Timestamp Query Transformer for Temporal Action Segmentation

Tieqiao Wang Sinisa Todorovic
Oregon State University
{wangtie, sinisa}@oregonstate.edu
https://github.com/tqosu/TQT

Abstract

This work addresses action segmentation in videos under sparse timestamp supervision, where only a single frame per action segment—referred to as a timestamp—is labeled during training. We propose the Timestamp Query Transformer (TQT) that treats timestamps as learnable class query tokens. While existing approaches rely on iterative, multi-step generation of framewise pseudo-labels, TQT directly predicts temporal segmentation masks by leveraging query-feature cross-attention. This design enables fully end-to-end learning and maximizes the utility of sparse labels from the entire training dataset, rather than relying on only a few local timestamps within each training video as in prior work. Experiments on the GTEA, 50Salads, and Breakfast datasets demonstrate that TQT outperforms SOTA methods by up to 5.8% in accuracy and 7.7% in F1@50. The model and code will be released.

1. Introduction

Action segmentation is a cornerstone of video understanding, with broad applications in video surveillance, robotics, and automation [11]. Existing fully supervised approaches typically require large training datasets with dense frame-level annotations, which are often prohibitively expensive to obtain [1–3, 16, 27, 31, 40, 41]. As a more scalable alternative, a substantial body of work has explored weakly supervised methods. However, these approaches often suffer from a significant performance gap compared to their fully supervised counterparts [5, 10, 13, 19–23, 30, 34, 36, 38, 39, 42]. Recently, this gap has been narrowed by more cost-effective methods that leverage sparse timestamp supervision, illustrated in Fig. 1, where only a single, randomly selected frame per action segment—referred to as a timestamp—is labeled during training [4, 12, 17, 26, 32, 44]. These methods typically adopt a three-step training strategy: (i) pretraining on the annotated timestamps, (ii) generating pseudo-labels for unlabeled frames while enforcing temporal smoothness, and (iii) training a fully supervised model using the generated pseudo-labels. Steps (ii) and (iii) are typically repeated for refinement.

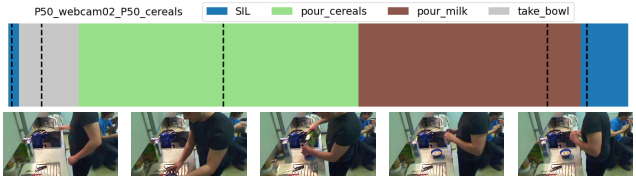

Figure 1. Timestamp setting: For each action instance, a single random frame is selected and annotated. The example above, from the Breakfast dataset, illustrates video annotations where different colors represent distinct action classes. Dashed vertical lines indicate the locations of the selected frames (displayed at the bottom).

Existing timestamp-supervised approaches face several limitations. First, they generate framewise pseudo-labels via frame clustering based on local feature similarities, lacking global information from all annotated timestamps across the dataset. This often leads to suboptimal action boundary detection. Second, their frame clustering is usually heuristic-based, which hinders end-to-end training. Third, they employ computationally inefficient training strategies—such as iterative pseudo-label refinement [26], teacher-student framework [44], or mask modeling [28].

Leveraging the aforementioned three-step training framework of prior work, we aim to improve the second step of generating pseudo-labels. We alleviate the need for multiple rounds of framewise pseudo-label refinement during training and avoid heuristic action boundary detection based on local feature similarities. To this end, we propose the *Timestamp Query Transformer* (TQT) that treats annotated timestamps as learnable, class-aware query tokens (or simply queries). TQT leverages query-feature cross-attention to directly predict temporal segmentation masks. Because these queries encode feature prototypes of action classes across the entire training dataset, the resulting query-feature similarities capture rich semantic context across the dataset. Importantly, TQT bypasses heuristic frame clustering used in prior work, enabling fully end-to-end learning.

Our design draws inspiration from object segmentation transformers, like Mask2Former [9], which use learnable query tokens to decode instance masks. However, Mask2Former and similar models are specialized for 2D spatial segmentation, and adapting them to the temporal domain of action segmentation under sparse timestamp supervision poses nontrivial challenges that remain unexplored.

Our contributions are threefold: (1) To the best of our knowledge, our approach is the first to replace iterative heuristic pipelines with end-to-end query-token learning for timestamp-supervised action segmentation; (2) A novel design of query tokens in a transformer where annotated timestamps serve as anchors for learnable class prototypes, capturing dataset-wide class-aware action representations; and (3) State-of-the-art (SOTA) results on benchmark datasets (GTEA, 50Salads, and Breakfast), surpassing prior methods by up to 5.8% in accuracy and 7.7% in F1@50.

2. Related Work

This section reviews closely related work.

Fully supervised approaches to action segmentation rely on dense, frame-wise annotations to train models for predicting action labels across video frames [4, 14, 24, 33, 43]. Early approaches relied on dilated temporal convolutional networks (e.g., MS-TCN [14]) to capture local dependencies but suffered from over-segmentation errors. Recent transformer-based models (e.g., ASFormer[43], FACT [31]) improved long-range modeling but require exhaustive frame-wise annotations. Our approach differs fundamentally in its problem setting, as we operate under timestamp supervision, requiring only a single annotated frame per action segment. Fully supervised methods assume access to dense annotations, which are costly and impractical for large-scale video datasets. Thus, a direct comparison with these methods is not applicable, as we target a more annotation-efficient setting.

Timestamp-supervised action segmentation minimizes annotation costs by requiring only one labeled frame per action segment [4, 12, 17, 26, 32, 44]. Most methods employ a two-phase pipeline: model initialization followed by iterative pseudo-label refinement. For example, Li et al. [26] and UVAST [4] generate pseudo-labels via local feature clustering (e.g., k-medoids), which is sensitive to feature quality and overlooks global context. EM-TSS [32] models label uncertainty using expectation-maximization, while Zhao and Song [44] uses a teacher-student framework. Both approaches require multiple refinement iterations, risking error accumulation and high computational costs. Liu et al. [28] mitigates label bias with masked timestamp prediction but retains a complex multi-step pipeline. These methods primarily rely on local or video-specific features, limiting their ability to capture dataset-wide patterns. In contrast, TQT treats timestamps as optimizable anchors for direct action segmentation, significantly closing the performance gap with fully supervised methods.

Query-based segmentation methods, which use learnable tokens to anchor object or action representations, are most related to our approach. In images, approaches like DETR [6] introduce object queries for end-to-end detection, while MaskFormer [8] and Mask2Former [9] extend this paradigm to segmentation by decoding regions or objects via learnable queries. For unsupervised object discovery, Slot Attention [29] demonstrates that a fixed set of learnable slots can decompose scenes without labels. In the video domain, UVAST [4] adapts queries for timestamp-supervised action segmentation but still relies on clustering rather than direct query-based boundary prediction. Fully supervised methods, like FACT [31] and BAFormer [40], depend on dense frame-level annotations to model temporal dependencies or boundaries with queries. In contrast, our approach is the first to leverage learnable, class-aware queries directly anchored at sparse timestamps for weakly supervised action segmentation, enabling efficient, end-to-end segmentation without iterative clustering or dense annotation requirements, and demonstrating the effectiveness of query-based models in annotation-limited video tasks.

3. Timestamp Query Transformer (TQT)

This section, first, formalizes the problem of timestamp-supervised action segmentation, and then specifies the TQT architecture and its training objective.

3.1. Problem Formulation

Following prior work [4, 12, 17, 26, 32, 44], for a video sequence of T RGB frames $\mathbf{I} = [\mathbf{I}_1, \ldots, \mathbf{I}_t, \ldots, \mathbf{I}_T]$, where $\mathbf{I}_t \in \mathbb{R}^{H \times W \times 3}$, we extract I3D features $\mathbf{x} \in \mathbb{R}^{d_{\text{in}} \times T}$ using a frozen backbone network, as described in [14]. The goal of action segmentation is to assign an action label $y_t \in \mathcal{Y}$ to each frame t, where $\mathcal{Y}$ is a predefined set of action classes. In timestamp-based supervision, a training video with N action segments, where $N \ll T$, is annotated with N ground-truth labels $\mathbf{y} = [y_{t_1}, \ldots, y_{t_N}]$ at frames $\mathbf{t} = [t_1, \ldots, t_N]$, with each timestamp t_n randomly selected from the nth action segment.

3.2. TQT Network Architecture

As shown in Fig. 2, TQT comprises three main modules: a backbone network, a frame decoder, and our proposed Timestamp Action Decoder. During training, all three modules are used in steps (i) and (ii)—pretraining on the sparse timestamps and generating pseudo-labels for all frames—while only the backbone and frame decoder are employed in step (iii), which involves full supervision using the generated pseudo-labels. At inference time on test videos, only the backbone and frame decoder are used for evaluation.

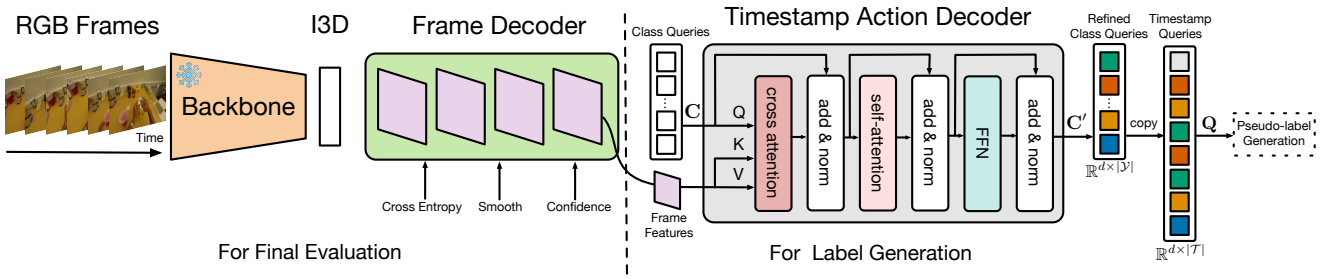

Figure 2. TQT comprises a backbone network, a frame decoder, and Timestamp Action Decoder (TAD). Annotated timestamps guide learning of class embeddings, which serve as action query tokens in TAD to generate pseudo-labels for unlabeled frames. This enables the subsequent fully supervised training on both timestamp and pseudo labels. In the fully supervised training stage, TAD is excluded, and only the frame decoder is updated. At inference time on test videos, only the backbone and frame decoder are used for evaluation.

3.2.1. Backbone and Frame Decoder

TQT is agnostic to the choice of backbone network and frame decoder, making it easily adaptable to a wide range of existing architectures. As illustrated in Fig. 2, we use a frozen I3D backbone [7] to extract spatiotemporal features $\mathbf{x} \in \mathbb{R}^{d_{\text{in}} \times T}$, following the protocol in [14]. These features are passed to a frame decoder, producing refined per-frame representations $\mathbf{z} = [\mathbf{z}_1, \dots, \mathbf{z}_T]$, which serve as *keys* and *values* in cross-attention of the subsequent module of TQT, called Timestamp Action Decoder. While TQT is compatible with any state-of-the-art frame decoder, in this work we evaluate two alternatives: a convolution-based frame decoder [26] and a transformer-based frame decoder [28].

3.2.2. Timestamp Action Decoder

As shown in Fig. 2, Timestamp Action Decoder (TAD) consists of a single transformer layer with cross- and self-attention. In training, TAD learns class-specific feature prototypes, $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_{|\mathcal{Y}|}] \in \mathbb{R}^{d \times |\mathcal{Y}|}$, where $|\mathcal{Y}|$ is the total number of action classes. These feature prototypes serve as class *query tokens* (or queries) in the cross-attention of TAD. The class queries $\mathbf{C}$ attend to frame features $\mathbf{z}$, which serve as keys and values in cross-attention, resulting in the updated query embeddings

$$\mathbf{C}' = \text{ActionDecoder}(\mathbf{C}, \mathbf{z}). \tag{1}$$

3.2.3. TQT Training for Pseudo-Label Generation

Before generating pseudo-labels for every frame, we train the frame decoder and TAD jointly in an end-to-end manner on the N annoted timestamps. Specifically, we first compute class scores for all frames as

$$\mathbf{p} = \mathbf{z}^\top \mathbf{C}' \in \mathbb{R}^{T \times |\mathcal{Y}|}, \tag{2}$$

and then supervise these predictions using the cross-entropy loss along with two additional regularization terms: smoothness loss and confidence loss. The total loss is a weighted sum of these three losses, with hyperparameters chosen as in [26]. Below, we detail each loss term.

The cross-entropy loss, $\mathcal{L}_{\text{ce}}$, is applied only at the N annotated timestamps as

$$\mathcal{L}_{\text{ce}} = -\frac{1}{N} \sum_{n=1}^{N} \log \mathbf{p}(t_n, y_{t_n}), \tag{3}$$

where $\mathbf{p}(t_n, y_{t_n})$ denotes the predicted score for the ground-truth class y_{t_n} of the timestamp frame t_n.

The smoothness loss, $\mathcal{L}_{\text{s}}$, encourages temporal consistency in the predicted class distributions along the video sequence as

$$\mathcal{L}_{\text{s}} = \frac{1}{T|\mathcal{Y}|} \sum_{t,y} \min\left((\log \mathbf{p}(t, y) - \log \mathbf{p}(t-1, y))^2, \alpha \right), \tag{4}$$

where α is a positive threshold.

The confidence loss [26] promotes higher confidence for frames near the timestamps as

$$\mathcal{L}_{\text{conf}} = \frac{1}{T} \sum_{n=1}^{N} \sum_{t=t_{n-1}}^{t_{n+1}} \delta_{t_n, t}, \tag{5}$$

where $t_0 = 1$ and $t_{N+1} = T$, and

$$\delta_{t_n,t} = \begin{cases} \max\left(0, \log \mathbf{p}(t, y_{t_n}) - \log \mathbf{p}(t{-}1, y_{t_n})\right), \text{if } t \geq t_n \\ \max\left(0, \log \mathbf{p}(t{-}1, y_{t_n}) - \log \mathbf{p}(t, y_{t_n})\right), \text{if } t < t_n \end{cases}. \tag{6}$$

3.2.4. Generating Pseudo-labels for Every Frame

Although the class scores $\mathbf{p}$ can be directly used to generate framewise pseudo-labels by assigning each frame t its highest-scoring class, $\hat{y}_t = \arg\max_{y \in \mathcal{Y}} \mathbf{p}(t, y)$, the resulting predictions may lack temporal smoothness and violate the key assumption that each interval $[t_n, t_{n+1}]$ between consecutive timestamps contains at most two action classes. Instead, Timestamp Action Decoder (TAD) associates annotated timestamps with learnable, class-aware query tokens to generate pseudo-labels for unannotated frames through the following steps.

First, as shown in Fig. 2, TAD initializes the N timestamp query tokens $\mathbf{Q} \in \mathbb{R}^{d \times N}$ by copying the corresponding refined class prototypes $\mathbf{C}'$, given by (1). Each timestamp query representing a timestamp t_n is assigned the refined class query token corresponding to the timestamp's ground-truth class y_{t_n}.

Second, for each interval $\tau_n = [t_n, t_{n+1}]$ between two consecutive timestamps, TAD computes framewise class scores for two possible action classes present in τ_n as

$$\mathbf{f}_{\tau_n} = \mathbf{z}_{\tau_n}^{\top} \mathbf{Q}_{\{n,n+1\}} \in \mathbb{R}^{|\tau_n| \times 2}, \tag{7}$$

where $\mathbf{z}_{\tau_n}$ are all frame features in the interval τ_n, and $\mathbf{Q}_{\{n,n+1\}}$ are the two timestamp query tokens corresponding to t_n and t_{n+1}.

Since there can be only one action boundary in τ_n, the boundary $t^* \in \tau_n$ is efficiently detected by maximizing the following linear function:

$$\begin{aligned} t^* &= \arg\max_{\xi \in \tau_n} b(\xi), \\ b(\xi) &= \left[\sum_{t=t_n}^{\xi} \log \mathbf{f}_{\tau_n}(t, y_{t_n}) + \sum_{t=\xi+1}^{t_{n+1}} \log \mathbf{f}_{\tau_n}(t, y_{t_{n+1}}) \right]. \end{aligned} \tag{8}$$

After estimating t^*, all frames $t \in [t_n, t^*]$ are assigned ground-truth label y_{t_n}, and frames $t \in [t^*+1, t_{n+1}]$ are assigned ground-truth label $y_{t_{n+1}}$. This enables the subsequent dense supervision using both the generated pseudo-labels and sparse timestamp annotations.

3.3. The More General SkipTag Setting

TQT can be readily extended to the more general SkipTag setting [32], where the annotated timestamps may not cover all action instances in the video. We refer to this extension as TQT-SkipTag. In this case, we identify and insert missing timestamps—referred to as pseudo-timestamps—into the annotated set, thereby enabling the subsequent action boundary detection and pseudo-label generation as described in Sec. 3.2.4.

Pseudo-timestamps are identified from among unannotated frames by analyzing their class scores $\mathbf{p} \in \mathbb{R}^{T \times |\mathcal{Y}|}$, given by (2), as follows. First, we compute the mean μ and standard deviation σ of the temporal lengths $|\tau_n| = |t_n - t_{n+1}|$ of all intervals between consecutive timestamps across the dataset. Long intervals whose temporal length exceeds $|\tau_n| > \mu + \sigma$ are selected as candidates for inserting pseudo-timestamps, while shorter intervals are assumed to be sufficiently covered by the existing annotations.

Second, within each selected long interval τ_n, a pseudo-timestamp and its class are identified as

$$(\hat{t}, \hat{y}_{\hat{t}}) = \arg\max_{t \in \tau_n,\ y \in \mathcal{Y}} \mathbf{p}(t, y), \tag{9}$$

and inserted only if both of the following two conditions are satisfied, ensuring reliable detection of a distinct new action instance within τ_n: (i) $\hat{y}_{\hat{t}}$ differs from the timestamp classes, $\hat{y}_{\hat{t}} \neq y_{t_n}$ and $\hat{y}_{\hat{t}} \neq y_{t_{n+1}}$; and (ii) all frames within a local temporal segment around $\hat{t}$ also share $\hat{y}_{\hat{t}}$ as their highest-scoring class, supporting the presence of a missed action instance in annotations within τ_n.

Once the pseudo-timestamp $\hat{t}$ is inserted, the original long interval $[t_n, t_{n+1}]$ is split into two subintervals: $[t_n, \hat{t}]$ and $[\hat{t}, t_{n+1}]$. This splitting procedure is applied recursively until the length of resulting subintervals no longer exceeds one standard deviation above the mean interval length, $(\mu + \sigma)$.

After augmenting the original timestamp set with pseudo-timestamps, we identify action boundaries, as in (8), which generates pseudo-labels for every frame. This is followed by the fully supervised training step of the frame decoder using both the pseudo and the ground-truth labels.

4. Experiments

Datasets. For evaluation, we use the following benchmark datasets: GTEA [15], 50Salads [37] and Breakfast [18]. The GTEA dataset has 28 videos showing 7 different activities and 11 action classes including background. The 50Salads dataset consists of 50 long videos of 19 different manipulative gestures for making a salad. The Breakfast dataset consists of 1,712 videos of people making breakfast with 10 cooking activities and 48 action classes. We use the same annotations as in Li et al. [26] for a fair comparison.

Metrics. Following [14], we evaluate: mean-over-frames (MoF), segment-wise edit score (Edit), and F1-scores with IoU thresholds of 0.10, 0.25 and 0.50 (F1@10,25,50).

Baselines. We compare with recent SOTA: Li *et al.*[26], GCN [17], Zhao *et al.* [44], Souri et al. [35], EM-TSS[32], UVAST [4], Du *et al.* [12] and D-TSTAS [28]. For qualitative comparisons, we follow D-TSTAS [28] and use Li *et al.* [26] as the baseline, as more recent methods—including Du *et al.* [12] and D-TSTAS [28]—do not provide publicly available code or model weights.

Implementation Details. For a fair comparison, we integrate TQT with the transformer frame decoder from ASFormer [43], as used in D-TSTAS [28]. We train the model for approximately 130 epochs, starting with class query learning followed by fully supervised training using pseudo-labels. The initial iteration comprises 60 epochs of class query learning and 50 epochs of pseudo-label training. Subsequent iterations consist of 10-20 epochs for both class query learning and pseudo-label refinement. The full set of training hyperparameters for each dataset is provided in our public GitHub[1] repository. The optimal number of iterations is 1 for GTEA and Breakfast, and 2 for 50Salads.

[1] https://github.com/tqosu/TQT

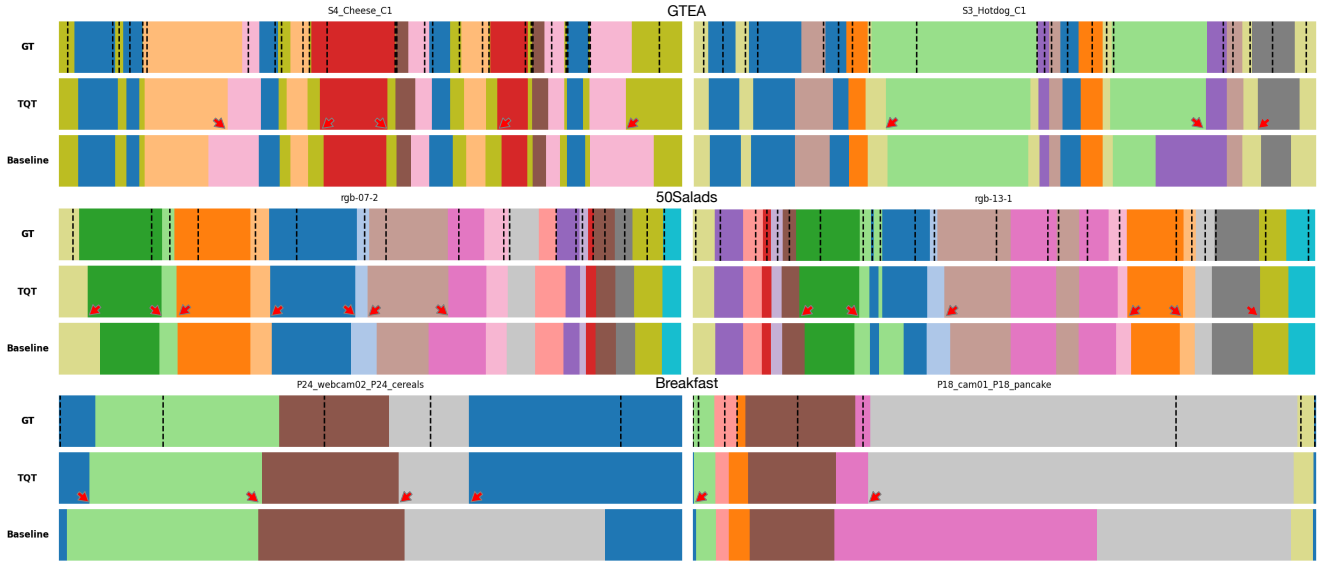

Figure 3. Qualitative comparison of label generation by TQT and the clustering-based baseline [26] on GTEA, 50Salads, and Breakfast datasets (top to bottom). Annotated timestamps are marked with dashed lines in the first row (GT). TQT consistently produces high-quality pseudo-labels, demonstrating robustness to the location of annotations. In contrast, the clustering-based baseline [26] often fails to accurately detect boundaries when they are relatively far away from the annotated timestamps, as highlighted by the red arrows.

4.1. Ablation Studies

Pseudo-Label Generation. Fig. 3 shows that TQT generates high-quality initial pseudo-labels, enabling us to bypass the prior work's multiple refinement rounds and teacher-student training, and instead proceed directly to fully supervised training with the pseudo-labels. As shown in Tab. 1, TQT achieves SOTA performance using only the initial pseudo-labels, in contrast to previous methods such as Li et al. [26] and D-TSTAS [28], which require 20 and 6 rounds of pseudo-label refinement, respectively.

Dataset	Refinement	F1@{10,25,50}			Edit	Acc
GTEA	0	89.6	88.4	77.6	88.8	75.3
	1	**91.8**	**89.8**	**78.9**	**90.8**	**77.1**
50Salads	0	87.1	84.7	76.1	81.2	83.5
	1	85.5	82.9	74.5	79.6	83.5
	2	**86.3**	**84.7**	**77.3**	**80.5**	**84.6**
Breakfast	0	77.2	71.6	56.8	76.3	71.8
	1	**78.0**	**72.8**	**58.4**	**77.1**	**72.8**

Table 1. TQT performance after pseudo-labels training, where 0 represents training with initial pseudo-labels without refinement.

# Layers	F1@{10,25,50}			Edit	Acc
0	85.4	83.2	75.1	79.1	83.1
1	**87.1**	**84.7**	**76.1**	**81.2**	83.5
2	86.3	84.3	75.4	79.9	**83.7**
3	86.8	84.7	76.6	79.8	83.5

Table 2. TQT performance on 50Salads as a function of the number of layers in the Timestamp Action Decoder.

Timestamp Action Decoder. In Tab. 2, we vary the number of layers in the Timestamp Action Decoder (TAD) from 0 to 3 and evaluate their effect on TQT performance. The results indicate that TQT without TAD (i.e., zero layers) achieves the lowest performance, while a single layer in TAD offers the best balance between accuracy and model complexity.

Frame Decoder. Timestamp-supervised methods commonly employ Temporal Convolutional Networks (TCN)

Datase	Method	F1@{10,25,50}			Edit	Acc
Frame Decoder: TCN						
GTEA	D-TSTAS [28]	84.0	79.0	60.4	79.5	67.8
	TQT	**80.9**	**75.7**	**59.0**	**77.1**	**74.9**
50salads	D-TSTAS [28]	72.0	69.0	58.6	64.6	75.6
	TQT	**80.6**	**78.1**	**68.4**	**72.4**	**80.7**
Breakfast	D-TSTAS [28]	-	-	-	-	-
	TQT	**66.2**	**60.9**	**48.0**	**68.6**	**67.7**
Frame Decoder: Transformer						
GTEA	D-TSTAS [28]	91.5	90.1	76.2	88.5	75.7
	TQT	**91.8**	**89.8**	**78.9**	**90.8**	**77.1**
50salads	D-TSTAS [28]	84.2	82.1	71.5	77.6	80.0
	TQT	**86.3**	**84.7**	**77.3**	**80.5**	**84.6**
Breakfast	D-TSTAS [28]	76.7	69.3	50.7	75.8	65.7
	TQT	**78.0**	**72.8**	**58.4**	**77.1**	**72.8**

Table 3. TQT performance when using the TCN-based or Transformer-based frame decoder: TQT uses the same backbone and frame decoder as in D-TSTAS [28].

Dataset	$\mathbf{p}$	$\mathbf{f}_{\tau_n}$	b	F1@{10,25,50}			Edit	Acc
GTEA	✓			79.7	76.6	62.1	75.8	66.7
	✓	✓		88.2	86.2	75.4	89.0	75.0
	✓	✓	✓	**89.6**	**88.4**	**77.6**	**88.8**	**75.3**
50salads	✓			77.9	74.3	65.1	70.8	79.8
	✓	✓		83.9	81.4	72.7	76.4	82.2
	✓	✓	✓	**87.1**	**84.7**	**76.1**	**81.2**	**83.5**
Breakfast	✓			72.2	66.7	52.4	72.2	70.0
	✓	✓		76.5	70.9	56.8	75.5	**72.8**
	✓	✓	✓	**77.2**	**71.6**	**56.8**	**76.3**	71.8

Table 4. TQT performance for three alternative ways of generating pseudo-labels: ($\mathbf{p}$) — direct "$\mathbf{p}$-based"; ($\mathbf{f}_{\tau_n}$) — direct "$\mathbf{f}_{\tau_n}$-based"; and (b) — "boundary-based" as described in Sec. 3.2.4.

Method	F1@{10,25,50}			Edit	Acc
Dataset: GTEA					
Li et al. (CVPR'21) [26]	72.4	66.2	43.6	71.4	59.3
EM-TSS (ECCV'22) [32]	-	-	-	-	-
D-TSTAS (CVPR'23) [28]	**90.1**	84.3	60.5	87.2	62.9
TQT	89.1	**86.0**	**67.5**	**87.9**	**70.3**
Dataset: 50Salads					
Li et al. (CVPR'21)[26]	60.8	48.5	23.1	60.6	52.3
EM-TSS(ECCV'22) [32]	62.9	50.5	25.0	63.9	52.0
D-TSTAS (CVPR'23) [28]	77.7	62.5	31.8	74.3	57.8
TQT	**79.9**	**69.9**	**40.0**	**74.8**	**63.5**
Dataset: Breakfast					
Li et al. (CVPR'21) [26]	65.5	52.2	28.0	70.4	51.2
EM-TSS (ECCV'22) [32]	57.3	46.9	25.0	61.7	48.5
D-TSTAS (CVPR'23) [28]	67.6	54.3	31.0	69.4	52.4
TQT	**74.5**	**64.0**	**38.6**	**75.1**	**64.6**

Table 5. Evaluation in the Start annotation setting. Comparison with SOTA methods on standard benchmarks under alternative timestamp supervision settings, where each action instance is annotated by a single frame at the action start, enabling evaluation of the model's ability to learn meaningful action representations from noisy, boundary features.

as frame decoders. We evaluate TQT using two alternative frame decoders: a Transformer-based decoder (our default configuration) and a TCN-based decoder with the same architecture as in D-TSTAS [28]. As shown in Tab. 3, TQT outperforms D-TSTAS with either decoder.

Alternative Ways of Pseudo-label Generation. Tab. 4 evaluates three alternative ways of generating pseudo-labels: "$\mathbf{p}$-based" — by directly assigning each frame t its highest-scoring class $\hat{y}_t = \arg\max_{y \in \mathcal{Y}} \mathbf{p}(t, y)$; "$\mathbf{f}_{\tau_n}$-based" — by considering each interval $\tau_n = [t_n, t_{n+1}]$ along the video sequence and assigning each frame $t \in \tau_n$ one of two possible classes $\hat{y}_t = \arg\max_{y \in \{y_{t_n}, y_{t_{n+1}}\}} \mathbf{f}_{\tau_n}(t, y)$; and "boundary-based" using the boundary detection as described in Sec. 3.2.4. As shown in Tab. 4, "$\mathbf{f}_{\tau_n}$-based" yields a 2.4% improvement in accuracy over "$\mathbf{p}$-based", while "boundary-based" achieves an additional performance gain of 1.3%.

Method	F1@{10,25,50}			Edit	Acc
Dataset: GTEA					
Li et al. (CVPR'21) [26]	76.5	73.0	55.6	68.7	63.8
EM-TSS (ECCV'22) [32]	-	-	-	-	-
D-TSTAS (CVPR'23) [28]	90.1	**88.8**	73.3	87.0	71.5
TQT	**90.1**	88.1	**74.8**	**89.2**	**73.2**
Dataset: 50Salads					
Li et al. (CVPR'21) [26]	74.2	71.0	59.2	68.3	74.3
EM-TSS (ECCV'22) [32]	78.4	76.0	63.5	71.1	77.1
D-TSTAS (CVPR'23) [28]	84.0	81.6	70.4	77.4	79.4
TQT	**87.2**	**85.8**	**77.2**	**80.1**	**83.8**
Dataset: Breakfast					
Li et al. (CVPR'21) [26]	70.8	63.5	45.4	71.1	65.3
EM-TSS (ECCV'22) [32]	-	-	-	-	-
D-TSTAS (CVPR'23) [28]	76.4	68.5	51.5	75.1	65.7
TQT	**77.7**	**71.8**	**56.6**	**77.6**	**72.6**

Table 6. Evaluation in the Center annotation setting. Comparison with SOTA methods on standard benchmarks under alternative timestamp supervision settings, where each action instance is annotated by a single frame at the action center, reducing the diversity of observed action features. This setup challenges the model's robustness and generalizability, particularly for out-of-distribution cases where frames are far from the center.

Dataset: Breakfast ($K = 6$)					
Method	F1@{10,25,50}			Edit	Acc
TQT	69.9	62.4	45.5	71.0	62.7
$\text{TQT}_{\mathbf{p}}$	69.3	63.4	49.8	70.6	67.3
TQT-SkipTag	**74.8**	**69.0**	**53.3**	**74.4**	**69.3**

Table 7. Ablation study under the SkipTag setting on the Breakfast dataset. TQT estimates a single boundary between adjacent annotations, while $\text{TQT}_{\mathbf{p}}$ directly assigns each frame the class with the highest predicted probability.

4.2. Alternative Timestamp Settings

We compare TQT with SOTA timestamp-supervised methods in three alternative timestamp settings: Start, Center and SkipTag.

The Start and Center settings place annotated timestamps at the beginning and midpoint of each action instance, respectively. These settings are widely regarded as more challenging than randomly sampling timestamps within action segments, since the start or center frame may not be representative of the full action class. Tab. 5 and 6 show that TQT outperforms SOTA methods in the Start and Center settings. On Breakfast, TQT improves accuracy by 12.2% and F1@25 by 9.7% over prior methods. For center-based annotations on the 50Salads dataset, TQT yields a 4.4% increase in accuracy and a 6.8% improvement in F1@50.

SkipTag Setting. The SkipTag setting introduces additional challenges by allowing annotations to miss some action instances, thereby simulating a realistic scenario when the an-

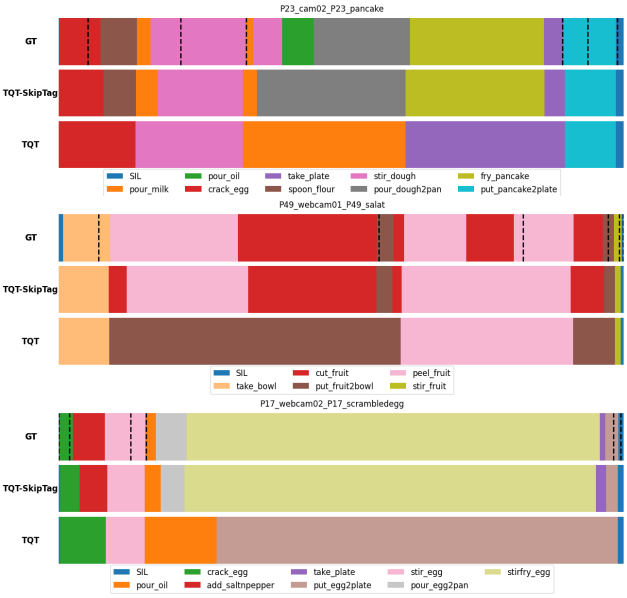

Figure 4. Visualization of pseudo-labels generated by TQT-SkipTag and TQT on Breakfast in the SkipTag setting, where each training video is annotated with only six randomly selected frames (marked with dashed lines), thereby missing to annotate several action instances. GT denotes ground truth action segments.

Method	F1@{10,25,50}			Edit	Acc
Dataset: GTEA ($K = 32$)					
Supervised Only [32]	-	64.2	43.4	65.8	63.1
Li et al. (CVPR'21) [26]	-	69.4	50.0	68.9	66.0
EM-TSS (ECCV'22) [32]	-	76.7	57.9	73.5	69.8
TQT	**89.0**	**86.8**	**73.6**	**88.7**	**73.0**
Dataset: 50Salads ($K = 19$)					
Supervised Only [32]	-	49.2	38.1	46.3	70.4
Li et al. (CVPR'21) [26]	-	54.3	40.1	54.6	70.7
EM-TSS (ECCV'22) [32]	-	68.1	54.9	64.3	74.4
TQT	**83.3**	**80.8**	**71.2**	**76.6**	**80.3**
Dataset: Breakfast ($K = 6$)					
Supervised Only [32]	-	27.0	18.8	36.4	59.8
Li et al. (CVPR'21) [26]	-	-	-	-	61.7
EM-TSS (ECCV'22) [32]	-	57.3	45.2	59.9	64.1
TQT	**74.8**	**69.0**	**53.3**	**74.4**	**69.3**

Table 8. Comparison of TQT-SkipTag with SOTA methods on standard benchmarks in the SkipTag setting [32], where the per-video annotation budget is enforced by randomly sampling K frames in each training video.

notation budget per video is limited. As shown in Tab. 7, our TQT — designed for the standard setting where all action instances are annotated with timestamps — is competitive compared to SOTA methods tailored for the SkipTag setting, but fails to always outperform them. In contrast, our extension TQT-SkipTag, described in Sec. 3.3, yields significantly improved pseudo-label generation and overall segmentation performance.

Fig. 4 illustrates pseudo-label generation by TQT and TQT-SkipTag on three sample training videos from Breakfast, highlighting the advantages of our approach. Although TQT is not designed for the SkipTag setting, its class-aware timestamp queries enable it to produce reasonable action segmentations. Its extension, TQT-SkipTag, further improves upon this by generating significantly more accurate pseudo-labels. For example, TQT-SkipTag is able to recover more than one missing action instance (see the top of example in Fig. 4).

Tab. 8 compares TQT-SkipTag with SOTA timestamp-supervised methods designed for the SkipTag setting. TQT-SkipTag achieves over 10% improvements in both F1 and Edit scores across multiple datasets despite the additional challenges posed by sparse and uneven annotations.

4.3. Comparison with the State of the Art

Tab. 9 compares TQT with SOTA methods in the standard timestamp annotation setting, where each action instance is annotated with a single, randomly selected frame. As shown in Tab. 9, TQT outperforms transformer-based SOTA methods, such as UVAST and D-TSTAS, across multiple datasets. On the Breakfast dataset, TQT achieves a 5.8% increase in accuracy and a 7.7% improvement in F1@50 compared to previous SOTA methods. Similarly, on the smaller GTEA dataset, TQT demonstrates robust performance, with gains attributed to its ability to leverage diverse training examples for generalizable feature learning. Fig. 5 compares action segmentation results on test videos between TQT and the clustering-based baseline [26]. While TQT occasionally struggles with precise action boundary localization, it consistently delivers superior segmentation quality. This improvement is due to the higher-quality pseudo-labels produced by its query-based framework (Fig. 3).

From a computational perspective, TQT remains efficient by removing heuristic per-segment prototype estimation and reducing refinement stages, while using a lightweight decoder with far fewer queries than frames. This yields negligible overhead on the backbone and keeps training and inference costs comparable to prior SOTAs.

5. Conclusion

We introduced Timestamp Query Transformer (TQT), a novel approach to timestamp-supervised action segmentation that assigns learnable, class-aware query tokens to annotated timestamps. This design enables end-to-end training and direct feed-forward inference, eliminating the need for iterative pseudo-label refinement or heuristic frame clustering used in prior work. By leveraging query–feature similarity, TQT facilitates effective knowledge transfer from sparse timestamp annotations across the entire dataset, rather than restricting analysis to local intervals between

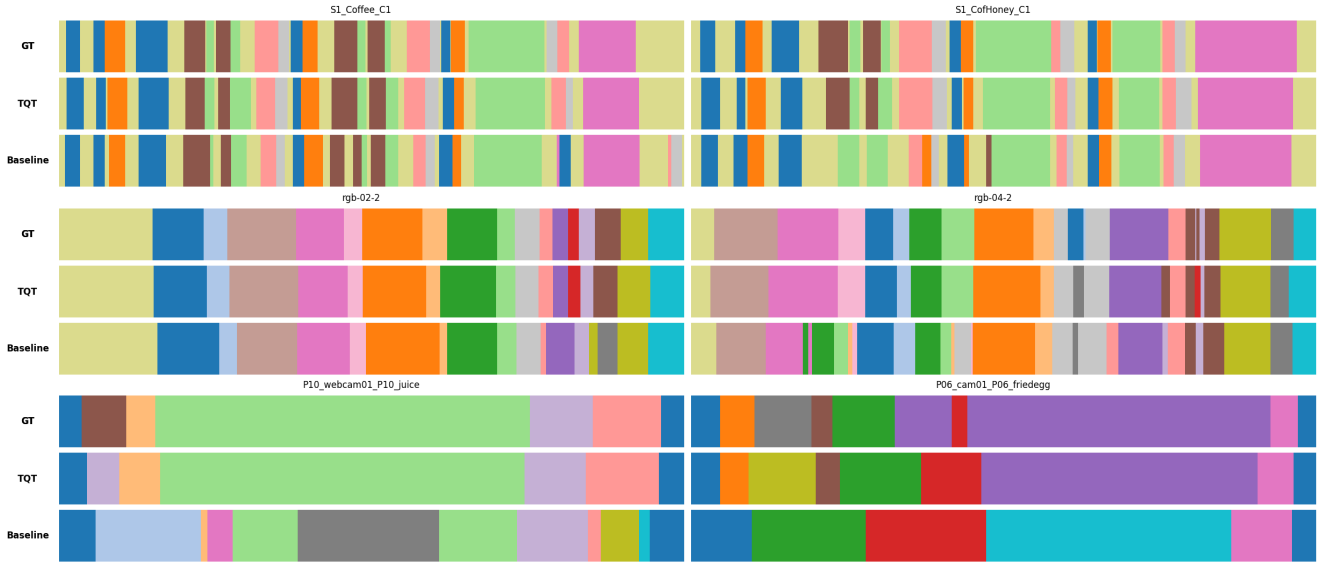

Figure 5. Qualitative comparison of final predictions on the GTEA, 50Salads, and Breakfast datasets (top to bottom) by TQT—trained using pseudo-labels generated from timestamp queries—and the clustering-based baseline [26]. Each action category is depicted with a distinct color.

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 [14]	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	66.5	67.6
	ASFormer (BMCV'21) [43]	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
	UVAST (ECCV'22) [4]	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
	DiffAct (ICCV'23) [27]	92.5	91.5	84.7	89.6	82.2	90.1	89.2	83.7	85.0	88.9	80.3	75.9	64.6	78.4	76.4
	LTContext (ICCV'23) [3]	-	-	-	-	-	89.4	87.7	82.0	83.2	87.7	77.6	72.6	60.1	77.0	74.2
	FACT (CVPR'24) [31]	93.5	92.1	84.1	91.4	86.1	-	-	-	-	-	81.4	76.5	66.2	79.7	76.2
Semi	ICC (5%) (AAAI'22) [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	35.5	56.6	65.3
	ICC (10%) (AAAI'22) [34]	83.7	81.9	66.6	76.4	73.3	63.7	64.9	49.2	56.9	68.6	64.6	59.0	42.2	61.9	68.8
Active	TSAL (ECCV'24) [38]	59.9	48.7	27.3	57.0	47.6	55.1	49.1	32.9	45.0	57.8	62.8	58.1	43.5	58.6	63.5
Timestamp	Li et al. (CVPR'21) [26]	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
	GCN (IROS'22) [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. (ICME'22) [44]	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	64.4
	Souri et al. (BMCV'22) [35]	-	-	-	-	-	77.0	74.2	62.2	69.8	79.3	71.5	64.3	47.3	70.9	62.9
	EM-TSS (ECCV'22) [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 (ECCV'22) [4]	80.7	83.7	66.0	89.3	70.5	83.0	79.6	65.5	77.2	77.0	71.3	63.3	48.3	74.1	60.7
	Du *et al.* (IJCAI'23) [12]	83.7	79.8	65.4	77.2	70.1	77.3	74.7	63.7	70.1	78.6	71.2	64.6	48.9	71.6	65.7
	D-TSTAS (CVPR'23) [28]	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
	TQT	**91.8**	89.8	**78.9**	**90.8**	**77.1**	**86.3**	**84.7**	**77.3**	**80.5**	**84.6**	**78.0**	**72.8**	**58.4**	**77.1**	**72.8**

Table 9. Comparison with state-of-the-art (SOTA) methods on standard benchmarks under different learning settings, including full supervision, semi-supervision, active learning, and timestamp supervision with a single random annotation per action instance.

consecutive timestamps as in previous methods. Extensive experiments on three benchmark datasets show that TQT not only narrows the gap with fully supervised methods but also sets new state-of-the-art results under timestamp supervision. Ablation studies of TQT's two main components—the frame decoder and the Timestamp Action Decoder (TAD)—along with evaluations of three principled strategies for framewise pseudo-label generation in training, validate our design choices. We also propose an extension, TQT-SkipTag, to address the more annotation-efficient SkipTag setting, demonstrating the generalizability of TQT across diverse annotation regimes. These findings highlight the promise of query-based representations for scalable and annotation-efficient video understanding.

Acknowledgement: This work has been supported by USDA NIFA award No.2021-67021-35344.

References

[1] Nicolas Aziere and Sinisa Todorovic. Multistage temporal convolution transformer for action segmentation. *Image and Vision Computing*, 128:104567, 2022. 1

[2] Nicolas Aziere and Sinisa Todorovic. Markov game video

augmentation for action segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13505–13514, 2023.

[3] Emad Bahrami, Gianpiero Francesca, and Juergen Gall. How much temporal long-term context is needed for action segmentation? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10351–10361, 2023. 1, 8

[4] 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. 1, 2, 4, 8

[5] Elena Bueno-Benito, Biel Tura Vecino, and Mariella Dimiccoli. Leveraging triplet loss for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4922–4930, 2023. 1

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 3

[8] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in neural information processing systems*, 34:17864–17875, 2021. 2

[9] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022. 2

[10] Guodong Ding and Angela Yao. Leveraging action affinity and continuity for semi-supervised temporal action segmentation. In *European Conference on Computer Vision*, pages 17–32. Springer, 2022. 1

[11] Guodong Ding, Fadime Sener, and Angela Yao. Temporal action segmentation: An analysis of modern techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):1011–1030, 2023. 1

[12] Dazhao Du, Enhan Li, Lingyu Si, Fanjiang Xu, and Fuchun Sun. Timestamp-supervised action segmentation in the perspective of clustering. *arXiv preprint arXiv:2212.11694*, 2022. 1, 2, 4, 8

[13] Zexing Du, Xue Wang, Guoqing Zhou, and Qing Wang. Fast and unsupervised action boundary detection for action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3323–3332, 2022. 1

[14] 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. 2, 3, 4, 8

[15] Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in egocentric activities. In *CVPR 2011*, pages 3281–3288. IEEE, 2011. 4

[16] Borui Jiang, Yang Jin, Zhentao Tan, and Yadong Mu. Video action segmentation via contextually refined temporal keypoints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13836–13845, 2023. 1

[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, 4, 8

[18] 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. 4

[19] Sateesh Kumar, Sanjay Haresh, Awais Ahmed, Andrey Konin, M Zeeshan Zia, and Quoc-Huy Tran. Unsupervised action segmentation by joint representation learning and online clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20174–20185, 2022. 1

[20] Jun Li and Sinisa Todorovic. Set-constrained viterbi for set-supervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10820–10829, 2020.

[21] Jun Li and Sinisa Todorovic. Action shuffle alternating learning for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12628–12636, 2021.

[22] Jun Li and Sinisa Todorovic. Anchor-constrained viterbi for set-supervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9806–9815, 2021.

[23] Jun Li, Peng Lei, and Sinisa Todorovic. Weakly supervised energy-based learning for action segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6243–6251, 2019. 1

[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*, pages 19880–19889, 2022. 2

[25] Shi-Jie Li, Yazan AbuFarha, Yun Liu, Ming-Ming Cheng, and Juergen Gall. Ms-tcn++: Multi-stage temporal convolutional network for action segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. 8

[26] 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

[27] Daochang Liu, Qiyue Li, Anh-Dung Dinh, Tingting Jiang, Mubarak Shah, and Chang Xu. Diffusion action segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10139–10149, 2023. 1, 8

[28] Kaiyuan Liu, Yunheng Li, Shenglan Liu, Chenwei Tan, and Zihang Shao. Reducing the label bias for timestamp supervised temporal action segmentation. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6503–6513, 2023. 1, 2, 3, 4, 5, 6, 8

[29] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *Advances in neural information processing systems*, 33:11525–11538, 2020. 2

[30] Zijia Lu and Ehsan Elhamifar. Set-supervised action learning in procedural task videos via pairwise order consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19903–19913, 2022. 1

[31] Zijia Lu and Ehsan Elhamifar. Fact: Frame-action cross-attention temporal modeling for efficient action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18175–18185, 2024. 1, 2, 8

[32] Rahul Rahaman, Dipika Singhania, Alexandre Thiery, and Angela Yao. A generalized & robust framework for timestamp supervision in temporal action segmentation. *arXiv preprint arXiv:2207.10137*, 2022. 1, 2, 4, 6, 7, 8

[33] Dipika Singhania, Rahul Rahaman, and Angela Yao. Coarse to fine multi-resolution temporal convolutional network. *arXiv preprint arXiv:2105.10859*, 2021. 2

[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*, pages 2262–2270, 2022. 1, 8

[35] Yaser Souri, Yazan Abu Farha, Emad Bahrami, Gianpiero Francesca, and Juergen Gall. Robust action segmentation from timestamp supervision. *arXiv preprint arXiv:2210.06501*, 2022. 4, 8

[36] Federico Spurio, Emad Bahrami, Gianpiero Francesca, and Juergen Gall. Hierarchical vector quantization for unsupervised action segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6996–7005, 2025. 1

[37] 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. 4

[38] Yuhao Su and Ehsan Elhamifar. Two-stage active learning for efficient temporal action segmentation. In *European Conference on Computer Vision*, pages 161–183. Springer, 2024. 1, 8

[39] Quoc-Huy Tran, Ahmed Mehmood, Muhammad Ahmed, Muhammad Naufil, Anas Zafar, Andrey Konin, and Zeeshan Zia. Permutation-aware activity segmentation via unsupervised frame-to-segment alignment. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6426–6436, 2024. 1

[40] Peiyao Wang, Yuewei Lin, Erik Blasch, Haibin Ling, et al. Efficient temporal action segmentation via boundary-aware query voting. *Advances in Neural Information Processing Systems*, 37:37765–37790, 2024. 1, 2

[41] Tieqiao Wang and Sinisa Todorovic. End-to-end action segmentation transformer. *arXiv preprint arXiv:2503.06316*, 2025. 1

[42] Ming Xu and Stephen Gould. Temporally consistent unbalanced optimal transport for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14618–14627, 2024. 1

[43] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. *arXiv preprint arXiv:2110.08568*, 2021. 2, 4, 8

[44] 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. 1, 2, 4, 8