trimmed and has a single action instance across all the frames. We report our experimental results on the first split.

Evaluation metrics: We employ two popular metrics used by the most researches in the region of spatio-temporal action detection to generate convincing evaluations. Following strictly the rule applied by the PASCAL VOC 2012 metric [38], frame-mAP measures the area under the precision-recall curve of the detections for each frame. On the other hand, video-mAP focuses on the action tubes [19]. If the mean per frame intersection-over-union with the ground truth across the frames of the whole video is greater than a threshold and in the meanwhile the action label is correctly predicted, then this detected tube is regarded as a correct instance. Finally, the average precision for each class is computed and the average over all classes is reported.

4.2. Ablation study

3D network, 2D network or both? Depending only on its own, neither 3D-CNN nor 2D-CNN can solve the spatiotemporal localization task independently. However, if they operate simultaneously, there is potential to benefit from one another. Results on comparing the performance of different architectures are reported in Table 1. We first observe that a single 2D network can not provide a satisfying result since it does not take temporal information

Model	UCF101-24	J-HMDB-21
2D	61.6	36.0
3D	70.5	41.5
2D + 3D	73.8	47.1
2D + 3D + CFAM	**79.2**	**64.9**

Table 1: Frame-mAP @ IoU 0.5 results on datasets UCF101-24 and J-HMDB-21 for different models. For all architectures, the input to 3D-CNNs is 8 frames clips with downsampling 1.

into account. A single 3D network is better at capturing motion information and the fusion of 2D and 3D networks (simple concatenation) can improve the performance by around 3% and 6% compared to 3D network on UCF101-24 and J-HMDB-21, respectively. This indicates that 2D-CNN learns finer spatial features and 3D-CNN concentrates more on the motion process yet the spatial drift of an action in the clip may lead to a lower localization accuracy. It is also shown that CFAM module further boosts the performance from 73.8% to 79.2% on UCF101-24 and from 47.1% to 64.9% on J-HMDB-21. This clearly shows the importance of the attention mechanism which strengthens the inter-dependencies among channels and helps aggregating features more reasonably.

Moreover, in order to explore the impact of each 2D-CNN, 3D-CNN and CFAM blocks, we investigate the localization and the classification performance of different architectures, which is given in Table 2. For localization, we look at the recall value, which is the ratio of the number of correctly localized actions to the total number of ground truth actions. For classification, we look at the classification accuracy of the correctly localized detections. For both datasets, 2D network is better at localization while 3D network performs better at classification. It is also obvious that CFAM module boosts both localization and classification performance.

We have also visualized the activations maps [39] for 2D and 3D backbones of the trained model, which is shown in Fig. 4. Conforming our findings in Table 2, 3D backbone focuses on the parts of the clip where a motion is occurring and 2D

	Model	Localization (recall)	Classif.
UCF101-24	2D	91.7	85.9
	3D	90.8	92.9
	2D + 3D	93.2	93.7
	2D + 3D + CFAM	**93.5**	**94.5**
J-HMDB-21	2D	94.3	50.6
	3D	76.3	69.3
	2D + 3D	94.5	63.0
	2D + 3D + CFAM	**97.3**	**76.1**

Table 2: Localization @ IoU 0.5 (recall) and classification results on UCF101-24 and J-HMDB-21. For all architectures, the input to 3D-CNNs is 8 frames clips with downsampling 1.

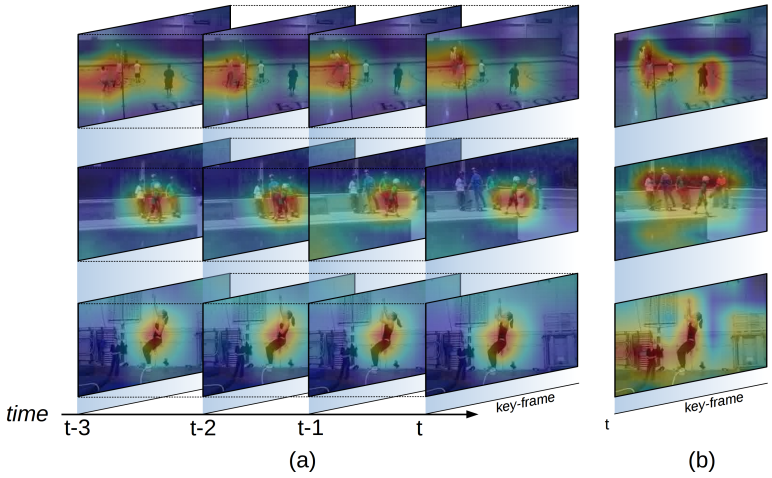

Figure 4: Activation maps for (a) 3D-CNN backbone and (b) 2D-CNN backbone. 3D-CNN backbone focuses on areas where there is a movement/action happening, whereas 2D-CNN backbone focuses on all the people in the key-frame. Examples are volleyball spiking (top), skate boarding (middle) and rope climbing (bottom).

Input	UCF101-24	J-HMDB-21
8-frames (d=1)	79.2	64.9
8-frames (d=2)	78.5	61.5
8-frames (d=3)	78.4	61.0
16-frames (d=1)	**80.4**	**74.4**
16-frames (d=2)	79.0	71.4

Table 3: Frame-mAP @ IoU 0.5 results on datasets UCF101-24 and J-HMDB-21 for different clip lengths and different downsampling rates d.

backbone focuses on fine spatial information on complete body parts of people. This validates that backbones of YOWO extract complementary features.

How many frames are suitable for temporal information? For 3D-CNN branch, different clip lengths with different downsampling rates can change the performance of overall YOWO architecture [40]. Therefore, we conduct experiments with 8-frames and 16-frames clips with different downsampling rates, which is given in Table 3. For example, 8-frames (d=3) refers to selecting 8 frames from 24 frames window with downsampling rate of 3. Specifically, we compare three downsampling rates $d = 1, 2, 3$ for clip length 8-frames and two downsampling rates $d = 1, 2$ for 16-frames clip length. As expected, we observe that the framework with input of 16 frames performs better than 8 frames since long frame sequence contains more temporal information. However, as downsampling rate is increased, the performance becomes worse. We conjecture that downsampling hinders capturing motion patterns properly and too long sequence may break the temporal contextual relationship. Especially for some quick motion classes, a long sequence may contain several unrelated frames, which can be viewed as noise.

Is it possible to save model complexity with more efficient networks? We have chosen 3D-ResNext-101 [13] since it has multiple cardinalities thus is able to learn more complicated features. However, it is a heavy-weighted

Model	GFLOPs	Frame-mAP (@ IoU 0.5)	
		UCF101-24	J-HMDB-21
3D-ResNext-101	27.7	**80.4**	**74.4**
3D-ResNet-101	42.4	78.1	70.8
3D-ResNet-50	27.1	77.8	61.3
3D-ResNet-18	22.2	72.6	57.5
3D-ShuffleNetV1 2.0x	1.6	71.3	54.8
3D-ShuffleNetV2 2.0x	1.4	71.4	55.3
3D-MobileNetV1 2.0x	1.8	67.3	48.5
3D-MobileNetV2 1.0x	1.8	66.6	52.5

Table 4: Performance comparison on datasets for different 3D backbones UCF101-24 and J-HMDB-21. For all architectures, Darknet-19 is used as 2D backbone. The number of floating point operation (FLOPs) are calculated for corresponding 3D backbones for 16 frames (d=1) clips with spatial resolution of 224 × 224.

backbone with a huge number of parameters and computational complexity. Therefore, we have replaced the 3D backbone with 3D-ResNet with different depths and with some other resource efficient 3D-CNN architectures [14]. Table 4 reports the achieved performance on both datasets together with the number of floating point operations (FLOPs) for each 3D backbone. We find that even with light-weight architecture in 3D backbones, our framework is still better than 2D network. However, Table 4 clearly shows the importance of the 3D backbone. The stronger 3D-CNN architecture we use, better the achieved results.

4.3. State-of-the-art comparison

We have compared YOWO with other state-of-the-art architectures on J-HMDB-21 and UCF101-24 datasets. For the sake of fairness, we have excluded VideoCapsuleNet [22] as it uses different video-mAP calculation without constructing action tubes via some linking strategies. However, YOWO still per-

Method	Frame-mAP	Video-mAP		
		0.2	**0.5**	**0.75**
Peng w/o MR [3]	56.9	71.1	70.6	48.2
Peng w/ MR [3]	58.5	74.3	73.1	-
ROAD [21]	-	73.8	72.0	44.5
T-CNN [2]	61.3	78.4	76.9	-
ACT [41]	65.7	74.2	73.7	52.1
P3D-CTN [42]	71.1	84.0	80.5	-
TPnet [43]	-	74.8	74.1	**61.3**
YOWO (16-frame)	74.4	87.8	85.7	58.1
YOWO+LFB*	**75.7**	**88.3**	**85.9**	58.6

Table 5: Performance on dataset J-HMDB-21 and comparison with SOTA results by frame-mAP under IOU threshold 0.5 and video-mAP under different IOU thresholds. * version of YOWO is non-causal.

forms around 9% and 8% better than VideoCapsuleNet in terms of frame-mAP @ 0.5 IoU on J-HMDB-21 and UCF101-24, respectively.

Performance comparison on J-HMDB-21 YOWO is compared with the previous state-of-the-art methods on J-HMDB-21 in Table 5. Using the standard metrics, we report the frame-mAP at IOU threshold 0.5 and the video-mAP at various IOU thresholds. YOWO (16-frame) consistently outperforms the state-of-the-art results on dataset J-HMDB-21, with a frame-mAP increase of 3.3% and a video-mAP increase of 3.8%, 5.2% at IOU thresholds of 0.2 and 0.5, respectively. Utilization of LFB brings further improvements on the performance. However, this improvement is marginal since the video duration of videos of J-HMDB-21 dataset is maximum 40 frames.

Performance comparison on UCF101-24 Table 6 presents the comparison of YOWO with the state-of-the-art methods on UCF101-24. YOWO (16-frame) achieves 80.4% with respect to frame-mAP metric, which is significantly better

Method	Frame-mAP	Video-mAP		
		0.1	**0.2**	**0.5**
Peng w/o MR [3]	64.8	49.5	41.2	-
Peng w/ MR [3]	65.7	50.4	42.3	-
ROAD [21]	-	-	73.5	46.3
T-CNN [2]	41.4	51.3	47.1	-
ACT [41]	69.5	-	77.2	51.4
MPS [44]	-	82.4	72.9	41.1
STEP [45]	75.0	83.1	76.6	-
YOWO (16-frame)	80.4	82.5	75.8	48.8
YOWO+LFB*	**87.3**	**86.1**	**78.6**	**53.1**

Table 6: Performance on dataset UCF101-24 and comparison with SOTA results by frame-mAP under IOU threshold 0.5 and video-mAP under different IOU thresholds. * version of YOWO is non-causal.

than the others by preceding the second best result with 5.4% improvement. As for video-mAP, our framework also produces very competitive results even though we just utilize a simple linking strategy. Utilization of LFB brings considerable improvement this time since the duration of UCF101-24 videos is much bigger than J-HMDB-21 videos. LBF further increases frame-mAP performance by around 7%.

Runtime comparison Most of the state-of-the-art methods are two stage architectures, which are computationally expensive to run in real time. YOWO is a unified architecture, which can be trained end-to-end. In addition, we do not employ optical flow, which is computationally burdensome. In Table 7, we compare runtime performance of YOWO with other state-of-the-art methods. YOWO's speed is calculated in terms of frames per second (fps) on a single NVIDIA Titan Xp GPU with a batch size of 8. It must be noted that YOWO's 2D and 3D backbones can be replaced with any arbitrary CNN model accord-

Model	Speed (fps)	F-mAP	V-mAP
Saha *et al.* [20]	4	-	36.4
ROAD (A) [21]	40	-	40.9
ROAD (A+RTF)[21]	28	-	41.9
ROAD (A+AF)[21]	7	-	46.3
YOWO (8-frames, d=1)	**62**	79.2	47.6
YOWO (16-frames, d=1)	34	**80.4**	**48.8**

Table 7: Run time and performance comparison on dataset UCF101-24 for F-mAP and V-mAP at 0.5 IoU threshold. For YOWO, ResNeXt-101 is used in its 3D backbone.

ing to desired runtime performance. Moreover, additional new backbones can be easily introduced for different information source such as *depth* or *infrared* modalities. The only thing to do is modification of CFAM block in order to accommodate new features.

4.4. Model visualization

In general, YOWO architecture performs a decent job at localizing actions in videos, which is illustrated in Fig. 5. However, YOWO also has some drawbacks. Firstly, since YOWO captures all the content of the key frame and the clip, it sometimes makes some false positive detections before the actions are performed. For example, in Fig. 5 first row last image, YOWO sees a person holding a ball at a basketball court and detects him very confidently although he is not shooting the ball yet. Secondly, YOWO needs enough temporal content to make correct action localization. If a person starts performing an action suddenly, localization at initial frames lacks temporal content and false actions are recognized consequently, as in Fig. 5 second row last image (climbing stair instead of running).

Figure 5: Sample action localizations for UCF101-24 and J-HMDB-21. Red bounding boxes are ground truth while green and orange are true and false positive localizations, respectively.

5. Conclusion

In this paper, we presented a novel unified architecture for spatiotemporal action localization in video streams. Our approach, YOWO, models the spatiotemporal context from successive frames for action understanding while extracting the fine spatial information from key frame to address the localization task in parallel. In addition, we make use of a channel fusion and attention mechanism for effective aggregation of these two kinds of information. Since we do not separate human detection and action classification procedures, the whole network can be optimized by a joint loss in an end-to-end framework. We have carried out a series of comparative evaluations on two challenging representative datasets UCF101-24 and J-HMDB-21. Our approach outperforms the other state-of-the-art results while retaining real-time capability, which makes it possible to deploy it on mobile devices.

Acknowledgements

We gratefully acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG) under Grant No. RI 658/25-2. We also acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in neural information processing systems, 2015, pp. 91–99.

[2] 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.

[3] X. Peng, C. Schmid, Multi-region two-stream r-cnn for action detection, in: European conference on computer vision, Springer, 2016, pp. 744–759.

[4] C.-Y. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, R. Girshick, Long-term feature banks for detailed video understanding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 284–293.

[5] C. Feichtenhofer, A. Pinz, A. Zisserman, Convolutional two-stream network fusion for video action recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1933–1941.

[6] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: Advances in neural information processing systems, 2014, pp. 568–576.

[7] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, L. Van Gool, Temporal segment networks: Towards good practices for deep action recognition, in: European conference on computer vision, Springer, 2016, pp. 20–36.

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

[9] S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, IEEE transactions on pattern analysis and machine intelligence 35 (1) (2012) 221–231.

[10] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Learning spatiotemporal features with 3d convolutional networks, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 4489–4497.

[11] J. Carreira, A. Zisserman, Quo vadis, action recognition? a new model and the kinetics dataset, in: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6299–6308.

[12] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1492–1500.

[13] K. Hara, H. Kataoka, Y. Satoh, Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, pp. 6546–6555.

[14] O. Kopuklu, N. Kose, A. Gunduz, G. Rigoll, Resource efficient 3d convolutional neural networks, in: The IEEE International Conference on Computer Vision (ICCV) Workshops, 2019.

[15] M. Zolfaghari, K. Singh, T. Brox, Eco: Efficient convolutional network for online video understanding, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 695–712.

[16] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.

[17] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European conference on computer vision, Springer, 2016, pp. 21–37.

[19] G. Gkioxari, J. Malik, Finding action tubes, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 759–768.

[20] M. S. P. T. Suman Saha, Gurkirt Singh, F. Cuzzolin, Deep learning for detecting multiple space-time action tubes in videos, in: Proceedings of the British Machine Vision Conference (BMVC), BMVA Press, 2016, pp. 58.1–58.13.

[21] G. Singh, S. Saha, M. Sapienza, P. H. Torr, F. Cuzzolin, Online real-time multiple spatiotemporal action localisation and prediction, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3637–3646.

[22] K. Duarte, Y. Rawat, M. Shah, Videocapsulenet: A simplified network for action detection, in: Advances in Neural Information Processing Systems, 2018, pp. 7610–7619.

[23] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical image computing and computer-assisted intervention, Springer, 2015, pp. 234–241.

[24] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, X. Tang, Residual attention network for image classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3156–3164.

[25] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, T.-S. Chua, Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5659–5667.

[26] S. Woo, J. Park, J.-Y. Lee, I. So Kweon, Cbam: Convolutional block attention module, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3–19.

[27] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, H. Lu, Dual attention network for scene segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3146–3154.

[28] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7132–7141.

[29] X. Wang, R. Girshick, A. Gupta, K. He, Non-local neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7794–7803.

[30] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.

[31] L. A. Gatys, A. S. Ecker, M. Bethge, A neural algorithm of artistic style, arXiv preprint arXiv:1508.06576.

[32] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.

[33] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.

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

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

[36] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, M. J. Black, Towards understanding action recognition, in: Proceedings of the IEEE international conference on computer vision, 2013, pp. 3192–3199.

[37] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, Hmdb: a large video database for human motion recognition, in: 2011 International Conference on Computer Vision, IEEE, 2011, pp. 2556–2563.

[38] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, International journal of computer vision 88 (2) (2010) 303–338.

[39] B. Zhou, A. Khosla, L. A., A. Oliva, A. Torralba, Learning Deep Features for Discriminative Localization., Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[40] O. Köpüklü, G. Rigoll, Analysis on temporal dimension of inputs for 3d convolutional neural networks, in: 2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS), IEEE, 2018, pp. 79–84.

[41] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, C. Schmid, Action tubelet detector for spatio-temporal action localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 4405–4413.

[42] J. Wei, H. Wang, Y. Yi, Q. Li, D. Huang, P3d-ctn: Pseudo-3d convolutional tube network for spatio-temporal action detection in videos, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE, 2019, pp. 300–304.

[43] G. Singh, S. Saha, F. Cuzzolin, Predicting action tubes, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 0–0.

[44] E. H. P. Alwando, Y.-T. Chen, W.-H. Fang, Cnn-based multiple path search for action tube detection in videos, IEEE Transactions on Circuits and Systems for Video Technology.

[45] X. Yang, X. Yang, M.-Y. Liu, F. Xiao, L. S. Davis, J. Kautz, Step: Spatio-temporal progressive learning for video action detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 264–272.