Noname manuscript No.
(will be inserted by the editor)

Temporal Action Detection with Structured Segment Networks

Yue Zhao · Yuanjun Xiong · Limin Wang · Zhirong Wu · Xiaoou Tang · Dahua Lin

Received: date / Accepted: date

Abstract This paper addresses the problem of detecting the temporal intervals of actions in untrimmed videos. To deal with this important yet challenging task, we present the structured segment network (SSN). It is built on temporal proposals of actions. SSN models the temporal structure of each action instance via a structured temporal pyramid. On top of the pyramid, we further introduce a decomposed discriminative model comprising two classifiers, respectively for classifying actions and determining completeness. This allows the framework to effectively distinguish positive proposals from background or incomplete ones, thus leading to both accurate recognition and precise localization. These components are integrated into a unified network that can be efficiently trained in an end-to-end manner. Additionally, a simple yet effective temporal action proposal scheme, dubbed temporal actionness grouping (TAG) is devised to generate high quality action proposals. We further studied the importance of the decomposed discriminative model and discovered a way to achieve similar accuracy using a single classifier, which is also complementary with the original SSN design. On two challenging benchmarks, THUMOS14 and ActivityNet, our method remarkably outperforms previous state-of-the-art methods, demonstrating superior accuracy and strong adaptivity in handling actions with various temporal structures.

Keywords Temporal Action Detection · Temporal Action Localization · Temporal Action Proposals · Human Action Recognition

Y. Zhao · Y. Xiong · Z. Wu · X. Tang · D. Lin
Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong
E-mail: zy317@ie.cuhk.edu.hk

Xiaoou Tang
E-mail: xtang@ie.cuhk.edu.hk

Dahua Lin
E-mail: dhlin@ie.cuhk.edu.hk

Yuanjun Xiong
Amazon Rekognition
E-mail: bitxiong@gmail.com

Limin Wang
Department of Computer Science and Technology, Naijing University, China
E-mail: lmwang.nju@gmail.com

Zhirong Wu
Microsoft Research Asia
E-mail: xavibrowu@gmail.com

1 Introduction

Temporal action detection has drawn increasing attention from the research community, owing to its numerous potential applications in surveillance, video analytics, interactive entertainment, and other areas Oneata et al (2013); Mettes et al (2015); Yeung et al (2016); Shou et al (2016). A temporal action detection system aims to detect human action instances' temporal intervals from untrimmed, and possibly very long videos. Compared to action recognition, this task is more challenging, as the model is expected to output not only the action category, but also the precise starting and ending time points.

Over the past few years, thanks to the advances in convolutional neural networks, the accuracy of action recognition has been significantly increased Simonyan and Zisserman (2014); Tran et al (2015); Fernando et al (2015); Wang et al (2015, 2016b). Yet, the performances of action detection methods remain unsatisfactory Yuan et al (2016); Yeung et al (2016); Singh and Cuzzolin (2016). Existing methods mostly use the two-stage detection approach, where the action proposals are first produced to reduce the search space and a variety of classifiers are constructed on the proposal to emit the final detection results. For the two-stage action

detection approaches, one major challenge in precise temporal localization is the large number of incomplete action fragments in the proposed temporal regions. Compared with image object detection, where the two-stage idea is also often used, there is an inherent difficulty of temporal action detection in adopting this idea. In untrimmed videos, an action instance can be very long or very short in time. And a small portion of it may have already conveyed enough information for a simple action classification task Schindler and Van Gool (2008). Traditional snippet based classifiers Simonyan and Zisserman (2014); Tran et al (2015); Wang et al (2016b), which are often used as the proposal classifiers. rely on these discriminative snippets of actions, which would make it very hard for them to distinguish short, incomplete proposals, from valid detections (see Fig. 1). Tackling this challenge requires the model's capability of temporal structure analysis. We propose to view the temporal structure as multiple stages of an action instance, *i.e. e.g. starting*, *course*, and *ending*. We argue that the capacity of clearly modeling this stage structure is essential in the success of a temporal action detection model to precisely locate the temporal intervals.

Structural analysis is not new in computer vision. It has been well studied in various tasks, *e.g.* image segmentation Lafferty et al (2001), scene understanding Hoiem et al (2008), and human pose estimation Andriluka et al (2009). Take the most related object detection for example, in deformable part based models (DPM) Felzenszwalb et al (2010), the modeling of the spatial configurations among parts is crucial. Even with the strong expressive power of convolutional networks Girshick et al (2014), explicitly modeling of spatial structures, in the form of spatial pyramids Lazebnik et al (2006); He et al (2014), remains an effective way to achieve improved performance, as demonstrated in a number of state-of-the-art object detection frameworks, *e.g.* Fast R-CNN Girshick (2015) and region-based FCN Li et al (2016).

In the context of video understanding, although temporal structures have played an crucial role in action recognition Niebles et al (2010); Wang et al (2014b); Pirsiavash and Ramanan (2014); Wang et al (2016c), their modeling in temporal action detection was not as common and successful. Snippet based methods Mettes et al (2015); Singh and Cuzzolin (2016) often process individual snippets independently without considering the temporal structures among them. Later works attempt to incorporate temporal structures, but are often limited to analyzing short clips. S-CNN Shou et al (2016) models the temporal structures via the 3D convolution, but its capability is restricted by the underlying architecture Tran et al (2015), which is designed to accommodate only 16 frames. The methods based on recurrent networks Donahue et al (2015); Montes et al (2016) rely on dense snippet sampling and thus are confronted with serious computational challenges when modeling long-term structures. Overall, existing works are limited in two key aspects. First, the tremendous amount of visual data in videos restricts their capability of modeling long-term dependencies in an end-to-end manner. Also, they neither provide *explicit* modeling of different stages in an activity (*e.g. starting* and *ending*) nor offer a mechanism to assess the *completeness*, which, as mentioned, is crucial for accurate action detection.

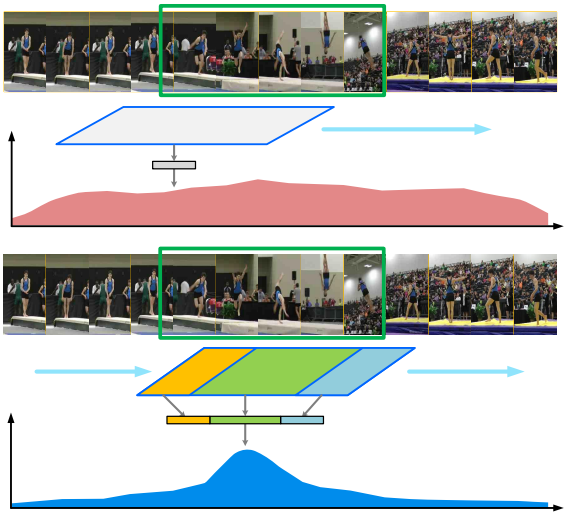

Fig. 1 Importance of modeling stage structures in action detection. We slide window detectors through a video clip with an action instance of "Tumbling" (green box). **Top**: The detector builds features without any stage structure of the action, *e.g.* average pooling throughout the window. It produces high responses whenever it sees any discriminative snippet related to tumbling, making it hard to localize the instance. **Bottom**: SSN detector utilizes stage structures (starting, course, and ending) via structured temporal pyramid pooling. Its response is only significant when the window is well aligned.

In this work, we aim to resolve these limitations and develop an effective technique for temporal action detection. Specifically, we adopt the proven paradigm of two-stage "proposal+classification", but take a significant step forward by utilizing explicit structural modeling in the temporal dimension. In our model, one complete activity instance is considered as a composition of three major stages, namely *starting*, *course*, and *ending*. We introduce structured temporal pyramid pooling to produce a global representation of the entire proposal. Then we introduce a decomposed discriminative model to jointly classify action categories and determine *completeness* of the proposals, which work collectively to output only complete action instances. These components are integrated into a unified network, called *structured segment network* (SSN). We adopt the sparse snippet sampling strategy Wang et al (2016b), which overcomes the computational issue for long-term modeling and enables efficient end-to-end training of SSN. Additionally, we propose to use multi-scale grouping upon the temporal actionness signal to generate action proposals, achieving higher temporal recall with less proposals to further boost the de-

tection performance. To further study the effect the design of SSN, we explored different ways of using a unified classifier for proposal classification, to substitute the decomposed classifiers in SSN. We demonstrated that through careful design we can achieve similar level of accuracy with it, we call this variant of SSN as U-SSN. By ensembling U-SSN and SSN, we are able to achieve higher detection accuracy. This again corroborates the structural modeling is the crucial component in accurate temporal action detection.

The proposed SSN framework excels in the following aspects: 1) It provides an effective mechanism to model the temporal structures of activities, and thus the capability of discriminating between complete and incomplete proposals. 2) It can be efficiently learned in an end-to-end fashion (5 to 15 hours over a large video dataset, *e.g.* ActivityNet), and once trained, can perform fast inference of temporal structures. 3) The method achieves superior detection performance on standard benchmark datasets, establishing new state-of-the-art for temporal action detection.

2 Related Work

In this section, we review previous works related to ours, which we categorize into three parts: (1) action recognition, (2) object detection in images, and (3) temporal action detection in untrimmed videos.

2.1 Action Recognition

Action recognition has been extensively studied in the past few years Laptev (2005); Wang and Schmid (2013); Simonyan and Zisserman (2014); Tran et al (2015); Wang et al (2015, 2016b); Zhang et al (2016). Earlier methods are mostly based on hand-crafted visual features Laptev (2005); Wang and Schmid (2013). In the past several years, the wide adoption of convolutional networks (CNNs) has resulted in remarkable performance gain. CNNs are first introduced to this task in Karpathy et al (2014). Later, two-stream architectures Simonyan and Zisserman (2014) and 3D-CNN Tran et al (2015) are proposed to incorporate both appearance and motion features. These methods are primarily frame-based and snippet-based, with simple schemes to aggregate results. There are also efforts that explore long-range temporal structures via temporal pooling or RNNs Wang et al (2015); Ng et al (2015); Donahue et al (2015). However, most methods assume well-trimmed videos, where the action of interest lasts for nearly the entire duration. Hence, they don't need to consider the issue of localizing the action instances. It is not until recently that there have been attempts Wang et al (2017a) to learn action recognition models from untrimmed videos in the absence of temporal annotations in a weakly-supervised manner.

2.2 Object Detection

The proposed SSN framework is closely related to object detection frameworks Felzenszwalb et al (2010); Girshick et al (2014); Ren et al (2015) in still images, where detection is performed by classifying object proposals into foreground classes and a background class. Traditional object proposal methods rely on dense sliding windows Felzenszwalb et al (2010) and bottom-up methods that exploit low-level boundary cues Van de Sande et al (2011); Zitnick and Dollár (2014). Recent proposal methods based on deep neural networks show better average recall while requiring less candidates Ren et al (2015). Deep models also introduce great modeling capacity for capturing object appearances. With strong visual features, spatial structural modeling Lazebnik et al (2006) remains a key component for detection. In particular, the operation of RoI pooling Girshick (2015) is introduced to model the spatial configuration of object with minimal extra cost. The idea is further reflected in R-FCN Li et al (2016) where the spatial configuration is handled with a position sensitive RoI pooling. The idea of embedding the proposal classifier into the feature extraction network is also adopted in Sec. 4.

2.3 Temporal Action Detection

Previous works on temporal activity detection mainly use sliding windows as candidates and focus on designing hand-crafted feature representations for classification Gaidon et al (2013); Tang et al (2013); Oneata et al (2013); Mettes et al (2015); Yuan et al (2016); Jain et al (2014). Sliding-window based proposals are generated by performing uniform sampling in multiple scales and temporal locations throughout the whole video. To get a high recall, this data-independent sampling will need a lot of scales and locations which lead to prohibitive computational cost. Therefore, one line of recent works are focused on efficient and data-dependent generation of action proposals specifically designed for long videos Caba Heilbron et al (2016); Escorcia et al (2016). Fast TAPCaba Heilbron et al (2016) ranks action candidate using a dictionary which is learned from trimmed action instances described by STIP Laptev (2005). DAP Escorcia et al (2016) and its successor SST Buch et al (2017) use a long short-term memory (LSTM) or gated recurrent unit(GRU) network to encode a sequence of video frames and predicts the location of a fixed number of proposals from the encoded visual contents in a single shot. Although these methods can run at a high FPS, a large amount of action proposals as many as 10^4 are required to achieve a reasonably high recall, which would still be costful for a subsequent classifier.

Another line of works incorporate deep networks into the detection frameworks and obtain improved performance Yeung et al (2016); Shou et al (2016); De Geest et al (2016). S-

CNN Shou et al (2016) proposes a multi-stage CNN which boosts accuracy via a localization network. A more precise temporal boundary can be achieved by applying temporal upsampling via a CDC filter on top Shou et al (2017). However, S-CNN relies on C3D Tran et al (2015) as the feature extractor, which is initially designed for snippet-wise action classification. Extending it to detection with possibly long action proposals needs enforcing an undesired large temporal kernel stride. Another work Yeung et al (2016) uses Recurrent Neural Network (RNN) to learn a glimpse policy for predicting the starting and ending points of an action. Such sequential prediction is often time-consuming for processing long videos and it does not support joint training of the underlying feature extraction CNN. Our method differs from these approaches in that it explicitly models the action structure via structural temporal pyramid pooling. By using sparse sampling, we further enable efficient end-to-end training. Note there are also works on spatial-temporal detection Gkioxari and Malik (2015); Weinzaepfel et al (2015); Mettes et al (2016); Wang et al (2016a); Peng and Schmid (2016); Gu et al (2018) and temporal video segmentation Hoai et al (2011); Li and Loy (2018), which are beyond the scope of this paper.

3 Structured Segment Network

The proposed structured segment network framework, as shown in Figure 2, takes as input a video and a set of temporal action proposals. It outputs a set of predicted *activity instances* each associated with a category label and a temporal range (delimited by a starting point and an ending point). From the input to the output, it takes three key steps. First, the framework relies on a proposal method to produce a set of *temporal proposals* of varying durations, where each proposal comes with a starting and an ending time. The proposal methods will be discussed in detail in Section 5. Our framework considers each proposal as a composition of three consecutive stages, *starting*, *course*, and *ending*, which respectively capture how the action starts, proceeds, and ends. Thus upon each proposal, structured temporal pyramid pooling (STPP) are performed by 1) splitting the proposal into the three stages; 2) building temporal pyramidal representation for each stage; 3) building global representation for the whole proposal by concatenating stage-level representations. Finally, two classifiers respectively for recognizing the activity category and assessing the completeness will be applied on the representation obtained by STPP and their predictions will be combined, resulting in a subset of *complete* instances tagged with category labels. Other proposals, which are considered as either *belonging to background* or *incomplete*, will be filtered out. All the components outlined above are integrated into a unified network, which will be trained in an end-to-end way. For training, we adopt the sparse snippet sampling strategy Wang et al (2016b) to approximate the temporal pyramid on dense samples. By exploiting the redundancy among video snippets, this strategy can substantially reduce the computational cost, thus allowing the crucial modeling of long-term temporal structures.

3.1 Three-Stage Structures

At the input level, a video can be represented as a sequence of T *snippets*, denoted as $(S_t)_{t=1}^{T}$. Here, one snippet contains several consecutive frames, which, as a whole, is characterized by a combination of RGB images and an optical flow stack Simonyan and Zisserman (2014). Consider a given set of N proposals $P = \{p_i = [s_i, e_i]\}_{i=1}^{N}$. Each proposal p_i is composed of a starting time s_i and an ending time e_i. The duration of p_i is thus $d_i = e_i - s_i$. To allow structural analysis and particularly to determine whether a proposal captures a *complete* instance, we need to put it in a context. Hence, we augment each proposal p_i into $p'_i = [s'_i, e'_i]$ with where $s'_i = s_i - d_i/2$ and $e'_i = e_i + d_i/2$. In other words, the augmented proposal p'_i doubles the span of p_i by extending beyond the starting and ending points, respectively by $d_i/2$. If a proposal accurately aligns well with a groundtruth instance, the augmented proposal will capture not only the inherent process of the activity, but also how it starts and ends. Following the three-stage notion, we divide the augmented proposal p'_i into three consecutive intervals: $p^s_i = [s'_i, s_i]$, $p^c_i = [s_i, e_i]$, and $p^e_i = [e_i, e'_i]$, which are respectively corresponding to the *starting*, *course*, and *ending* stages.

3.2 Structured Temporal Pyramid Pooling

As mentioned, the structured segment network framework derives a global representation for each proposal via temporal pyramid pooling. This design is inspired by the success of spatial pyramid pooling Lazebnik et al (2006); He et al (2014) in object recognition and scene classification. Specifically, given an augmented proposal p'_i divided into three stages p^s_i, p^c_i, and p^e_i, we first compute the stage-wise feature vectors $\mathbf{f}^s_i$, $\mathbf{f}^c_i$, and $\mathbf{f}^e_i$ respectively via temporal pyramid pooling, and then concatenate them into a global representation.

Specifically, a stage with interval $[s, e]$ would cover a series of snippets, denoted as $\{S_t | s \le t \le e\}$. For each snippet, we can obtain a feature vector $\mathbf{v}_t$. Note that we can use any feature extractor here. In this work, we adopt the effective two-stream feature representation first proposed in Simonyan and Zisserman (2014). Based on these features, we construct a L-level temporal pyramid where each level evenly divides the interval into B_l parts. For the i-th part of the l-th level, whose interval is $[s_{li}, e_{li}]$, we can derive a pooled

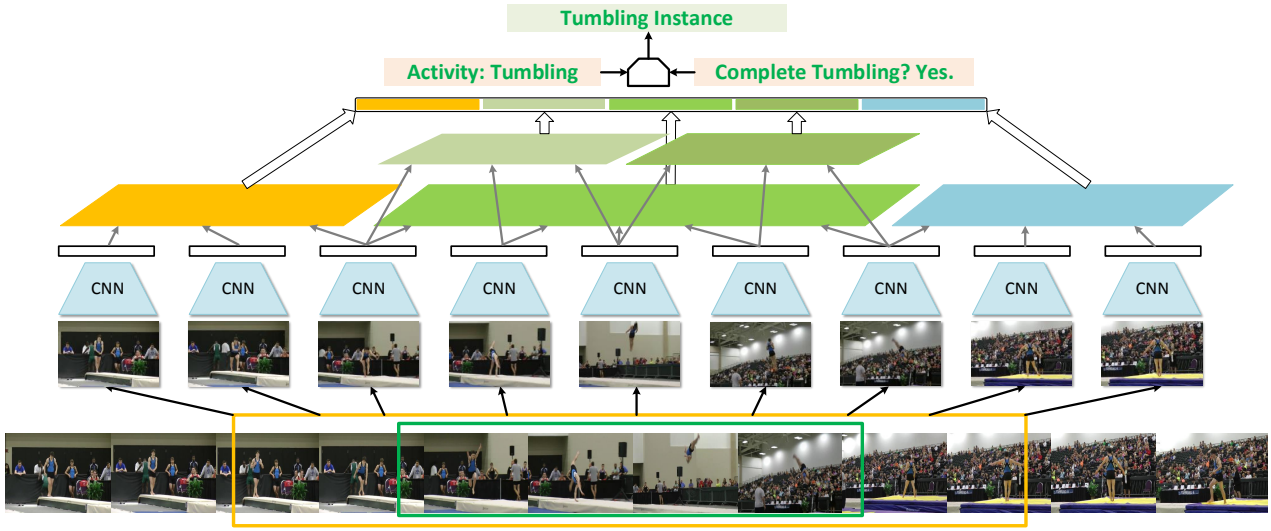

Fig. 2 An overview of the structured segment network framework. On a video from ActivityNet Caba Heilbron et al (2015) there is a candidate region (green box). We first build the augmented proposal (yellow box) by extending it. The augmented proposal is divided into starting (orange), course (green), and ending (blue) stages. An additional level of pyramid with two sub-parts is constructed on the course stage. Features from CNNs are pooled within these five parts and concatenated to form the global region representations. The activity classifier and the completeness classifier operate on the the region representations to produce activity probability and class conditional completeness probability. The final probability of the proposal being positive instance is decided by the joint probability from these two classifiers. During training, we sparsely sample $L = 9$ snippets from evenly divided segments to approximate the dense temporal pyramid pooling.

feature as

$$\mathbf{u}_i^{(l)} = \frac{1}{|e_{li} - s_{li} + 1|} \sum_{t=s_{li}}^{e_{li}} \mathbf{v}_t. \tag{1}$$

Then the overall representation of this stage can be obtained by concatenating the pooled features across all parts at all levels as $\mathbf{f}_i^c = (\mathbf{u}_i^{(l)} | l = 1, \ldots, L,\ i = 1,\ \ldots, B_l)$.

We treat the three stages differently. Generally, we observed that the *course* stage, which reflects the activity process itself, usually contains richer structure *e.g.* this process itself may contain sub-stages. Hence, we use a two-level pyramid, *i.e.* $L = 2, B_1 = 1$, and $B_2 = 2$, for the *course* stage, while using simpler one-level pyramids (which essentially reduce to standard average pooling) for *starting* and *ending* pyramids. We found empirically that this setting strikes a good balance between expressive power and complexity. Finally, the stage-wise features are combined via concatenation. Overall, this construction explicitly leverages the structure of an activity instance and its surrounding context, and thus we call it *structured temporal pyramid pooling* (STPP).

3.3 Activity and Completeness Classifiers

On top of the structured features described above, we introduce two types of classifiers, an *activity classifier* and a set of *completeness classifiers*. Specifically, the *activity classifier* A classifies input proposals into $K+1$ classes, *i.e.* K activity classes (with labels $1, \ldots, K$) and an additional *"background"* class (with label 0). This classifier restricts its scope to the *course* stage, making predictions based on the corresponding feature $\mathbf{f}_i^c$. The *completeness classifiers* $\{C_k\}_{k=1}^K$ are a set of binary classifiers, each for one activity class. Particularly, C_k predicts whether a proposal captures a *complete* activity instance of class k, based on the global representation $\{\mathbf{f}_i^s, \mathbf{f}_i^c, \mathbf{f}_i^e\}$ induced by STPP. In this way, the *completeness* is determined not only on the proposal itself but also on its surrounding context.

Both types of classifiers are implemented as linear classifiers on top of high-level features. Given a proposal p_i, the activity classifier will produce a vector of normalized responses via a softmax layer. From a probabilistic view, it can be considered as a conditional distribution $P(c_i|p_i)$, where c_i is the class label. For each activity class k, the corresponding completeness classifier C_k will yield a probability value, which can be understood as the conditional probability $P(b_i|c_i, p_i)$, where b_i indicates whether p_i is *complete*. Both outputs together form a joint distribution. When $c_i \geq 1$, $P(c_i, b_i|p_i) = P(c_i|p_i) \cdot P(b_i|c_i, p_i)$. Hence, we can define a *unified classification loss* jointly on both types of classifiers. With a proposal p_i and its label c_i:

$$\mathscr{L}_{cls}(c_i, b_i; p_i) = -\log P(c_i|p_i) - 1_{(c_i \geq 1)} \log P(b_i|c_i, p_i). \tag{2}$$

Here, the *completeness* term $P(b_i|c_i, p_i)$ is only used when $c_i \geq 1$, *i.e.* the proposal p_i is not considered as part of the background. Note that these classifiers together with STPP are integrated into a single network that is trained in an *end-to-end* way.

During training, we collect three types of proposal samples: (1) *positive proposals*, *i.e.* those overlap with the clos-

Fig. 3 An illustration of three types of proposals. The ground-truth action instance is denoted in a green box for reference. A positive, background, and incomplete proposal is denoted in a blue, yellow, and red box respectively. We emphasize two points: 1) The positive and incomplete proposal are likely to have similar appearance if we consider only one or two frames inside. Therefore, a structured modeling is necessary; 2) Both background and incomplete proposals are seen as negative samples but they have different appearance. Therefore they should be treated differently.

est groundtruth instances with at least 0.7 IoU; (2) *background proposals*, *i.e.* those that do not overlap with any groundtruth instances; and (3) *incomplete proposals*, *i.e.* those that satisfy the following criteria: 80% of its own span is contained in a groundtruth instance, while its IoU with that instance is below 0.3 (in other words, it just covers a small part of the instance).

Fig An illustration For these proposal types, we respectively have $(c_i > 0, b_i = 1)$, $c_i = 0$, and $(c_i > 0, b_i = 0)$. Each mini-batch is ensured to contain all three types of proposals.

3.4 Location Regression and Multi-Task Loss

With the structured information encoded in the global features, we can not only make categorical predictions, but also refine the proposal's temporal interval itself by *location regression*. We devise a set of location regressors $\{R_k\}_{k=1}^{K}$, each for an activity class. We follow the design in RCNN Girshick et al (2014), but adapting it for 1D temporal regions. Particularly, for a *positive proposal* p_i, we regress the relative changes of both the interval center μ_i and the span ϕ_i (in log-scale), using the closest groundtruth instance as the target. With both the classifiers and location regressors, we define a multi-task loss over an training sample p_i, as:

$$\mathscr{L}_{cls}(c_i, b_i; p_i) + \lambda \cdot 1_{(c_i \geq 1 \ \& \ b_i = 1)} \mathscr{L}_{reg}(\mu_i, \phi_i; p_i). \quad (3)$$

Here, $\mathscr{L}_{reg}$ uses the smooth L_1 loss function Girshick (2015).

3.5 Unifying Activity and Completeness Classifiers

Starting from the original formation of two types of classifiers separately, we take one step further by unifying the $(K+1)$ - class activity classifier and a set of K completeness classifiers stated above as one single classifier. Both incomplete and background proposals are treated as negative samples, *i.e.* the 0-th class, while the positive samples remain the same. We show in Sec. 8 that such unification is non-trivial since simply using a single classifier incurs inferior performance. To remedy such loss of performance, we apply the following practices. First, the model parameters are initialized with the model pre-trained on Kinetics Carreira and Zisserman (2017), a larger video dataset. Such pre-trained model provides stronger discriminative capacity of activity modeling. Second, a fully-connected layer is added between the STPP and the final classifier to reduce feature dimension for ease of training. Third, due to the imbalance between incomplete and background proposals, we apply a sample mining method similar to online hard example mining (OHEM) Shrivastava et al (2016), which only calculates gradients of samples with the highest loss value within a minibatch. We term this approach as unified structured segment network (U-SSN).

4 Efficient Training and Inference with SSN

The huge amount of frames pose a serious challenge in computational cost to video analysis. Our structured segment network also faces this challenge. This section presents two techniques which we use to reduce the cost and enable end-to-end training.

4.1 Training with sparse sampling

The structured temporal pyramid, in its original form, relies on densely sampled snippets. This would lead to excessive computational cost and memory demand in end-to-end training over long proposals – in practice, proposals that span over hundreds of frames are not uncommon. However, dense sampling is generally unnecessary in our framework. Particularly, the *pooling* operation is essentially to collect feature statistics over a certain region. Such statistics can be well approximated via a subset of snippets, due to the high redundancy among them.

Motivated by this, we devise a *sparse snippet sampling scheme*. Specifically, given a augmented proposal p_i', we evenly divide it into $L = 9$ segments, randomly sampling only one snippet from each segment. Structured temporal pyramid

pooling is performed for each pooling region on its corresponding segments. This scheme is inspired by the segmental architecture in Wang et al (2016b), but differs in that it operates within STPP instead of a global average pooling. In this way, we fix the number of features needed to be computed regardless of how long the proposal is, thus effectively reducing the computational cost, especially for modeling long-term structures. More importantly, this enables end-to-end training of the entire framework over a large number of long proposals.

4.2 Inference with reordered computation

In testing, we sample video snippets with a fixed interval of 6 frames, and construct the temporal pyramid thereon. The original formulation of temporal pyramid first computes pooled features and then applies the classifiers and regressors on top which is not efficient. Actually, for each video, hundreds of proposals will be generated, and these proposals can significantly overlap with each other – therefore, a considerable portion of the snippets and the features derived thereon are shared among proposals.

To exploit this redundancy in the computation, we adopt the idea introduced in position sensitive pooling Li et al (2016) to improve testing efficiency. Note that our classifiers and regressors are both linear. So the key step in classification or regression is to multiply a weight matrix $\mathbf{W}$ with the global feature vector $\mathbf{f}$. Recall that $\mathbf{f}$ itself is a concatenation of multiple features, each pooled over a certain interval. Hence the computation can be written as $\mathbf{W}\mathbf{f} = \sum_j \mathbf{W}_j \mathbf{f}_j$, where j indexes different regions along the pyramid. Here, $\mathbf{f}_j$ is obtained by *average pooling* over all snippet-wise features within the region r_j. Thus, we have

$$\mathbf{W}_j \mathbf{f}_j = \mathbf{W}_j \cdot \mathbb{E}_{t\sim r_j}[\mathbf{v}_t] = \mathbb{E}_{t\sim r_j}[\mathbf{W}_j \mathbf{v}_t]. \quad (4)$$

$\mathbb{E}_{t\sim r_j}$ denotes the average pooling over r_j, which is a linear operation and therefore can be exchanged with the matrix multiplication. Eq (4) suggests that the linear responses *w.r.t.* the classifiers/regressors can be computed *before* pooling. In this way, the heavy matrix multiplication can be done in the CNN for each video over all snippets, and for each proposal, we only have to pool over the network outputs. This technique can reduce the processing time after extracting network outputs from around 10 seconds to less than 0.5 second per video on average.

5 Temporal Proposals via Actionness Grouping

In general, SSN accepts arbitrary proposals, *e.g.* sliding windows Shou et al (2016); Yuan et al (2016). Yet, an effective proposal method can produce more accurate proposals, and thus allowing a small number of proposals to reach a certain level of performance. In this work, we devise an effective proposal method called *temporal actionness grouping (TAG)*.

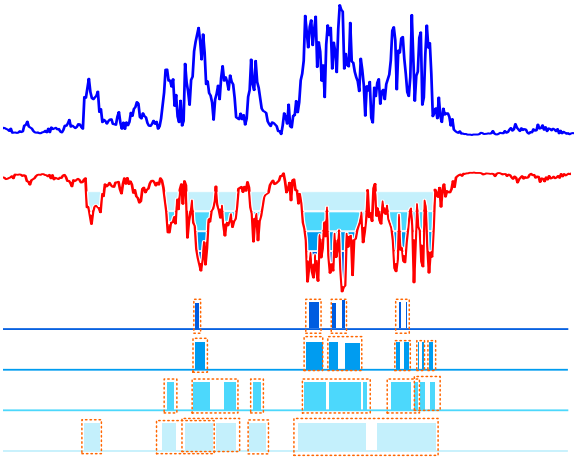

Fig. 4 Visualization of the temporal actionness grouping process for proposal generation. **Top**: Actionness probabilities as a 1D signal sequence. **Middle**: The complement signal. We flood it with different levels γ. **Bottom**: Regions obtained by different flooding levels. By merging the regions according to the grouping criterion, we get the final set of proposals (in orange color).

This method uses an actionness classifier to evaluate the binary *actionness probabilities* for individual snippets. The use of binary actionness for proposals is first introduced in spatial action detection by Wang et al (2016a). Here we utilize it for temporal action detection.

Our basic idea is to find those continuous temporal regions with mostly high actionness snippets to serve as proposals. To this end, we repurpose a classic watershed algorithm Roerdink and Meijster (2000), applying it to the 1D signal formed by a sequence of *complemented* actionness values, as shown in Figure 4. Imagine the signal as 1D terrain with heights and basins. This algorithm floods water on this terrain with different *"water level"* (γ), resulting in a set of *"basins"* covered by water, denoted by $G(\gamma)$. Intuitively, each "basin" corresponds to a temporal region with high actionness. The ridges above water then form the blank areas between basins, as illustrated in Fig. 4.

Given a set of basins $G(\gamma)$, we devise a grouping scheme similar to Pont-Tuset et al (2017), which tries to connect small basins into proposal regions. The scheme works as follows: it begins with a seed basin, and consecutively absorbs the basins that follow, until the fraction of the basin durations over the total duration (*i.e.* from the beginning of the first basin to the ending of the last) drops below a certain threshold τ. The absorbed basins and the blank spaces between them are then grouped to form a single proposal. We treat each basin as seed and perform the grouping procedure to obtain a set of proposals denoted by $G'(\tau,\gamma)$. Note that we do not choose a specific combination of τ and γ. Instead

we uniformly sample τ and γ in the range $(0,1)$ with an even step of 0.05. The combination of these two thresholds leads to multiple sets of regions. We then take the *union* of them. Finally, we apply non-maximal suppression (NMS) to the union set with IoU threshold 0.95, to filter out highly overlapped proposals. The retained proposals will be fed to the SSN framework.

Algorithm 1 lists the procedural details. First, assume that we have obtained a sequences of T snippet-level actionness scores, denoted as $\{A_1, A_2, \ldots, A_T\}$, here T is the length of video snippets. In order to produce proposals of different granularities, we vary the *actionness thresholds* $\{\tau_i\}$ and the *tolerance thresholds* $\{\gamma_j\}$. Each iteration of the procedure will generate a collection of proposals by grouping fragments, based on a certain setting of τ_i and γ_j. In particular, the fragment grouping operation is described in Algorithm 2.

Algorithm 1 Temporal Actionness Proposal Grouping

INPUT: $\{A_t\}_{t=1}^{M}$, $\{\tau_i\}_{i=1}^{N}$, $\{\gamma_j\}_{j=1}^{K}$
  $B \leftarrow \{\}$
  for $i \leftarrow 1, N$ **do**
    for $t \leftarrow 1, T$ **do**
      $L_t \leftarrow \delta(S_m^t >= \tau_i)$
    end for
    for $j \leftarrow 1, K$ **do**
      $B \leftarrow B + AtomicGroup(\{L_i\}_{i=1}^{T}, \gamma_j)$
    end for
  end for
  $B \leftarrow NMS(B, 0.95)$
OUTPUT: B

Algorithm 2 Atomic Grouping Process

procedure ATOMICGROUP($\{L_t\}_{t=1}^{T}, \gamma$)
  $Box \leftarrow \{\}$
  $g \leftarrow 0$, $C^+ \leftarrow 0$, $C^- \leftarrow 0$, $s \leftarrow 0, e \leftarrow 0$
  for $t \leftarrow 1, T$ **do**
    if $L_t = 1$ **then**
      $s \leftarrow t \times (1-g) + s \times g$, $e \leftarrow t$
      $g \leftarrow 1$, $C^+ \leftarrow C^+ + 1$
    else if $g = 1$ **then**
      $C^- \leftarrow C^- + 1$
      if $C^-/(t-s+1) \geq \gamma$ **then**
        $Box \leftarrow Box + (s, e)$
        $g \leftarrow 0, C^+ \leftarrow 0, C^- \leftarrow 0$, $s \leftarrow 0$
      end if
    end if
  end for
end procedure

6 Experimental Settings

In this section, we describe the detailed experimental setting. The experiments are conducted on two large-scale action detection benchmark datasets: *ActivityNet* Caba Heilbron et al (2015) and *THUMOS14* Jiang et al (2014). First, we introduce these datasets and their experimental setup. Next, we describe the implementation details of our methods.

6.1 Datasets and Evaluation Protocol

The **ActivityNet Dataset** Caba Heilbron et al (2015) [1] has two versions, *v1.2* and *v1.3*. The former contains 9682 videos in 100 classes, while the latter, which is a superset of v1.2 and was used in the ActivityNet Challenge 2016, contains 19994 videos in 200 classes. In each version, the dataset is divided into three disjoint subsets, training, validation, and testing, by 2:1:1. The annotation for the testing subset is held out by the test server, while the training and validation subsets are publicly accessible. We train our model on the training set and evaluate it on the validation set except that the model submitted to the test server is trained using both the training and validation subset.

Following the conventions of the ActivityNet 2016 Challenge, we report mean average precision (mAP) at different IoU thresholds $0.5, 0.75, 0.95$ on both versions of ActivityNet dataset. The average of the mAP values (average mAP) with IoU thresholds $[0.5 : 0.05 : 0.95]$ is used to compare the performance between different methods.

The **THUMOS14 Dataset** Jiang et al (2014) [2] has 1010 videos for validation and 1574 videos for testing. This dataset does not provide the training set by itself. Instead, the UCF101 dataset Soomro et al (2012), a trimmed video dataset is appointed as the official training set. Following the standard practice, we train out models on the validation set and evaluate them on the testing set since the ground-truth of test data is made available has been the competition. On these two subsets, 220 and 212 videos have temporal annotations in 20 classes, respectively. 2 falsely annotated videos ("270","1496") in the test set are excluded in evaluation.

Following the conventions of the THUMOS Challenge 2014, we report mean average precision (mAP) at IoU thresholds ranging from 0.1 to 0.5 with a step of 0.1. The mAP at IoU of 0.5 is used for comparing results between different methods.

In the following experiments, we compare our method with the states of the art on both *THUMOS14* and *ActivityNet v1.3*, and perform ablation studies on *ActivityNet v1.2*.

[1] http://activity-net.org/index.html
[2] http://crcv.ucf.edu/THUMOS14/

6.2 Implementation Details

We train the structured segment network in an end-to-end manner, with raw video frames and action proposals as the input. Two-stream CNNs Simonyan and Zisserman (2014) are used for feature extraction: the spatial and temporal streams are used to harness both the appearance and motion features. To generate the inputs to the temporal stream, we adopt the TV-L1 optical flow algorithm Zach et al (2007) which is implemented in OpenCV with CUDA. The x-axis and y-axis components are then linearly rescaled to a range of 0-255 and compressed into a grey-scale image using JPEG respectively.

The binary actionness classifiers underlying the TAG proposals are trained with TSN Wang et al (2016b) on the training subset of each dataset. In the stage of classification for proposal, the activity and completeness classifiers are trained likewise. We use SGD to learn CNN parameters in our framework, with batch size 128 and momentum 0.9. We initialize the CNNs with pre-trained models from ImageNet Deng et al (2009). The initial learning rates are set to 0.001 for RGB networks and 0.005 for optical flow networks. In each minibatch, we keep the ratio of three types of proposals, namely *positive*, *background*, and *incomplete*, to be 1:1:6. For the completeness classifiers, only the samples with loss values ranked in the first $1/6$ of a minibatch are used for calculating gradients, which resembles online hard negative mining Shrivastava et al (2016). On both versions of ActivityNet, the RGB and optical flow branches of the two-stream CNN are respectively trained for $9.5K$ and $20K$ iterations, with learning rates scaled down by 0.1 after every $4K$ and $8K$ iterations, respectively. On THUMOS14, these two branches are respectively trained for $1K$ and $6K$ iterations, with learning rates scaled down by 0.1 per 400 and 2500 iterations. For the temporal stream, gradient over 20 will be clipped to ensure convergence especially at the beginning of the training, since the optical flow extracted from untrimmed videos contains more noise than that extracted from trimmed videos.

7 Experimental Results on Temporal Action Proposals

In this section, we analyze the performance of our proposed action proposal method via actionness grouping. The quality of temporal action proposals is assessed by the metrics from Hosang et al (2016) following the previous practice in Escorcia et al (2016). Specifically, we calculate Average Recall (AR) at IoU thresholds ranging from 0.5 to 1 with a step of 0.05, given a certain number of proposals. Different from object detection proposals in 2D dimension, the temporal action proposals only vary in temporal dimension. A proposal at IOU threshold of 0.5 may not be precise enough for localization. Therefore, we also measure the recall rate at different IOU with N proposals, denoted by $Recall@N$.

Table 1 Evaluation of different grouping parameters for TAG on ActivityNet v1.2. "AR" refers to the average recall rates.

AR (# of proposals)		τ		
		0.1:0.2:0.9	0.05:0.1:0.95	0.05:0.05:0.95
γ	0.1:0.2:0.9	58.9 (22)	63.9 (45)	66.5 (80)
	0.1:0.1:0.9	60.6 (24)	65.2 (50)	67.8 (85)
	0.1:0.05:0.9	61.8 (26)	66.3 (52)	68.7 (89)

We mainly focus on three aspects, *i.e.* (1) the performance of our approach with various grouping parameters, (2) the quality of temporal action proposals compared with other methods, and (3) the ability to generalize to unseen action classes. As of applying the generated proposals to the framework of temporal action detection, we leave it to Sec. 8.

Evaluation of Grouping Parameters. The proposal generation includes two sets of hyper-parameters, *i.e.*, the actionness thresholds γ and tolerance thresholds τ. We vary the sampling step of γ and τ from 0.05 to 0.1 and 0.2 and obtain the results in Table 1. We find TAG to be robust to such variations and achieves reasonable average recall rate ($\sim 58.9\%$) with an amount of proposals as small as 22 on average.

Comparison with Other Proposal Approaches. We compare our TAG scheme on ActivityNet v1.2 with common sliding windows as well as other state-of-the-art proposal methods, including SCNN-prop, a proposal networks presented in Shou et al (2016), Sparse-prop Caba Heilbron et al (2016), DAP Escorcia et al (2016). For the sliding window scheme, we use 20 exponential scales starting from 0.3 second long and step sizes of 0.4 times of window lengths.

We first evaluate the average recall rates, which are summarized in Table 2. We can see that TAG proposal have higher recall rates with the same number of proposals. Then we investigate the quality of its proposals by plotting the recall rates from different proposal methods at different IoU thresholds in Fig. 5. We can see TAG retains relatively high recall at high IoU thresholds, demonstrating that the proposals from TAG are generally more accurate.

On the THUMOS14 dataset, we compare our TAG scheme with more recent proposal methods, including Sparse-prop Caba Heilbron et al (2016), BoFrag Mettes et al (2015), SCNN-prop Shou et al (2016), DAP Escorcia et al (2016), and SST Buch et al (2017). A representative spatio-temporal proposal method, APT Van Gemert et al (2015) is also included by simply projecting spatio-temporal proposals to the temporal only. The results are illustrated in Fig. 6. From the AR curve, our TAG method outperforms other methods by a large margin. In terms of average recall computed over a higher temporal

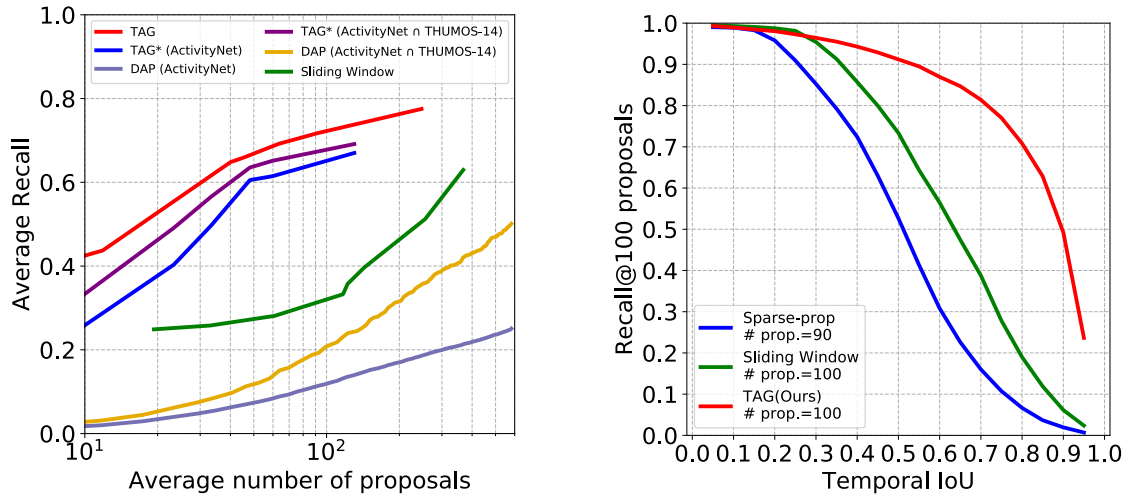

Fig. 5 Comparison of the TAG proposal with state-of-the-art proposal generation methods. The results are reported on the ActivityNet v1.2. **(left)** TAG achieves higher average recall with same amount of proposals. TAG* denotes the model that is trained on THUMOS-14 and tested on the whole ActivityNet v1.2, *i.e.* the *ActivityNet* line or the shared categories of both datasets *i.e.* the *ActivityNet*$\cap$*THUMOS-14* line. **(right)** Recall rate at different tIoU thresholds with around 100 proposals. High recall rates at high IoU thresholds (> 0.7) indicate better proposal quality.

Table 2 Comparison between different temporal action proposal methods with same number of proposals. "AR" refers to the average recall rates. "-" indicates the result is not available.

Proposal Method	**THUMOS14**		**ActivityNet v1.2**	
	# Prop.	AR	# Prop.	AR
Sliding Windows	204	21.2	100	34.8
SCNN-prop Shou et al (2016)	200	20.0	-	-
TAP Caba Heilbron et al (2016)	200	23.0	90	14.9
DAP Escorcia et al (2016)	200	37.0	100	12.1
TAG	200	**48.9**	100	**71.7**

IOU range (0.7-0.95), the performance is more visible. From the *Recall@1000* curve shown in the lower part of Fig. 6, we can clearly see that TAG achieves competitive results against other methods at lower tIOU thresholds (0-0.5) and notable performance gain at higher tIOU thresholds (0.8-0.95). The largest improvement achieved at tIOU = 0.8 reaches nearly 20%, verifying the preciseness of our temporal action proposals.

Generalization Ability to Unseen Action Classes. The actionness can be seen as a generic measurement to a wide variety of activities. To verify it, we test the generalization ability of the actionness classifier by applying the actionness classifier trained on ActivityNet v1.2 directly on THUMOS14 and ActivityNet v1.3. We can still achieve a reasonable average recall of 39.6%, while the one trained specifically on THUMOS14 achieves 48.9% in Table 2. To have a better understanding, we compare the average recall on overlapped classes, *i.e.* those seen in ActivityNet v1.2 and unseen ones in Table 3. We see that in general the AR values do not severely decrease when the actionness classifier is used to propose candidates for unseen classes. In particular, on the overlapped part of THUMOS14 dataset[3], TAG trained from ActivityNet v1.2 only drops a little (48.9% to 46.6%) compared to that both trained and evaluated on THUMOS14. The recall across the remaining 10 unseen classes decreases more but is also competitive with previous methods which see all action classes. For ActivityNet v1.3, the difference between the two parts (68.1% vs. 66.4%) is smaller probably because ActivityNet v1.3 shares similar action types and data distribution with its subset.

Previous works Escorcia et al (2016) also test the generalization ability of proposal generation by training on THUMOS14 and testing on ActivityNet v1.2. Hence, in Fig. 5, we show the average recall curve of our method (denoted by TAG* to show the difference) against DAP Escorcia et al (2016) on two sets: *ActivityNet* (all 100 categories on v1.2)

[3] To be specific, the 10 classes are: "Clean and Jerk", "Hammer Throw", "High jump", "Javelin Throw", "Long Jump", "Pole Vault", "Shotput", "Tennis Swing", "Throw Discus", "Volleyball Spiking". Note that THUMOS14 has two classes named "Cricket Bowling" and "Cricket Shot" while ActivityNet v1.2 also has one called "Cricket". However we categorize the two classes into the unseen part since the single label in ActivityNet is unable to distinguish these two specific actions.

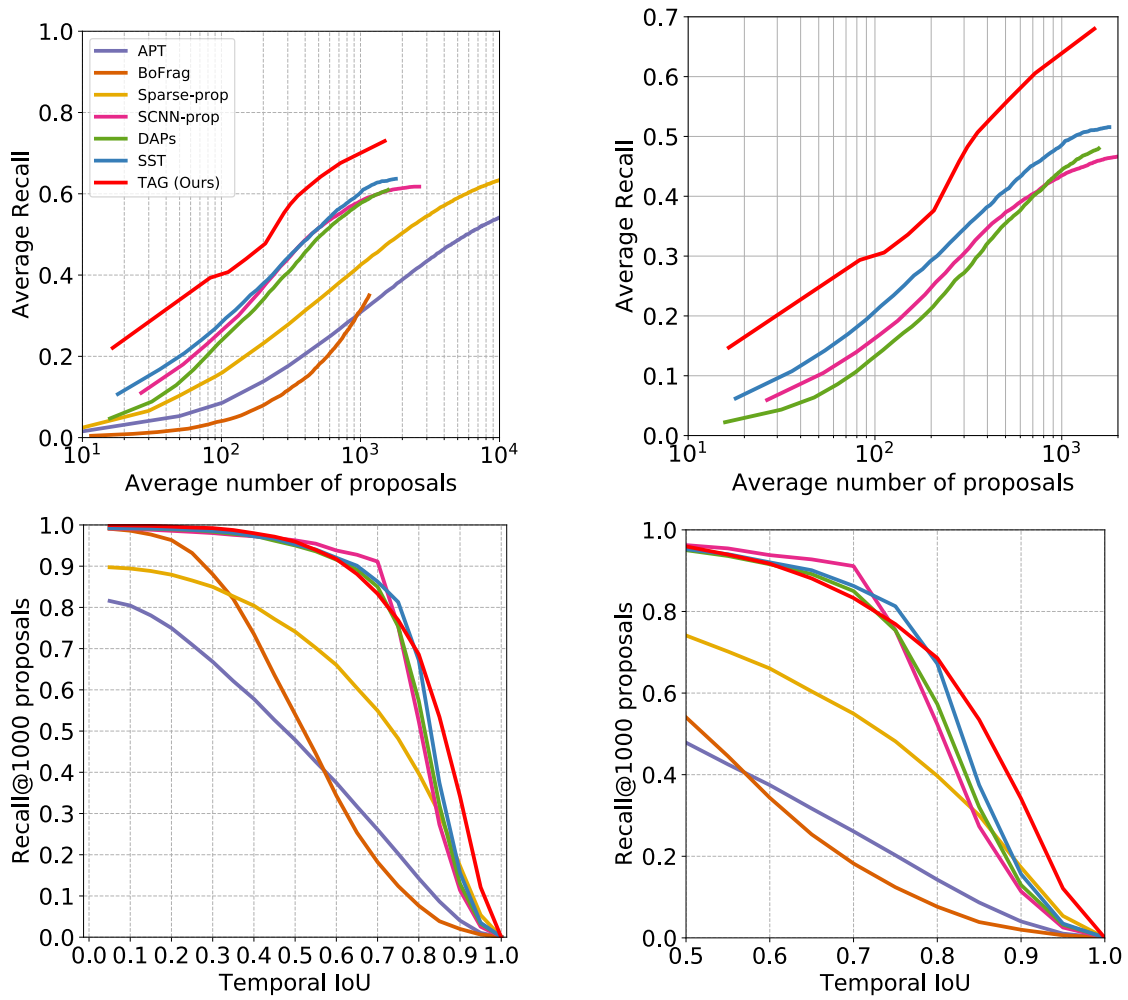

Fig. 6 Comparison of the TAG proposal with state-of-the-art proposal generation methods. The results are reported on the THUMOS14. **(upper left)** TAG achieves higher average recall with same amount of proposals. **(lower left)** The performance gain is larger when the average recall is computed over a higher temporal IOU (0.7-0.95). **(lower left)** Recall rate at different tIoU thresholds with around 1000 proposals. **(lower right)** Recall rate at higher tIoU thresholds (0.5-1.0) with around 1000 proposals.

Table 3 The comparison of average recall on overlapped classes and unseen ones of THUMOS14 and ActivityNet v1.3 using an actionness classifier trained on ActivityNet v1.2 only.

	THUMOS14		ActivityNet v1.3	
	Overlapped	Unseen	Overlapped	Unseen
	(10 classes)	(10 classes)	(100 classes)	(100 classes)
AR	46.6	28.3	68.1	66.4

and *ActivityNet∩THUMOS-14* (on categories shared by both datasets). Compared with TAG trained on ActivityNet v1.2, we can observe that TAG* trained on THUMOS14 only incurs a slight drop of performance on both the whole and overlapped dataset. The gap is closer as the average number of proposals increases. This show the generalization ability of the proposed TAG method.

8 Experimental Results on Temporal Action Detection

In this section, we report the performance of our method on the datasets stated in Sec. 6. First, we investigate the impact of different components in the framework via a set of ablation studies. Then we compare the performance of SSN with winning entries in the challenge as well as other state-of-the-art approaches. Finally, some qualitative results on both datasets are presented for visualization.

8.1 Ablation Studies

Temporal Action Proposals. We evaluate the performance of action detection using different proposal methods. The detection mAP values using sliding window proposals and TAG proposals are shown in Table 5. The results confirm

Table 4 Comparison between different temporal pooling settings. The setting (1,2)-1 is used in the SSN framework. Please refer to Sec. 8.1 for the definition of these settings.

Average mAP (%)	(1)-0	(1,2)-0	(1)-1	(1,2)-1
Max Pool	13.1	13.5	18.3	18.4
Average Pool	4.48	4.34	24.3	24.6

that, in most cases, the improved proposals can result in improved detection performance.

Structured Temporal Pyramid Pooling. Here we study the influence of different pooling strategies in STPP. We denote one pooling configuration as $(B_1,\ldots,B_K)-A$, where K refers to the number of pyramid levels for the course stage and $B_1,\ldots,B_K$ the number of regions in each level. $A=1$ indicates we use augmented proposal and model the starting and ending stage, while $A=0$ indicates we only use the original proposal (without augmentation). Additionally we compare two within-region pooling methods: average and max pooling. The results are summarized in Table 4. Note that these configurations are evaluated in the stage-wise training scenario. We observe that cases where $A=0$ have inferior performance, showing that the introduction of the stage structure is very important for accurate detection. Also, increasing the depth of the pyramids for the course stage can give slight performance gain. Based on these results, we fix the configuration to $(1,2)-1$ in later experiments.

Classifier Design. In this work, we introduced the activity and completeness classifiers which work together to classify the proposal. We verify the importance of this decomposed design by studying another design that replaces it with a single set of classifiers, for which both *background* and *incomplete* samples are uniformly treated as negative. We perform similar negative sample mining for this setting. The results are summarized in Table 5. We observe that using only one classifier to distinguish *positive* samples from both *background* and *incomplete* would lead to worse result even with negative mining, where mAP decreased from 23.7% to 17.9%. We attribute this performance gain to the different natures of the two negative proposal types, which require different classifiers to handle.

Location Regression & Multi-Task Learning. Because of the contextual information contained in the starting and ending stages of the global region features, we are able to perform location regression. We measure the contribution of this step to the detection performance in Table 5. From the results we can see that the location regression and multi-task learning, where we train the classifiers and the regressors together in an end-to-end manner, always improve the detection accuracy.

Table 5 Ablation study on ActivityNet Caba Heilbron et al (2015) v1.2. Overall, end-to-end training is compared against stage wise training. We evaluate the performance using both sliding window proposals ("SW") and TAG proposals ("TAG"), measured by mean average precision (mAP). Here, "STPP" refers to structure temporal pyramid pooling. "Act. + Comp." refers to the use of two classifiers design. "Loc. Reg" denotes the use the location regression.

	Stage-Wise				End-to-End	
STPP		✓	✓	✓	✓	✓
Act. + Comp.			✓	✓	✓	✓
Loc. Reg.				✓		✓
SW	0.56	5.99	16.4	18.1	-	-
TAG	4.82	17.9	24.6	24.9	24.8	25.9

Training: Stage-wise v.s. End-to-end. While the structured segment network is designed for end-to-end training, it is also possible to first densely extract features and train the classifiers and regressors with SVM and ridge regression, respectively. We refer to this training scheme as stage-wise training. The linear SVM for the stage-wise training are from the implementation provided by *scikit-learn* Pedregosa et al (2011). We compare the performance of end-to-end training and stage-wise training in Table 5. We observe that models from end-to-end training can slightly outperform those learned with stage-wise training under the same settings. This is remarkable as we are only sparsely sampling snippets in end-to-end training, which also demonstrates the importance of jointly optimizing the classifiers and feature extractors and justifies our framework design. Besides, end-to-end training has another major advantage that it does not need to store the extracted features for the training set, which could become quite storage intensive as training data grows.

Pre-training on large-scale video dataset. The **Kinetics Human Action Video dataset** Carreira and Zisserman (2017) contains 300,000 10-second-long video clips in 400 classes collected from YouTube. It was shown in Carreira and Zisserman (2017) that deep models pre-trained on this dataset can bring considerable improvement on smaller-scale datasets such as UCF-101 Soomro et al (2012) in the action recognition task. Here, we investigate whether models pre-trained on this dataset for the purpose of action recognition can be adapted to the domain of temporal action detection.

We choose the base model to be a BN-Inception network Ioffe and Szegedy (2015) as well as an Inception-V3 network Szegedy et al (2016) initialized with parameters learned from ImageNet Deng et al (2009) and then trained with TSN Wang et al (2017b) on Kinetics Carreira and Zisserman (2017). The results are shown in Table 6. It is observed that models pre-trained from Kinetics Carreira and Zisserman (2017) show considerable improvement over those pre-trained from ImageNet Deng et al (2009) only. This indicates that large-scale video datasets, such as Kinetics Car-

Table 6 Comparison of performance of pre-training on the Kinetics dataset. The results are reported on the ActivityNet v1.2 val set.

Base model	Pre-train	Average mAP
BN-Inception	ImageNet	26.8%
	ImageNet + Kinetics	28.6%
Inception-V3	ImageNet	27.4%
	ImageNet + Kinetics	29.4%

Table 7 Comparison of using original SSN and unified SSN (U-SSN). The results are reported on ActivityNet v1.3 val set using ResNet-101 He et al (2016) as the underlying architecture.

method	average mAP
SSN	29.2%
U-SSN	28.0%
SSN + U-SSN	29.7%

reira and Zisserman (2017), serve as a better source for temporal action detection compared to image datasets.

Unifying Activity and Completeness Classifiers In previous discussion, we find that simply combining activity and completeness classifiers into a single classifier drastically reduces performance (from 24.6% to 17.9% using TAG proposals). To remedy such loss of performance, we apply the following practices. First, the parameter is initialized with the model pre-trained on Kinetics Carreira and Zisserman (2017) to ease training. Second, a fully-connected layer is inserted between the STPP module and the activity classifier for dimension reduction. No performance gain is observed as we increase the number of fc layers. What's more, OHEM Shrivastava et al (2016)-like operation of updating losses is applied amongst all types of proposals *i.e.*, positive, incomplete and background proposals. Here we present the results of the unified SSN in Table 7. We summarize the following observations: First, the gap between the original SSN and the unified SSN is greatly narrowed down with the modifications stated above; Second, the detection results of SSN and U-SSN can be fused by simply averaging the detection scores and surpass either single model.

8.2 Comparison with the State of the Art

Finally, we compare our method with other state-of-the-art temporal action detection methods on THUMOS14 Jiang et al (2014) and ActivityNet v1.3 Caba Heilbron et al (2015), and report the performances using the metrics described above. Note that the average action duration in THUMOS14 and ActivityNet are 4 and 50 seconds. And the average video duration are 233 and 114 seconds, respectively. This reflects the distinct natures of these datasets in terms of the granularities and temporal structures of the action instances. Hence, strong adaptivity is required to perform consistently well on both datasets.

THUMOS14. On THUMOS14, we compare with the contest results Wang et al (2014a); Oneata et al (2014); Richard and Gall (2016) and those from recent works, including the methods that use segment-based 3D CNN Shou et al (2016), score pyramids Yuan et al (2016), recurrent reinforcement learning Yeung et al (2016), convolutional-de-convolutional (CDC) network Shou et al (2017) and region-based C3D Xu et al (2017). The results are shown in Table 8. In most cases, the proposed method outperforms previous state-of-the-art methods by over 10% in absolute mAP values. By pre-training on the Kinetics dataset, consistent improvements can further be observed.

ActivityNet. The results on the validation and testing set of ActivityNet v1.3 are shown in Table 9 and Table 10. For references, we list the performances of highest ranked entries in the ActivityNet 2016 challenge[4] as well as the results reported in recent papers such as Convolutional-De-Convolutional Networks(CDC) Shou et al (2017), R-C3D Xu et al (2017), and temporal context network Dai et al (2017). We submit our results to the test server of ActivityNet v1.3 and report the detection performance on the testing set. The proposed framework, using a single model with Inception-V3 Szegedy et al (2016) as the backbone network, is able to achieve an average mAP of 28.28% and performs well at high IOU thresholds, *i.e.*, 0.75 and 0.95. This clearly demonstrates the superiority of our method. By fine-tuning the model pre-trained on the Kinetics dataset, the single model with BN-Inception Ioffe and Szegedy (2015) as the backbone network can further increase the average mAP to 29.34%, which is a remarkable improvement due to smaller model complexity. In the final submission, which composes an ensemble of SSN and U-SSN with multiple backbone networks, the detection average mAP reaches 31.86%.

8.3 Visualization and Analysis of Detection Results

We visualize some detection results obtained on the validation set of ActivityNet v1.2 dataset and the testing set of THUMOS'14 dataset in Fig. 9 and Fig. 10, respectively. The qualitative results validate that our methods can produce temporal bounding boxes with high accuracy. It is also interesting to see that the temporal extents of action instances in the two datasets differ a lot and our framework is capable of detecting actions of different durations.

[4] The ActivityNet 2016 challenge summary is provided here: `http://activity-net.org/challenges/2016/data/anet_challenge_summary.pdf`

Table 8 Action detection results on THUMOS14, measured by mAP at different IoU thresholds α. The upper half of the table shows challenge results back in 2014. "SSN*" indicates metrics calculated in the PASCAL-VOC style used by ActivityNet Caba Heilbron et al (2015). "SSN†" indicates metrics calculated originally used by THUMOS Challenge 14 Jiang et al (2014).

THUMOS14, **mAP@α**					
Method	0.1	0.2	0.3	0.4	0.5
Wang *et. al.* Wang et al (2014a)	18.2	17.0	14.0	11.7	8.3
Oneata *et. al.* Oneata et al (2014)	36.6	33.6	27.0	20.8	14.4
Richard *et. al.* Richard and Gall (2016)	39.7	35.7	30.0	23.2	15.2
S-CNN Shou et al (2016)	47.7	43.5	36.3	28.7	19.0
Yeung *et. al.* Yeung et al (2016)	48.9	44.0	36.0	26.4	17.1
Yuan *et. al.* Yuan et al (2016)	51.4	42.6	33.6	26.1	18.8
CDC Shou et al (2017)	-	-	40.1	29.4	23.3
R-C3D Xu et al (2017)	54.5	51.5	44.8	35.6	28.9
SSN† (ImageNet) Zhao et al (2017)	**60.3**	**56.2**	**50.6**	**40.8**	**29.1**
SSN* (ImageNet) Zhao et al (2017)	**66.0**	**59.4**	**51.9**	**41.0**	**29.8**
SSN† (ImageNet+Kinetics)	**62.2**	**58.7**	**53.3**	**44.5**	**33.3**
SSN* (ImageNet+Kinetics)	**69.3**	**63.6**	**55.2**	**44.8**	**34.3**

Table 9 Action detection results on ActivityNet v1.3 validation set, measured by mean average precision (mAP) for different IoU thresholds α and the average mAP of IoU thresholds from 0.5 to 0.95. The upper half of the table shows challenge results in 2016, mostly reported in the challenge summary.

ActivityNet v1.3 (validation), **mAP@α**				
Method	0.5	0.75	0.95	Average
Wang *et. al.* Wang and Tao (2016)	43.65	-	-	-
Montes *et. al.* Montes et al (2016)	22.51	-	-	-
Singh *et. al.* Singh and Cuzzolin (2016)	34.47	-	-	-
R-C3D Xu et al (2017)	26.8	-	-	-
TCN Dai et al (2017)	36.17	21.12	3.89	-
CDC Shou et al (2017)	45.3	26.0	0.2	23.8
SSN (ImageNet+Kinetics)	**45.88**	**28.56**	**6.39**	**28.85**

Table 10 Action detection results on ActivityNet v1.3 testing dataset, measured by mean average precision (mAP) for different IoU thresholds α and the average mAP of IoU thresholds from 0.5 to 0.95. The upper half of the table shows challenge results in 2016.

ActivityNet v1.3 (testing), **mAP@α**				
Method	0.5	0.75	0.95	Average
Wang *et. al.* Wang and Tao (2016)	42.48	2.88	0.06	14.62
Montes *et. al.* Montes et al (2016)	22.37	14.88	4.45	14.81
Singh *et. al.* Singh et al (2016)	28.67	17.78	2.88	17.68
Singh *et. al.* Singh and Cuzzolin (2016)	36.40	11.05	0.14	17.83
R-C3D Xu et al (2017)	28.4	-	-	-
TCN Dai et al (2017)	37.49	23.47	4.47	23.58
CDC Shou et al (2017)	43.0	25.7	0.2	22.9
SSN (ImageNet) Zhao et al (2017)	**43.26**	**28.70**	**5.63**	**28.28**
SSN (ImageNet+Kinetics), single model	-	-	-	**29.34**
SSN + U-SSN (ImageNet+Kinetics), ensemble	-	-	-	**31.86**

It may be of interest for readers to see the performance on individual classes. Hence, we first plot the per-class average AP of SSN on the ActivityNet v1.2 val set in Fig. 7. For comparison, detection results produced by SSN with proposals generated from a sliding window (486 proposals, AR=71%) and a TAG (100 proposals per video, AR=67%) method are listed in parallel, showing that TAG-SSN achieves a higher average AP on most of the classes.

We also plot the top 10 action classes that are easiest and hardest to detect in Fig. 8. We see that the framework can do fairly well on actions which last long and have distinguishable appearance. Different types of dancing, *i.e.* zumba, tango, belly dance, and cumbia, are typical examples. For those ac-

tions which are inconspicuous (for example, drinking coffee/beer), the proposed framework is still far from being satisfactory. Another characterization is that action instances in these poor performing classes have a short duration. The shorter the action instance is, the less likely it will be precisely detected in terms of a high tIOU. This may be alleviated by proposals with more granularities and predictions with higher temporal resolutions. Last but not least, it has to be pointed out that there are certain action pairs that are likely to confuse, for example, "drinking coffee"-"drinking beer", "long jump"-"triple jump", and "bungee jumping"-"platform diving". The confusion of these pairs are understandable to a large extent since they share similar overall appearance. Such issue also occurs in the field of action recognition, which motivates us to design a better model for recognizing these confusing actions.

9 Conclusion

In this paper, we present a generic framework for temporal action detection, which combines a structured temporal pyramid with two types of classifiers, respectively for predicting activity class and completeness. In addition, a simple yet effective temporal action proposal method is invented to generate action proposals with high quality at a small amount in a bottom-up manner.

With this framework, we achieve significant performance gain over state-of-the-art methods on both ActivityNet and THUMOS14. Moreover, we demonstrated that our method is both accurate and generic, being able to localize temporal boundaries precisely and working well for activity classes with very different temporal structures.

Acknowledgements This work is partially supported by the Big Data Collaboration Research grant from SenseTime Group (CUHK Agreement No. TS1610626), the General Research Fund (GRF) of Hong Kong (No. 14236516) and the Early Career Scheme (ECS) of Hong Kong (No. 24204215).

References

Andriluka M, Roth S, Schiele B (2009) Pictorial structures revisited: People detection and articulated pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, pp 1014–1021

Buch S, Escorcia V, Shen C, Ghanem B, Niebles JC (2017) SST: Single-stream temporal action proposals. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, pp 6373–6382

Caba Heilbron F, Escorcia V, Ghanem B, Niebles JC (2015) Activitynet: A large-scale video benchmark for human activity understanding. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 961–970

Caba Heilbron F, Niebles JC, Ghanem B (2016) Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1914–1923

Carreira J, Zisserman A (2017) Quo vadis, action recognition? a new model and the kinetics dataset. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, pp 4724–4733

Dai X, Singh B, Zhang G, Davis LS, Chen YQ (2017) Temporal context network for activity localization in videos. In: The IEEE International Conference on Computer Vision (ICCV), pp 5727–5736

De Geest R, Gavves E, Ghodrati A, Li Z, Snoek C, Tuytelaars T (2016) Online action detection. In: European Conference on Computer Vision (ECCV), Springer, pp 269–284

Deng J, Dong W, Socher R, Li L, Li K, Li F (2009) ImageNet: A large-scale hierarchical image database. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 248–255

Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2625–2634

Escorcia V, Caba Heilbron F, Niebles JC, Ghanem B (2016) Daps: Deep action proposals for action understanding". In: European Conference on Computer Vision (ECCV), pp 768–784

Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(9):1627–1645

Fernando B, Gavves E, M JO, Ghodrati A, Tuytelaars T (2015) Modeling video evolution for action recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 5378–5387

Gaidon A, Harchaoui Z, Schmid C (2013) Temporal localization of actions with actoms. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(11):2782–2795

Girshick R (2015) Fast r-cnn. In: The IEEE International Conference on Computer Vision (ICCV), pp 1440–1448

Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 580–587

Gkioxari G, Malik J (2015) Finding action tubes. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 759–768

Gu C, Sun C, Vijayanarasimhan S, Pantofaru C, Ross DA, Toderici G, Li Y, Ricco S, Sukthankar R, Schmid C, et al (2018) Ava: A video dataset of spatio-temporally localized atomic visual actions. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

He K, Zhang X, Ren S, Sun J (2014) Spatial pyramid pooling in deep convolutional networks for visual recognition. In: European Conference on Computer Vision (ECCV), Springer, pp 346–361

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 770–778

Hoai M, Lan ZZ, De la Torre F (2011) Joint segmentation and classification of human actions in video. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, pp 3265–3272

Hoiem D, Efros AA, Hebert M (2008) Putting objects in perspective. International Journal of Computer Vision (IJCV) 80(1):3–15

Hosang J, Benenson R, Dollár P, Schiele B (2016) What makes for effective detection proposals? IEEE Transactions on Pattern Analysis and Machine Intelligence 38(4):814–830

Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (ICML), pp 448–456

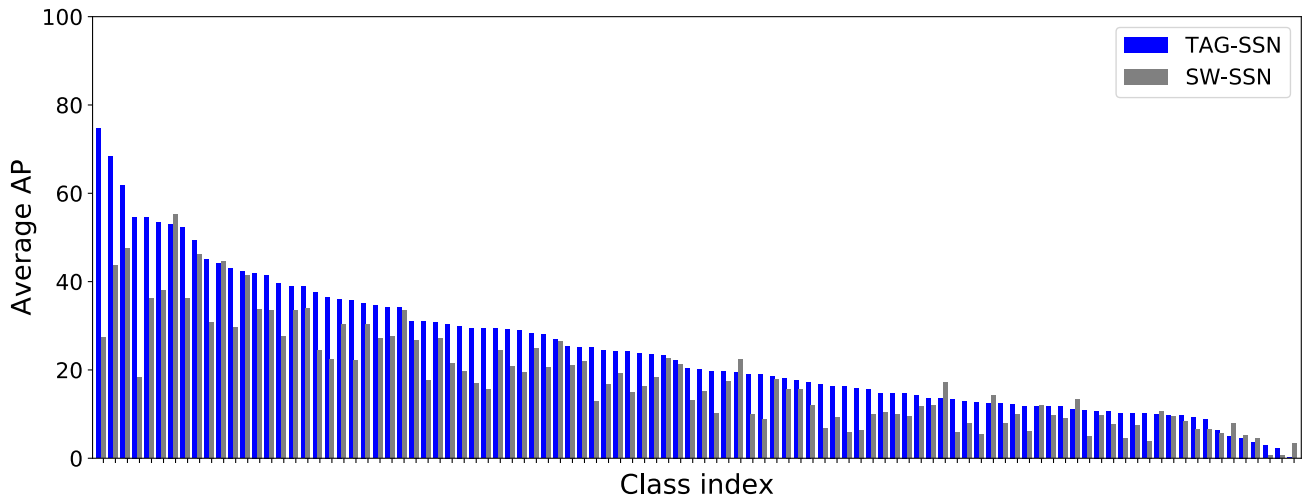

Fig. 7 Per-class average AP@$\{0.5 : 0.05 : 0.95\}$ of SSN using a TAG (TAG-SSN, colored in blue bars) and sliding window (SW-SSN, colored in gray) methods.

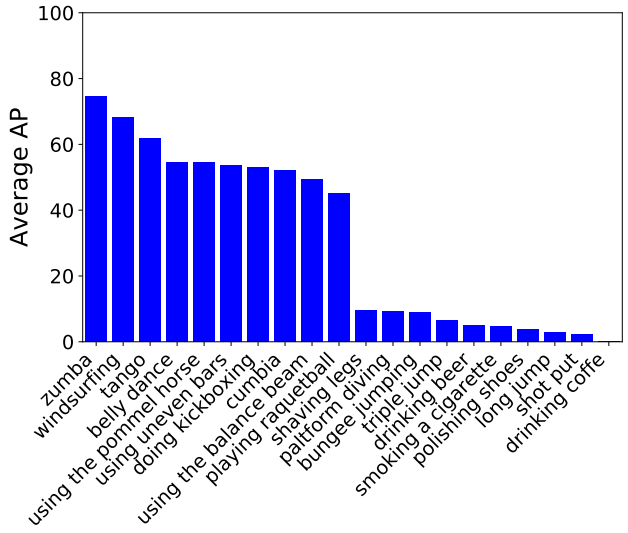

Fig. 8 Ten classes that TAG-SSN performs best and worst in the ActivityNet v1.2 val set with its corresponding average per-class AP@$\{0.5 : 0.05 : 0.95\}$.

Jain M, van Gemert JC, Jégou H, Bouthemy P, Snoek CGM (2014) Action localization by tubelets from motion. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Jiang YG, Liu J, Roshan Zamir A, Toderici G, Laptev I, Shah M, Sukthankar R (2014) THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`

Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1725–1732

Lafferty J, McCallum A, Pereira F, et al (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning (ICML), vol 1, pp 282–289

Laptev I (2005) On space-time interest points. International Journal of Computer Vision (IJCV) 64(2-3)

Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, vol 2, pp 2169–2178

Li X, Loy CC (2018) Video object segmentation with joint re-identification and attention-aware mask propagation. arXiv preprint arXiv:180304242

Li Y, He K, Sun J, et al (2016) R-fcn: Object detection via region-based fully convolutional networks. In: Neural Information Processing Systems (NIPS), pp 379–387

Mettes P, van Gemert JC, Cappallo S, Mensink T, Snoek CG (2015) Bag-of-fragments: Selecting and encoding video fragments for event detection and recounting. In: ACM International Conference on Multimedia Retrieval (ICMR)s, pp 427–434

Mettes P, van Gemert JC, Snoek CG (2016) Spot on: Action localization from pointly-supervised proposals. In: European Conference on Computer Vision (ECCV), Springer, pp 437–453

Montes A, Salvador A, Pascual S, Giro-i Nieto X (2016) Temporal activity detection in untrimmed videos with recurrent neural networks. In: NIPS Workshop on Large Scale Computer Vision Systems

Ng JYH, Hausknecht M, Vijayanarasimhan S, Vinyals O, Monga R, Toderici G (2015) Beyond short snippets: Deep networks for video classification. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 4694–4702

Niebles JC, Chen CW, Fei-Fei L (2010) Modeling temporal structure of decomposable motion segments for activity classification. In: European Conference on Computer Vision (ECCV), Springer, pp 392–405

Oneata D, Verbeek J, Schmid C (2013) Action and event recognition with fisher vectors on a compact feature set. In: The IEEE International Conference on Computer Vision (ICCV), pp 1817–1824

Oneata D, Verbeek J, Schmid C (2014) The lear submission at thumos 2014. In: THUMOS Action Recognition Challenge

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12:2825–2830

Peng X, Schmid C (2016) Multi-region two-stream r-cnn for action detection. In: European Conference on Computer Vision (ECCV),

Springer
Pirsiavash H, Ramanan D (2014) Parsing videos of actions with segmental grammars. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 612–619
Pont-Tuset J, Arbelaez P, Barron JT, Marques F, Malik J (2017) Multiscale combinatorial grouping for image segmentation and object proposal generation. IEEE Transactions on Pattern Analysis and Machine Intelligence 39(1):128–140
Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. In: Neural Information Processing Systems (NIPS), pp 91–99
Richard A, Gall J (2016) Temporal action detection using a statistical language model. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 3131–3140
Roerdink JB, Meijster A (2000) The watershed transform: Definitions, algorithms and parallelization strategies. Fundamenta informaticae 41(1, 2):187–228
Van de Sande KE, Uijlings JR, Gevers T, Smeulders AW (2011) Segmentation as selective search for object recognition. In: The IEEE International Conference on Computer Vision (ICCV), pp 1879–1886
Schindler K, Van Gool L (2008) Action snippets: How many frames does human action recognition require? In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, pp 1–8
Shou Z, Wang D, Chang SF (2016) Temporal action localization in untrimmed videos via multi-stage CNNs. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1049–1058
Shou Z, Chan J, Zareian A, Miyazawa K, Chang SF (2017) CDC: convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1417–1426
Shrivastava A, Gupta A, Girshick R (2016) Training region-based object detectors with online hard example mining. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 761–769
Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: Neural Information Processing Systems (NIPS), pp 568–576
Singh B, Marks TK, Jones M, Tuzel O, Shao M (2016) A multi-stream bi-directional recurrent neural network for fine-grained action detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1961–1970
Singh G, Cuzzolin F (2016) Untrimmed video classification for activity detection: submission to activitynet challenge. CoRR abs/1607.01979
Soomro K, Zamir AR, Shah M (2012) UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv:12120402
Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2818–2826
Tang K, Yao B, Fei-Fei L, Koller D (2013) Combining the right features for complex event recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2696–2703
Tran D, Bourdev LD, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3D convolutional networks. In: The IEEE International Conference on Computer Vision (ICCV), pp 4489–4497
Van Gemert JC, Jain M, Gati E, Snoek CG, et al (2015) Apt: Action localization proposals from dense trajectories. In: The British Machine Vision Conference (BMVC), vol 2, p 4
Wang H, Schmid C (2013) Action recognition with improved trajectories. In: The IEEE International Conference on Computer Vision (ICCV), pp 3551–3558
Wang L, Qiao Y, Tang X (2014a) Action recognition and detection by combining motion and appearance features. In: THUMOS Action Recognition Challenge
Wang L, Qiao Y, Tang X (2014b) Latent hierarchical model of temporal structure for complex activity classification. IEEE Transactions on Image Processing 23(2):810–822
Wang L, Qiao Y, Tang X (2015) Action recognition with trajectory-pooled deep-convolutional descriptors. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 4305–4314
Wang L, Qiao Y, Tang X, Van Gool L (2016a) Actionness estimation using hybrid fully convolutional networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2708–2717
Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2016b) Temporal segment networks: Towards good practices for deep action recognition. In: European Conference on Computer Vision (ECCV), pp 20–36
Wang L, Xiong Y, Lin D, Van Gool L (2017a) Untrimmednets for weakly supervised action recognition and detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2017b) Temporal segment networks for action recognition in videos. arXiv preprint arXiv:170502953
Wang P, Cao Y, Shen C, Liu L, Shen HT (2016c) Temporal pyramid pooling based convolutional neural network for action recognition. IEEE Transactions on Circuits and Systems for Video Technology
Wang R, Tao D (2016) UTS at activitynet 2016. In: AcitivityNet Large Scale Activity Recognition Challenge 2016
Weinzaepfel P, Harchaoui Z, Schmid C (2015) Learning to track for spatio-temporal action localization. In: The IEEE International Conference on Computer Vision (ICCV), pp 3164–3172
Xu H, Das A, Saenko K (2017) R-c3d: Region convolutional 3d network for temporal activity detection. In: The IEEE International Conference on Computer Vision (ICCV), vol 6, p 8
Yeung S, Russakovsky O, Mori G, Fei-Fei L (2016) End-to-end learning of action detection from frame glimpses in videos. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2678–2687
Yuan J, Ni B, Yang X, Kassim AA (2016) Temporal action localization with pyramid of score distribution features. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 3093–3102
Zach C, Pock T, Bischof H (2007) A duality based approach for realtime tv-L^1 optical flow. In: 29th DAGM Symposium on Pattern Recognition, pp 214–223
Zhang B, Wang L, Wang Z, Qiao Y, Wang H (2016) Real-time action recognition with enhanced motion vector CNNs. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2718–2726
Zhao Y, Xiong Y, Wang L, Wu Z, Tang X, Lin D (2017) Temporal action detection with structured segment networks. In: The IEEE International Conference on Computer Vision (ICCV), vol 8, pp 2914–2923
Zitnick CL, Dollár P (2014) Edge boxes: Locating object proposals from edges. In: European Conference on Computer Vision (ECCV), pp 391–405

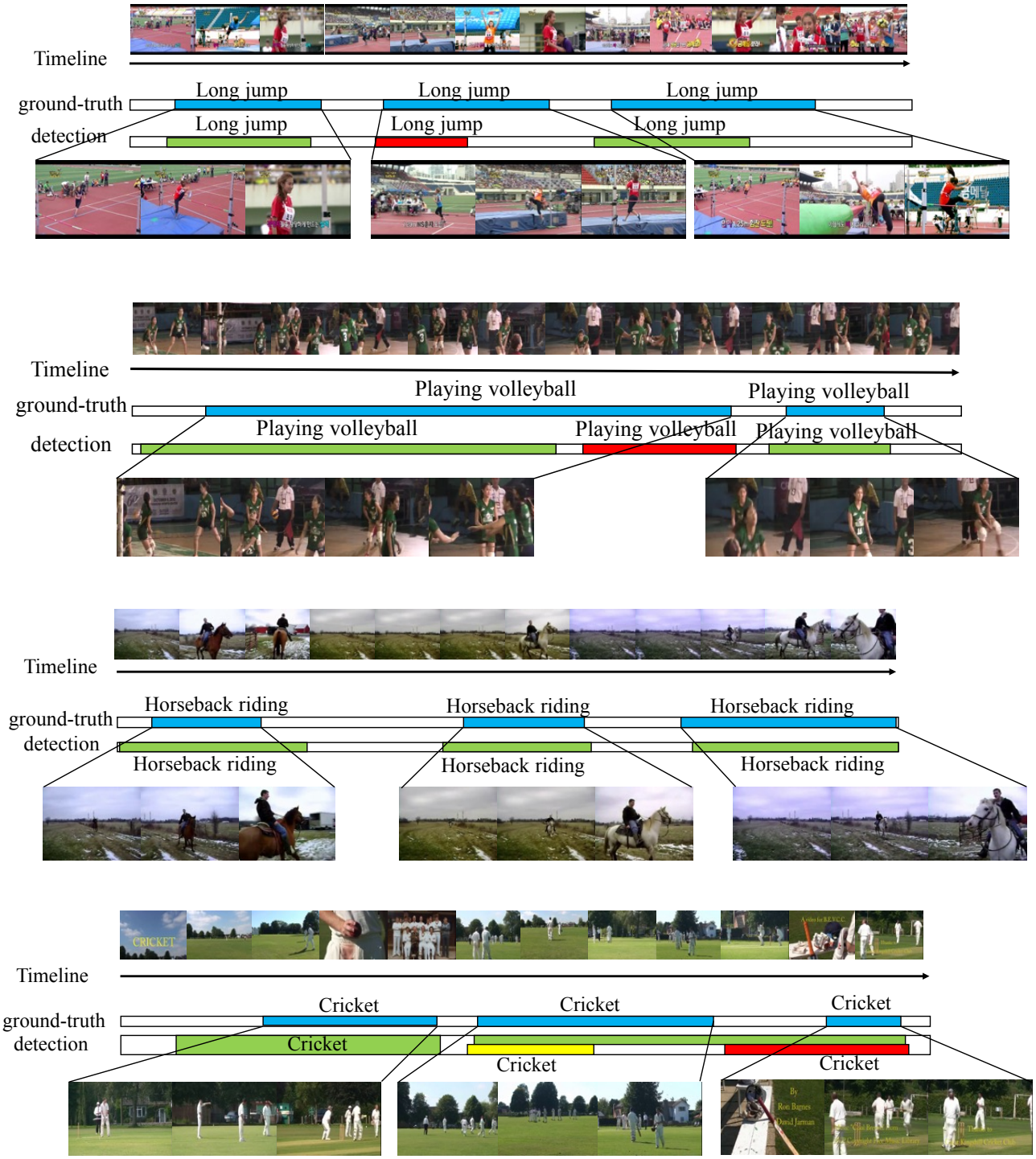

Fig. 9 Examples of detection results on the ActivityNet v1.2 validation set. In each group, the video is shown as sequences of frames on top. The upper bar in each group with blue boxes denotes the annotated ground-truth instances, whose sampled frames are also illustrated at bottom. The detection results from SSN are shown in the lower bar, filled with different colors. A green box denotes a correct detection on condition that IoU ≥ 0.5. Other colors, namely red and yellow, denote the cases of bad localization (IoU < 0.5) and multiple detection, respectively.

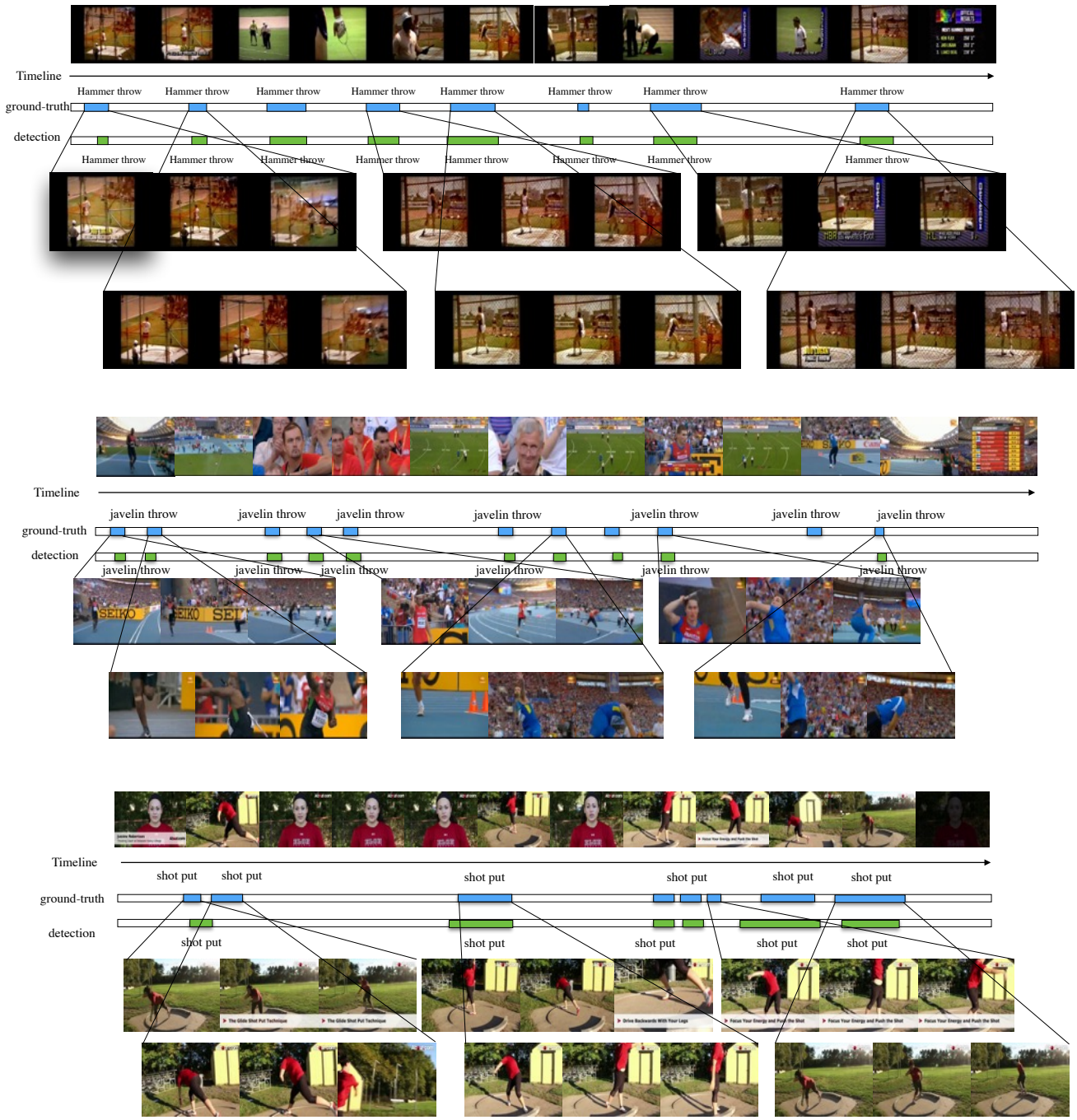

Fig. 10 Examples of detection results on the THUMOS14 testing set. In each group, the video is shown as sequences of frames on top. The upper bar in each group with blue boxes denotes the annotated ground-truth instances, whose sampled frames are also illustrated at bottom. The detection results from SSN are shown in the lower bar, where a green box denotes a correct detection on condition that IoU ≥ 0.1. Note that the durations of action instances in THUMOS14 are much different from those in ActivityNet.