Reducing the Label Bias for Timestamp Supervised Temporal Action Segmentation

Kaiyuan Liu[1*] Yunheng Li[1*] Shenglan Liu[1†] Chenwei Tan[1] Zihang Shao[1]
[1]School of Computer Science, Dalian University of Technology, China
{1154864382,liyunheng,liusl,tcw2000,1317588161}@mail.dlut.edu.cn

Abstract

Timestamp supervised temporal action segmentation (TSTAS) is more cost-effective than fully supervised counterparts. However, previous approaches suffer from severe label bias due to over-reliance on sparse timestamp annotations, resulting in unsatisfactory performance. In this paper, we propose the Debiasing-TSTAS (D-TSTAS) framework by exploiting unannotated frames to alleviate this bias from two phases: 1) Initialization. To reduce the dependencies on annotated frames, we propose masked timestamp predictions (MTP) to ensure that initialized model captures more contextual information. 2) Refinement. To overcome the limitation of the expressiveness from sparsely annotated timestamps, we propose a center-oriented timestamp expansion (CTE) approach to progressively expand pseudo-timestamp groups which contain semantic-rich motion representation of action segments. Then, these pseudo-timestamp groups and the model output are used to iteratively generate pseudo-labels for refining the model in a fully supervised setup. We further introduce segmental confidence loss to enable the model to have high confidence predictions within the pseudo-timestamp groups and more accurate action boundaries. Our D-TSTAS outperforms the state-of-the-art TSTAS method as well as achieves competitive results compared with fully supervised approaches on three benchmark datasets.

1. Introduction

Analyzing and understanding human actions in videos is very important for many applications, such as human-robot interaction [14] and healthcare [33]. Recently, several approaches have been very successful in locating and analyzing activities in videos, including action localization [12,29,37,50], action segmentation [7,11,20,23], and action recognition [9,27,39,41,47].

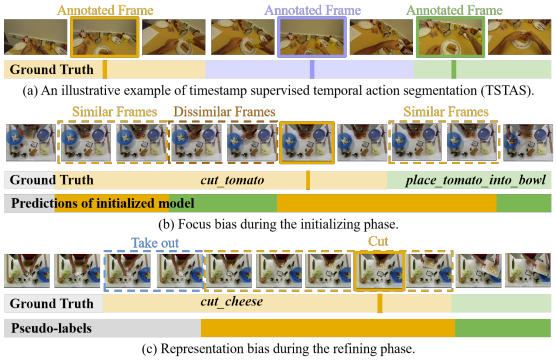


Figure 1. (a) The TSTAS task aims to segment actions in videos by timestamp annotations. (b) An example of focus bias: existing initialization methods prefer to predict frames similar to the annotated timestamp, leading to incorrectly identifying dissimilar frames within the segment and similar frames outside the segment. (c) An example of representation bias: the frames of complex action share large semantic variance, e.g. the process of cutting cheese includes both taking out and cutting, resulting in the models that rely on sparse timestamps to produce biased pseudo-labels.

Despite the success of previous approaches, they rely on fully temporal supervision, where the start and end frames of each action are annotated. As a lightweight alternative, many researchers have started exploring timestamp supervised temporal action segmentation (TSTAS), where each action segment is annotated with only one frame in the untrimmed video, as shown in Fig. 1 (a).

Most previous approaches follow a two-step pipeline by initializing and then iteratively refining the model with generated pseudo-labels. However, relying only on supervised signals of sparse single-timestamp annotations, these methods fail to learn the semantics of entire action segments, which is referred to as ***label bias*** in this paper, including ***focus bias*** and ***representation bias***. Specifically, to initialize the segmentation model, the previous methods [17, 28, 32] adopt the Naive approach proposed in [28] (referred to as Naive), which computes the loss at the annotated frames for training. However, utilizing only the annotated frames leads to ***focus bias***, where the initialized model tends to fo-

*These authors contributed equally to this work.
†Corresponding author.

cus on frames distributed over segments of various action categories similar to the annotated frames [51] (shown in Fig. 1 (b)).

During refining the segmentation model, to utilize the unlabeled frames, typical methods generate hard [17,28,51] or soft weighted [32] pseudo-labels that are used to refine the model like fully supervised methods. To generate pseudo-labels, the above methods depend on the timestamps that contain semantic or relative position information. Despite the success of such refined models, these approaches suffer from ***representation bias*** that single-frame fails to represent the entire action segment for complex actions with the large semantic variance of various frames (illustrated in Fig. 1 (c)). Moreover, when refining the segmentation model by these biased pseudo-labels, representation bias can be accumulated and expanded, resulting in unsatisfactory performance that is still a gap compared to fully supervised approaches.

In this paper, we propose a novel framework called Debiasing-TSTAS (D-TSTAS) to reduce label bias, containing masked timestamp predictions and center-oriented timestamp expansion approaches. During the initialization phase, we aim to alleviate the focus bias problem by propagating timestamp supervision information to the contextual frames of the timestamps. In contrast to previous approaches, we propose the Masked Timestamp Predictions (MTP) approach that masks the input features of timestamps to remove the dependencies of the model on annotated frames. In this way, the initialized model is forced to reconstruct the annotated frames in corresponding output features and then predict their action categories by contextual information of timestamps. Furthermore, to capture the semantics of both timestamps and contextual frames, we adopt the Naive after our MTP.

While our initialized model can reduce focus bias by capturing more contextual information, it does not guarantee that the generated pseudo-labels avoid representation bias during refining. Inspired by query expansion, we propose a Center-oriented Timestamp Expansion (CTE) approach for obtaining potential trustworthy unlabeled frames, which we refer to as pseudo-timestamps, to progressively expand the pseudo-timestamp groups that contain more semantics than single annotated timestamps. More specifically, it consists of three steps: 1) In the generating step, we generate pseudo-labels by current pseudo-timestamp groups and the model output. 2) In the updating step, we choose the semantically dissimilar center frames of each segment in the pseudo-labels as pseudo-timestamps to expand the pseudo-timestamp groups. 3) In the segmenting step, the model is refined by pseudo-labels and our segmental confidence loss, which smooths the predicted probabilities in each action segment and maintains high confidence within pseudo-timestamp groups. The above steps are executed several times alternately to improve the model prediction in a lazy manner instead of generating pseudo-labels per epoch in previous methods [28,51].

Our contributions can be summarised as follows:

- We study the label bias in the TSTAS task and propose a novel D-TSTAS framework to reduce both focus and representation bias.
- Our masked timestamp predictions approach is the first attempt to alleviate the dependencies on timestamps, promoting the model to capture contextual information. Coupling MTP and Naive as a general solution is used to initialize the model in the TSTAS.
- Compared to sparsely annotated timestamps, our center-oriented timestamp expansion approach progressively expands pseudo-timestamp groups to contain semantic-rich motion representations of action segments.
- The proposed D-TSTAS not only outperforms state-of-the-art TSTAS approaches but also achieves competitive results compared with fully supervised approaches on three benchmark datasets.

2. Related Work

2.1. Temporal Action Segmentation

Fully Supervised Action Segmentation has been attracting increasing attention. In contrast to action recognition where trimmed videos are classified [5, 22, 30, 39, 48], temporal action segmentation requires modeling long-range dependencies to predict all the frames in the video. To achieve a larger receptive field, many approaches utilized dilated temporal convolutional networks to capture local and global dependencies [7, 25]. Despite the success of these approaches, they suffer from an over-segmentation error. To alleviate this problem, current state-of-the-art methods follow the architecture that refines the initial predictions [1, 15, 26, 40, 49] or integrates text prompts [24]. The above fully supervised approaches rely on frame-wise annotations, which are expensive to obtain.

Timestamp Supervised Action Segmentation only requires a single annotated frame for each action segment. Most previous methods [17, 28, 32, 51] apply a two-phase training scheme, initializing by Naive [28] (also called the warmup phase [51]) and then iteratively refining the learned model with generated pseudo-labels. Unlike these methods, UVAST [3] directly refines the model by generating pseudo-labels from the constrained k-medoids algorithm based on input features. Moreover, in the specific action field (surgical phase recognition), Ding et al. [6] generate only partial pseudo-labels. While the above methods for generating pseudo-labels are highly dependent on annotated

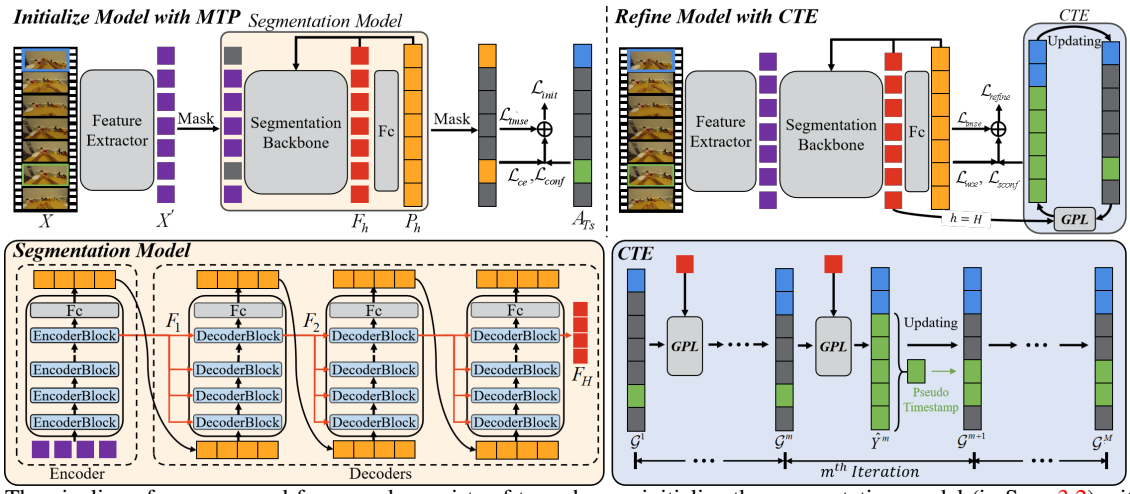


Figure 2. The pipeline of our proposed framework consists of two phases: initialize the segmentation model (in Sec. 3.2) with masked timestamp predictions (MTP in Sec. 3.3) and refine the model with center-oriented timestamp expansion (CTE in Sec. 3.4). The overall losses are introduced in Sec. 3.5. *GPL* is the abbreviation for generating pseudo-labels.

timestamps, the semantic representation of the timestamps is limited. Different from the previous solutions, we propose MTP to enable the initialized model to capture contextual information and CTE to enrich pseudo-timestamp groups instead of single timestamps.

2.2. Self-Supervised Learning

In image and video tasks, self-supervised learning has been successful for its ability to learn information feature representations without human supervision. Most of the previous works were based on the spatial-temporal orders of videos by designing an auxiliary task related to the main task where the labels can be self-annotated [2,18,21,44,46]. Inspired by the success of the vision transformer, masked modeling is widely used in image [13, 42] and video modeling [10, 38]. Unlike these works, our proposed MTP predicts categories of annotated frames by using contextual information, aiming to address the problem of focus bias for timestamp supervised action segmentation.

3. The Debiasing-TSTAS Framework

In this section, we present the details of our D-TSTAS. In Sec. 3.1, we first introduce the task of timestamp supervision and the pipeline of D-TSTAS, and the segmentation model is described in Sec. 3.2. Then, the masked timestamp predictions and the center-oriented timestamp expansion are proposed in Sec. 3.3 and Sec. 3.4, respectively. Finally, we provide the details of the loss functions in Sec. 3.5.

3.1. Problem statement and pipeline

Given a video $X = \{x_t\}_{t=1}^{T}$ with T frames and N action segments ($N \ll T$), the fully supervised temporal action segmentation requires high-cost frame-wise annotations for training and aims to predict a sequence of frame-wise action labels. In contrast to the fully supervised approaches, we consider efficient timestamp supervised segmentation, which provides a set of annotated timestamps $A_{Ts} = \{(a_n, t_n)\}_{n=1}^{N}$, where frame t_n belongs to the n-th action segment and a_n is the label n-th of action segment. While the annotation time of timestamp supervision is much less than that of the fully supervised setup, there is still a large gap in performance.

To achieve comparable performance to the fully supervised methods, we propose a framework named D-TSTAS, including an Initialization phase and a Refinement phase, as shown in Fig. 2. Specifically, following the previous [7, 16, 40, 49], we extract I3D [4] features $X' = \{x'_t\}_{t=1}^{T}$ as the input to our segmentation model. Then, to reduce focus bias, we initialize our model by masked timestamp predictions (MTP) and Naive [49] for capturing more contextual information. To further reduce representation bias, we propose the center-oriented timestamp expansion (CTE) to expand the progressive pseudo-timestamp groups and refine the segmentation model by generating pseudo-labels.

3.2. The segmentation model

The architecture of our segmentation model is shown in Fig. 2. We build a Transformer backbone of the cascaded encoder-decoder network structure [49], which contains an encoder and H-1 decoders. The linear attention [45] is applied in our backbone to balance training efficiencies and performance, which is beneficial for modeling long sequences. More details of the segmentation model are described in the supplementary material. We also explore the effect of Temporal Convolutional Network (TCN) [17, 28] or our backbone and compare the different attentions in the backbone as illustrated in Sec. 4.7.

3.3. Reducing the focus bias by MTP

As mentioned before, the previous initialization [17, 28, 32] suffers from focus bias. In this paper, we propose masked timestamp predictions (MTP) to reduce this bias as shown in Fig. 2. Specifically, given the input feature X', we mask the I3D feature x'_{t_n} of annotated timestamps by a shared learnable token to reduce the dependency of the model on annotated timestamps. The input feature of the unannotated frames is used to generate output feature F_h and predictions of timestamps P_h, where $h = 1, \ldots, H$, and H denotes the total number of stages (encoder & decoder). To enable our model to involve the semantic information of annotated timestamps, Naive is employed to train the segmentation model after the MTP. The obtained output of our initialized model captures temporal feature representations at different distances from the timestamps by propagating the timestamp supervision information to contextual frames, which is superior to the feature representations obtained by using a Naive alone in typical approaches. Unlike the self-supervised methods [10, 13, 38, 43] that attain a generic pre-trained model by reconstructing input or handcrafted feature descriptor of input, the MTP maps to annotated frames with high-level semantic information (more analyses in Sec. 4.4).

3.4. Reducing the representation bias by CTE

Due to the limitation of the expressiveness of the single-frame timestamp, the model tends to produce expressiveness bias for complex action segments, generating incorrect pseudo-labels. Inspired by query expansion, we first propose to use progressive pseudo-timestamp groups $\mathcal{G}$ to represent the action segments. Compared to the original single annotated frames, pseudo-timestamp groups provide a semantically richer representation of each action segment. We further propose the center-oriented timestamp expansion (CTE) to expand $\mathcal{G}$ while refining the model. As illustrated in Alg.1, the annotated timestamp set A_{T_s} is first used to initialize $\mathcal{G} = \{G_n\}_{n=1}^{N}$, where $G_n = \{(a_n, t_n^1)\}$ is the timestamp group of n-th action segment and the t_n^1 indicates the first annotated timestamp of the group G_n. Then, the generating step, the segmenting step, and updating step perform M iterations to expand progressive pseudo-timestamp groups $\mathcal{G}$ and refine the model by generated pseudo-labels $\hat{Y}$. The framework of the refined model with the proposed CTE approach is shown in Fig. 2 and we describe the process of the m-th iteration as follows.

Generating step. Obtain output features of final decoder $F_H = \{f_t\}_{t=1}^{T}$ and current timestamps groups $\mathcal{G}$, generating the frame-wise pseudo-labels can be reduced to estimate the action boundaries between the last timestamp $(a_i, t_i^{\Omega(i)})$ of G_i and the first timestamp (a_{i+1}, t_{i+1}^1) of G_{i+1} in temporal order, where $i = 1, \ldots, N-1$. Only $N-1$ boundaries of pseudo-labels are required to be estimated, and $\Omega(i)$ denotes the number of timestamps in the timestamps group G_i. Specifically, the stamp-to-stamp energy function as proposed in [28] is used to detect the action change between $t_i^{\Omega(i)}$ and t_{i+1}^1, and the boundaries of pseudo-label t_{b_i} can be denoted by:

$$
\begin{aligned}
& t_{b_i} = \arg\min_{\hat{t}} \sum_{t=t_i^{\Omega(j)}}^{\hat{t}} d\left(f_t, e_i\right) + \sum_{t=\hat{t}+1}^{t_{j+1}^1} d\left(f_t, e_{i+1}\right), \\
& s.t. \\
& e_i = \frac{1}{\hat{t}-t_i^{\Omega(i)}+1} \sum_{t=t_i^{\Omega(i)}}^{\hat{t}} f_t, \\
& e_{i+1} = \frac{1}{t_{i+1}^1-\hat{t}} \sum_{t=\hat{t}+1}^{t_{i+1}^1} f_t, \\
& t_i^{\Omega(i)} \leq \hat{t} < t_{i+1}^1,
\end{aligned}
\tag{1}
$$

where $d(\cdot,\cdot)$ is the Euclidean distance, e_i is the average of the embedding features between $t_i^{\Omega(i)}$ and the estimate $\hat{t}$, and e_{i+1} is the average of the segmentation feature between the estimate $\hat{t}$ and t_{i+1}^1. In Eq. (1), the pseudo-boundaries are estimated from the start of the video. The same argument holds if we estimate the pseudo-boundaries at the end of the video. The final pseudo-boundaries for t_{b_i} are the average of these two estimates, which can be expressed as follows:

$$
\begin{aligned}
& t_{b_i} = \tfrac{t_{b_i,FW} + t_{b_i,BW}}{2}, \\
& s.t. \\
& t_{b_i,FW} = \arg\min_{\hat{t}} \sum_{t=t_{b_{i-1}}}^{\hat{t}} d\left(f_t, e_i\right) + \sum_{t=\hat{t}+1}^{t_{i+1}^1} d\left(f_t, e_{i+1}\right), \\
& t_{b_i,BW} = \arg\min_{\hat{t}} \sum_{t=t_i^{\Omega(i)}}^{\hat{t}} d\left(f_t, e_i\right) + \sum_{t=\hat{t}+1}^{t_{b_{i+1}}} d\left(f_t, e_{i+1}\right), \\
& t_i^{\Omega(i)} \leq \hat{t} < t_{i+1}^1,
\end{aligned}
\tag{2}
$$

thus, we get the pseudo-boundaries $B = \{t_{b_j}\}_{j=0}^{N}$, which $t_{b_0} = 0$ and $t_{b_N} = T$. To generate the pseudo-labels $\hat{Y} = \{\hat{y}_t\}_{t=1}^{T}$, we assign the frames between $t_{b_{n-1}}$ and t_{b_n} to the same action label a_n.

Updating step. Our approach relies on the progressive pseudo-timestamp groups $\mathcal{G}$ that contain more semantic and positional information to detect pseudo-boundaries. To enrich the $\mathcal{G}$, we need to find potentially trustworthy pseudo-timestamps that are not semantically similar to original timestamps. For finding potentially trustworthy pseudo timestamps, we intuitively choose the center frames of each segment in the pseudo-labels, which are expressed as $C = \{(a_n, t_{c_n})\}_{n=1}^{N}$, and

$$
t_{c_n} = \frac{t_{b_{n-1}} + t_{b_n}}{2}. \tag{3}
$$

Since adjacent frames usually have similar semantic information in the video, new pseudo-timestamps close to the G_n might fail to reduce representation bias. To alleviate this problem, we keep a minimum temporal threshold D between the new pseudo-timestamps and the latest

Algorithm 1 Refine Model with Center-oriented Timestamp Expansion

Input: I3D features X', annotated timestamps A_{Ts} and segmentation model $\mathcal{M}$.
Output: Refined segmentation model $\mathcal{M}$.

▷ ***Initialize*** $\mathcal{G}$ ***and*** R ***by*** A_{Ts}.
for $m = 1, ..., M$ **do**
  ▷ ***Generating step.***
  $F_H \leftarrow ExtractFeature(X', \mathcal{M})$
  $B \leftarrow EstimateBoundaries(F_H, \mathcal{G})$ see Eq.2
  $\hat{Y} \leftarrow GeneratePseudoLabels(B, \mathcal{G})$
  ▷ ***Updating step.***
  $(\mathcal{G}, R) \leftarrow Update(\hat{Y}, \mathcal{G}, R)$
  ▷ ***Segmenting step.***
  for $k = 1, ..., K$ **do**
    $\mathcal{M} \leftarrow RefineModel(\mathcal{M}, X', \hat{Y}, \mathcal{G}, \mathcal{L}_{refine})$
  end for
end for

added pseudo-timestamps, i.e., $|t_{c_n} - t_{r_n}| > D$, where $R = \{t_{r_n}\}_{n=1}^{N}$ denotes the location of the latest added pseudo-timestamps each iteration (initially, $t_{r_n} = t_n$). Besides, the value of the new pseudo timestamp t_{c_n} is assigned to t_{r_n}. Finally, we obtain the new pseudo-timestamp groups $\mathcal{G}$ that are utilized for the next iteration.

Segmenting step. To further improve the quality of the predictions, we refine the segmentation model with the pseudo-labels $\hat{Y}$, new pseudo-timestamp groups $\mathcal{G}$, and loss functions $\mathcal{L}_{refine}$ (described in Sec. 3.5) through K training epochs.

3.5. Loss Functions

As shown in Fig. 2, we use the same loss as baseline [28] to initialize the model with MTP and Naive, which can be formulated as follows:

$$\mathcal{L}_{init} = \mathcal{L}_{ce} + \lambda \mathcal{L}_{tmse} + \beta \mathcal{L}_{conf}, \tag{4}$$

where $\mathcal{L}_{ce}$, $\mathcal{L}_{tmse}$, and $\mathcal{L}_{conf}$ are the cross-entropy loss based on the annotated timestamps, smoothing loss [7], and confidence loss [28], respectively. λ and β are hyper-parameters utilized to balance the contribution of each loss.

During the refinement of the model with CTE, we adapt weighted cross-entropy loss $\mathcal{L}_{wce}$ to replace cross-entropy loss $\mathcal{L}_{cls}$, which can alleviate the imbalance problem in the frequency of different action segments for each action category [15]. Unlike the [15], to get the class weight, we calculate the median of class frequencies divided by each class frequency through pseudo-labels that approximate ground truth. The previous methods [17,28,32] typically apply confidence loss proposed in [28] so that the probability of prediction decreases monotonically with increasing distance from the timestamp. However, due to focus monotonicity outside the segments, the confidence loss leads to the blurring of the boundaries of the action segments.

Segmental confidence loss. Unlike confidence loss $\mathcal{L}_{conf}$, we propose segmental confidence loss $\mathcal{L}_{sconf}$ to smooth the predicted probabilities within the action segments of pseudo-labels.

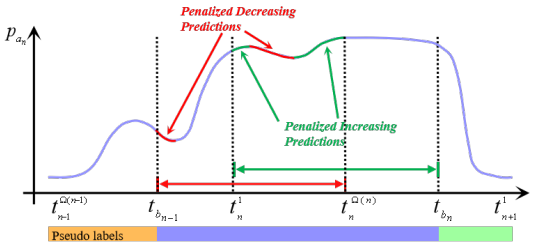


Figure 3. The segmental confidence loss penalizes decreasing predictions within the red range ($t_{b_{n-1}}$ to $t_n^{\Omega(n)}$ and increasing predictions in the green range (t_n^1 to t_{b_n}) for label a_n.

$$\begin{aligned}
\mathcal{L}_{sconf} &= \left(\sum_{n=1}^{N} \sum_{t=t_{b_{n-1}}}^{t_n^{\Omega(n)}} \delta_{a_n,1} \right) / T_1 + \left(\sum_{n=1}^{N} \sum_{t=t_n^1}^{t_{b_n}} \delta_{a_n,2} \right) / T_2, \\
\delta_{a_n,1} &= \max\left(0, p_{a_n,t-1} - p_{a_n,t}\right), \\
\delta_{a_n,2} &= \max\left(0, p_{a_n,t} - p_{a_n,t-1}\right), \\
T_1 &= \sum_{n=1}^{N} \left(t_n^{\Omega(n)} - t_{b_{n-1}} \right), \\
T_2 &= \sum_{n=1}^{N} \left(t_{b_n} - t_n^1 \right),
\end{aligned} \tag{5}$$

where $p_{a_n,t}$ is the probability of action a_n at time t. $\delta_{a_n,1}$ and $\delta_{a_n,2}$ are decreasing penalized predictions and increasing penalized predictions as illustrated in Fig. 3. T_1 and T_2 are the total sum of frames of $\delta_{a_n,1}$ and $\delta_{a_n,2}$, respectively. The effect of our loss function is visualized as shown in Fig. 5, which can prevent boundary blur caused by prediction monotonicity outside the action segments and keeps consistent high confidence between t_n^1 and $t_n^{\Omega(n)}$.

The overall loss function to refine the segmentation model with CTE is as follows:

$$\mathcal{L}_{refine} = \mathcal{L}_{wce} + \lambda \mathcal{L}_{tmse} + \gamma \mathcal{L}_{sconf}, \tag{6}$$

where λ and γ are hyper-parameters to balance the contribution of each loss.

4. Experiments

4.1. Datasets and evaluation

Datasets. Following the previous works [17, 32, 51], we evaluate our approach on three datasets: 50Salads [36], Breakfast [19], and Georgia Tech Egocentric Activities (GTEA) [8]. We perform 4-fold cross-validation on Breakfast and GTEA, and 5-fold cross-validation on 50Salads.

The **50Salads** dataset contains 50 videos (0.6M frames) with actors preparing different kinds of salads. It consists of 17 action classes which are recorded from the top view. On average, the video duration is 6.4 minutes long and each video has 20 action instances. The **Breakfast** dataset consists of 1712 third-person view videos (3.6M frames) related to breakfast preparation activities. There are 48 different action classes that were recorded in 18 different kitchens. The average video duration ranges from a few

seconds to several minutes and each video contains 6 action instances on average. The **GTEA** dataset contains 58 videos (32K frames) with actors performing various kinds of daily activities. It consists of 11 action classes. The average video duration is 1 minute long.

For all datasets, our experimental results are reported for three random seeds used for annotating frames in videos. Only in Tab. 8, we use the same annotations that Li *et al.* [28] provide for a fair comparison.

Evaluation Metrics. Following previous works [7, 26], we use the standard metrics for fully supervised temporal action segmentation and report the frame-wise accuracy (Acc), segmental edit distance (Edit) and segmental F1 scores with overlapping thresholds of 10%, 25%, and 50%, denoted by $F1@\{10, 25, 50\}$.

4.2. Implementation details

We set $H = 4$ as in [49] and train our model for 290 epochs, including 80 epochs for MTP, 30 epochs for Naive, and 180 ($K = 30, M = 6$) epochs for CTE. D is set to $\{5, 20, 10\}$ for GTEA, 50Salads, and Breakfast, which is related to the average action duration. The batch size is 8 and an AdamW optimizer with a learning rate of 0.0005 and weight decay of 0.05 is used. For the loss function, we set $\lambda = 0.15, \beta = 0.075$ as in [28], and set $\gamma = 1$. All experiments in the comparison study use the above setting and are conducted with NVIDIA GeForce RTX 2080 Ti GPUs.

4.3. Effectiveness of MTP

We first compare the initialization of the MTP and Naive [28] on TCN backbone [28] and our backbone. As shown in Tab. 1, our approach outperforms Naive with a large margin F1 and edit scores. This is because the Naive initialized model by only using sparse timestamps, the model produces focus bias, incorrectly recognizing frames in various action categories similar to annotated frames. On the contrary, our approach reduces focus bias by masking the timestamps and predicting the category of annotated frames using contextual information. Moreover, we combine MTP and Naive to generate better initialized predictions by exploiting semantic information with annotated timestamps and the context of annotated frames. These are also reflected in the performance as illustrated in Fig. 4. We also evaluate the order of MTP and Naive in the supplementary.

To further explore the effectiveness of the MTP, we explore the impact of different initialization strategies on the refinement model, as shown in Tab. 2. To conduct a fair comparison, the same refinement approach in [28], containing 20 epochs of training, is used to refine the model. While our MTP gives an additional boost in performance, the best performance is achieved when MTP and Naive are combined. This illustrates that a good initialization model is beneficial for refining the model. While the MTP of the

	Initialization	GTEA					50Salads				
		F1@{10, 25, 50}			Edit	Acc	F1@{10, 25, 50}			Edit	Acc
TCN	Naive	59.7	55.3	39.6	51.1	56.5	47.9	43.3	34.0	37.2	69.6
	MTP	63.4	59.7	45.9	56.4	53.4	59.8	55.1	44.9	52.1	70.1
	MTP+Naive	74.4	68.1	52.1	67.2	62.8	60.2	55.7	45.5	52.8	70.5
ours	Naive	47.4	42.1	33.0	45.1	47.6	41.3	36.1	25.3	40.0	37.9
	MTP	67.6	62.7	47.3	61.0	60.5	55.4	51.1	40.9	46.8	68.7
	MTP+Naive	77.0	72.4	56.1	70.5	65.4	62.6	58.1	47.4	52.6	74.0

Table 1. Impact of different initialization strategies and backbones without refinement on GTEA and 50Salads datasets.

	Initialization	GTEA					50Salads				
		F1@{10, 25, 50}			Edit	Acc	F1@{10, 25, 50}			Edit	Acc
TCN	Naive	78.9	73.0	55.4	72.3	66.4	73.9	70.9	60.1	66.8	64.1
	MTP	81.4	76.9	58.5	75.9	66.8	71.5	68.3	58.8	65.5	73.6
	MTP+Naive	84.0	79.0	60.4	79.5	67.8	72.0	69.0	58.6	64.6	75.6
ours	Naive	76.0	71.4	55.3	71.7	69.0	72.0	67.9	54.7	64.9	73.7
	MTP	82.6	79.6	61.6	77.5	70.2	73.9	69.4	58.2	67.0	75.3
	MTP+Naive	86.2	83.2	66.3	82.7	72.0	74.5	71.1	60.0	67.3	77.6

Table 2. Impact of different initialization strategies and backbones with refinement [28] on GTEA and 50Salads datasets.

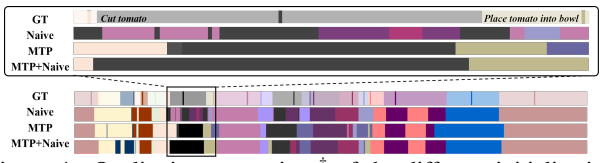


Figure 4. Qualitative comparison[‡] of the different initialization strategies with TCN backbone [28] on the 50Salads dataset. Naive only correctly identifies frames close to the timestamp, which are also similar frames to annotated timestamps. MTP reduces this bias by directing the model to capture contextual information, but the boundaries of segments are not accurate. Adding Naive after MTP generates better predictions as the model exploits semantic information with annotated timestamps and the context of annotated frames.

TCN backbone on 50Salads achieves good frame-wise accuracy, it suffers from over-segmentation during refining as evidenced by the low F1 and Edit scores. Meanwhile, our backbone based on Transformer can alleviate this problem.

4.4. Impact of mapping target for MTP

Our approach initializes the model by mapping the masked features to the label of timestamps (Feature-to-Label), which is described in Sec. 3.3. To analyze the impact of the mapping target, we train another model (Feature-to-Feature) that minimizes the l_2 distance between the output feature f_{t_n} and the input I3D feature x'_{t_n}. Specifically, we use an additional linear layer to adjust the dimensions of f_{t_n} to match the dimensions of the x'_{t_n}. For a fair comparison, the Feature-to-Feature model follows the same initialization method (MTP+Native) and refinement method [28] based on our backbone as Feature-to-Label. As shown in Tab. 2 and Tab. 3, compared to the Naive, Feature-to-Feature slightly improves the performance, which validates that the masking strategy achieves better initialization

[‡]Since there is no timestamp on the test set, we visualize the focus bias clearly by timestamps and predictions on the training set.

Dataset	Target	F1@{10,25,50}			Edit	Acc
GTEA	Feature-to-Feature	75.6	72.4	56.8	65.5	71.2
	Feature-to-Label	86.2	83.2	66.3	82.7	72.0
50Salads	Feature-to-Feature	72.3	69.1	58.9	65.8	75.9
	Feature-to-Label	74.5	71.1	60.0	67.3	77.6

Table 3. Impact of mapping target for MTP on GTEA and 50Salads datasets.

Refinement	F1@{10, 25, 50}			Edit	Acc
baseline [28] (epoch = 20)	74.5	71.1	60.0	67.3	77.6
CTE ($K = 10, M = 2$)	76.4	73.1	62.6	69.6	77.5

Table 4. Comparison with the baseline in the refining phase on the 50Salads dataset.

Impact of K	F1@{10, 25, 50}			Edit	Acc	Acc_{PL}	Acc_{TS}
CTE ($K = 10, M = 6$)	80.1	77.8	66.3	72.9	78.0	87.1	97.3
CTE ($K = 20, M = 6$)	82.8	80.6	69.6	75.7	79.1	87.3	97.2
CTE ($K = 30, M = 6$)	84.4	82.2	71.2	77.5	80.2	87.4	97.2
CTE ($K = 40, M = 6$)	83.5	81.3	71.2	76.7	80.0	87.2	97.0
Impact of M	F1@{10, 25, 50}			Edit	Acc	Acc_{PL}	Acc_{Ti}
CTE ($K = 30, M = 2$)	81.0	78.4	68.0	74.1	79.7	88.6	97.9
CTE ($K = 30, M = 4$)	83.2	80.6	70.4	76.3	79.8	87.8	97.4
CTE ($K = 30, M = 6$)	84.4	82.2	71.2	77.5	80.2	87.4	97.2
CTE ($K = 30, M = 8$)	83.7	81.9	70.3	77.8	79.2	86.4	97.0
CTE ($K = 30, M = 10$)	83.7	81.0	69.5	76.5	78.8	85.6	96.4

Table 5. Impact of epoch-per-segmentation step K and the number of iterations M in CTE on 50Salads dataset. Acc_{PL} and Acc_{TS} denote the accuracy of pseudo-labels and timestamps for each video, respectively.

of the model. However, it performed worse than Feature-to-Label. This is mainly because Feature-to-Label promotes the model to capture high-level semantic information, which is beneficial for the propagation of supervised signals of timestamps to unlabeled frames in the TSTAS task. We also discuss the number of masked frames with Feature-to-Label and Feature-to-Feature (please refer to the supplementary).

4.5. Effectiveness of CTE

To verify the effectiveness of the proposed CTE during the refining phase, we compare the CTE with the baseline that employs the refining approach [28] and perform the comparison experiments based on our initialized model. As shown in Tab. 4, our approach outperforms the baseline. This is because our method captures more semantic information within the action segment compared to the baseline by expanding the timestamp groups. As shown in Tab. 5, the addition of pseudo-labels is trustworthy, and pseudo-timestamp groups are used to generate better pseudo-labels, as indicated by the high accuracy of pseudo-labels and timestamps. The effect of CTE can also be seen in the qualitative result, please refer to the supplementary.

4.6. Ablation Study of CTE

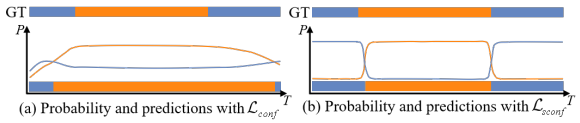
The effect of the center-oriented timestamp expansion is controlled by two hyper-parameters: K and M. In this section, we study the impact of the two parameters with different values of K and M, as shown in Tab. 5.

	Strategies	GTEA				
		F1@{10, 25, 50}			Edit	Acc
Local attention	Naive	69.7	62.7	45.4	62.0	59.8
	MTP	75.5	72.4	53.3	72.0	64.8
	MTP+Naive	77.4	73.6	54.9	71.1	66.3
	MTP+Naive+CTE	89.1	86.8	74.5	85.5	73.2
Linear attention	Naive	47.4	42.1	33.0	45.1	47.6
	MTP	67.6	62.7	47.3	61.0	60.5
	MTP+Naive	77.0	72.4	56.1	70.5	65.4
	MTP+Naive+CTE	91.5	90.1	76.2	88.5	75.7

Table 6. Impact of different attention architectures in our backbone on the GTEA dataset.

$\mathcal{L}_{ce}$	$\mathcal{L}_{wce}$	$\mathcal{L}_{conf}$	$\mathcal{L}_{sconf}$	F1@{10, 25, 50}			Edit	Acc
✓		✓		82.5	79.9	68.5	76.2	78.7
✓			✓	83.8	81.1	70.6	76.6	79.4
	✓	✓		82.9	80.4	65.8	76.8	75.6
	✓		✓	84.2	82.1	71.5	77.6	80.0

Table 7. Comparing different loss functions on the 50Salads dataset.



(a) Probability and predictions with $\mathcal{L}_{conf}$ (b) Probability and predictions with $\mathcal{L}_{sconf}$

Figure 5. Quality comparison of confidence loss [28] and segmental confidence loss.

Impact of epoch-per-segmentation stage K. The parameter of K affects the refinement model by using the same pseudo-labels in the segmenting step. Our default value is $K = 30$. Increasing the value K from 10 to 30 achieves better performance because sufficient training epochs lead to better convergence. However, setting $K = 40$ results in a slight drop in performance compared with a default value. This drop in performance is due to the over-fitting problem.

Impact of the number of iterations M. This hyper-parameter defines the number of times to supplement the timestamp groups and generate pseudo-labels. Reducing M to 2 still improves the performance but is not as good as the default value of $M = 6$. Increasing its value to $M = 10$ also causes a degradation in performance. This drop in performance is because adding the incorrect pseudo-timestamp results in a decrease in the accuracy of the pseudo-labels, which affects the predictions of the refinement model.

4.7. Impact of the backbones

We compare our backbone with TCN [28] backbone as shown in Tab. 1 and Tab. 2. When the Naive is used alone, our backbone causes worse performance in the initialization and refinement phase compared with the TCN backbone. This is because the linear attention in our backbone with the Naive approach incorrectly identifies more frames with semantic similarity to timestamps by capturing global dependencies. In contrast, MTP can improve the performance of our backbone by a large margin, which indicates

Supervision	Method	GTEA					50Salads					Breakfast				
		F1@{10, 25, 50}			Edit	Acc	F1@{10, 25, 50}			Edit	Acc	F1@{10, 25, 50}			Edit	Acc
Fully	MS-TCN [7]	87.5	85.4	74.6	81.4	79.2	76.3	74.0	64.5	67.9	80.7	52.6	48.1	37.9	61.7	66.3
	MS-TCN++ [25]	88.8	85.7	76.0	83.5	80.1	80.7	78.5	70.1	74.3	83.7	64.1	58.6	45.9	65.6	67.6
	BCN [40]	88.5	87.1	77.3	84.4	79.8	82.3	81.3	74.0	74.3	84.4	68.7	65.5	55.0	66.2	70.4
	ASRF [16]	89.4	87.8	79.8	83.7	77.3	84.9	83.5	77.3	79.3	84.5	74.3	68.9	56.1	72.4	67.6
	ASFormer [49]	90.1	88.8	79.2	84.6	79.7	85.1	83.4	76.0	79.6	85.6	76.0	70.6	57.4	75.0	73.5
	ETSN [26]	91.1	90.0	77.9	86.2	78.2	85.2	83.9	75.4	78.8	82.0	74.0	69.0	56.2	70.3	67.8
	ICC [34]	91.4	89.1	80.5	87.8	82.0	83.8	82.0	74.3	76.1	85.0	72.4	68.5	55.9	68.6	75.2
	UVAST [3]	92.7	91.3	81.0	92.1	80.2	89.1	87.6	81.7	83.9	87.4	76.9	71.5	58.0	77.1	69.7
	DPRN [31]	92.9	92.0	82.9	90.9	82.0	87.8	86.3	79.4	82.0	87.2	75.6	70.5	57.6	75.1	71.7
	Br-Prompt+ ASFormer [23]	94.1	92.0	83.0	91.6	81.2	89.2	87.8	81.3	83.8	88.1	-	-	-	-	-
Semi	ICC(5%) [34]	77.9	71.6	54.6	71.4	68.2	52.9	49.0	36.6	45.6	61.3	60.2	53.5	35.6	56.6	65.3
	ICC(10%) [34]	83.7	81.9	66.6	76.4	73.3	67.3	64.9	49.2	56.9	68.6	64.6	59.0	42.2	61.9	68.8
Timestamp	Li et al. [28]	78.9	73.0	55.4	72.3	66.4	73.9	70.9	60.1	66.8	75.6	70.5	63.6	47.4	69.9	64.1
	Khan et al. [17]	81.5	77.5	60.8	75.6	66.1	75.1	72.3	61.0	67.6	75.1	67.9	61.0	45.3	67.0	61.4
	Zhao et al. [51]	84.3	81.7	64.8	79.8	74.4	78.5	75.5	63.4	71.8	77.7	73.1	66.5	49.4	72.6	64.6
	EM-TSS [32]	-	82.7	66.5	82.3	70.5	-	75.9	64.7	71.6	77.9	-	63.7	49.8	67.2	**67.0**
	UVAST+ alignment decoder [3]	70.8	63.5	49.2	88.2	55.3	75.7	70.6	58.2	78.4	67.8	72.0	64.1	48.6	74.3	60.2
	UVAST+Viterbi [3]	87.2	83.7	66.0	**89.3**	70.5	83.0	79.6	65.9	78.2	77.0	71.3	63.3	48.3	74.1	60.7
	UVAST+FIFA [3]	80.7	75.2	57.4	88.7	66.0	80.2	74.9	61.6	**78.6**	72.5	72.0	64.2	47.6	74.1	60.3
	D-TSTAS	**91.5**	**90.1**	**76.2**	88.5	**75.7**	**84.2**	**82.1**	**71.5**	77.6	**80.0**	**76.7**	**69.3**	**50.7**	**75.8**	65.7

Table 8. Comparison with different levels of supervision on all three datasets. D-TSTAS uses the backbone that combines the cascaded encoder-decoder network structure [49] and the linear attention architecture [45].

that MTP is effective in alleviating focus bias. Moreover, MTP and transformer backbone are more complementary and can help to capture contextual information.

We further explore the impact of different attention architectures, including linear attention in Flowformer [45] and local attention in ASFormer [49]. The MTP and CTE are added to the local attention model separately by setting the batch size to 1. Tab. 6 shows that our MTP and CTE can improve the performance of the model. With both MTP and CTE, the linear attention model outperforms the local attention model. Moreover, we compare the training time for the linear attention model and local attention model in the supplementary. To balance training time and performance, we use linear attention in our backbone.

4.8. Comparing Different Loss Functions

Tab. 7 shows the comparison of our loss function with the previous loss function [28]. Our proposed segmental confidence loss achieves better F1 and edit scores compared with the confidence loss [28]. This is because $\mathcal{L}_{conf}$ blurs the boundaries of the action segmentation when it forces the predicted probabilities to remain monotonic between adjacent annotated timestamps. In contrast, our loss function truncates smoothing through the pseudo-labels at the action boundary. This effect can also be seen in the qualitative result shown in Fig. 5. Moreover, $\mathcal{L}_{wce}$ replacing $\mathcal{L}_{ce}$ also improves performance by alleviating the imbalance of segments for action categories.

4.9. Comparison with the state-of-the-Arts

We compare our method with recent approaches under different levels of supervision on three datasets. By reducing the label bias, D-TSTAS outperforms current state-of-the-art approaches under timestamp supervision and semi-supervised, as shown in Tab. 8. Please refer to the supplementary for the qualitative results of our method and the baseline. Our approach further reduces the gap to fully supervised approaches and only involves single-frame annotations for each action segment. Moreover, our approach performs better in rare annotation cases compared with other methods [28, 32] (please refer to the supplementary).

5. Conclusion and Limitation

In this paper, we propose a Debiasing-TSTAS framework to alleviate the issue of label bias in the TSTAS task. Instead of the commonly used Naive, we additionally use masked timestamp prediction to initialize the model, which models contextual information of timestamps. Moreover, we utilize a center-oriented timestamp expansion to capture a more efficient representation of action segments. We further introduced a segmental confidence loss that improves prediction quality. Our approach outperforms state-of-the-art TSTAS methods on three datasets and achieves comparable performance to the fully supervised setup.

Limitation. The annotation frames for each action segment are essential to our approach, which limits the annotator not to miss any action segment [32, 35]. To release this limitation, we consider replacing the pseudo-labeling of hard weight in the cross-entropy loss with soft weight as mentioned in [32] and detecting pseudo-timestamps from the soft weight.

Acknowledgments: This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant DUT22JC14.

References

[1] Hyemin Ahn and Dongheui Lee. Refining action segmentation with hierarchical video representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16302–16310, October 2021. 2

[2] Unaiza Ahsan, Rishi Madhok, and Irfan Essa. Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 179–189. IEEE, 2019. 3

[3] Nadine Behrmann, S Alireza Golestaneh, Zico Kolter, Juergen Gall, and Mehdi Noroozi. Unified fully and timestamp supervised temporal action segmentation via sequence to sequence translation. In *European Conference on Computer Vision*, pages 52–68. Springer, 2022. 2, 8

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 3

[5] Ishan Rajendrakumar Dave, Chen Chen, and Mubarak Shah. Spact: Self-supervised privacy preservation for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20164–20173, June 2022. 2

[6] Xinpeng Ding, Xinjian Yan, Zixun Wang, Wei Zhao, Jian Zhuang, Xiaowei Xu, and Xiaomeng Li. Less is more: Surgical phase recognition from timestamp supervision. *IEEE Transactions on Medical Imaging*, 2023. 2

[7] Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3575–3584, 2019. 1, 2, 3, 5, 6, 8

[8] Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in egocentric activities. In *CVPR 2011*, pages 3281–3288. IEEE, 2011. 5

[9] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 203–213, 2020. 1

[10] Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. Masked autoencoders as spatiotemporal learners. In *Advances in Neural Information Processing Systems*. 3, 4

[11] Shang-Hua Gao, Qi Han, Zhong-Yu Li, Pai Peng, Liang Wang, and Ming-Ming Cheng. Global2local: Efficient structure search for video action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16805–16814, 2021. 1

[12] Guoqiang Gong, Xinghan Wang, Yadong Mu, and Qi Tian. Learning temporal co-attention models for unsupervised video action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9819–9828, 2020. 1

[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 3, 4

[14] Tariq Iqbal and Laurel D Riek. Human-robot teaming: Approaches from joint action and dynamical systems. *Humanoid robotics: A reference*, pages 2293–2312, 2019. 1

[15] Yuchi Ishikawa, Seito Kasai, Yoshimitsu Aoki, and Hirokatsu Kataoka. Alleviating over-segmentation errors by detecting action boundaries. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2322–2331, January 2021. 2, 5

[16] Yuchi Ishikawa, Seito Kasai, Yoshimitsu Aoki, and Hirokatsu Kataoka. Alleviating over-segmentation errors by detecting action boundaries. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2322–2331, 2021. 3, 8

[17] Hamza Khan, Sanjay Haresh, Awais Ahmed, Shakeeb Siddiqui, Andrey Konin, M Zeeshan Zia, and Quoc-Huy Tran. Timestamp-supervised action segmentation with graph convolutional networks. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10619–10626. IEEE, 2022. 1, 2, 3, 4, 5, 8

[18] Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8545–8552, 2019. 3

[19] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 780–787, 2014. 5

[20] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017. 1

[21] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[22] Dong Li, Zhaofan Qiu, Yingwei Pan, Ting Yao, Houqiang Li, and Tao Mei. Representing videos as discriminative sub-graphs for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3310–3319, June 2021. 2

[23] Muheng Li, Lei Chen, Yueqi Duan, Zhilan Hu, Jianjiang Feng, Jie Zhou, and Jiwen Lu. Bridge-prompt: Towards ordinal action understanding in instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19880–19889, 2022. 1, 8

[24] Muheng Li, Lei Chen, Yueqi Duan, Zhilan Hu, Jianjiang Feng, Jie Zhou, and Jiwen Lu. Bridge-prompt: Towards ordinal action understanding in instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19880–19889, June 2022. 2

[25] Shi-Jie Li, Yazan AbuFarha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. Ms-tcn++: Multi-stage temporal convolu-

tional network for action segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2, 8

[26] Yunheng Li, Zhuben Dong, Kaiyuan Liu, Lin Feng, Lianyu Hu, Jie Zhu, Li Xu, Shenglan Liu, et al. Efficient two-step networks for temporal action segmentation. *Neurocomputing*, 454:373–381, 2021. 2, 6, 8

[27] Yan Li, Bin Ji, Xintian Shi, Jianguo Zhang, Bin Kang, and Limin Wang. Tea: Temporal excitation and aggregation for action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2020. 1

[28] Zhe Li, Yazan Abu Farha, and Jurgen Gall. Temporal action segmentation from timestamp supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8365–8374, 2021. 1, 2, 3, 4, 5, 6, 7, 8

[29] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning salient boundary feature for anchor-free temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3320–3329, 2021. 1

[30] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3202–3211, June 2022. 2

[31] Junyong Park, Daekyum Kim, Sejoon Huh, and Sungho Jo. Maximization and restoration: Action segmentation through dilation passing and temporal reconstruction. *Pattern Recognition*, 129:108764, 2022. 8

[32] Rahul Rahaman, Dipika Singhania, Alexandre Thiery, and Angela Yao. A generalized and robust framework for timestamp supervision in temporal action segmentation. In *European Conference on Computer Vision*, pages 279–296. Springer, 2022. 1, 2, 4, 5, 8

[33] Laurel D Riek. Healthcare robotics. *Communications of the ACM*, 60(11):68–78, 2017. 1

[34] Dipika Singhania, Rahul Rahaman, and Angela Yao. Iterative contrast-classify for semi-supervised temporal action segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2262–2270, 2022. 8

[35] Yaser Souri, Yazan Abu Farha, Emad Bahrami, Gianpiero Francesca, and Juergen Gall. Robust action segmentation from timestamp supervision. 2022. 8

[36] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738, 2013. 5

[37] Praveen Tirupattur, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. Modeling multi-label action dependencies for temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1460–1470, 2021. 1

[38] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems*. 3, 4

[39] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. Tdn: Temporal difference networks for efficient action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1895–1904, 2021. 1, 2

[40] Zhenzhi Wang, Ziteng Gao, Limin Wang, Zhifeng Li, and Gangshan Wu. Boundary-aware cascade networks for temporal action segmentation. In *European Conference on Computer Vision*, pages 34–51. Springer, 2020. 2, 3, 8

[41] Zhengwei Wang, Qi She, and Aljosa Smolic. Action-net: Multipath excitation for action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13214–13223, 2021. 1

[42] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14668–14678, 2022. 3

[43] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14668–14678, June 2022. 4

[44] Donglai Wei, Joseph J. Lim, Andrew Zisserman, and William T. Freeman. Learning and using the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3

[45] Haixu Wu, Jialong Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Flowformer: Linearizing transformers with conservation flows. In *International Conference on Machine Learning*, pages 24226–24242. PMLR, 2022. 3, 8

[46] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[47] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 591–600, 2020. 1

[48] Jiewen Yang, Xingbo Dong, Liujun Liu, Chao Zhang, Jiajun Shen, and Dahai Yu. Recurring the transformer for video action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14063–14073, June 2022. 2

[49] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. In *The British Machine Vision Conference (BMVC)*, 2021. 2, 3, 6, 8

[50] Da Zhang, Xiyang Dai, and Yuan-Fang Wang. Metal: Minimum effort temporal activity localization in untrimmed videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3882–3892, 2020. 1

[51] Yang Zhao and Yan Song. Turning to a teacher for timestamp supervised temporal action segmentation. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 01–06. IEEE, 2022. 2, 5, 8