# Finding Action Tubes with a Sparse-to-Dense Framework

**Yuxi Li,[1] Weiyao Lin,[1,2*] Tao Wang,[1] John See,[3]**
**Rui Qian,[1] Ning Xu,[4] Limin Wang,[5] Shugong Xu[2]**
[1]School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, China
[2]Shanghai Institute for Advanced Communication and Data Science, Shanghai University, China
[3]Faculty of Computing and Informatics, Multimedia University, Malaysia
[4]Adobe Research, USA
[5]State Key Laboratory for Novel Software Technology, Nanjing University, China

## Abstract

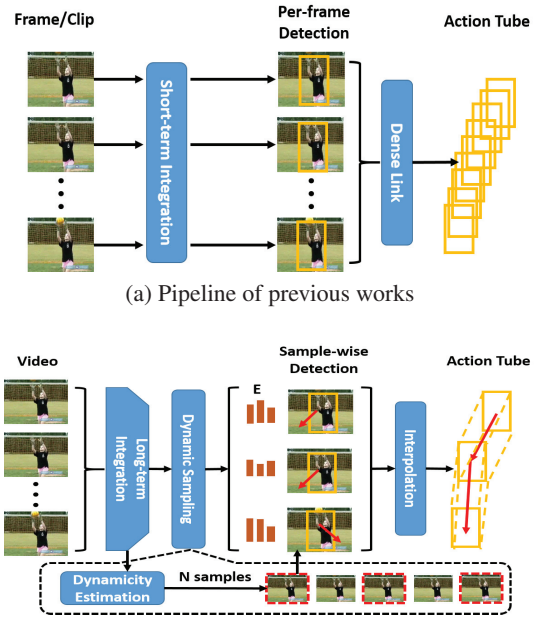
The task of spatial-temporal action detection has attracted increasing attention among researchers. Existing dominant methods solve this problem by relying on short-term information and dense serial-wise detection on each individual frames or clips. Despite their effectiveness, these methods showed inadequate use of long-term information and are prone to inefficiency. In this paper, we propose for the first time, an efficient framework that generates action tube proposals from video streams with a single forward pass in a sparse-to-dense manner. There are two key characteristics in this framework: (1) Both long-term and short-term sampled information are explicitly utilized in our spatio-temporal network, (2) A new dynamic feature sampling module (DTS) is designed to effectively approximate the tube output while keeping the system tractable. We evaluate the efficacy of our model on the UCF101-24, JHMDB-21 and UCFSports benchmark datasets, achieving promising results that are competitive to state-of-the-art methods. The proposed sparse-to-dense strategy rendered our framework about 7.6 times more efficient than the nearest competitor.

## Introduction

Spatial-temporal action detection is an essential technology for video understanding applications. In contrast to action recognition or temporal localization, where a video-level class label or temporal proposals are to be assigned spatial-temporal localization involves three sub-tasks: determining the interval of action occurrence, localizing the actors within the interval and correctly categorizing the action. This makes for an extremely complex task.

To the best of our knowledge, previous approaches (Peng and Schmid 2016; Saha et al. 2016; Singh et al. 2017; Hou, Chen, and Shah 2017; Yang, Gao, and Nevatia 2017) conform to a dense detection paradigm (Fig. 1(a)), which detects dense bounding boxes with actor detectors (Liu et al. 2016; Ren et al. 2017) first on with short-term information from frame or snippet level and then link the proposals together via certain heuristic linking algorithms. Relying on powerful convolution neural networks, these methods achieved good performance in both localization and classification. However, there are several shortcomings of such pipelines. Firstly, the input of most methods only contain brief temporal information, which makes it hard to identify actions that are somewhat similar through most parts of the video, e.g. both the action "Long Jump" and "Pole Vault" involve actors running at the beginning and are only different at the last part of these videos. Secondly, by simple but brute force means, it is inefficient to generate dense frame level proposals considering the computational complexity.

With these consideration, we propose an novel framework for spatial-temporal action detection in a sparse-to-dense manner. In contrast to the pipelines of previous works, we first generate box proposals at sparsely sampled frames (Fig. 1(b)), then we get the dense tube by interpolating the




(a) Pipeline of previous works

(b) Pipeline of our work

Figure 1: Comparison between previous pipelines and our pipeline

*Corresponding Author, Email: wylin@sjtu.edu.cn

sparse proposals across a given detected time interval.

We solve the issues of previous works in two aspects. Firstly, we introduce a long-term feature augmentation module (LFA) to combine both the long-term and short-term information in a single forward pass. We exploit this combined feature to generate more reliable spatial and temporal proposals. Additionally, the augmented feature is further utilized to generate embedding and shift vectors (symbolized by 'E' and red arrows in Fig. 1(b)) for each bounding box, facilitating the tube generation process.

Secondly, we propose an adaptive dynamic temporal sampling module (DTS) to achieve sparse detection and decrease computation complexity. We argue that if the number of sampled frames is sufficient and the detection on each sampled frame is accurate, then the interpolation result over each bounding box forms an approximated tube proposal, as shown in Fig. 1(b). Additionally, we hypothesize that the number of sampled frames required to achieve good approximation varies and depends on the dynamic extent of actions, e.g. when actors stand still and do not move, then the bounding box stays the same over time, therefore boxes on two sampled frames are sufficient to reconstruct the action tube. Based on these observation, we guide our network to estimate the level of action dynamics, or *dynamicity* of each tube proposal, and DTS adaptively maps 3D features to 2D form for further detection. With the DTS module, our framework can achieve a good trade-off between model complexity and accuracy of localization.

In summary, our contributions are three folds.

1. We propose an general framework for the task of spatial-temporal action detection. In this framework, tube proposals are generated by a *sparse-to-dense* mechanism with a single forward pass. Competitive results on three benchmark datasets are obtained with an inference speed that is $7.6\times$ faster than the nearest competitor. [1]
2. In this framework, we design a long-term feature augmentation module (LFA) for enhanced feature representation. We further exploit the augmented feature to generate reliable proposals and auxiliary vectors for box association.
3. For sparse detection, we introduce an adaptive sampling module called dynamic temporal sampling (DTS) to boost the accuracy of tube construction at a much lower computation cost.

## Related Works

**Action recognition.** Owing to its successful application of deep neural networks for image classification (Simonyan and Zisserman 2014b; 2014b; Huang et al. 2017), some works have started to focus on the utilization of CNNs for video action recognition. Simonyan & Zisserman (Simonyan and Zisserman 2014a) proposed a two-stream framework, which used two networks to extract features from appearance and optical flow input respectively, and combined them to obtain the final decision. (Tran et al. 2015) developed a 3D ConvNet to automatically extract the feature representation for actions. The I3D network (Carreira and Zisserman 2017) further inflated the networks pretrained on ImageNet (2D) (Deng et al. 2009) to form an efficient 3D network for action recognition. Despite these works achieving good results on standard benchmarks, these methods only focused on labels at video level and is not capable of providing more detailed information as when the action starts or how the actors move.

**Temporal action detection.** Some early works on action detection mainly focused on the temporal localization of video actions, which aims at finding the time interval of action instances. A few recent works (Lin, Zhao, and Shou 2017; Zhao et al. 2017; Chao et al. 2018) extracted frame level deep features and apply either RPN or pyramidal structure for 1-dimensional action detection. In (Xu, Das, and Saenko 2017; Zhang et al. 2018), a 3D convNet was first utilized to extract 3D features for the proposal network or action detectors (Ren et al. 2017; Liu et al. 2016). However, none of these approaches yielded precise spatial-temporal tubes for action instances.

**Spatial-temporal action detection.** The earliest pipeline devised for action tube detection was proposed in (Gkioxari and Malik 2015), where the R-CNN structure (Girshick et al. 2014) was applied on each frame for action bounding box detection and the results were linked by viterbi algorithm; however, it cannot determine the interval of actions. (Saha et al. 2016) addressed the problem by introducing an extra labeling operation after linking. The following methods mainly strove for better feature representation by involving contextual information (Peng and Schmid 2016), extracting spatial-temporal features from short time snippets (Saha, Singh, and Cuzzolin 2017; Kalogeiton et al. 2017; Hou, Chen, and Shah 2017), or resorting to recurrent neural networks (Huang et al. 2018). All of these works follow a common detect-and-link framework, the output spatial proposals are linked either by viterbi algorithm or other improved strategies (Singh et al. 2017; Huang et al. 2018). Such pipelines require dense detections for each video. This inefficiency worsens when optical flow computation is taken into account. Further, the input model only contains local short time information and can be ambiguous during some short intervals.

In contrast to the works mentioned above, our framework is the first pipeline to generate action tube proposals in a sparse-to-dense manner and handles both long and short term information in a single forward pass . Long-term temporal context is considered by capturing characteristics of action instances for final association. Besides, the dynamic temporal sampling (DTS) module further decreases the search space to achieve good approximation of the ground truth bounding boxes.

## Method

Given a video $\mathcal{V}$, our goal is to find a tube set $\mathcal{A} = \{\Phi_j | j = 1 \dots N\}$, where the symbol $\Phi_j$ denotes a certain action tube. For each tube $\Phi_j$, a class label $c_j$ and a spatial bounding box set $\mathcal{B}_j = \{b_t | t \in [s_j, e_j]\}$ is assigned to it, where $b_t$ denotes

[1] The related project page will be posted at: http://min.sjtu.edu.cn/lwydemo/Tube.html

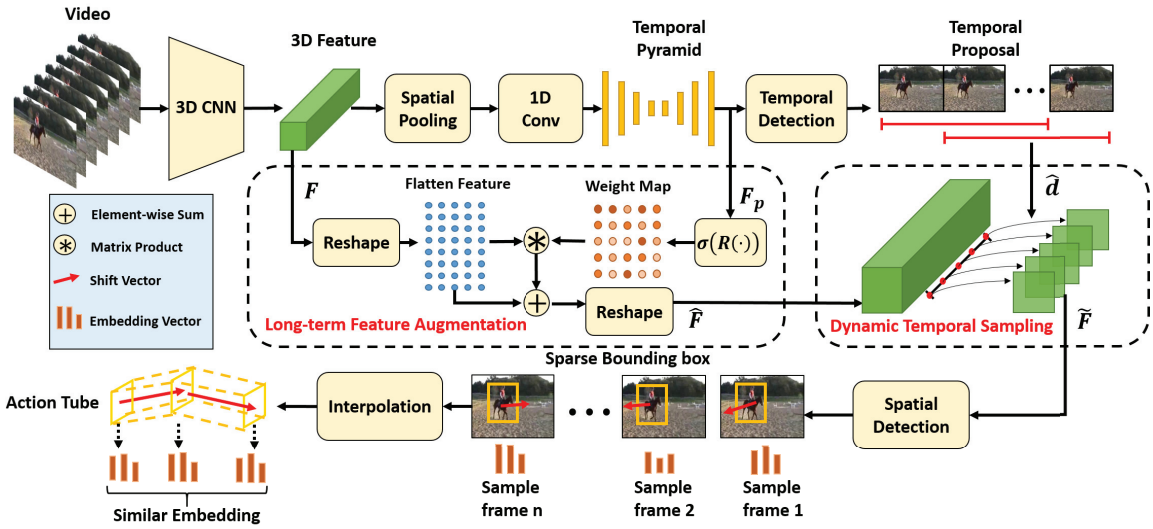



Figure 2: Overview of the proposed framework.

the four dimensions of the bounding box and $s_j, e_j$ denote the starting and ending frame indices.

Our framework is illustrated in Fig. 2. At first, an input video stream is temporally sampled to form a fixed-length tensor of $T$ frames. The tensor is fed into a 3D ConvNet for feature extraction. The network transforms the feature into a 1D temporal feature pyramid to capture long-term information and generate temporal proposals. Since the original 3D features have limited temporal receiptive field, the following long-term feature augmentation module (LFA) boosts the 3D features with long-term weighted recombination. Next, the dynamic temporal sampling module (DTS) takes the augmented feature and temporal proposals to generate sparse 2D feature samples over time. For each temporal proposal,the sampled features are utilized to generate spatial bounding boxes, which are accompanied with embedding and shift vectors. With the guidance of these vectors, the boxes are associated and interpolated to form final tube proposals along corresponding temporal proposal.

## Temporal Long-term Integration

To extract long-term information and integrate it with short-term features, we design a temporal pyramid and long-term augmentation module to handle features at temporal scale.

**Temporal pyramid and proposals.** The construction of the temporal pyramid structure is illustrated in Fig. 2. We first squeeze the input 3D feature into a single dimension via spatial average pooling. To obtain 1D temporal features, we first shrink the temporal resolution by convolution with stride 2, and then reversely enlarge the temporal size with deconvolution operators. These manipulations construct a two-path temporal feature pyramid, i.e. downsample path and upsample path.

We then predict temporal proposals from the pyramid. Similar to (Zhang et al. 2018), on each level of the upsample path, we assign the pixels with a set of temporal anchors of different scales. For each anchor, the temporal detector predicts the actioness score and regresses the offsets relative to the center and length of it via 1D temporal convolution.

**Long-term feature augmentation.** The LFA module works by explicitly recombining long-term information via an attention mechanism and fuses it with short-term features extracted earlier. The details are illustrated in Fig. 2. We take the feature $\mathcal{F}_p$ from the last layer of upsample path (of size $C \times T_f$) as the output of temporal pyramid, where $C$ is the channel number and $T_f$ is the temporal size of feature. The model learns a mapping function $\mathcal{R}(\cdot)$ that maps $\mathcal{F}_p$ into a subspace and normalizes it with a sigmoid function $\sigma(\cdot)$,

$$\mathcal{S} = \sigma\left(\mathcal{R}(\mathcal{F}_p; \theta)\right) \tag{1}$$

where $\mathcal{S}$ is a weight map of size $T_f \times T_f$, and $\theta$ represent the learnable parameters. In implementation, the function $\mathcal{R}(\cdot)$ is constructed by stacking three 1-D convolutional blocks followed by batch normalization and ReLU function. Finally, we obtain the new 3D feature representation via weighted combination and fusion process in terms of Eq. 2, which can be simply accomplished via matrix multiplication and a residual connection:

$$\hat{\mathcal{F}}_{c,t,h,w} = \mathcal{F}_{c,t,h,w} + \frac{1}{T_f} \sum_{k=1}^{T_f} \mathcal{F}_{c,k,h,w} \mathcal{S}_{t,k} \tag{2}$$

where the tensor $\mathcal{F}$ is the original 3D feature input and $T_f$ is its temporal size, $t, h, w$ are the temporal and spatial coordinates of each pixel in feature map. Note that Eq. 2 can be regarded as a recombination of each temporal component of input feature. At each time index of the output feature, the component is a weighted combination of all other time indices. With this module, we can guarantee that the time component of each 3D feature is encoded with long-term information.

We argue that our design of the long-short term integration mechanism has the following advantages: (i) Our network localizes the temporal interval at an early stage, thus

**Algorithm 1:** Ground-truth dynamic level generation

**Input:** Tube $\Phi$, threshold $\epsilon$
**Output:** dynamic level $d$

get bounding box set $\mathcal{B} = \{b_i | i = 1 \dots T\}$ from tube $\Phi$ ;
**for** $d = 1$ *to* $T$ **do**
  $r = 0$;
  uniformly sample $d$ bounding boxes from $\mathcal{B}$ and obtain sampled box set $\mathcal{P} = \{b_{t_k} | t_k = \lfloor kT/d \rfloor, k = 1 \dots d\}$;
  Interpolate over $\mathcal{P}$ to obtain $\hat{\mathcal{B}} = \{\hat{b}_i | i = 1 \dots T\}$;
  **for** $k = 1$ *to* $T$ **do**
    $r = r + IOU(b_k, \hat{b}_k)$;
  **end**
  $r = r/T$;
  **if** $r \geq \epsilon$ **then**
    **return** $d$;
  **end**
**end**

avoiding the temporal labelling process (Saha et al. 2016) used in other works as a post-processing step. (ii) The feature pyramid guides the long-term recombination and fusion of original 3D features via LFA, which leads to more representative 3D features. The augmented features can be utilized to generate better detection results.

## Dynamic Temporal Sampling

The next step is to temporally sample the augmented 3D feature into 2D form for subsequent sparse detection. We design an adaptive dynamic temporal sampling (DTS) module to obtain sufficient feature samples at an appropriate sampling rate.

The key mechanism of DTS is to generate an appropriate ground-truth dynamic level $d$ to guide the prediction of $\hat{d}$. Intuitively, for a scalar function $f(x)$, if its sampled values $Y = \{f(x_t) | t = 1 \dots\}$ are sufficient, then the piecewise interpolation between $(x_i, f(x_i))$ and $(x_{i+1}, f(x_{i+1}))$ is a good approximation to the corresponding original segment $f(x)$. Based on this intuition, we define $d$ as the minimum number of uniform samples such that the interpolation between samples form a tube as close as possible to the ground-truth. This process is depicted in Algorithm 1.

To obtain an appropriate estimation, for each temporal proposal, we estimate the dynamic level $\hat{d}$ via an additional convolution layer by considering the intensity of motion variation. During training stage, for each temporal proposal that matches a ground-truth tube, we assign the corresponding $d$ to it and train with a weighted smooth L1 loss (Ren et al. 2017):

$$L_d = smooth\_L_1(d - \hat{d}; \gamma) \tag{3}$$

where the negative part of loss function is suppressed by a factor of $\gamma$. Since having insufficient samples (small $n$ value) is more harmful to final approximation than large ones, we place a stronger penalty on the positive side of loss function.

During inference time, we set the sample number $n = \min(\lceil \hat{d} \rceil, N_{max})$, where the hyperparameter $N_{max}$ avoids sample numbers that are too large. Given a normalized temporal proposal $(s, e)$, where $s$ and $e$ are the starting and ending time indices, the 2D spatial features can be sampled from 3D features via linear interpolation:

$$\widetilde{\mathcal{F}}^i_{c,h,w} = \sum_{k=1}^{T_f} \hat{\mathcal{F}}_{c,k,h,w} \max\left(0, 1 - \left| s + \frac{e-s}{n-1} i - k \right| \right) \tag{4}$$

where $\widetilde{\mathcal{F}}^i$ is the $i$-th 2D feature map output. Compared with dense frame-level detection, the design of DTS is able to decrease the number of detection operations to a large extent and make the framework more efficient during inference.

## Sparsely Sampled Bounding Box Detection

Given the sparsely sampled 2D features, we detect actor bounding boxes on each feature map with a spatial detector network. The spatial detector network consists of two branches: the box detection branch and the association branch (shown in Fig. 3). In the box detection branch, we apply 2D convolution over each sampled feature map to obtain classification score and regress coordinate offsets, which is similar to (Liu et al. 2016). For the association branch, 3D convolution is applied on the stacked sampled features to capture the temporal relation among adjacent key frames, at the end of the path, the network outputs an embedding vector $\mathbf{f}_i$ and a potential shift vector $\delta_i = [d_{x,i}, d_{y,i}]^T$ for each anchor box $b_i$. These vectors encode the appearance and spatial relationship between boxes in sparsely sampled frames of the same action instance.

Intuitively, the boxes belonging to the same action instance are expected to exhibit similar appearance feature, while the potential shift vector should indicate the direction and distance in which the bounding box of the current sample could shift to the corresponding box in the next sample. With this consideration, we design two losses $L_a$ and $L_m$ to regularize the embedding and shift vectors of positive anchors. To be specific, we sample the corresponding ground-truth tubes with the same sampling rate as the feature and treat the $i$-th sampled bounding box $b_{t_i} = (x_{t_i}, y_{t_i}, w_{t_i}, h_{t_i})$ from tube $\Phi$ as the ground-truth for the $i$-th sampled feature map. The loss term $L_a$ guides the form of embedding vectors to optimize the cluster-like distance in feature space:

$$L_a = \sum_i C_i \mathbf{d}_2(\bar{\mathbf{f}}, \mathbf{f}_i) + (1 - C_i)[\alpha - \mathbf{d}_2(\bar{\mathbf{f}}, \mathbf{f}_i)]_+ \tag{5}$$

where $C_i$ is 1 if anchor $i$ matches current ground-truth action instance, otherwise it equals 0. $\mathbf{d}_2(\cdot, \cdot)$ is the squared $L_2$ distance, the symbol $\bar{\mathbf{f}}$ is the mean embedding vector over all anchors matching the instance. $[\cdot]_+$ is the ReLU operator and $\alpha$ is a hyperparameter to control the margin between different cluster centers.

The other loss term $L_m$ guides the prediction of box offsets to the next ground-truth sample, which is expressed as follows:

$$L_m = \sum_{k=2}^{n} \sum_i C_i \mathbf{d}_2(\delta_i, \delta_{gt,k}) \tag{6}$$

where $n$ is the number of sampled features, and $\delta_{gt,k} = [x_{t_{k+1}} - x_{t_k}, y_{t_{k+1}} - y_{t_k}]^T$ indicates the offset from the center of current sampled box to the next one.

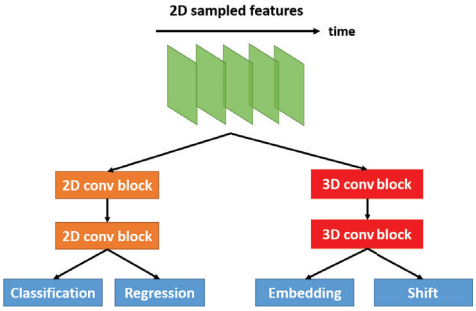



Figure 3: Illustration of spatial detection network. Each convolutional block comprises of a convolutional layer, a batch normalization layer and a ReLU layer.

## Dense Tube Generation from Sparse Proposals

While the preceding steps predict the time interval of actions and their associated sparse bounding box proposals, the final step combines these proposals to form the dense action tube along the temporal dimension. In this scenario, we simply design a greedy strategy to generate tube proposals.

Given a temporal proposal $(s, e)$ and its confidence probability $p$, we can obtain $n$ sets of bounding boxes via DTS and the spatial detection network, which is denoted as $\mathcal{U} = \{\hat{\mathcal{B}}_t | t = 1 \dots n\}$, where $\hat{\mathcal{B}}_t$ is the box proposal set from the $t$-th sampled 2D feature. Note that in Fig. 3, the spatial detection network does not only predict the boxes, but also assign embedding vector and shift vectors for each anchor. We hereby define $p_{tj}$ as the probability score for class $c$ from box proposal $\hat{b_{tj}} \in \hat{\mathcal{B}}_t$, its corresponding embedding vector as $\mathbf{f}_{tj}$ and potential shift vector as $\delta_{tj}$. For each pair of bounding boxes from adjacent samples, we compute their appearance distance $D_{a,tij}$ and spatial distance $D_{s,tij}$ at $t \in [1, n-1]$ between $\hat{b}_{ti} \in \hat{\mathcal{B}}_t$ and $\hat{b}_{(t+1)j} \in \hat{\mathcal{B}}_{t+1}$ as:

$$\begin{aligned} D_{a,tij} &= ||\mathbf{f}_{ti} - \mathbf{f}_{(t+1)j}||_2 \\ D_{s,tij} &= ||\delta_{ti} - \overline{\delta}_{tij}||_2 \end{aligned} \tag{7}$$

where $\overline{\delta}_{tij}$ denotes the center offsets from box $\hat{b}_{ti}$ to box $\hat{b}_{(t+1)j}$. We formulate the connectivity between the two adjacent boxes as:

$$\mathcal{C}\left(\hat{b}_{ti}, \hat{b}_{(t+1)j}\right) = \exp\left(-\frac{D_{a,tij} + D_{s,tij}}{2}\right) \tag{8}$$

Selection of box proposals begin by first picking the starting box from set $\hat{\mathcal{B}}_1$ according to the classification probability:

$$\hat{b_1} = \arg\max_{\hat{\mathcal{B}}_1} p_{1j} \tag{9}$$

Subsequently, for the rest of the box proposal sets, we select boxes with the largest connectivity to the last box of current tube.

$$\hat{b}_{(t+1)} = \arg\max_{\hat{\mathcal{B}}_{(t+1)}} \mathcal{C}\left(\hat{b}_t, \hat{b}_{(t+1)j}\right) \tag{10}$$

Finally, we perform linear interpolation between contiguous boxes in sample set $\{\hat{b}_t\}$ to reconstruct the tube. The sampling rate for interpolation can be expressed as $f_s = \frac{n}{(e-s)T} f_v$, where $f_v$ is the sample frequency from original raw video to fixed length tensor. After the association and interpolation steps, we obtain the dense frame level bounding boxes for each action tube. The score of the action tube is the average score across all predicted boxes multiplied by the actioness score $p$ of the temporal proposal.

# Experimental results

## Experimental Settings

**Datasets.** We conduct our experiment on three common datasets – UCF101-24, UCFSports and JHMDB-21 datasets. Although the recent AVA (Gu et al. 2018) dataset also contains bounding box annotations, it mainly concentrates on the task of detecting actors on a single key frame. Their measurement with frame level mAP, which does not include temporal interval prediction, is slightly different from the task we are focusing here. Hence, we did not conduct our experiments on the AVA dataset.

The UCF101-24 dataset (Soomro, Zamir, and Shah 2012) contains 3,207 untrimmed videos with frame level bounding box annotations for 24 sports classes. The dataset is challenging due to the frequent camera shake besides the actor movements. Following previous works (Saha et al. 2016), we report results for the first split. The JHMDB-21 is a subset of HMDB-51 dataset (Jhuang et al. 2013), which contains a total of 928 videos with 21 types of actions. All video sequences are temporally trimmed. The results are reported as the average performance over 3 train-test splits. The UCF-Sports dataset (Rodriguez, Ahmed, and Shah 2008) contains 150 videos of 10 sport action classes. All videos in this dataset are temporally trimmed. We report the result on the standard split. Note that although the last two datasets are trimmed temporally, their samples are still suitable for our framework as they comprise of actions spanning the whole video.

**Metric.** We adopt the standard video-mAP (v-mAP) (Gkioxari and Malik 2015) as our metric for spatial-temporal action detection on all three datasets. A proposal is regarded as positive only when its tube overlap with an undetected ground-truth is larger than threshold $\Delta$.

**Implementation details.** Our model is implemented on an NVIDIA Titan 1080 GPU. We use the I3D network (Carreira and Zisserman 2017) as our 3D feature extractor. During training, we use length $T$ of videos sampled to 96 frames for UCF101, 32 frames for JHMDB-21 and 48 frames for UCFSports. We set the hyperparameters as: $\alpha = 2, \epsilon = 0.7, \gamma = 0.1$. We train the network in two stages: First, we fix the weights in the spatial detection network and LFA, then train the backbone and temporal detection network for the purpose of learning the proposals and dynamic levels. Next, we jointly train the network end-to-end to learn the final action tubes. We use the SGD solver and train our network with an accumulative batch size of 8. In inference stage, to handle con-current actions, the association process is conducted 4 times (the maximum possible number of ac-

tion instances in all of the three datasets) and a following tube-wise NMS is applied to remove duplicated tubes.

## Ablation Studies

We conduct ablation studies on the UCF101-24 dataset. Note that all experiments are using only RGB data input.

**Long-term feature augmentation.** We first analyze the impact of the LFA since it is the key operation for long-term information integration. For comparison, we remove the module from our model and directly sample from the original 3D feature. In Table 1, the performance of our framework dropped $8.9\%$ in video-mAP without the long-term integration, which indicates that the temporal context is essential to final sample-wise detection.

In Fig. 4, we show the per-class AP value at the threshold of $\Delta = 0.3$ on UCF101-24 dataset. To make comparison, we also reported results with the removal of LFA module. It can be observed for most action classes, LFA module can boost the detection performance. Especially, for some short-term ambiguous classes like "Long Jump" versus "PoleVault" and "Cricket Bowling", the enhancement is more obvious. (increased by $38\%, 29\%$ and $25\%$ respectively.)

**Embedding and shift vectors.** We also study how the embedding and shift vectors from association path helps action localization. In this experiment, we experimented with the removal of different vector components (and combinations of them) while training with the remaining leftover parts. For experiments without both components, we adopt a score-based strategy which always chooses the box with largest score in sample. The results are shown in Table 1. We find that when the embedding vector is ignored, the result dropped $2.2\%$. Meanwhile, ignoring the potential shift vector resulted in drop of $1.5\%$ v-mAP. On the other hand, we find the baseline method with the largest score strategy performs worse than our method. These observation indicates that both the embedding and shift information are helpful towards the final box association process.

| LFA | | ✓ | ✓ | ✓ | ✓ |
|---|---|---|---|---|---|
| Embedding | ✓ | | | ✓ | ✓ |
| Shift | ✓ | | ✓ | | ✓ |
| v-mAP@0.3 | 62.2 | 67.4 | 68.9 | 69.6 | **71.1** |

Table 1: Results of applying the LFA module and association path vectors from the spatial detection network.

| scheme | v-mAP@0.3 | per-video time (s) |
|---|---|---|
| fixed point-avg | 67.4 | 0.551 |
| fixed point-4 | 63.3 | 0.412 |
| fixed point-10 | 67.5 | 0.654 |
| fixed step-avg | 66.9 | 0.571 |
| DTS | **71.1** | **0.569** |

Table 2: Results of different sampling schemes

**Dynamic temporal sample**. To demonstrate the effectiveness of our dynamic temporal sample module, we conduct an experiment to compare between different sample strategies. (i) Fixed point sampling: For each temporal proposal, we sample a fixed number of 2D features for spatial detection. For comparison, we adopt the average sample number of algorithm 1; we denote this scheme with suffix *'avg'*. We also test the performance with a larger $(10)$ and smaller $(4)$ number of samples, and they are denoted with suffixes *'10'* and *'4'* respectively. (ii) Fixed step sampling: Given the normalized interval $(s, e)$, we sample features according to a fixed step size. Here, we set the step size as the average sample frequency of Algorithm 1 during experiments, which is denoted with suffix *'avg'*. (iii) The proposed dynamic temporal sampling *DTS* strategy.

Table 2 shows the experimental results with different sampling schemes. By comparison, we make the following observations: (i) By comparing *DTS* and *fixed point-avg*, we find that the two scheme achieved similar time costs, while *DTS* outperformed the latter strategy in v-mAP terms. This is because *fixed point-avg* is not adaptive to different dynamic levels, thus some samples could be redundant for simple action or scarce for complex actions. (ii) *fixed point-10* performed worse than *DTS* and took more time to process the video, which shows that our strategy is more efficient and effective. On the other hand, *fixed point-4* is worser than its 10-point counterpart, which indicates that less number of samples could be harmful to final tube detection. (iii) *DTS* and *fixed step-avg* both consumes similar time costs, but *DTS* outperformed *fixed step-avg*. This indicates that being adaptive to the duration of action is less optimal than being adaptive to the complexity of the action.

## Comparison with State-of-the-art

In Table 3, we compare the detection performance of our framework with other state-of-the-art methods on three benchmark datasets with only RGB image as input.

On UCF101-24, our method outperforms (Saha, Singh, and Cuzzolin 2017), which is another RGB-only method, at all tested thresholds in this table. Compared with other methods that utilize both RGB and optical flow inputs, our methods also achieves state-of-the-art results at both $\Delta = 0.3$ and $\Delta = 0.5$. This indicates that our framework is effective and competitive even when the actions contain large motion and length variations. On the UCFSports and JHMDB-21 datasets, our method is also able to outperform many recent works (Saha, Singh, and Cuzzolin 2017; Kalogeiton et al. 2017; Singh et al. 2017; Zhao and Snoek 2019). Overall, the comparisons on these three datasets indicate that our approach may be suited for both short trimmed clips and long videos of variable length. Although there is still a margin between our method and state-of-the-art ConvNet + LSTM method (Li et al. 2018) on UCF-Sports and JHMDB, their improvement mainly comes from the introduction of flow data, which is not involved in our work due to the efficiency consideration. This is most obvious from the fact that our approach is superior to (Li et al. 2018) solely on RGB input.

**Inference time.** We also report the runtime cost for action tube detection. Following (Saha et al. 2016), we run our model with data from JHMDB-21 for one epoch and compute the average inference time. The result is reported

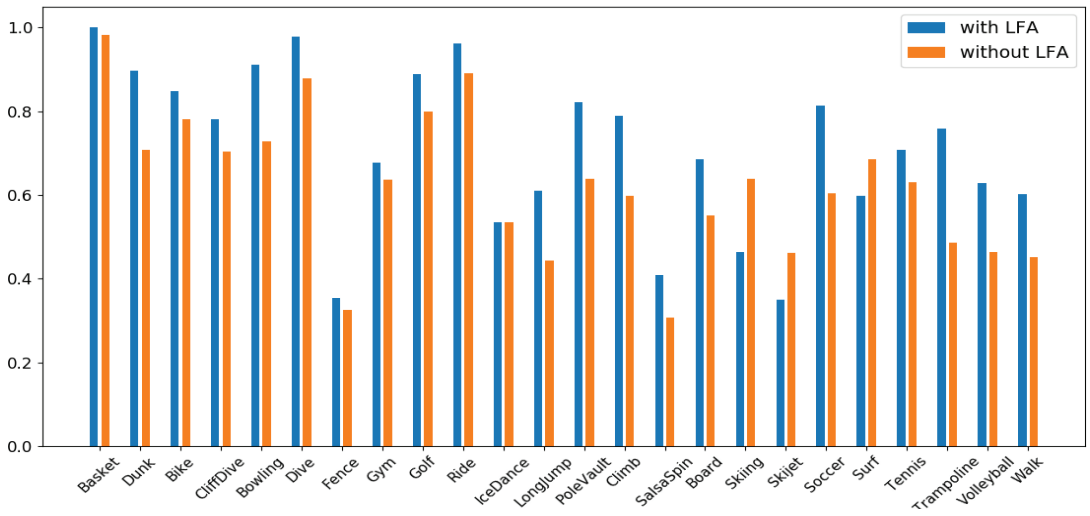



Figure 4: Per-class AP on the UCF101-24 dataset with and without LFA module.

| method | input | JHMDB-21 | | | UCFSports | | | UCF101-24 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\Delta$ | | 0.2 | 0.5 | 0.75 | 0.2 | 0.5 | 0.75 | 0.3 | 0.5 | 0.75 |
| (Saha et al. 2016) | RGB+Flow | 72.6 | 71.5 | - | - | - | - | 54.9 | 35.9 | - |
| (Peng and Schmid 2016) | RGB+Flow | 74.3 | 73.1 | 48.2 | 94.8 | 94.7 | 47.3 | 65.7 | 32.1 | 2.7 |
| (Kalogeiton et al. 2017) | RGB+Flow | 74.2 | 73.7 | 52.1 | 92.7 | 92.7 | 78.4 | - | 49.2 | 19.7 |
| (Hou, Chen, and Shah 2017) | RGB+Flow | 78.4 | 76.9 | - | 95.2 | 95.2 | - | 69.4 | - | - |
| (Singh et al. 2017) | RGB+Flow | 73.8 | 72.0 | 44.5 | - | - | - | - | 46.3 | 15.0 |
| (Yang, Gao, and Nevatia 2017) | RGB+Flow | - | - | - | - | - | - | 60.7 | 37.8 | - |
| (Song et al. 2019) | RGB+Flow | 74.1 | 73.4 | 52.5 | - | - | - | - | 52.9 | 21.8 |
| (Zhao and Snoek 2019) | RGB+Flow | - | 58.0 | 42.8 | - | 92.7 | **83.4** | - | 48.3 | **22.1** |
| (Li et al. 2018) | RGB+Flow | **82.3** | **80.5** | - | **97.8** | **97.8** | - | **70.9** | - | - |
| (Saha et al. 2016) | RGB | 52.9 | 51.3 | - | - | - | - | 48.3 | 30.7 | - |
| (Saha, Singh, and Cuzzolin 2017) | RGB | 57.8 | 55.3 | - | - | - | - | 51.7 | 33.0 | 0.5 |
| (Li et al. 2018) | RGB | - | 61.7 | - | - | 87.6 | - | - | - | - |
| Ours | RGB | **76.1** | **74.3** | **56.4** | **94.3** | **93.8** | 79.5 | **71.1** | **54.0** | 21.8 |

Table 3: Comparison with state-of-the-art methods (video-mAP), '-' denotes that the result is not available

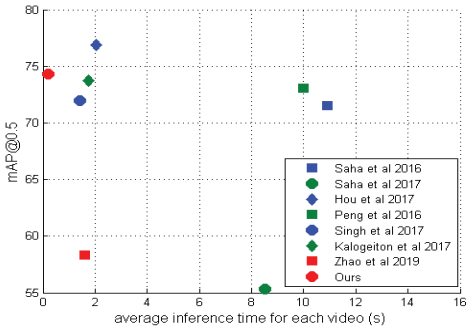



Figure 5: Time cost comparison between different pipelines

in Fig. 5. Our framework infers each video in only 0.21 seconds on average, achieving at most 7.6× faster than the nearest competitor. In terms of FPS, our approach runs at approximately 168 FPS, faster than other works reported in FPS speed (Singh et al. 2017; Kalogeiton et al. 2017; Yang et al. 2019).

## Conclusion

In this paper, we depart from previous pipelines to propose a new framework for spatial-temporal action detection. We design the long-term augmentation mechanism and dynamic temporal sampling module to facilitate the detection process. We demonstrate the effectiveness of our approach on benchmarks of UCF101-24, JHMDB-21 and UCFSports. Besides, our model can process at a quicker speed, about 7.6× faster than the nearest competitor.

## Acknowledgement

The paper is supported in part by the following grants: China Major Project for New Generation of AI Grant(No. 2018AAA0100400), National Natural Science Foundation of China (No. 61971277, 61921066), CREST Malaysia Grant T03C1-17, and Adobe Research Gift.

## References

Carreira, J., and Zisserman, A. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 6299–6308.

Chao, Y.-W.; Vijayanarasimhan, S.; Seybold, B.; Ross, D. A.; Deng, J.; and Sukthankar, R. 2018. Rethinking the faster r-cnn architecture for temporal action localization. In *CVPR*, 1130–1139.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*, 248–255. IEEE.

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 580–587.

Gkioxari, G., and Malik, J. 2015. Finding action tubes. *CVPR*.

Gu, C.; Sun, C.; Vijayanarasimhan, S.; Pantofaru, C.; Ross, D. A.; Toderici, G.; Li, Y.; Ricco, S.; Sukthankar, R.; Schmid, C.; and Malik, J. 2018. Ava: A video dataset of spatio-temporally localized atomic visual actions. *CVPR* 6047–6056.

Hou, R.; Chen, C.; and Shah, M. 2017. An end-to-end 3d convolutional neural network for action detection and segmentation in videos. *arXiv preprint arXiv:1712.01111*.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *CVPR*, 4700–4708.

Huang, J.; Li, N.; Zhong, J.; Li, T. H.; and Li, G. 2018. Online action tube detection via resolving the spatio-temporal context pattern. In *ACM MM*, 993–1001. ACM.

Jhuang, H.; Gall, J.; Zuffi, S.; Schmid, C.; and Black, M. J. 2013. Towards understanding action recognition. In *ICCV*, 3192–3199.

Kalogeiton, V.; Weinzaepfel, P.; Ferrari, V.; and Schmid, C. 2017. Action tubelet detector for spatio-temporal action localization. In *ICCV*, 4405–4413.

Li, D.; Qiu, Z.; Dai, Q.; Yao, T.; and Mei, T. 2018. Recurrent tubelet proposal and recognition networks for action detection. In *ECCV*, 303–318.

Lin, T.; Zhao, X.; and Shou, Z. 2017. Single shot temporal action detection. In *ACM MM*, 988–996. ACM.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *ECCV*, 21–37. Springer.

Peng, X., and Schmid, C. 2016. Multi-region two-stream r-cnn for action detection. In *ECCV*, 744–759. Springer.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2017. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1137–1149.

Rodriguez, M. D.; Ahmed, J.; and Shah, M. 2008. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 1–8.

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

Saha, S.; Singh, G.; and Cuzzolin, F. 2017. Amtnet: Action-micro-tube regression by end-to-end trainable deep architecture. In *ICCV*, 4414–4423.

Simonyan, K., and Zisserman, A. 2014a. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 568–576.

Simonyan, K., and Zisserman, A. 2014b. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Singh, G.; Saha, S.; Sapienza, M.; Torr, P. H.; and Cuzzolin, F. 2017. Online real-time multiple spatiotemporal action localisation and prediction. In *ICCV*, 3637–3646.

Song, L.; Zhang, S.; Yu, G.; and Sun, H. 2019. Tacnet: Transition-aware context network for spatio-temporal action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11987–11995.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild.

Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; and Paluri, M. 2015. Learning spatiotemporal features with 3d convolutional networks. *ICCV*.

Xu, H.; Das, A.; and Saenko, K. 2017. R-c3d: Region convolutional 3d network for temporal activity detection. In *ICCV*, 5783–5792.

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

Yang, Z.; Gao, J.; and Nevatia, R. 2017. Spatio-temporal action detection with cascade proposal and location anticipation. *arXiv preprint arXiv:1708.00042*.

Zhang, D.; Dai, X.; Wang, X.; and Wang, Y.-F. 2018. S3d: Single shot multi-span detector via fully 3d convolutional networks. *arXiv preprint arXiv:1807.08069*.

Zhao, J., and Snoek, C. G. 2019. Dance with flow: Two-in-one stream action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9935–9944.

Zhao, Y.; Xiong, Y.; Wang, L.; Wu, Z.; Tang, X.; and Lin, D. 2017. Temporal action detection with structured segment networks. In *ICCV*, 2914–2923.