Online Detection of Action Start in Untrimmed, Streaming Videos

Zheng Shou*[1], Junting Pan*[1,2], Jonathan Chan[1], Kazuyuki Miyazawa[3], Hassan Mansour[3], Anthony Vetro[3], Xavi Gir-i-Nieto[2], Shih-Fu Chang[1]

[1]Columbia University, [2]Universitat Politecnica de Catalunya, [3]Mitsubishi Electric

Abstract. We aim to tackle a novel task in action detection - Online Detection of Action Start (ODAS) in untrimmed, streaming videos. The goal of ODAS is to detect the start of an action instance, with high categorization accuracy and low detection latency. ODAS is important in many applications such as early alert generation to allow timely security or emergency response. We propose three novel methods to specifically address the challenges in training ODAS models: (1) hard negative samples generation based on Generative Adversarial Network (GAN) to distinguish ambiguous background, (2) explicitly modeling the temporal consistency between data around action start and data succeeding action start, and (3) adaptive sampling strategy to handle the scarcity of training data. We conduct extensive experiments using THUMOS'14 and ActivityNet. We show that our proposed methods lead to significant performance gains and improve the state-of-the-art methods. An ablation study confirms the effectiveness of each proposed method.

Keywords: Online Detection; Action Start; Generative Adversarial Network; Evaluation Protocol

1 Introduction

In this paper, we investigate a novel task - **Online Detection of Action Start (ODAS)** in untrimmed, streaming videos:

i. **Online detection** requires continuously monitoring the live video stream in real time. When a new video frame arrives, online detection system processes it immediately, without any side information or access to the future frames.

ii. We refer the start/onset of an action instance as its **Action Start (AS)**, which is a time point and is associated with one action instance.

iii. Following recent online detection works [21,19], we target **untrimmed, long, unconstrained** videos with large amounts of complex background streams. ODAS differs from prior works on early event detection such as [26,27], which targeted relatively simple videos and assumed that each video contained only one action instance and which the action class is going to happen is known beforehand. ODAS targets the more practical setting that each video can contain

* indicates equal contributions.

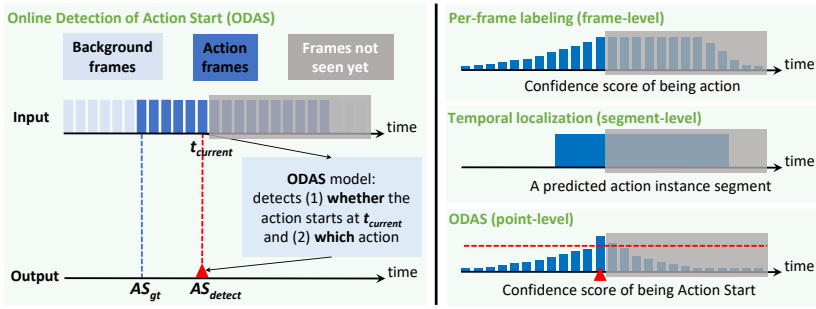


Fig. 1: *Left*: Illustration of the novel **Online Detection of Action Start (ODAS)** task; *Right*: comparisons with the conventional action detection tasks

multiple action instances and the action class in the testing video is not known in advance.

As illustrated in Fig. 1 left, ODAS aims to detect the occurrence and class of AS as soon as the action happens. ODAS is very important in many practical application scenarios, such as early alert generation. For example, the surveillance camera monitoring system needs to detect AS and then issue an alert as soon as possible to allow timely security response; autonomous driving car needs to detect AS of accidents happening in front of it as soon as possible so that the car can slow down or change course timely to avoid collision; robot looking after walking-impaired people shall detect AS of falling as soon as possible to provide assistance before the person has fallen down already. Consequently, in each of such scenarios, it is important to detect AS timely and accurately.

As illustrated in Fig. 1 right, ODAS differs from the conventional action detection tasks. Recent online detection works [21,19] target the **per-frame labeling** task which aims to correctly classifying every frame into either the background class or certain action classes. Recent offline detection works [25,14,45,70,72,48,49,
target the **temporal localization** task which aims to predict a set of action segment instances in a long, untrimmed video. Despite lacking the need to correctly classify every frame (as in per-frame labeling) or localize the complete action segment instance (as in temporal localization), ODAS explicitly focuses on detecting AS, which is quite challenging as discussed in the next paragraph. Traditional methods for per-frame labeling and temporal localization can indeed be adapted for ODAS. But since they were originally designed to address different problems, the challenges in detecting AS have not been specifically considered and deeply investigated. Therefore, methods excelling at per-frame labeling or temporal localization might not perform well in ODAS.

In this paper, we identify three **challenges** in training a good ODAS model and accordingly propose three novel **solutions**. **(Challenge 1)** As the example shown in Fig. 2, it is important to learn and detect characteristics that can correctly distinguish the start window from the background, which precedes AS and may share very similar scenes but without the actual occurrence of ac-

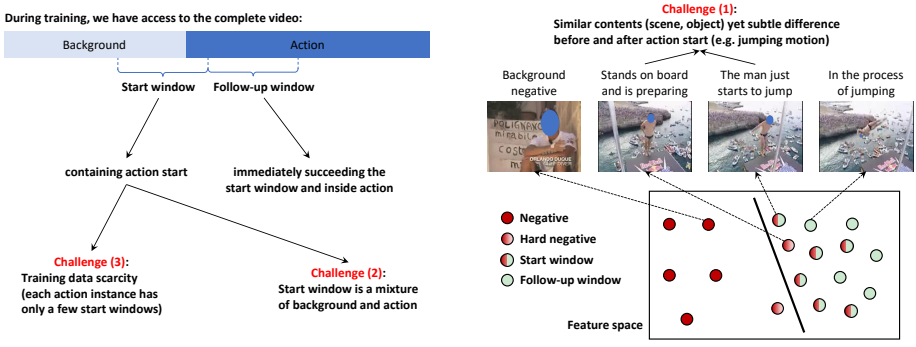


Fig. 2: We identify three challenges in training a good ODAS model

tions. Note that we follow the state-of-the-art video classification models such as C3D [57,58], TSN [64], I3D [9] to accept short temporal sliding window as the network input. To address this challenge, we introduce an auxiliary generative network trained in an adversarial process to automatically **generate hard negative samples** during training. Although hard negative data may be rare in the training videos, our generator directly learns to model the distribution of hard negatives and thus can generate a much larger pool of hard negatives. **(Challenge 2)** We define the start window and its follow-up window in Fig. 2. A start window contains both action frames and background frames. Background preceding action can provide temporal contextual information but can also be confusing. Due to the shared contents (background scene and object), the feature of the start window may be closer to the preceding background window than the actual action window after the start. To remedy this issue, since the follow-up window is completely inside action, we propose to model the **temporal consistency** between the start window and its follow-up window during training. **(Challenge 3)** It is important to accurately classify start windows in ODAS. But each action instance only has a few training samples of start windows, and thus the number of training samples for start windows is much more scarce than others such as background windows and windows fully inside action. To address this issue, we design an **adaptive sampling** strategy to increase the percentage of start windows in each training batch. Our experiments in Sec. 6 will prove the effectiveness and necessity of each proposed method and putting three methods together results in significant performance gains.

In summary, we make three contributions in this paper:

(a) We propose a novel task ODAS in untrimmed, unconstrained, streaming videos to specifically focus on detecting AS timely and accurately.

(b) We design three novel methods for training effective ODAS models: (1) generating hard negative samples based on GAN to assist ODAS models in discriminating start windows from negatives, (2) modeling the temporal consistency between the start window and its follow-up window to encourage their feature

similarity, and finally (3) adaptively sampling start windows more frequently to address the training sample unbalance issue.

(c) Extensive comparisons on THUMOS'14 and ActivityNet demonstrate the superiority of our approach over the conventional methods designed for online detection, per-frame labeling, temporal localization, and shot boundary detection in specifically solving the ODAS problem.

2 Related Works

Action Classification. Given a video clip, the goal of the classification task is to recognize the action categories contained in the whole video. Impressive progress [64,9,62,63,42,30,57,16,51,15,68] has been made to address this problem. Various network architectures have been proposed, such as 3D ConvNets, two-stream network [51,64], I3D [9], etc. Detailed review can be found in surveys [66,43,2,10,4,32].

Temporal Action Localization. Given a long, untrimmed video, temporal action localization needs to temporally localize each action instance: we not only predict its category but also detect when it starts and ends. This task has raised a lot of interest in recent [25,14,45,70,72,48,49,38,24,74,20,67,13,11,18,7,6,73,56,50,69,17]. Shou et al. [49] proposed a Convolutional-De-Convolutional (CDC) network to detect precise segment boundary; Zhao et al. [74] presented the Structured Segment Network (SSN) to model the temporal structure of each action segment; Buch et al. [6] designed an end-to-end system to stream the video and detect action segments ending at the current time.

Although these methods for temporal localization were originally designed for the offline setting, some of them can be adapted to conduct temporal localization in an online manner. However, besides detecting AS, temporal localization requires detecting the Action End (AE) as well. To this end, many methods have to wait for seeing AE in order to localize the action segment as a whole so that can determine AS, resulting in high latency. Also, a good temporal localization method may excel at AE detection but perform poorly in AS detection (considering a detected segment that overlaps with the ground truth segment with IoU 0.7 and has the same AE as the ground truth). ODAS focuses on AS specifically.

Early Recognition and Detection. Similar to ODAS, early recognition and detection also aim to detect action as soon as it happens in streaming videos. Early recognition was effectively formulated as partial action classification [34,71,8,46,35,3 the videos used in early recognition literatures are usually relatively short; during testing, they cut each video to only keep its first certain portion of the whole video, and then classify the cut video into the pre-defined action classes.

ODAS is more related to early detection. Hoai and De la Torre [26,27] made attempts to detect actions in an online manner yet under a simplified setting (e.g., one action instance per video). Huang et al. [28] worked on a scenario that

the background contents are simple (i.e. the person is standing and keeping still). In this paper, like [21,19], we focus on the realistic videos that are unconstrained and contain complex backgrounds of large variety. Ma et al. [40] approached the early detection task by cutting the first certain portion of the whole testing video and then conducting temporal localization on the cut video. Hence, besides detecting AS, this work also focused on detecting whether the action ends or not.

Online Action Detection. Recent works on online action detection are very close to ODAS. De Geest et al. [21] first simulated the online action detection problem using untrimmed, realistic videos and benchmarked the existing models. Gao et al. [19] designed a training strategy to encourage a LSTM-based Reinforced Encoder-Decoder (RED) Network to make correct frame-level label predictions as early as possible. But both of them formulated online action detection as online per-frame labeling task, which requires correctly classifying every frame rather than just detecting AS. A good per-frame labeling method might not be necessarily good at detecting AS (considering a per-frame labeling method correctly classifying frames in the 30%-100% portion of each action instance but mis-classifying frames in its 0%-30% portion). Consequently, as for the applications that detecting AS is the most important task, ODAS is the best fit.

In addition, there are also works on spatio-temporally localizing actions in an online manner but also limited to short videos [52,54]. Li et al. [37] and Liu et al. [39] leveraged Kinect sensors and performed detection based on the tracked skeleton information. Vondrick et al. [60] targeted future prediction, which is a more ambitious online detection task.

Adversarial Learning. The idea of training in an adversarial process was first proposed in [22] and has been adopted in many applications [41,61,29,75,59]. Generative Adversarial Network (GAN) [22,44] consists of two networks trained simultaneously to compete with each other: a generator network G that learns to generate fake samples indistinguishable from real data and a discriminator network D which is optimized to recognize whether input data samples are real or fake. To the best of our knowledge, we are the first to explore GAN for action detection.

3 Framework

In this Section, we introduce our ODAS framework as shown in Fig. 1. We follow the state-of-the-art video classification networks like C3D [57,58], TSN [64], I3D [9] to accept temporal sliding windows as input. In particular, we set the window length to 16 frames and use C3D as our backbone network in Sec. 3 and Sec. 4 to help illustrate technical ideas.

We outline our ODAS framework by walking through the testing pipeline. During testing, when a new frame arrives at the current time t, we immediately feed the streaming window ending at t into our network. The network output

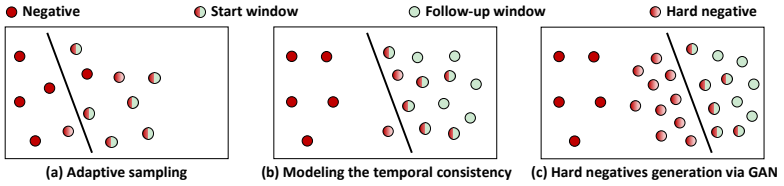


Fig. 3: The effects of our proposed three training methods for ODAS. Each rectangle represents data distribution in the high-level feature space after training

at t consists of the semantic class c_t which could be either background or action $1, \ldots, K$ (K is the total number of action classes) and the confidence score s_t. In order to detect AS, we compare the network outputs at $t-1$ and t. We generate an AS point prediction whenever the following conditions are all satisfied: (1) c_t is action; (2) $c_t \neq c_{t-1}$; (3) s_t exceeds the threshold obtained by grid search on the training set. Such an AS point prediction is associated with the time point t, the predicted class (set to c_t) and the confidence score (set to s_t).

As alternatives, we have also studied the approach of adding a proposal stage specifically for detecting action start and then classifying the action class. We found such an alternative approach is not as effective as the one outlined above. Quantitative comparisons can be found in the supplementary material.

4 Our Methods

As for training our ODAS model, the complete videos are available during training. We slide windows over time with a stride of 1 frame to first construct a set of training windows to be fed into the network. For each window, we assign its label as the action class of the last frame of the window. In this section, we propose three novel methods to improve the capability of the backbone networks at detecting action in a timely manner. We first illustrate our intuition of designing these methods and then present their formulations. We close this section by summarizing the full objective used for training. Implementation details can be found in the supplementary material.

4.1 Intuition

Adaptively sample the training data. Since we want to detect actions as soon as possible, it is important for ODAS to accurately classify start windows. This is a challenging task because the start window contains various background contents, and the number of start windows is quite scarce. If we construct each training batch via randomly sampling out of all windows, the model might not see enough start windows during training in order to generalize to the testing data well. To solve this issue, we guide the ODAS model to pay more attention to start windows via adaptively sampling more start windows during each training batch. Using this strategy, the ODAS model can learn a better classification

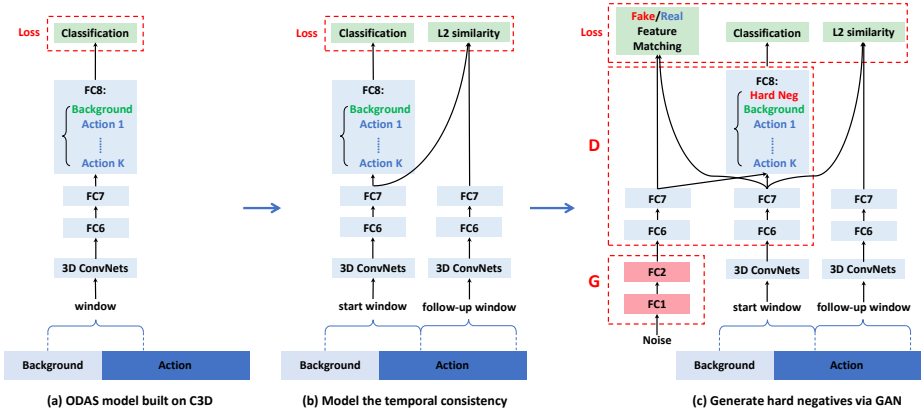


Fig. 4: The network architectures of our ODAS models built on C3D [57] and the proposed training objectives. (a) Our basic ODAS model consists of 3D ConvNets from `Conv1` to `Pool5` and 3 fully connected layers (`FC6`, `FC7`, `FC8`). We keep the same backbone network architecture as C3D while setting the number of nodes in `FC8` to $K+1$, standing for K actions and background. The output of `FC8` is used for calculating multi-class classification softmax loss. (b) We model the temporal consistency between the start window and its paired follow-up window by adding a temporal consistency loss term to minimize the L2 similarity computed using their `FC7` activations. Two streams in this siamese network share the same parameters. (c) Further, we design a GAN-based framework to automatically generate hard negative samples to help our model more accurately distinguish actions against negatives. G is generator and D is discriminator. G accepts random noise as input and output fake `Pool5` features. We add an additional class in `FC8` for fake samples. All blue blocks of the same name are the same layer and share weights. More details of GAN can be found in Section 4.4

boundary to distinguish start windows against negatives more accurately, as illustrated in Fig. 3 (a).

Model the temporal consistency. As illustrated in Fig. 3 (a), the start window is a mixture of action frames and background frames. Therefore, in the feature space, start windows could be close to or even mixed with negatives. It is important to accurately distinguish start windows and negatives in ODAS so that the model can more timely detect action start when the video stream switches from negative to action. As illustrated in Fig. 3 (b), since the follow-up windows are completely inside action, they are far away from negative data in the feature space. Thus we explicitly model the **Temporal Consistency (TC)** between each start window and its follow-up window to encourage their feature similarity. Using this training method, start windows move closer to follow-up windows and thus become more separable from negatives.

Generate hard negative samples via GAN. As exampled in Fig. 2, it is important to train the ODAS model to capture the subtle differences that can serve as evidences for discriminating start windows from negatives preceding AS. As illustrated in Fig. 3 (b), hard negatives that have subtle differences with start windows might be close to start windows in the feature space. In order to learn a better decision boundary, we aim to identify such hard negative samples in the training set during training. However, exhaustively finding such samples is time-consuming because such hard negatives are rare and may even not exist in the training data. Therefore, we propose to train a model to automatically synthesize samples that are hard to distinguish from true start windows. As illustrated in Fig. 3 (c), equipped by these synthesized hard negatives, we can learn an even better ODAS model to discriminate start windows from negatives.

4.2 Adaptively Sample the Training Data

Concretely, we randomly sample half of the training batch from start windows and randomly sample the other half batch from the remaining windows, which can be backgrounds or windows completely inside actions. After each training batch is constructed, we can feed them into our network as shown in Fig. 4 (a) and train the network via minimizing the multi-class classification softmax loss $\mathcal{L}_{\text{classification}}$. We denote the set of start windows as $p_{\text{start}} = \{(x_s, y)\}$ where x_s is the start window to be fed into our model and y is its corresponding ground truth label. Similarly, we express the set of remaining windows as $p_{\text{notstart}} = \{(x_{ns}, y)\}$. The label space of y is $1, \ldots, K+1$ where the first K classes are actions and the $K+1$-th class stands for background. Our network takes x as input and predicts a vector $\{o_1, \ldots, o_{K+1}\}$ of $K+1$ dimension. Finally we apply the softmax function to obtain the normalized probability of being class i: $P_{\text{model}}(i|x) = \frac{e^{o_i}}{\sum_{k=1}^{K+1} e^{o_k}}$. We use $\mathbb{E}[\cdot]$ to represent expectation. The classification loss is defined as:

$$\begin{aligned}\mathcal{L}_{\text{classification}} = \quad & \mathbb{E}_{(x_s,y)\sim p_{\text{start}}}\left[-\log\left(P_{\text{model}}\left(y|x_s\right)\right)\right] \\ & +\mathbb{E}_{(x_{ns},y)\sim p_{\text{notstart}}}\left[-\log\left(P_{\text{model}}\left(y|x_{ns}\right)\right)\right].\end{aligned} \tag{1}$$

4.3 Model the Temporal Consistency

Formally, we denote the training set of the paired start window and follow-up window as $p_{\text{startfollowup}} = \{(x_s, x_f, y)\}$, where x_s represents the start window and x_f is its associated follow-up window and y is still the ground truth label. We model the temporal consistency via minimizing the similarity $\mathcal{L}_{\text{similarity}}$ measured by L2 distance of the feature representation between x_s and x_f:

$$\mathcal{L}_{\text{similarity}} = \mathbb{E}_{(x_s,x_f,y)\sim p_{\text{startfollowup}}}\left\|F\left(x_s\right) - F\left(x_f\right)\right\|_2^2, \tag{2}$$

where the function $F(\cdot)$ indicates extracting feature representation. As shown in Fig. 4 (b), we set $F(\cdot)$ to be the output of `FC7` because it is also the input to the final layer `FC8` for classification. Trained with this temporal consistency loss, the features of start windows become more distinguishable from negatives, which leads to the ODAS performance improvements as shown in Sec. 6.1.

4.4 Generate Hard Negative Samples via GAN

In order to separate start windows and hard negatives which have subtle differences from start windows, we design a GAN-based framework which synthesizes hard negative samples to assist ODAS model training.

Generator (G). Since directly generating video is very challenging, we use GAN to generate features rather than raw videos. As shown in Fig. 4 (c), our GAN model has a fixed 3D ConvNets (from `Conv1` to `Pool5`) to extract real `Pool5` features from raw videos and also has a G to generate fake `Pool5` features. Upper layers serve as **Discriminator (D)**, which will be explained later.

G accepts a random noise z as input and learns to capture the true distribution of real start windows. Consequently, G has the potential to generate various fake `Pool5` samples which may not exist in the real training set but may appear during testing. This enables our model to continuously explore more discriminative classification boundary in the high-level feature space. Following [44], z is a 100-dimensional vector randomly drawn from the standard normal distribution. In practice, we find that a simple G consisting of two fully connected layers `FC1` and `FC2` works well. Each fully connected layer is followed by a BatchNorm layer and a ReLU layer.

When training G, the principle is to generate **hard** negative samples that are similar to real start windows. Conventional GANs utilize a binary real/fake classifier to provide supervision signal for training G. However, this method usually encounters an instability issue. Following [47], instead of adding a binary classifier, we require G to generate fake data matching the statistics of the real data. Specifically, the feature matching objective is forcing G to match the expectation of the real features on an intermediate layer of D (we use `FC7` layer as indicated in Fig. 4 (c)). Formally, we denote the feature extraction part of using the fixed 3D ConvNets as $\phi(\cdot)$ and the process from `Pool5` to `FC7` as $\psi(\cdot)$. The feature matching loss is defined as follows:

$$\mathcal{L}_{\text{matching}} = \left\| \mathbb{E}_{(x_s,y)\sim p_{\text{start}}} \left[\psi\left(\phi\left(x_s\right)\right) \right] - \mathbb{E}_{z\sim noise} \left[\psi\left(G\left(z\right)\right) \right] \right\|_2^2, \tag{3}$$

where $G(\cdot)$ denotes the generator, $p_{\text{start}} = \{(x_s, y)\}$ is the training set of start windows, x_s represents the start window, and y is the ground truth label.

Discriminator (D). The principle for designing D is that the generated samples should be still separable from real start windows despite their similarity, so that the generated samples can be regarded as hard **negatives**. As shown in Fig. 4 (c), D consists of `FC6`, `FC7`, and `FC8`. Instead of adding a binary real/fake classifier, we add an additional node in `FC8` layer to represent the hard negative class, which is the ground truth label for the generated samples. Note that this additional class is used during training only and is removed during testing.

Similarly, some previous works also replaced the binary discriminator with a multi-class classifier that has an additional class for the fake samples [47,55,12].

However, their motivation is mainly extending GAN to the semi-supervised setting: the unlabeled real samples could belong to any class except fake. But in this paper, we focus on generating hard negatives which should be similar to actions but dissimilar to backgrounds; meanwhile our D needs to distinguish hard negatives from not only actions but also from backgrounds.

Given a `Pool5` feature $\phi(x)$ either extracted from real data or generated by G, D accepts $\phi(x)$ as input and predicts a vector $\{o_1, \ldots, o_{K+2}\}$ which goes through a softmax function to get class probabilities: $P_{\mathrm{D}}(i|\phi(x)) = \frac{e^{o_i}}{\sum_{k=1}^{K+2} e^{o_k}}$, where $i \in \{1, \ldots, K+2\}$. Regarding the real samples, we can calculate their corresponding classification loss $\mathcal{L}_{\mathrm{real}}$ via extending $\mathcal{L}_{\mathrm{classification}}$ defined in Eq. 1:

$$\begin{aligned}\mathcal{L}_{\mathrm{real}} = \quad & \mathbb{E}_{(x_s,y)\sim P_{\mathrm{start}}}\left[-\log P_D\left(y|\phi\left(x_s\right)\right)\right] \\ & +\mathbb{E}_{(x_{ns},y)\sim P_{\mathrm{notstart}}}\left[-\log P_D\left(y|\phi\left(x_{ns}\right)\right)\right].\end{aligned} \tag{4}$$

As for the generated fake samples, the loss is:

$$\mathcal{L}_{\mathrm{fake}} = \mathbb{E}_{z\sim noise}\left[-\log P_D\left(K+2|G\left(z\right)\right)\right], \tag{5}$$

where $K+2$ represents the hard negative class.

4.5 The Full Objective

We first pre-train the whole network via minimizing $\mathcal{L}_{\mathrm{classification}} + \lambda \cdot \mathcal{L}_{\mathrm{similarity}}$, which combines the classification loss (Eq. 1) and the temporal consistency loss (Eq. 2) together with the weighting parameter λ. Based on such initialization, we train G and D in an alternating manner during each iteration: **When training G**, we fix D and train G so that the full objective is $\min_G \mathcal{L}_{\mathrm{G}}$. $\mathcal{L}_{\mathrm{G}}$ contains only the feature matching loss (Eq. 3): $\mathcal{L}_{\mathrm{G}} = \mathcal{L}_{\mathrm{matching}}$. **When training D**, we fix G and train D so that the full objective is $\min_D \mathcal{L}_{\mathrm{D}}$. $\mathcal{L}_{\mathrm{D}}$ contains the classification loss for both the real and fake samples (Eq. 4 and Eq. 5) and also the temporal consistency loss (Eq. 2): $\mathcal{L}_{\mathrm{D}} = \mathcal{L}_{\mathrm{real}} + \mathcal{L}_{\mathrm{fake}} + \lambda \cdot \mathcal{L}_{\mathrm{similarity}}$, where λ is the weight.

5 Evaluation

5.1 Conventional Protocols and Metrics

Hoai and De la Torre [26,27] first worked on early detection and proposed three evaluation protocols to respectively evaluate classification accuracy, detection timeliness, and localization precision. As comprehensively discussed in [21], their protocols do not suit online detection in realistic, unconstrained videos, because their protocols were designed for a simplified setting: each video contains only one action instance of interest.

Therefore, as mentioned in Sec. 2, recent online detection works [21,19] effectively worked on the per-frame labeling task and evaluated the frame-level classification mean Average Precision (mAP) or its calibrated version. In addition, temporal localization methods detect both the start and end times of each

action instance and evaluate the segment-level detection mAP. As for ODAS, the performance of detecting AS can indeed affect both the frame-level mAP and the segment-level mAP. However, since the frame-level mAP is mainly used to evaluate the accuracy of classifying every frame and the segment-level mAP involves evaluating the correctness of detecting the end, both metrics are not exactly evaluating the performance in detecting action starts. Detailed discussions and examples can be found in the supplementary material.

5.2 Proposed New Protocol and Metrics

In order to specifically evaluate ODAS performance, we propose a new evaluation protocol. We evaluate ODAS at the point level for each AS instance. As mentioned in Sec. 3, ODAS system outputs a rank list of detected AS points. Each AS point prediction is associated with the time point t, the predicted action class and the confidence score.

We aim to evaluate the detection accuracy and timeliness of ODAS system compared to the human annotated ground truths [1]. As for the timeliness, inspired by the segment-level mAP which measures the temporal overlap between the ground truth segment and the predicted segment, we measures the temporal offset (absolute distance) between the ground truth AS point and the predicted AS point. We discuss the temporal offset in detail in the supplementary material.

We propose the **point-level AS detection mAP** to evaluate ODAS results. Each AS point prediction is counted as correct only when its action class is correct and its offset is smaller than the evaluation threshold. We evaluate the point-level AP for each action class and average over all action classes to get the point-level mAP. We do not allow duplicate detections for the same ground truth AS point.

6 Experiments

In order to simulate the ODAS setting, we employ standard benchmarks consisting of untrimmed videos to simulate the sequential arrival of video frames.

6.1 Results on THUMOS'14

Dataset. THUMOS'14 [31] involves 20 actions and videos over 20 hours: 200 validation videos (3,007 action instances) and 213 test videos (3,358 action instances). These videos are untrimmed and contain at least one action instance. Each video has 16.8 action instances in average. We use the validation videos for training and use the test videos to simulate streaming videos for testing ODAS.

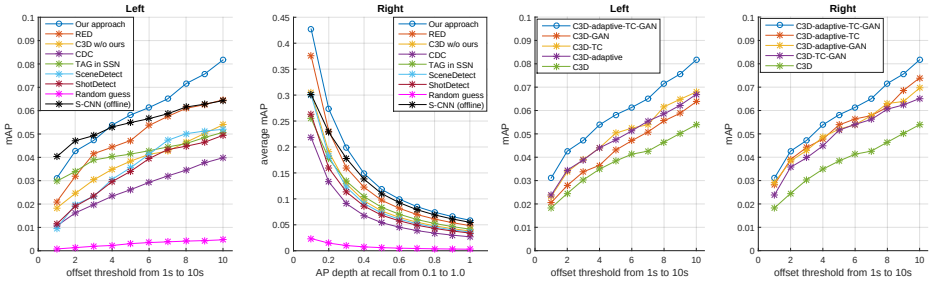
[1] According to the human annotations on THUMOS'14 test set [31], people usually can form agreement on when the action starts: out of 3,358 action instances, 3,259 instances (97%) have their AS time annotations agreed by multiple human annotators. The instances with ambiguous start times are excluded during evaluation.

Metrics. We evaluate the point-level AS detection mAP at different temporal offset thresholds to compare ODAS systems under the different user tolerances or application requirements. Note that *AP depth at recall X%* means averaging the precisions at points on the P-R curve with the recall ranging from 0% to X%. The aforementioned default detection mAP is evaluated at AP depth at recall 100%. In order to look at the precision of top ranked predictions, we can evaluate detection mAP at different AP depth. At each AP depth, we average the detection mAP under different offset thresholds to obtain the average mAP.

Comparisons. As for **our approach**, our network architecture can be found in Fig. 4. Since we build our model upon C3D [57] which has been pre-trained on Sports-1M [33], we use it to initialize models shown in Fig. 4. Since `Pool5` output has 8,192 dimensions, we use 4,096 nodes for both `FC1` and `FC2` in G. We compare with the following baselines. (1) **Random guess**: we replace the network output scores for K actions and background mentioned in Sec. 3 via randomly splitting score 1 into $K+1$ numbers all within $[0, 1]$. (2) **C3D w/o ours**: we use the C3D model which has exactly the same network architecture as our model used during testing but is trained without our proposed methods. (3) **RED**: Gao et al. [19] achieved the state-of-the-art performances on THUMOS'14 in online action detection task by encouraging the LSTM network to make correct frame-level predictions at the early part of a sequence. We obtained the results from the authors and evaluated based on our proposed protocol. (4) Per-frame labeling method - **CDC**: Shou et al. [49] designed a Convolutional-De-Convolutional Network to operate on the testing video in an online manner to output the per-frame classification scores, which can be used to determine AS points following the same pipeline as proposed in Sec. 3. (5) Temporal localization method - **TAG in SSN**: Zhao et al. [74] proposed an effective segment proposal method called temporal actionness grouping (TAG). Based on the actionness score sequence, TAG can be operated in an online manner to detect the start of a segment and then also detect its end. Thus TAG can be used for ODAS to generate class-agnostic AS point proposals. For fair comparisons, we determine the action score of each proposal by applying the AS classifier obtained by our best model (trained with all three methods). (6) Shot boundary detection methods [5,65,53] detect the change boundaries in the video which can be considered as AS proposals. Then we utilize our best classifier to classify each AS proposal. We employ two popular open-source shot detection methods **ShotDetect**[2] and **SceneDetect**[3] respectively for comparisons. (7) Offline detection method: **S-CNN** [48] uses the same C3D network architecture as our model but performs testing in offline.

Since the duration of action instance varies from <1s to >20s and thus during evaluation we vary the offset threshold from 1s to 10s. As shown in Fig. 5, when using the proposed training strategies specifically designed to tackle ODAS, **our approach** improves the **C3D w/o ours** baseline by a large margin. Qualita-

[2] https://github.com/johmathe/Shotdetect
[3] https://github.com/Breakthrough/PySceneDetect



(a) Comparisons with SoA (b) Ablation study

Fig. 5: Experimental results on THUMOS'14. (a) Comparisons with state-of-the-art methods. ***Left***: y-axis is the point-level AS detection mAP and x-axis is varying the offset threshold. ***Right***: y-axis is the average mAP averaged over offsets from 1s to 10s at AP depth at X% recall and x-axis is varying X% from 0.1 to 1. (b) Ablation study of our methods. Point-level AS detection mAP vs. offset threshold. ***Left***: Using one of proposed training methods only. ***Right***: removing one method out of the whole three combination

tive comparisons can be found in the supplementary material. Notably, **our approach** is far better than **random guess** and also outperforms **RED**, which is a state-of-the-art method developed very recently to detect action as soon as possible. Also, our method is better than other per-frame labeling method (i.e. **CDC**) and some class-agnostic start point detection methods (i.e. **TAG in SSN**, **ShotDetect**, **SceneDetect**). Furthermore, **our approach** can even achieve better results under high offset thresholds than the offline **S-CNN** method.

Ablation study of individual proposed method. We conduct in-depth study on THUMOS'14 to analyze the performance gain contributed by each proposed training method. In Fig. 5 (b) left, we report the results on THUMOS'14 when training with only one of our proposed methods. All approaches in the following have the same network architecture during testing: **C3D** is trained without any proposed methods; **C3D-adaptive** is trained with the adaptive sampling strategy; **C3D-TC** is trained with modeling the temporal consistency; **C3D-GAN** is trained within our proposed GAN-based framework; **C3D-adaptive-TC-GAN** combines all three proposed methods together during training and achieves the best performance. These results indicate that all three proposed methods are effective in improving the ODAS model.

In Fig. 5 (b) right, we report the results on THUMOS'14 when training without one of our proposed methods. We add additional approaches of the same network architecture during testing: **C3D-TC-GAN** is trained without the adaptive sampling strategy; **C3D-adaptive-GAN** is trained without modeling the temporal consistency; **C3D-adaptive-TC** is trained without our pro-

Table 1: AS detection mAP (%) on ActivityNet when varying the offset threshold

offset threshold (s)	10	50	100
Random guess	0.06	0.14	0.17
SceneDetect	4.71	18.93	25.84
ShotDetect	6.10	24.35	33.76
TSN w/o ours	8.18	31.39	44.15
Our approach	**8.33**	**33.08**	**46.97**

posed GAN-based framework. These results indicate that all three proposed methods are necessary for training a good ODAS model.

6.2 Results on ActivityNet

Dataset. ActivityNet [23,1] v1.3 involves 200 actions and untrimmed videos over 800 hours: around 10K training videos (15K instances) and 5K validation videos (7.6K instances). Each video has 1.7 action instances in average. We train on the training videos and evaluate ODAS using the validation videos.

Comparisons. As for our approach, given the superior performances of TSN on ActivityNet video classification task [64], following [18], for each window of 16 frames, we use TSN to extract a feature vector of 3,072 dimensions to serve as input to our network. Our backbone network for ActivityNet consists of three fully connected layers (i.e. `FC6`, `FC7`, `FC8`) that are the same as these in C3D, but we train this network directly from scratch. As for G, since the dimension of fake samples here is 3,072, we set `FC1` and `FC2` in G to be 2,048 dimensions.

The duration of action instance varies from <1s to >200s in ActivityNet and thus during evaluation we vary the temporal offset from 10s to 100s. As shown in Table 1, **our approach** significantly outperform the baseline methods again and improves **TSN w/o ours** which indicates that it also accepts TSN features as input and has the same testing network architecture as **our approach** but is trained without our proposed methods.

6.3 Efficiency

In terms of testing speed, unlike offline detection which evaluates how many frames can be processed per second simultaneously, it is important for ODAS to evaluate the detection delay which is the time duration between the system receives a new video frame and the system outputs the prediction for this frame. Our model in Fig. 4 (c) is able to respond within 0.16s on one single Titan X GPU. Further, our method can maintain similar mAP results even when the striding distance of the input window is increased to 8 frames, thus allowing real-time implementations.

7 Conclusion and Future Works

In this paper, we have proposed a novel Online Detection of Action Start task in a practical setting involving untrimmed, unconstrained videos. Three training methods have been proposed to specifically improve the capability of ODAS models in detecting action timely and accurately. Our methods can be applied to any existing video backbone network. Extensive experiments demonstrate the effectiveness of our approach. In the future, since the developed methods can address the specific challenges in detecting action starts, it would be interesting to further apply them to help address other online action detection tasks such as per-frame labeling and temporal localization.

References

1. Activitynet challenge 2016. http://activity-net.org/challenges/2016/ (2016)
2. Aggarwal, J.K., Ryoo, M.S.: Human activity analysis: A review. In: ACM Computing Surveys (2011)
3. Aliakbarian, M.S., Saleh, F., Salzmann, M., Fernando, B., Petersson, L., Andersson, L.: Encouraging lstms to anticipate actions very early. In: ICCV (2017)
4. Asadi-Aghbolaghi, M., Clapés, A., Bellantonio, M., Escalante, H.J., Ponce-López, V., Baró, X., Guyon, I., Kasaei, S., Escalera, S.: A survey on deep learning based approaches for action and gesture recognition in image sequences. In: FG (2017)
5. Boreczky, J.S., Rowe, L.A.: Comparison of video shot boundary detection techniques. Journal of Electronic Imaging (1996)
6. Buch, S., Escorcia, V., Ghanem, B., Fei-Fei, L., Niebles, J.C.: End-to-end, single-stream temporal action detection in untrimmed videos. In: BMVC (2017)
7. Buch, S., Escorcia, V., Shen, C., Ghanem, B., Niebles, J.C.: Sst: Single-stream temporal action proposals. In: CVPR (2017)
8. Cao, Y., Barrett, D., Barbu, A., Narayanaswamy, S., Yu, H., Michaux, A., Lin, Y., Dickinson, S., Mark Siskind, J., Wang, S.: Recognize human activities from partially observed videos. In: CVPR (2013)
9. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017)
10. Cheng, G., Wan, Y., Saudagar, A.N., Namuduri, K., Buckles, B.P.: Advances in human action recognition: A survey (2015), http://arxiv.org/abs/1501.05964
11. Dai, X., Singh, B., Zhang, G., Davis, L.S., Chen, Y.Q.: Temporal context network for activity localization in videos. In: ICCV (2017)
12. Dai, Z., Yang, Z., Yang, F., Cohen, W.W., Salakhutdinov, R.: Good semi-supervised learning that requires a bad gan. In: NIPS (2017)
13. Dave, A., Russakovsky, O., Ramanan, D.: Predictive-corrective networks for action detection. In: CVPR (2017)
14. Escorcia, V., Heilbron, F.C., Niebles, J.C., Ghanem, B.: Daps: Deep action proposals for action understanding. In: ECCV (2016)
15. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: CVPR (2016)
16. Gan, C., Wang, N., Yang, Y., Yeung, D.Y., Hauptmann, A.G.: Devnet: A deep event network for multimedia event detection and evidence recounting. In: CVPR (2015)
17. Gao, J., Sun, C., Yang, Z., Nevatia, R.: Tall: Temporal activity localization via language query. In: ICCV (2017)
18. Gao, J., Yang, Z., Nevatia, R.: Cascaded boundary regression for temporal action detection. In: BMVC (2017)
19. Gao, J., Yang, Z., Nevatia, R.: Red: Reinforced encoder-decoder networks for action anticipation. In: BMVC (2017)
20. Gao, J., Yang, Z., Sun, C., Chen, K., Nevatia, R.: Turn tap: Temporal unit regression network for temporal action proposals. In: ICCV (2017)
21. Geest, R.D., Gavves, E., Ghodrati, A., Li, Z., Snoek, C., Tuytelaars, T.: Online action detection. In: ECCV (2016)
22. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS (2014)
23. Heilbron, F.C., Escorcia, V., Ghanem, B., Niebles, J.C.: Activitynet: A large-scale video benchmark for human activity understanding. In: CVPR (2015)

24. Heilbron, F.C., Barrios, W., Escorcia, V., Ghanem, B.: Scc: Semantic context cascade for efficient action detection. In: CVPR (2017)
25. Heilbron, F.C., Niebles, J.C., Ghanem, B.: Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In: CVPR (2016)
26. Hoai, M., De la Torre, F.: Max-margin early event detectors. In: CVPR (2012)
27. Hoai, M., De la Torre, F.: Max-margin early event detectors. In: IJCV (2014)
28. Huang, D., Yao, S., Wang, Y., De La Torre, F.: Sequential max-margin event detectors. In: ECCV (2014)
29. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2017)
30. Jégou, H., Douze, M., Schmid, C., Pérez., P.: Aggregating local descriptors into a compact image representation. In: CVPR (2010)
31. Jiang, Y.G., Liu, J., Zamir, A.R., Toderici, G., Laptev, I., Shah, M., Sukthankar, R.: THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/` (2014)
32. Kang, S.M., Wildes, R.P.: Review of action recognition and detection methods. arXiv preprint arXiv:1610.06906 (2016)
33. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR (2014)
34. Kong, Y., Kit, D., Fu, Y.: A discriminative model with multiple temporal scales for action prediction. In: ECCV (2014)
35. Kong, Y., Tao, Z., Fu, Y.: Deep sequential context networks for action prediction. In: CVPR (2017)
36. Lan, T., Chen, T.C., Savarese, S.: A hierarchical representation for future action prediction. In: ECCV (2014)
37. Li, Y., Lan, C., Xing, J., Zeng, W., Yuan, C., Liu, J.: Online human action detection using joint classification-regression recurrent neural networks. In: ECCV (2016)
38. Lin, T., Zhao, X., Shou, Z.: Single shot temporal action detection. In: ACM MM (2017)
39. Liu, C., Li, Y., Hu, Y., Liu, J.: Online action detection and forecast via multitask deep recurrent neural networks. In: ICASSP (2017)
40. Ma, S., Sigal, L., Sclaroff, S.: Learning activity progression in lstms for activity detection and early detection. In: CVPR (2016)
41. Odena, A., Olah, C., Shlens, J.: Conditional image synthesis with auxiliary classifier gans. In: ICML (2017)
42. Perronnin, F., Sánchez, J., Mensink., T.: Improving the fisher kernel for large-scale image classification. In: ECCV (2010)
43. Poppe, R.: A survey on vision-based human action recognition. In: Image and vision computing (2010)
44. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
45. Richard, A., Gall, J.: Temporal action detection using a statistical language model. In: CVPR (2016)
46. Ryoo, M.S.: Human activity prediction: Early recognition of ongoing activities from streaming videos. In: ICCV (2011)
47. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: NIPS (2016)
48. Shou, Z., Wang, D., Chang, S.F.: Temporal action localization in untrimmed videos via multi-stage cnns. In: CVPR (2016)

49. Shou, Z., Chan, J., Zareian, A., Miyazawa, K., Chang, S.F.: Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In: CVPR (2017)
50. Sigurdsson, G.A., Divvala, S., Farhadi, A., Gupta, A.: Asynchronous temporal fields for action recognition. In: CVPR (2017)
51. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NIPS (2014)
52. Singh, G., Saha, S., Cuzzolin, F.: Online real time multiple spatiotemporal action localisation and prediction on a single platform. In: ICCV (2017)
53. Smeaton, A.F., Over, P., Doherty, A.R.: Video shot boundary detection: Seven years of trecvid activity. Computer Vision and Image Understanding (2010)
54. Soomro, K., Idrees, H., Shah, M.: Predicting the where and what of actors and actions through online action localization. In: CVPR (2016)
55. Springenberg, J.T.: Unsupervised and semi-supervised learning with categorical generative adversarial networks. In: ICLR (2016)
56. Sun, C., Shetty, S., Sukthankar, R., Nevatia, R.: Temporal localization of fine-grained actions in videos by domain transfer from web images. In: ACM MM (2015)
57. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV (2015)
58. Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M.: Convnet architecture search for spatiotemporal feature learning. arXiv preprint arXiv:1708.05038 (2017)
59. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: CVPR (2017)
60. Vondrick, C., Pirsiavash, H., Torralba, A.: Anticipating the future by watching unlabeled video. In: CVPR (2016)
61. Vondrick, C., Torralba, A.: Generating the future with adversarial transformers. In: CVPR (2017)
62. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Action Recognition by Dense Trajectories. In: CVPR (2011)
63. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: ICCV (2013)
64. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Gool, L.V.: Temporal segment networks: Towards good practices for deep action recognition. In: ECCV (2016)
65. Warhade, K., Merchant, S.N., Desai, U.B., et al.: Video Shot Boundary Detection. River Publishers (2011)
66. Weinland, D., Ronfard, R., Boyer, E.: A survey of vision-based methods for action representation, segmentation and recognition. In: Computer Vision and Image Understanding (2011)
67. Xu, H., Das, A., Saenko, K.: R-c3d: Region convolutional 3d network for temporal activity detection. In: ICCV (2017)
68. Xu, Z., Yang, Y., Hauptmann, A.G.: A discriminative cnn video representation for event detection. In: CVPR (2015)
69. Yang, Z., Gao, J., Nevatia, R.: Spatio-temporal action detection with cascade proposal and location anticipation. In: BMVC (2017)
70. Yeung, S., Russakovsky, O., Mori, G., Fei-Fei, L.: End-to-end learning of action detection from frame glimpses in videos. In: CVPR (2016)
71. Yu, G., Yuan, J., Liu, Z.: Predicting human activities using spatio-temporal structure of interest points. In: ACM MM (2012)

72. Yuan, J., Ni, B., Yang, X., Kassim, A.: Temporal action localization with pyramid of score distribution features. In: CVPR (2016)
73. Yuan, Z., Stroud, J.C., Lu, T., Deng, J.: Temporal action localization by structured maximal sums. In: CVPR (2017)
74. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., Lin, D.: Temporal action detection with structured segment networks. In: ICCV (2017)
75. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV (2017)