Unified Fully and Timestamp Supervised Temporal Action Segmentation via Sequence to Sequence Translation

Nadine Behrmann[1,*], S. Alireza Golestaneh[1,*], Zico Kolter[1], Juergen Gall[2], and Mehdi Noroozi[1]

[1] Bosch Center for Artificial Intelligence
[2] University of Bonn, Germany

Abstract. This paper introduces a unified framework for video action segmentation via sequence to sequence (seq2seq) translation in a fully and timestamp supervised setup. In contrast to current state-of-the-art frame-level prediction methods, we view action segmentation as a seq2seq translation task, *i.e.,* mapping a sequence of video frames to a sequence of action segments. Our proposed method involves a series of modifications and auxiliary loss functions on the standard Transformer seq2seq translation model to cope with long input sequences opposed to short output sequences and relatively few videos. We incorporate an auxiliary supervision signal for the encoder via a frame-wise loss and propose a separate alignment decoder for an implicit duration prediction. Finally, we extend our framework to the timestamp supervised setting via our proposed constrained k-medoids algorithm to generate pseudo-segmentations. Our proposed framework performs consistently on both fully and timestamp supervised settings, outperforming or competing state-of-the-art on several datasets.

Keywords: Video Understanding, Action Segmentation, Timestamp Supervised Learning, Transformers, Auto-Regressive Learning

1 Introduction

The ability to analyze, comprehend, and segment video content at a temporal level is crucial for many computer vision, video understanding, robotics, and surveillance applications. Recent state-of-the-art methods for action segmentation mainly formalize the task as a frame-wise classification problem; that is, the objective is to assign an action label to each frame, based upon the full sequence of video frames. We illustrate this general approach in Fig. 1 (a). However, this formulation suffers several drawbacks, such as over-segmentation when trained on relatively small datasets (which typically need to consist of expensive frame-level annotations).

In this work, we propose an alternative approach to the action segmentation task. Our approach involves a transformer-based seq2seq architecture that aims to map from

* Equal contribution. JG has been supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) GA1927/4-2 (FOR 2535 Anticipating Human Behavior), MKW NRW iBehave, and the ERC Consolidator Grant FORHUE (101044724).

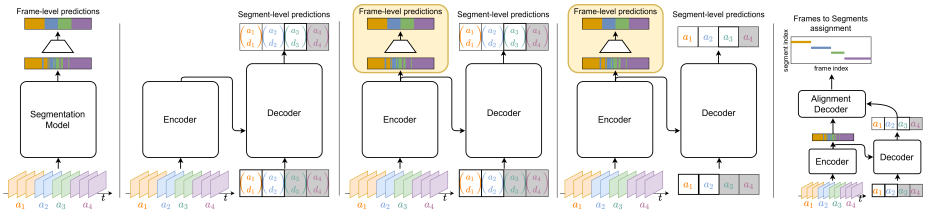

Fig. 1. Using Transformers for Action Segmentation. Instead of frame-level predictions, which are prone to over-segmentation (a), we propose a seq2seq transformer model for segment-level predictions (b). To provide more direct feedback to the encoder we apply a frame-wise loss (c); the resulting features enhance the decoder predictions. However, duration prediction still suffers, so we focus on transcript prediction (d) and use a separate alignment decoder to fuse encoder and decoder features to arrive at an implicit form of duration prediction (e).

the video frames directly to a *higher-level sequence* of action segments, *i.e.,* a sequence of action label / duration pairs that describes the full predicted segmentation.

The basic structure of our model follows traditional Transformer-based seq2seq models: the encoder branch takes as input a sequence of video frames and maps them to a set of features with the same length; the decoder branch then takes these features as input and generates a predicted sequence of high-level action segments in an auto-regressive manner. This approach, illustrated in Fig. 1 (b), is a natural fit for action segmentation because it allows the decoder to directly output sequences in the higher-level description space. The main advantage over the frame-level prediction is that it is less prone to over-segmentation.

However, this seemingly natural approach does not immediately perform well on the action segmentation task by itself. In contrast to language translation, action segmentation typically involves long input sequences of very similar frames opposed to short output sequences of action segments. This difference together with the relatively small amount of training videos, makes it challenging for the encoder and decoder to keep track of the full information flow that is necessary to predict the high-level segmentation alone. For this reason, we incorporate several modifications and additional loss terms into our system, which together make this approach compete with or improve upon the state-of-the-art.

First, to provide more immediate feedback to the encoder, we employ a frame-wise loss that linearly classifies each frame with the corresponding action label given the encoder features, Fig. 1 (c). As a result, the encoder performs frame-wise classification with high localization performance, *i.e.,* high frame-wise accuracy, but low discrimination performance, *i.e.,* over-segmentation with low Edit distance to the ground truth. Nonetheless, its features provide the decoder an informative signal to predict the sequence of actions more accurately. This immediate auxiliary supervision signal allows the decoder to learn more discriminative features for different actions. While the frame-wise loss improves the transcript prediction, the decoder still suffers from low localization performance for duration prediction. As the next step, we fuse the decoder predic-

tions with the encoder, for which we propose two solutions. First, we propose to fuse the discriminative features of the decoder with the encoder features via a cross-attention mechanism in an alignment decoder, Fig. 1 (d,e). Second, the high performance of our decoder on predicting transcripts and the high performance of our encoder on localizing actions allows us to effectively utilize the common post-processing algorithm such as FIFA [33] and Viterbi [30,21].

Finally, we further extend our proposed framework when only a weaker form of timestamp supervision is available. As mentioned before, the frame-wise prediction is vital in our Transformer model to cope with small datasets and long sequences of frames. In this case, when the frame-level annotations are not fully available, we assign a label to each frame by a constrained k-medoids clustering algorithm that takes advantage of timestamp supervision. Our simple proposed clustering method achieves a frame-wise accuracy of up to 81% on the training set, which can be effectively used to train our seq2seq model. We further show that the clustering method can also be used in combination with frame-wise prediction methods such as ASFormer [42].

We evaluate our model on three challenging action segmentation benchmarks: 50Salads [35], GTEA [12], and Breakfast [19]. While our method achieves competitive frame-wise accuracies compared to the state-of-the-art, our method substantially outperforms other approaches in predicting the action sequence of a video, which is measured by the Edit distance. By using Viterbi [30,21] or FIFA [33] as post-processing, our approach also achieves state-of-the-art results in terms of segmental F1 scores. To the best of our knowledge, this work is the first that utilizes Transformers in an autoregressive manner for action segmentation and is applicable to both the fully and timestamp supervised setup.

2 Related Work

Fully Supervised Action Segmentation. Early approaches for action segmentation are based on sliding window and non-maximum suppression [31,18]. Other traditional approaches use hidden Markov Models (HMM) for high-level temporal modeling [20,36]. [28] use a language and length model to model the probability of action sequences and convert the frame-wise probabilities into action segments using dynamic programming.

More recent approaches are based on temporal convolutions: [22] propose temporal convolutional networks (TCN) with temporal pooling to capture long-range dependencies within the video. However, such temporal pooling operations struggle to maintain fine-grained temporal information. Therefore, [1,25] use multi-stage TCNs, which maintain a high temporal resolution, with a smoothing loss and refinement modules. These methods solve the action segmentation task by predicting an action class for each frame, which is prone to over-segmentation and requires refinement modules and smoothing or expensive inference algorithms. [17] address this issue by adding a boundary regression branch to detect action boundaries, which are used during inference to refine the segmentation. [16] propose a graph-based temporal reasoning module that can be built on top of existing methods to refine predicted segmentations.

Weakly Supervised Action Segmentation. To avoid the costly frame-wise annotations, many methods have been proposed that rely on a weaker form of supervision

[5,29,24,34,9], such as transcript supervision [5]: Here, only the ordered sequence of actions in the video are given. [15] extend the connectionist temporal classification framework, originally introduced for speech recognition, to videos to efficiently evaluate all possible frame-to-action alignments. [10] propose an iterative soft boundary assignment strategy to generate frame-wise pseudo-labels from transcripts. [30] generate frame-wise pseudo-labels with the Viterbi algorithm. [24] extend this work by adding a loss that discriminates between valid and invalid segmentations. [34] use a two-branch neural network with a frame classification branch and a segment generation branch and enforce the two representations to be consistent via a mutual consistency loss. Similar to our method, their segment generation branch also predicts the transcript in an auto-regressive manner and achieves high Edit scores, validating our aspiration for segment-level predictions. While transcript supervision reduces the annotation cost significantly, the performance suffers. As an alternative, timestamp supervision [26] has been proposed, where for each action segment a single frame is annotated. The annotation cost for such timestamps is comparable to transcript annotations [26] but provides stronger supervision as it gives information about the rough location of the segments.

Transformers. Transformers [38] originally emerged in the field of natural language processing, and solely rely on the attention mechanism to capture contextual information from the entire sequence. Recently, Transformers have also seen wide adoption in vision-related tasks, *e.g.,* image classification [11], segmentation [43,40] and action classification [4,2]. Current standard Transformer-based models are unable to process very long sequences [3,37,27,8,44]. One reason for this is the self-attention operation, which scales quadratically with the sequence length. [3] showed that using sliding window attention can reduce the time and memory complexity of the Transformer while preserving the performance. Recently, ASFormer [42] leveraged multi-stage TCNs [1] and transformer-based models for action segmentation. For each dilated temporal convolutional layer of MS-TCN, an additional self-attention block with instance normalization is added. The first stage is the encoder while the later stages are the decoders, which take the concatenated features of the encoder and the features at the end of the previous stage as input. While we use a similar encoder as ASFormer [42], our decoder is very different. While ASFormer and MS-TCN perform frame-level prediction as illustrated in Fig. 1 (a), our decoder predicts the action segments in an autoregressive manner as illustrated in Fig. 1 (d,e).

3 Method

In this section, we introduce our **U**nified **V**ideo **A**ction **S**egmentation model via **T**ransformers (***UVAST***). The goal of action segmentation is to temporally segment long, untrimmed videos and classify each of the obtained segments. Current state-of-the-art methods are based on *frame-level* predictions – they assign an action label to each individual frame – which are prone to *over-segmentation*: The video is not accurately segmented into clean, continuous segments, but fragmented into many shorter pieces of alternating action classes. We challenge this view of frame-level predictions and propose a novel approach that directly predicts the segments. By focusing on *segment-level* predictions – an alternative but equivalent representation of segmentations – our method overcomes the deep-rooted over-segmentation problem of frame-level predictions.

3.1 Transformer for Auto-Regressive Segment Prediction

In this work, we view action segmentation from a sequence-to-sequence (seq2seq) perspective: mapping a sequence of video frames to a sequence of action segments, *e.g.,* as pairs of action label and segment duration. The Transformer model [38] has emerged as a particularly powerful tool for seq2seq tasks and may seem like the natural fit. The vanilla Transformer model consists of an encoder module that captures long-range dependencies within the input sequence and a decoder module that translates the input sequence to the desired output sequence in an auto-regressive manner. In contrast to language translation tasks, action segmentation faces a strong mismatch between input and output sequence lengths, *i.e.,* inputs are long and untrimmed videos with various sequence lengths, while outputs are relatively short sequences of action segments. Therefore, we incorporate several modifications to address these issues, which we will go over in more detail in the following.

Notation. Given an input sequence of T frame-wise features x_t, for frame $t \in \{1, \dots, T\}$, our goal is to temporally segment and classify the T frames. The ground-truth labels of a segmentation can be represented in two equivalent forms: 1) a sequence of frame-wise action labels $\hat{y}_t \in \mathcal{C}$ for frame t, where $\mathcal{C}$ is the set of action classes, 2) a sequence of segment-wise annotations, which consists of ground-truth segment action classes $\hat{a}_i \in \mathcal{C}$ (also known as *transcript*), and segment durations $\hat{u}_i \in \mathbb{R}_+$ for each segment $i \in \{1, \dots, N\}$.

Transformer Encoder. Our input sequence $X \in \mathbb{R}^{T \times d}$ consists of T frame-wise features x_t, where d denotes the feature dimension. We embed them using a linear layer and then feed them to the Transformer encoder, which consists of several layers and allows the model to capture long-range dependencies within the video via the self-attention mechanism. The output of the encoder, $E \in \mathbb{R}^{T \times d'}$, is a sequence of frame-wise features e_t, which will be used in the cross-attention module of the decoder. To provide direct feedback to the encoder, we apply a linear layer to obtain frame-level predictions for e_t. This enables the encoder to accurately localize the action classes within the video and provides more informative features to the decoder. In practice, we use a modified version of the encoder proposed in [42], which locally restricts the self-attention mechanism and uses dilated convolutions (see supplemental material for more details).

Transformer Decoder. Given a sequence of frame-wise features $E \in \mathbb{R}^{T \times d'}$, we use a Transformer decoder to auto-regressively predict the transcript, *i.e.,* the action labels of the segments. Starting with a *start-of-sequence (sos)* token, we feed the sequence of segments $S \in \mathbb{R}^{N \times d'}$ – embedded using learnable class tokens and positional encoding – up until segment i to the decoder. Via the cross-attention between the current sequence of segments and frame-wise features, the decoder determines the next segment $i + 1$ in the video. In principle, the decoder could predict the segment duration as well (Fig. 1 (c)), however, in practice we found that the decoder's duration prediction suffers from low localization performance, see Table 4. While it is sufficient to pick out a single or few frames in the cross-attention mechanism for predicting the correct action class of a segment, the duration prediction is more difficult since it requires to assign frames to a segment and count them. Since the number of segments is much smaller than the number of frames, the cross-attention mechanism tends to assign only a subset of the frames to the correct segment. To address this issue, we propose a separate

decoder module, which fuses the discriminative decoder features with the highly localized encoder features to obtain a more accurate duration prediction, which we describe in Section 3.3.

3.2 Training Objective

Although our ultimate goal is segment-level predictions, we provide feedback to both the encoder and decoder model to make the best use of the labels. To that end, we apply a frame-wise cross-entropy loss on the frame-level predictions of the encoder:

$$\mathcal{L}_{\text{frame}} = -\frac{1}{T}\sum_{t=1}^{T}\log(y_{t,\hat{c}}), \tag{1}$$

where $y_{t,c}$ denotes the predicted probability of label c at time t, and $\hat{c}$ denotes the ground-truth label of frame t. Analogously, we apply a segment-wise cross-entropy loss on the segment-level predictions of the decoder:

$$\mathcal{L}_{\text{segment}} = -\frac{1}{N}\sum_{i=1}^{N}\log(a_{i,\hat{c}}), \tag{2}$$

where $a_{i,c}$ denotes the predicted probability of label c at segment i, and $\hat{c}$ denotes the ground-truth label of segment i.

Regularization via Grouping. To regularize the encoder and decoder predictions, we additionally apply *group-wise* cross-entropy losses. To that end, we group the frames and segments by ground-truth labels $L = \{c \in \mathcal{C} | c \in \{\hat{a}_1, \dots, \hat{a}_n\}\}$ that occur in the video: $T_c = \{t \in \{1, \dots, T\} | \hat{y}_t = c\}$ are the indices of frames with class c, and $N_c = \{i \in \{1, \dots, N\} | \hat{a}_i = c\}$ the indices of segments with class c. We apply a cross-entropy loss to the averaged prediction of each group:

$$\mathcal{L}_{\text{g-frame}} = -\frac{1}{|L|}\sum_{c\in L}\log(\frac{1}{|T_c|}\sum_{t\in T_c} y_{t,c}) \tag{3}$$

$$\mathcal{L}_{\text{g-segment}} = -\frac{1}{|L|}\sum_{c\in L}\log(\frac{1}{|N_c|}\sum_{i\in N_c} a_{i,c}) \tag{4}$$

3.3 Cross-Attention Loss

We utilize a loss through a cross-attention mechanism between the encoder and decoder features to allow further interactions between them. Let us assume that T video frames and corresponding N actions in the encoder and decoder are represented by their features $E \in \mathbb{R}^{T\times d'}$ and $D \in \mathbb{R}^{N\times d'}$, respectively. The cross-attention loss involves obtaining a cross-attention matrix $M = \texttt{softmax}(\frac{ED^T}{\tau'\sqrt{d'}})$, where τ' is a stability temperature, and each row of M includes a probability vector that assigns each encoder feature (frame) to decoder features (actions). We then use M in the following cross-entropy loss function:

$$\mathcal{L}_{\text{CA}}(M) = -\frac{1}{T}\sum_{t}\log(M_{t,\hat{n}}), \tag{5}$$

where $\hat{n}$ is the ground-truth segment index to which frame t belongs. We use this loss in our transcript decoder (main decoder) and alignment decoder in the following.

Cross-Attention Loss for the Transcript Decoder. The cross-attention loss, when applied to the transcript decoder, provides more intermediate feedback to the decoder about the action location in the input sequence, see Fig. 4. We found this loss function especially effective on smaller datasets such as 50Salads (see Table 5). Our main objective for the encoder and the transcript decoder is:

$$\mathcal{L} = \mathcal{L}_{\text{frame}} + \mathcal{L}_{\text{segment}} + \mathcal{L}_{\text{g-frame}} + \mathcal{L}_{\text{g-segment}} + \mathcal{L}_{\text{CA}}(M), \tag{6}$$

Cross-Attention Loss for the Alignment Decoder. While the transcript decoder generates the sequence of actions in a video, it does not predict the duration of each action. Although it is possible to predict the duration as well, as illustrated in Fig. 1 (c), the transcript decoder still struggles to localize actions through direct duration prediction as shown in Table 4. One reason for this could be the high mismatch between input and output sequence length and the relatively small number of training videos. While picking up a single segment frame is sufficient to predict the action class, the duration prediction effectively requires counting the number of frames in the segment, resulting in a more challenging task. Therefore, we design an alternative alignment decoder for predicting segment durations implicitly.

A high Edit score of our decoder indicates that it has already learned discriminative features of the actions. The motivation for our alignment decoder is to align the encoder features to the highly discriminative features of the decoder, which can be further used for the duration prediction (see Fig 1 (e)). In essence, our proposed alignment decoder is a one-to-many mapping from the decoder features to the encoder features. The alignment decoder takes the encoder and decoder features $E \in \mathbb{R}^{T \times d'}$ and $D \in \mathbb{R}^{N \times d'}$ with positional encoding as input and generates the aligned features $A \in \mathbb{R}^{T \times d'}$. Since the alignment decoder aims to explore the dependencies between the encoder features and the decoder features, we employ a cross-attention mechanism in its architecture similar to the transcript decoder. To this end, we compute an assignment matrix $\overline{M} \in \mathbb{R}^{T \times N}$ via cross-attention between the alignment decoder features (A) and positional encoded features of the transcript decoder (D) by $\overline{M} = \texttt{softmax}(\frac{AD^T}{\tau})$ with a small value of τ. Note that with a small value of τ each row of $\overline{M}$ will be close to a one-hot-encoding indicating the segment index the frame is assigned to. The positional encoding for D resolves ambiguities if the same action occurs at several locations in the video.

In contrast to the decoder from the previous section, the alignment decoder is not auto-regressive since the full sequences of frame-wise and segment-wise features are already available from the previous encoder and decoder. During inference, we compute the segment durations by taking the sum over the assignments:

$$u_i = \sum_t \overline{M}_{t,i}, \tag{7}$$

where $i \in \{1, ..., n\}$ and $\overline{M}_{t,i}$ denotes whether frame t is assigned to segment i. We found that training the alignment decoder using only the loss for $\overline{M}$ (7) in a separate stage on top of the frozen encoder and decoder features results in a more robust model that suffers less from overfitting.

Algorithm 1: Constrained K-medoids to generate temporally continuous clusters.

Input: T features x_t, timestamps $[t_1, \dots, t_n]$
Init: $m_i = x_{t_i}$ `# initialize medoids`
repeat
  $D_{i,j} = \text{dist}(m_i, x_j)$ `# pairwise costs`
  $b_0 = 0; b_n = T$ `# compute boundaries`
  for $i = 1, \dots, n-1$ **do**
    $b_i = \text{argmin}_l(\sum_{j=t_i}^{l} D_{i,j} + \sum_{j=l+1}^{t_{i+1}} D_{i+1,j})$
  end
  for $i = 1, \dots, n$ **do**
    $t_i = \text{argmin}_l(\sum_{j=b_{i-1}+1}^{b_i} \text{dist}(x_l, x_j))$
    $m_i = x_{t_i}$ `# new medoids`
  end
until *until convergence*;
return $l_i = b_i - b_{i-1}$

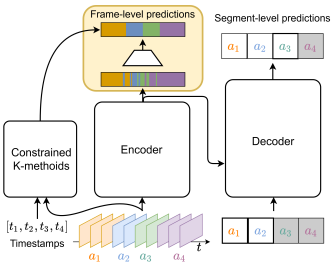

Fig. 2. Constrained K-medoids. Given frame-wise features and timestamps, k-medoids generates a pseudo-segmentation that guides the encoder during the training instead of ground truth frame-level labels in a fully supervised setup.

3.4 Timestamp Supervision

In this section, we show how our proposed framework can be extended to the timestamp supervised setting. In this setting, we are given a single annotated frame for each segment in the video, *i.e.,* frame annotations are reduced dramatically, and ground-truth segment durations are no longer available for all frames. As we extensively discussed before, our proposed framework relies on the frame-level supervisory signal on top of the encoder. However, it turns out that a noisy frame-level annotation provides a solid signal to the encoder. To obtain such frame-level annotations, we propose a constrained k-medoids algorithm that propagates the timestamp supervision to all frames.

A typical k-medoids algorithm starts with random data points as the cluster centers. It iteratively updates the cluster centers chosen from the data points and the assignments based on their similarity to the cluster center. Having access to the timestamp supervision, we can use them as initialization and cluster the input features. However, in a standard k-medoids algorithm, a temporally continuous set of clusters are not taken for granted. We call our method constrained k-medoids because we force the clusters to be temporally continuous. This can be simply achieved by modifying the assignment step of the k-medoids algorithm. Instead of assigning pseudo-labels to each frame, we find the temporal boundaries of each cluster. In the assignment step, we update the boundaries such that the accumulative distance of each cluster to the current center is minimized. Alg. 1 summarizes the steps of our clustering method. In principle, we can apply k-medoids using the frame-wise input features x_t, the encoder features e_t, or a combination of both. In practice, we found that using input features alone gives surprisingly accurate segmentations, see Table 3 or supplemental material for more analyses.

4 Experiments

4.1 Datasets

We evaluate the performance of our proposed model extensively on three challenging action segmentation datasets (50Salads [35], GTEA [12], and Breakfast [19]). We follow previous work [17,42,25,1,41,7] and perform 4-fold cross-validation on Breakfast and GTEA and 5-fold cross-validation on 50Salads.

4.2 Evaluation Metrics

For evaluation, following previous works, we report the frame-wise accuracy (Acc), segmental edit score (Edit), and the segmental F1 score at overlapping thresholds 10%, 25%, and 50%, denoted by F1@$\{10, 25, 50\}$ [22]. The overlapping threshold is determined based on the intersection over union (IoU) ratio. Although frame-wise accuracy is the most commonly used metric for action segmentation, it does not portray a thorough picture of the performance of action segmentation models. A major disadvantage of frame-wise accuracy is that long action classes have a higher impact than short action classes and dominate the results. Furthermore, over-segmentation errors have a relatively low impact on Acc, which is particularly problematic for applications such as video summarization. On the other hand, Edit and F1 scores establish more comprehensive measures of the quality of the segmentations [22]; Edit measures the quality of the predicted transcript of the segmentation, while F1 scores penalize over-segmentation and are also insensitive to the duration of the action classes. Our proposed method performs particularly well on the Edit, and F1 scores on all datasets and in fully and timestamp supervised setups, achieving state-of-the-art results in most cases.

4.3 Implementation Details and Training

We follow the standard training strategy from existing algorithms [1,25,17,42,32] and train our main network (Section 3.1) end-to-end with batch size of 1. We train our model for at most 800 epochs using Adam optimizer with learning rate 0.0005 and the loss (6). In the cross-attention loss, Eq. (7), we set $\tau = 1$ during training to ensure training stability, and $\tau = 0.0001$ during inference. As input for our model, we use the same I3D [6] features that were used in many previous works. For the encoder, we used a modified version of the encoder proposed in [42]. For the decoder, we use a standard decoder architecture [38], with two layers and single head attention. Due to a strong imbalance in the segment durations, we propose a *split-segment* approach for improved training: longer action segments are split up into several shorter ones so that segment durations are more uniformly distributed; for details and ablations, see supplemental material. During the inference, we do not use any split-segment and use the entire video.

For the alignment decoder (Section 3.3), we use a single layer, single head decoder. To train this model, we use similar hyper-parameters and optimizers while freezing the encoder-decoder model from Section 3.1 and only train the alignment decoder with our cross-attention loss. For positional encoding, we use the standard sinusoidal positional

Table 1. Fully supervised results on all three datasets. Best and second best results are shown in bold and underlined, respectively. With the assistance of Viterbi/FIFA our method outperforms state-of-the-art in terms of Edit and F1 scores on all datasets.

		Breakfast					50Salads					GTEA				
		F1@{10,25,50}			Edit	Acc	F1@{10,25,50}			Edit	Acc	F1@{10,25,50}			Edit	Acc
TDRN [23]		-	-	-	-	-	72.9	68.5	57.2	66.0	68.1	79.2	74.4	62.7	74.1	70.1
SSA-GAN [13]		-	-	-	-	-	74.9	71.7	67.0	69.8	73.3	80.6	79.1	74.2	76.0	74.4
MuCon [34]		73.2	66.1	48.4	76.3	62.8	-	-	-	-	-	-	-	-	-	-
DTGRM [39]		68.7	61.9	46.6	68.9	68.3	79.1	75.9	66.1	72.0	80.0	87.3	85.5	72.3	80.7	77.5
Gao *et al.* [14]		74.9	69.0	55.2	73.3	70.7	80.3	78.0	69.8	73.4	82.2	89.9	87.3	75.8	84.6	78.5
MS-TCN++ [25]		64.1	58.6	45.9	65.6	67.6	80.7	78.5	70.1	74.3	83.7	88.8	85.7	76.0	83.5	80.1
BCN [41]		68.7	65.5	55.0	66.2	70.4	82.3	81.3	74.0	74.3	84.4	88.5	87.1	77.3	84.4	79.8
SSTDA [7]		75.0	69.1	55.2	73.7	70.2	83.0	81.5	73.8	75.8	83.2	90.0	89.1	78.0	86.2	79.8
Singhania *et al.* [32]		70.1	66.6	56.2	68.2	**73.5**	76.6	73.0	62.5	69.2	80.1	90.5	88.5	77.1	87.3	**80.3**
ASRF [17]		74.3	68.9	56.1	72.4	67.6	84.9	83.5	77.3	79.3	84.5	89.4	87.8	79.8	83.7	77.3
ASFormer [42]		76.0	70.6	57.4	75.0	**73.5**	85.1	83.4	76.0	79.6	85.6	90.1	88.8	79.2	84.6	79.7
UVAST (Ours)	w/o duration	-	-	-	76.9	-	-	-	-	**83.9**	-	-	-	-	**92.1**	-
	+ alignment decoder	76.7	70.0	56.6	**77.2**	68.2	86.2	81.2	70.4	**83.9**	79.5	77.1	69.7	54.2	90.5	62.2
	+ Viterbi	75.9	70.0	57.2	76.5	66.0	**89.1**	**87.6**	**81.7**	**83.9**	**87.4**	**92.7**	**91.3**	**81.0**	**92.1**	80.2
	+ FIFA	**76.9**	**71.5**	**58.0**	77.1	69.7	88.9	87.0	78.5	**83.9**	84.5	82.9	79.4	64.7	90.5	69.8

encoding [38]. Furthermore, we use random dropping of the features as an augmentation method, where we randomly drop $\sim 1\%$ of the features in the sequence.

4.4 Performance Evaluation

Here, we provide the overall performance comparison of our proposed method, ***UVAST***, on three challenging action segmentation datasets with different levels of supervision. We demonstrate the effectiveness of our proposed method for both the fully supervised and timestamp supervised setup and achieve competitive results on both settings. We provide the results of our proposed model for four scenarios: Transcript prediction of our encoder-decoder architecture (referred to as "w/o duration") and three different approaches to obtain durations for the segments, namely alignment decoder from Section 3.3 ("+ alignment decoder"), Viterbi ("+ Viterbi"), and FIFA [33] ("+ FIFA"). We only report the Edit score for "w/o duration", as it does not provide segment durations. A significant advantage of our method is that a predicted transcript is readily available and can be used in these inference algorithms instead of the previous methods, which need to iterate over the training transcripts. Furthermore, we can optionally use the predicted duration of the alignment decoder to initialize the segment lengths in FIFA.

Fully Supervised Comparison. Table 1 shows the performance of our method in the fully supervised setting compared with state-of-the-art methods. At the bottom of Table 1 we provide the results of our proposed model for the four scenarios explained above. ***UVAST*** achieves significantly better Edit score on transcript prediction ("w/o duration") than all other existing methods on all three datasets, which demonstrates the effectiveness of our model to capture and summarize the actions occurring in the video. In the last three rows of Table 1, we use three different approaches to compute the duration of the segments. Combining ***UVAST*** with the alignment decoder from Section 3.3 achieves competitive results. However, it is important to note that Transformers are very data-hungry and training them on small datasets can be challenging. We observe

Table 2. Timestamp supervision results on all three datasets. UVAST, ASFormer [42], and MSTCN [1] are trained via our constrained k-medoids pseudo-labels. Best result shown in bold. ***UVAST*** outperforms SOTA on all datasets and metrics except for Acc on Breakfast. The performance in terms of Edit distance is significant, and is comparable to the fully supervised setup.

	Breakfast					50Salads					GTEA				
	F1@{10,25,50}			Edit	Acc	F1@{10,25,50}			Edit	Acc	F1@{10,25,50}			Edit	Acc
Li et al. [26]	70.5	63.6	47.4	69.9	**64.1**	73.9	70.9	60.1	66.8	75.6	78.9	73.0	55.4	72.3	66.4
MS-TCN [1]	56.1	50.0	36.8	61.7	62.5	74.4	70.4	57.7	66.8	72.8	82.8	80.3	63.5	79.5	67.7
MS-TCN [1] + Viterbi	43.3	37.2	25.6	43.5	35.9	74.0	70.0	55.5	68.2	72.8	82.6	79.7	61.6	81.0	68.1
ASFormer [42]	70.9	62.9	44.0	71.6	61.3	76.6	72.1	59.6	70.0	76.9	**87.2**	83.1	67.5	83.0	68.8
ASFormer [42] + Viterbi	71.3	63.1	44.3	71.1	60.7	76.3	72.1	59.4	68.8	**77.0**	87.1	83.1	**68.2**	83.0	69.1
+ alignment decoder	**72.0**	64.1	**48.6**	**74.3**	60.2	75.7	70.6	58.2	78.4	67.8	70.8	63.5	49.2	88.2	55.3
UVAST (Ours) + Viterbi	71.3	63.3	48.3	74.1	60.7	**83.0**	**79.6**	**65.9**	78.2	**77.0**	**87.2**	**83.7**	66.0	**89.3**	**70.5**
+ FIFA	**72.0**	**64.2**	47.6	74.1	60.3	80.2	74.9	61.6	**78.6**	72.5	80.7	75.2	57.4	88.7	66.0

that ***UVAST*** with alignment decoder outperforms other methods in terms of Edit score. While the F1 scores are comparable to the state-of-the-art on the Breakfast dataset, the small size of the GTEA dataset hinders the training of the alignment decoder.

Moreover, with frame-wise predictions and transcript prediction available, our method conveniently allows applying inference algorithms at test time, such as FIFA and Viterbi, without the need to expensively iterate over the training transcripts. Combining our method with Viterbi outperforms the existing methods on GTEA and 50Salads in terms of Edit and F1 scores, and achieves competitive results on Breakfast. We also provide the results of ***UVAST*** with FIFA, where we initialize the duration with the predicted duration. It achieves strong performance on Breakfast and 50Salads. Note that although FIFA is a fast approximation of Viterbi, it achieves better results on the Breakfast dataset. This is due to the fact that the objective function that is minimized by FIFA/Viterbi does not optimize the evaluation metrics directly, *i.e.,* the global optimum of the Viterbi objective function does not guarantee the global optimum of the evaluation metrics. This observation is consistent with the results reported in [33].

The comparison to ASFormer [42] is also interesting. While ASFormer performs like most other approaches frame-level prediction, Fig. 1 (a), ***UVAST*** predicts the action segments in an autoregressive manner, Fig. 1 (d,e). As expected, ASFormer achieves in general a better frame-wise accuracy while ***UVAST*** achieves a better Edit score. Since ASFormer uses a smoothing loss and multiple refinement stages to address over-segmentation similar to MS-TCN [1,25], it has $\sim$ 1.3M learnable parameters, whereas our proposed model has $\sim$ 1.1M parameters. Our approach with Viterbi achieves similar F1 scores on the Breakfast dataset, but higher F1 scores on the other datasets.

Overall, we find that our method achieves strong performance in terms of Edit and F1 scores, while Acc is compared to the state-of-the-art lower on Breakfast. Note that Acc is dominated by long segments and less sensitive to over-segmentation errors. Lower Acc and higher Edit/F1 scores indicate that ***UVAST*** localizes action boundaries, which are difficult to annotate precisely, less accurately. It is therefore an interesting research direction to improve the segment boundaries, *e.g.,* by using an additional refinement like ASFormer.

Timestamp Supervision Comparison. We use our proposed constrained k-medoids to generate pseudo-segmentation using the frame-wise input features and ground truth timestamps. The output consists of continuous segments, which can be identified with the transcript to yield a pseudo-segmentation. While this approach can be applied both to the input features and encoder features in principle, we find that using the input features already gives a surprisingly good performance; we report Acc and F1 scores in Table 3 averaged over all splits. Note that this is not a temporal segmentation method as it requires timestamp supervision as input. We use the resulting pseudo-segmentation as the auxiliary signal to our encoder during the training where we have access to the timestamp supervision.

In Table 2, we compare our proposed timestamp model with the recently proposed method [26] on the three action segmentation datasets. To the best of our knowledge, [26] is the first work that proposed and applied timestamp supervision for the temporal action segmentation task. Although other weakly supervised methods exist, they are based on *transcript* supervision, a weaker form of supervision; therefore, we additionally train MS-TCN [1] and ASFormer [42] with our constrained k-medoids. To get more thorough and fair comparisons, we further show their performance when combined with the Viterbi algorithm during inference.

Table 2 shows that: I) our method largely outperforms the other methods by achieving the best performance on 13 out of 15 metrics. Analogously to the fully supervised case, we observe the strong performance of our alignment decoder in terms of Edit and F1 scores on Breakfast; with FIFA and Viterbi, we outperform the method of [26] on 50Salads and GTEA. Notably, ***UVAST*** achieves significantly higher performance in terms of Edit distance, which is comparable to the fully supervised setup. II) ASFormer and MSTCN perform reasonably well in the timestamp supervision setup when trained on the pseudo-labels of our constrained k-medoids algorithm, which demonstrates one more time the effectiveness of our proposed constrained k-medoids algorithm. III) ASFormer and MSTCN do not benefit from the Viterbi algorithm in this case. This is due to the relatively lower Edit distance of these methods. Namely, Viterbi hurts MSTCN on Breakfast as it achieves significantly lower Edit distance compared to ours.

4.5 Qualitative Evaluation

We show qualitative results of two videos from the Breakfast dataset in the fully supervised and timestamp supervised setting in Fig. 3. We visualize the ground truth segmentations (first row) as well as the predicted segmentations of our encoder (second row) and decoder with alignment decoder, FIFA or Viterbi for duration prediction (last three rows). The encoder predictions demonstrate well the common problem of over-segmentation with frame-level predictions; the segment-level predictions of our decoder on the other hand yield coherent action segments.

4.6 Ablations Studies

Duration Prediction. As discussed in Section 1, the vanilla Transformer model, Fig. 1 (b), does not generalize to the action segmentation task, see Table 4. We train this model using $\mathcal{L}_{\text{segment}}$ and MSE between predicted and ground truth durations, which

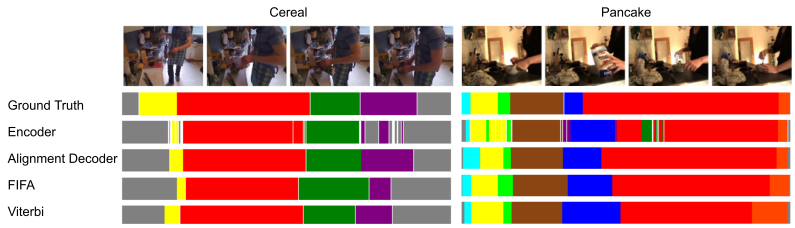

Fig. 3. Qualitative results. We show ground truth and predicted segmentation of fully supervised (left) and timestamp supervised (right) ***UVAST*** of two videos from the Breakfast dataset.

are scaled to $[0, 1]$ by dividing by the total number of frames T. Our first modification involves applying a frame-wise loss to the encoder features, which drastically improves the results. However, this explicit duration prediction still struggles to accurately localize the segments. Predicting duration implicitly via our alignment decoder instead, Fig. 1 (d)+(e), on the other hand improves the localization, increasing Acc and F1.

Table 3. Constrained K-medoids results. We evaluate the pseudo-segmentations of our constrained k-medoids algorithm, Alg. 1, given the frame-wise input features and ground truth timestamps.

	F1@{10,25,50}			Acc
Breakfast	95.5	87.5	70.0	76.9
50Salads	97.5	90.4	75.6	81.3
GTEA	99.8	97.7	83.0	75.3

Table 4. Explicit duration prediction on Breakfast split 1. We show the results of different steps described in Section 1 from explicit duration prediction via the vanilla Transformer to implicit duration prediction with our alignment decoder.

	F1@{10,25,50}			Edit	Acc
Vanilla Transformer, Fig. 1 (b)	48.1	42.3	26.7	52.9	35.0
+ Frame-wise Loss, Fig. 1 (c)	70.7	63.5	44.4	73.9	59.1
+ Alignment Decoder, Fig. 1 (d)+(e)	73.5	68.3	54.3	75.2	67.7

Impact of the Loss Terms. In Table 5 we investigate the impact of the different loss terms (Section 3.2) on split 1 of Breakfast and 50Salads. In the first row of Table 5, we evaluate the encoder when trained only using the frame-wise loss, *i.e.,* following the frame-wise prediction design as previous works. As expected, solely relying on the frame-wise loss leads to over-segmentation and poor performance. The rest of Table 5 shows the performance of our proposed model when using both encoder and decoder as explained in Sections 3.1 and 3.2, and reflect the key idea of our method to directly predict the segments. While the most basic version using the segment-wise loss (2) improves over frame-wise predictions, we observe that using both the frame-wise (1) and segment-wise (2) loss term increases the performance drastically. Moreover, we observe that adding the cross-attention loss (5) further improves the results, demonstrating its effectiveness for longer sequences with many action segments, such as 50Salads. While adding the group-wise loss terms (3) and (4) individually improves the performance moderately, the real benefit is revealed when combining them all together.

To shed more light on the contribution of our cross-attention loss we visualize its impact in Fig. 4. Given the ground truth segmentation, Fig. 4 (a), of a video,

Table 5. Loss terms. Contribution of different loss terms on Breakfast and 50Salads (split 1).

	Breakfast				50Salads			
	F1@{10,25,50}			Edit	F1@{10,25,50}			Edit
$\mathcal{L}_{\text{frame}}$	8.9	7.7	5.9	14.1	13.5	12.8	10.8	11.4
$\mathcal{L}_{\text{segment}}$	49.5	39.7	22.9	55.6	20.1	16.3	8.6	29.2
$\mathcal{L}_{\text{frame}}+\mathcal{L}_{\text{segment}}$	71.8	66.3	52.6	73.4	55.0	52.4	37.5	45.3
$\mathcal{L}_{\text{frame}}+\mathcal{L}_{\text{segment}}+\mathcal{L}_{\text{CA}}$	73.8	67.0	54.8	74.5	74.2	71.0	58.4	65.5
$\mathcal{L}_{\text{frame}}+\mathcal{L}_{\text{segment}}+\mathcal{L}_{\text{g-frame}}$	73.3	65.8	52.8	73.6	56.6	53.4	40.2	44.0
$\mathcal{L}_{\text{frame}}+\mathcal{L}_{\text{segment}}+\mathcal{L}_{\text{g-segment}}$	72.8	64.3	53.7	73.2	59.1	56.1	42.8	51.6
$\mathcal{L}_{\text{frame}}+\mathcal{L}_{\text{segment}}+\mathcal{L}_{\text{g-frame}}+\mathcal{L}_{\text{g-segment}}$	73.5	67.9	55.0	73.1	57.0	54.5	40.4	42.4
$\mathcal{L}_{\text{frame}}+\mathcal{L}_{\text{segment}}+\mathcal{L}_{\text{g-frame}}+\mathcal{L}_{\text{g-segment}}+\mathcal{L}_{\text{CA}}$	75.1	68.9	54.9	76.1	73.6	71.5	55.3	78.4

Fig. 4 (b) shows our expected target activations (output of softmax) of the decoder's cross-attention map; we hypothesize that activations should be higher in areas that belong to the corresponding segment. Fig. 4 (c) shows the output of the cross-attention when using our cross-attention loss. We observe that this loss indeed guides the cross-attention to have higher activations in the regions that belong to the related segment for an action. Fig. 4 (d) shows that lack of our cross-attention loss causes the attention map to be noisy; it's unclear which region is used for the segment classification.

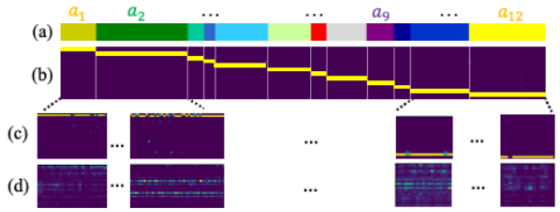

Fig. 4. Impact of the Cross-Attention Loss for the Transcript Decoder. (a) A ground truth example of a video with 13150 frames and 12 segments from the 50Salads dataset. (b) The target cross-attention map after softmax with dimension 12×13150. (c) and (d) show the zoomed-in segments of the cross-attention map of the decoder when using the cross-attention loss (top) or not using it (bottom). In (b-d) brighter color means higher values of the activations.

5 Conclusion

We presented ***UVAST***, a new unified design for fully and timestamp supervised temporal action segmentation via Transformers in a seq2seq style. While the segment-level predictions of our model effectively address the over-segmentation problem, this new design entails a new challenge: predicting the duration of segments explicitly does not work out of the box. Therefore, we proposed three different approaches to alleviate this problem, enabling our model to achieve competitive performance on all three datasets.

References

1. Abu Farha, Y., Gall, J.: MS-TCN: Multi-stage temporal convolutional network for action segmentation. In: CVPR (2019) 3, 4, 9, 11, 12
2. Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: ViViT: A video vision transformer. In: ICCV (2021) 4
3. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150 (2020) 4
4. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding? In: ICML (2021) 4
5. Bojanowski, P., Lajugie, R., Bach, F.R., Laptev, I., Ponce, J., Schmid, C., Sivic, J.: Weakly supervised action labeling in videos under ordering constraints. In: ECCV (2014) 4
6. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: CVPR (2017) 9
7. Chen, M.H., Li, B., Bao, Y., AlRegib, G., Kira, Z.: Action segmentation with joint self-supervised temporal domain adaptation. In: CVPR (2020) 9, 10
8. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q.V., Salakhutdinov, R.: Transformer-xl: Attentive language models beyond a fixed-length context. arXiv preprint arXiv:1901.02860 (2019) 4
9. Ding, L., Xu, C.: Weakly-supervised action segmentation with iterative soft boundary assignment. In: CVPR (2018) 4
10. Ding, L., Xu, C.: Weakly-supervised action segmentation with iterative soft boundary assignment. In: CVPR (2018) 4
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021) 4
12. Fathi, A., Ren, X., Rehg, J.M.: Learning to recognize objects in egocentric activities. In: CVPR (2011) 3, 9
13. Gammulle, H., Denman, S., Sridharan, S., Fookes, C.: Fine-grained action segmentation using the semi-supervised action gan. Pattern Recognition (2020) 10
14. Gao, S.H., Han, Q., Li, Z.Y., Peng, P., Wang, L., Cheng, M.M.: Global2local: Efficient structure search for video action segmentation. In: CVPR (2021) 10
15. Huang, D.A., Fei-Fei, L., Niebles, J.C.: Connectionist temporal modeling for weakly supervised action labeling. In: ECCV (2016) 4
16. Huang, Y., Sugano, Y., Sato, Y.: Improving action segmentation via graph-based temporal reasoning. In: CVPR (2020) 3
17. Ishikawa, Y., Kasai, S., Aoki, Y., Kataoka, H.: Alleviating over-segmentation errors by detecting action boundaries. In: WACV (2021) 3, 9, 10
18. Karaman, S., Seidenari, L., Bimbo, A.D.: Fast saliency based pooling of fisher encoded dense trajectories. In: ECCV Workshops (2014) 3
19. Kuehne, H., Arslan, A., Serre, T.: The language of actions: Recovering the syntax and semantics of goal-directed human activities. In: CVPR (2014) 3, 9
20. Kuehne, H., Gall, J., Serre, T.: An end-to-end generative framework for video segmentation and recognition. In: WACV (2016) 3
21. Kuehne, H., Richard, A., Gall, J.: A hybrid rnn-hmm approach for weakly supervised temporal action segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **42**(4), 765–779 (2020) 3
22. Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D.: Temporal convolutional networks for action segmentation and detection. In: CVPR (2017) 3, 9

23. Lei, P., Todorovic, S.: Temporal deformable residual networks for action segmentation in videos. In: CVPR (2018) 10
24. Li, J., Lei, P., Todorovic, S.: Weakly supervised energy-based learning for action segmentation. In: ICCV (2019) 4
25. Li, S.J., Abu Farha, Y., Liu, Y., Cheng, M.M., Gall, J.: MS-TCN++: Multi-stage temporal convolutional network for action segmentation. TPAMI (2020) 3, 9, 10, 11
26. Li, Z., Abu Farha, Y., Gall, J.: Temporal action segmentation from timestamp supervision. In: CVPR (2021) 4, 11, 12
27. Nawrot, P., Tworkowski, S., Tyrolski, M., Kaiser, Ł., Wu, Y., Szegedy, C., Michalewski, H.: Hierarchical transformers are more efficient language models. arXiv preprint arXiv:2110.13711 (2021) 4
28. Richard, A., Gall, J.: Temporal action detection using a statistical language model. In: CVPR (2016) 3
29. Richard, A., Kuehne, H., Gall, J.: Action sets: Weakly supervised action segmentation without ordering constraints. In: CVPR (2018) 4
30. Richard, A., Kuehne, H., Iqbal, A., Gall, J.: NeuralNetwork-Viterbi: A framework for weakly supervised video learning. In: CVPR (2018) 3, 4
31. Rohrbach, M., Amin, S., Andriluka, M., Schiele, B.: A database for fine grained activity detection of cooking activities. In: CVPR (2012) 3
32. Singhania, D., Rahaman, R., Yao, A.: Coarse to fine multi-resolution temporal convolutional network. arXiv preprint arXiv:2105.10859 (2021) 9, 10
33. Souri, Y., Abu Farha, Y., Despinoy, F., Francesca, G., Gall, J.: FIFA: Fast inference approximation for action segmentation. In: GCPR (2021) 3, 10, 11
34. Souri, Y., Fayyaz, M., Minciullo, L., Francesca, G., Gall, J.: Fast weakly supervised action segmentation using mutual consistency. TPAMI (2021) 4, 10
35. Stein, S., McKenna, S.J.: 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 (2013) 3, 9
36. Tang, K., Fei-Fei, L., Koller, D.: Learning latent temporal structure for complex event detection. In: CVPR (2012) 3
37. Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., Metzler, D.: Long range arena : A benchmark for efficient transformers. In: ICLR (2021) 4
38. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017) 4, 5, 9, 10
39. Wang, D., Hu, D., Li, X., Dou, D.: Temporal relational modeling with self-supervision for action segmentation. arXiv preprint arXiv:2012.07508 (2020) 10
40. Wang, H., Zhu, Y., Adam, H., Yuille, A., Chen, L.C.: MaX-DeepLab: End-to-end panoptic segmentation with mask transformers. In: CVPR (2021) 4
41. Wang, Z., Gao, Z., Wang, L., Li, Z., Wu, G.: Boundary-aware cascade networks for temporal action segmentation. In: ECCV (2020) 9, 10
42. Yi, F., Wen, H., Jiang, T.: ASFormer: Transformer for action segmentation. In: BMVC (2021) 3, 4, 5, 9, 10, 11, 12
43. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., Zhang, L.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: CVPR (2021) 4
44. Zhu, C., Ping, W., Xiao, C., Shoeybi, M., Goldstein, T., Anandkumar, A., Catanzaro, B.: Long-short transformer: Efficient transformers for language and vision. NeurIPS (2021) 4