Permutation-Aware Activity Segmentation via Unsupervised Frame-to-Segment Alignment

Quoc-Huy Tran* Ahmed Mehmood* Muhammad Ahmed Muhammad Naufil
Anas Zafar Andrey Konin M. Zeeshan Zia

Retrocausal, Inc., Redmond, WA
www.retrocausal.ai

Abstract

This paper presents an unsupervised transformer-based framework for temporal activity segmentation which leverages not only frame-level cues but also segment-level cues. This is in contrast with previous methods which often rely on frame-level information only. Our approach begins with a frame-level prediction module which estimates framewise action classes via a transformer encoder. The frame-level prediction module is trained in an unsupervised manner via temporal optimal transport. To exploit segment-level information, we utilize a segment-level prediction module and a frame-to-segment alignment module. The former includes a transformer decoder for estimating video transcripts, while the latter matches frame-level features with segment-level features, yielding permutation-aware segmentation results. Moreover, inspired by temporal optimal transport, we introduce simple-yet-effective pseudo labels for unsupervised training of the above modules. Our experiments on four public datasets, i.e., 50 Salads, YouTube Instructions, Breakfast, and Desktop Assembly show that our approach achieves comparable or better performance than previous methods in unsupervised activity segmentation.

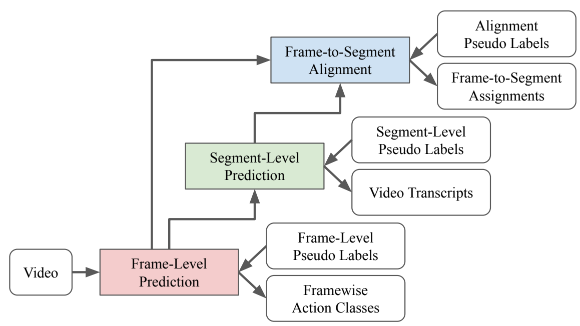


Figure 1. Prior works often use only frame-level cues via frame-level prediction modules (i.e., red) to predict framewise action classes. We adopt a segment-level prediction module and a frame-to-segment alignment module (i.e., green/blue), which exploit segment-level cues for permutation-aware results. Also, we introduce simple-yet-effective pseudo labels for unsupervised training.

1. Introduction

Temporal activity segmentation [5, 8, 11, 17, 31, 34, 37, 43, 54] aims to associate each frame in a video capturing a human activity with one of the action/sub-activity classes. Temporally segmenting human activities in videos plays an important role in several computer vision, robotics, healthcare, manufacturing, and surveillance applications. Examples include visual analytics [2, 22, 23] (i.e., compute time and motion statistics such as average cycle time from video recordings), ergonomics risk assessment [40, 41] (i.e., segment actions of interest in videos for analyzing ergonomics risks), and task guidance [7, 19, 39] (i.e., offer instructions to workers based on expert demonstration videos).

Considerable efforts have been made in designing fully-supervised methods [12,17,32,33,37] or weakly-supervised methods [5, 8, 13, 25, 29, 34, 38, 43, 44, 46, 51] for temporal activity segmentation due to their great performance. However, acquiring dense framewise labels or weak annotations such as transcripts [29] and timestamps [38] is generally hard and expensive especially for a large number of videos. Therefore, we are interested in unsupervised approaches for temporal activity segmentation, which simultaneously extract actions and segment all video frames into clusters with each cluster representing one action. Early unsupervised methods [30, 36, 49, 57, 59] separate representation learning from clustering, preventing effective feedback between them, while using offline clustering, resulting in memory inefficiency. To address that, UDE [54] and TOT [31] develop joint representation learning and online clustering ap-

* indicates joint first author.
{huy,ahmedraza,ahmed,naufil,anas,andrey,zeeshan}@retrocausal.ai.

proaches. The above methods often leverage frame-level information only (i.e., red block in Fig. 1), while not explicitly utilizing high-level information such as transcript, which is crucial for handling permutations of actions, missing actions, and repetitive actions.

In this work, we present an unsupervised activity segmentation framework which is based on transformers [56] and exploits both frame-level cues and segment-level cues. Motivated by the strong performance of supervised transformer-based architectures [5, 14] in supervised activity segmentation, our unsupervised model includes a transformer encoder and a transformer decoder. The former performs self-attention to learn dependencies within the video sequence, while the latter relies on cross-attention to learn dependencies between the video sequence and the transcript sequence, resulting in effective contextual features. In addition to the frame-level prediction module for exploiting frame-level cues, we include a segment-level prediction module and a frame-to-segment alignment module (i.e., green and blue blocks in Fig. 1) to leverage segment-level cues, yielding permutation-aware segmentation results. For unsupervised training of the above modules, we propose simple-yet-effective pseudo labels based on temporal optimal transport [31]. We demonstrate comparable or superior performance of our approach over previous unsupervised activity segmentation methods on four public datasets.

In summary, our contributions include:

- We introduce a novel combination of modules and unsupervised losses to exploit both frame-level cues and segment-level cues for permutation-aware activity segmentation.
- We propose simple-yet-effective pseudo labels based on temporal optimal transport, enabling unsupervised training of the segment-level prediction module and the frame-to-segment alignment module.
- Extensive evaluations on 50 Salads, YouTube Instructions, Breakfast, and Desktop Assembly datasets show that our approach performs on par with or better than prior methods in unsupervised activity segmentation.

2. Related Work

Fully-Supervised Activity Segmentation. Early works in fully-supervised activity segmentation often rely on sliding temporal window with non-maximum suppression [24, 47] or structured temporal modeling via hidden Markov models [28, 55], while recent methods are mostly based on temporal convolutional networks (TCNs) [12, 17, 32, 33, 37]. Lea et al. [32] develop the first TCN-based solution, which includes an encoder-decoder architecture with temporal convolutions and deconvolutions to capture long-range temporal dependencies. TricorNet [12] replaces the above decoder by a bi-directional LSTM, while TDRN [33] employs deformable temporal convolutions instead. Since these methods downsample videos to a temporal resolution, they fail to capture fine-grained details. Thus, multi-stage TCNs [17, 37] are introduced to maintain a high temporal resolution. However, due to performing framewise prediction, the above methods suffer from over-segmentation. To address that, refinement techniques, e.g., graph-based reasoning [20] and boundary detection [21], are proposed.

Weakly-Supervised Activity Segmentation. Weakly-supervised activity segmentation methods utilize different forms of weak labels, including the ordered list of actions appearing in the video, i.e., transcript supervision [8, 13, 29, 34, 44, 46], or the set of actions occurring in the video, i.e., set supervision [18, 35, 45]. Recently, timestamp supervision [25, 38, 43, 51], which requires labeling a single frame per action segment, has attracted research interests, since it has similar annotation costs as transcript supervision but it yields better results thanks to the additional approximate segment location information in timestamp labels. More recently, Behrmann et al. [5] introduce a unified fully-supervised and timestamp-supervised method, achieving competitive results. The above methods need either framewise labels for full supervision or weak labels for weak supervision, whereas our approach does not.

Unsupervised Activity Segmentation. Early attempts [3, 50] in unsupervised activity segmentation often utilize the narrations accompanied with the videos, however, these narrations are not always provided. That motivates the development of methods with only visual inputs [30, 31, 36, 49, 54, 57, 59]. Mallow [49] learns an appearance model and a temporal model of the activity in an alternating manner. CTE [30] first learns a temporal embedding and then clusters the embedded features with K-Means. To improve CTE, VTE [57] adds a visual embedding, while ASAL [36] adds an action-level embedding. SSCAP [59] first uses a video-based self-supervised model for feature extraction and then performs co-occurrence action parsing to capture the temporal structure of the activity. The aforementioned methods separate representation learning from offline clustering, preventing effective feedback between them, whereas we follow recent approaches, i.e., UDE [54] and TOT [31], to perform joint representation learning and online clustering. Furthermore, unlike UDE [54] and TOT [31], which exploit frame-level cues only, we propose modules for exploiting segment-level cues and pseudo labels for unsupervised training, yielding improved results.

Transformers in Activity Segmentation. After successes of transformers [56] in natural language processing, there has been a wide adoption of transformers in computer vision [4, 6, 9, 14]. Transformers focus on attention mechanism to extract contextual information over the entire sequence. Recently, a few methods [5, 60] have applied transformers for temporal activity segmentation. ASFormer [60] consists of encoder blocks, each of which includes a dilated temporal convolution and a self-attention layer, and decoder blocks, where cross-attention is used to gather information from encoder blocks. Due to making framewise prediction, ASFormer suffers from over-segmentation. To address that, UVAST [5] uses a transformer decoder to predict the transcript and exploit segment-level cues. In this work, we adopt the transformer encoder of ASFormer [60] and the transformer decoder of UVAST [5]. However, our overall architecture is different from them. Also, they require labels for supervised training, whereas we propose pseudo labels for unsupervised training.

3. Our Approach

We present below our main contribution, an unsupervised transformer-based framework for temporal activity segmentation. Fig. 2 shows an overview of our approach.

Notations. Let us first represent the encoder function and the decoder function as $f_{\boldsymbol{\theta}}$ and $g_{\boldsymbol{\phi}}$ respectively (with learnable parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$). Our approach takes as input a sequence of B frames, represented as $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_B]^\top$. The encoder features of $\boldsymbol{X}$ are expressed as $\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_B]^\top \in \mathbb{R}^{B\times d}$ with $\boldsymbol{e}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \in \mathbb{R}^d$ (d is the feature dimension). Next, let us denote $\boldsymbol{A} = [1, 2, \ldots, K]^\top \in \mathbb{R}^K$ as the sequence of K action classes in the activity. Our approach learns a group of K prototypes, represented as $\boldsymbol{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_K]^\top \in \mathbb{R}^{K\times d}$ with $\boldsymbol{c}_j \in \mathbb{R}^d$ corresponding to the j-th action class in $\boldsymbol{A}$. We denote $\boldsymbol{T} = [a_1, a_2, \ldots, a_N]^\top \in \mathbb{R}^N$ (with $a_j \in \boldsymbol{A}$) as the transcript which contains the sequence of actions appearing in $\boldsymbol{X}$, and $\boldsymbol{S} \in \mathbb{R}^{N\times d}$ as the transcript features. The decoder features are written as $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_N]^\top \in \mathbb{R}^{N\times d}$ with $\boldsymbol{d}_j \in \mathbb{R}^d$ corresponding to a_j in $\boldsymbol{T}$. Finally, we represent $\boldsymbol{P}_f \in \mathbb{R}^{B\times K}$, $\boldsymbol{P}_s \in \mathbb{R}^{N\times K}$, and $\boldsymbol{P}_a \in \mathbb{R}^{B\times K}$ as the *predicted* assignment probabilities (i.e., predicted "codes") at the frame-level prediction module (i.e., between frames and actions), the segment-level prediction module (i.e., between transcript positions and actions), and the frame-to-segment alignment module (i.e., between frames and actions) respectively. Similarly, $\boldsymbol{Q}_f \in \mathbb{R}^{B\times K}$, $\boldsymbol{Q}_s \in \mathbb{R}^{N\times K}$, and $\boldsymbol{Q}_a \in \mathbb{R}^{B\times K}$ denote the corresponding *pseudo-label* assignment probabilities (i.e., pseudo-label "codes") for $\boldsymbol{P}_f$, $\boldsymbol{P}_s$, and $\boldsymbol{P}_a$ respectively.

3.1. Unsupervised Frame-Level Prediction

Here we describe our frame-level prediction module. In particular, we adopt the joint representation learning and online clustering method of [31]. Unlike [31], we include modules and unsupervised losses in Secs. 3.2 and 3.3 for exploiting segment-level cues. Also, instead of the MLP encoder of [31], we utilize the transformer encoder of [5] to capture long-range dependencies via self-attention.

The input frames $\boldsymbol{X}$ are first fed to the transformer encoder $f_{\boldsymbol{\theta}}$ to yield the encoder features $\boldsymbol{E}$. The frame-level predicted codes $\boldsymbol{P}_f$ (with $\boldsymbol{P}_f^{ij}$ denoting the probability that the i-th frame in $\boldsymbol{X}$ is assigned to the j-th action in $\boldsymbol{A}$) are then computed as $\boldsymbol{P}_f = \text{softmax}\left(\frac{1}{\tau}\boldsymbol{E}\boldsymbol{C}^\top\right)$ with a temperature τ. We follow [31] to obtain the frame-level pseudo-label codes $\boldsymbol{Q}_f$ by solving the below fixed-order temporal optimal transport problem:

$$\max_{\boldsymbol{Q}\in\mathcal{Q}} \quad Tr(\boldsymbol{Q}^\top \boldsymbol{E}\boldsymbol{C}^\top) - \rho KL(\boldsymbol{Q}||\boldsymbol{M}_{\boldsymbol{A}}), \tag{1}$$

$$\mathcal{Q} = \left\{\boldsymbol{Q}: \quad \boldsymbol{Q}\boldsymbol{1}_K = \frac{1}{B}\boldsymbol{1}_B, \boldsymbol{Q}^\top\boldsymbol{1}_B = \frac{1}{K}\boldsymbol{1}_K\right\}, \tag{2}$$

where ρ is a balancing parameter, and $\boldsymbol{1}_B$ and $\boldsymbol{1}_K$ are vectors of ones with B and K dimensions respectively. The first term in Eq. 1 measures the similarity between the features $\boldsymbol{E}$ and the prototypes $\boldsymbol{C}$, while the second term denotes the Kullback-Leibler divergence between $\boldsymbol{Q}_f$ and the prior distribution $\boldsymbol{M}_{\boldsymbol{A}}$ [53]. In particular, $\boldsymbol{M}_{\boldsymbol{A}}$ assumes the *fixed order* of actions $\boldsymbol{A}$, and enforces initial frames in $\boldsymbol{X}$ to be assigned to initial actions in $\boldsymbol{A}$ and subsequent frames in $\boldsymbol{X}$ to be assigned to subsequent actions in $\boldsymbol{A}$. In Sec. 3.2, we will discuss relaxing the above fixed-order prior by introducing the transcript $\boldsymbol{T}$ and enabling permutations of actions. Eq. 2 represents the *equal partition* constraint, which imposes that each action in $\boldsymbol{A}$ is assigned the same number of frames in $\boldsymbol{X}$ to avoid a trivial solution. As mentioned in [31], the method works relatively well for activities with various action lengths since the above equal partition constraint is applied on soft assignments. The solution for the above fixed-order temporal optimal transport problem is:

$$\boldsymbol{Q}_f = \text{diag}(\boldsymbol{u}) \exp\left(\frac{\boldsymbol{E}\boldsymbol{C}^\top + \rho\log\boldsymbol{M}_{\boldsymbol{A}}}{\rho}\right) \text{diag}(\boldsymbol{v}), \tag{3}$$

where $\boldsymbol{u} \in \mathbb{R}^B$ and $\boldsymbol{v} \in \mathbb{R}^K$ are renormalization vectors [10]. Fig. 3 shows an example of $\boldsymbol{M}_{\boldsymbol{A}}$ and $\boldsymbol{Q}_f$, where the red boxes highlight the fixed order of actions $\{3, 4, 5\}$. We minimize the below cross-entropy loss with respect to $\boldsymbol{\theta}$ and $\boldsymbol{C}$ (note that we do not backpropagate through $\boldsymbol{Q}_f$):

$$L_f = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{K} \boldsymbol{Q}_f^{ij} \log \boldsymbol{P}_f^{ij}. \tag{4}$$

3.2. Unsupervised Segment-Level Prediction

The above module leverages frame-level cues and the fixed-order prior. In this section, we describe the segment-level prediction module to exploit segment-level cues and allow permutations of actions. In particular, we introduce the transcript $\boldsymbol{T}$, which indicates the sequence of actions

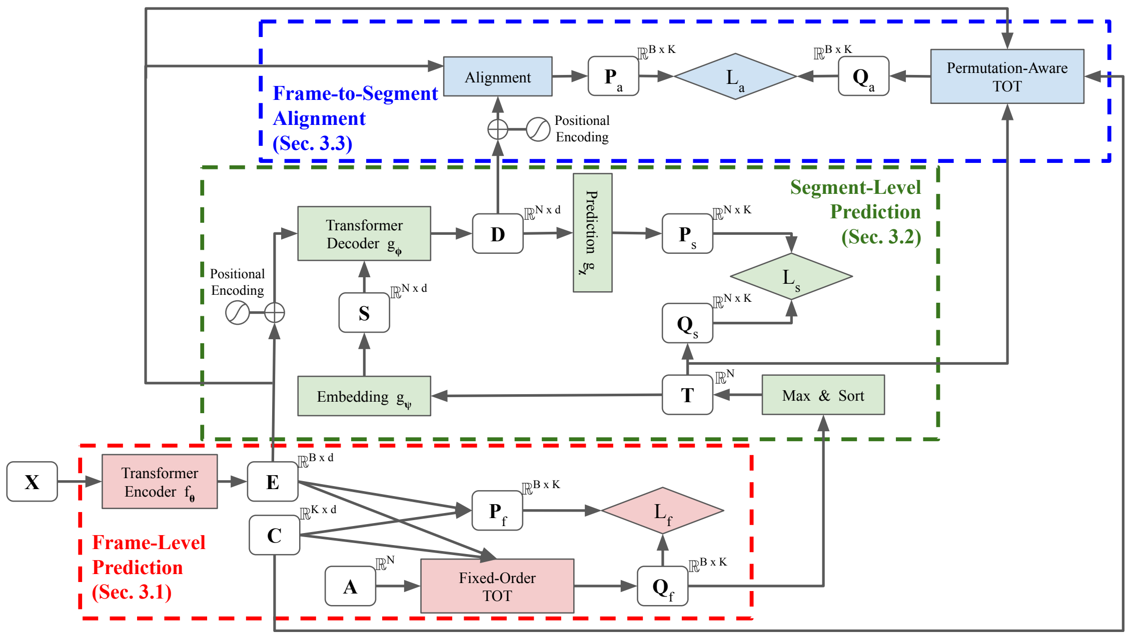
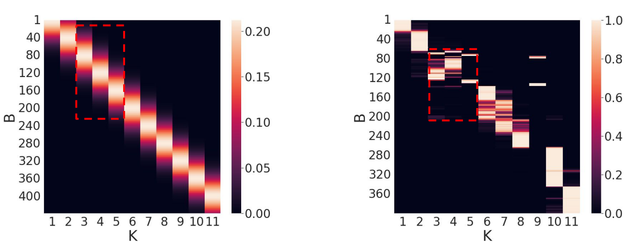
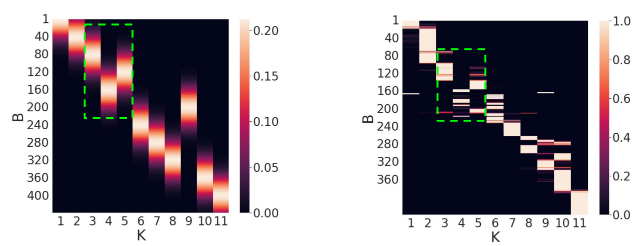


Figure 2. Our approach includes a frame-level prediction module (i.e., red) which extracts frame-level features $\boldsymbol{E}$ via a transformer encoder and uses temporal optimal transport to compute frame-level pseudo labels $\boldsymbol{Q}_f$ for unsupervised training. To exploit segment-level information, we utilize a segment-level prediction module (i.e., green), which extract segment-level features $\boldsymbol{D}$ via a transformer decoder, and a frame-to-segment alignment module (i.e., blue), which matches frame-level features $\boldsymbol{E}$ and segment-level features $\boldsymbol{D}$. In addition, we introduce segment-level pseudo labels $\boldsymbol{Q}_s$ and alignment-level pseudo labels $\boldsymbol{Q}_a$ for unsupervised training of the above modules.



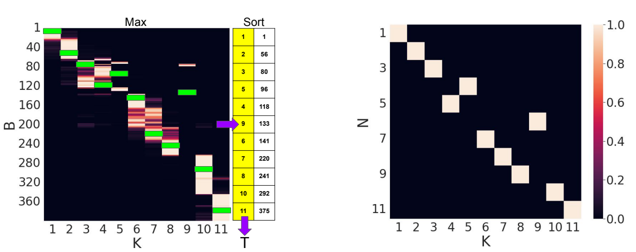
(a) $\boldsymbol{M_A}$ (b) $\boldsymbol{Q}_f$

Figure 3. (a) Fixed-order prior distribution $\boldsymbol{M_A}$. (b) Frame-level pseudo-label codes $\boldsymbol{Q}_f$.



(a) $\boldsymbol{M_T}$ (b) $\boldsymbol{Q}_a$

Figure 5. (a) Permutation-aware prior distribution $\boldsymbol{M_T}$. (b) Alignment-level pseudo-label codes $\boldsymbol{Q}_a$.



(a) $\boldsymbol{T}$ (b) $\boldsymbol{Q}_s$

Figure 4. (a) Permutation-aware transcript $\boldsymbol{T}$. (b) Segment-level pseudo-label codes $\boldsymbol{Q}_s$.

of $\boldsymbol{A}$ occurring in the input sequence $\boldsymbol{X}$. For example, let us assume $\boldsymbol{A} = [1,2,3,4,5]$, it is possible that $\boldsymbol{T} = [1,3,2,5,4]$, which is a permutation of $\boldsymbol{A}$. We will discuss later how $\boldsymbol{T}$ is estimated for unsupervised training.

Assuming the transcript $\boldsymbol{T}$ is given, we first pass it to the embedding layer $g_{\boldsymbol{\psi}}$ to obtain the transcript features $\boldsymbol{S}$, which are then fed to the transformer decoder $g_{\boldsymbol{\phi}}$. In addition, we also feed the encoder features $\boldsymbol{E}$ (after positional encoding) to the transformer decoder $g_{\boldsymbol{\phi}}$, which performs cross-attention between $\boldsymbol{E}$ and $\boldsymbol{S}$ to yield the decoder features $\boldsymbol{D}$. The segment-level predicted codes $\boldsymbol{P}_s$ (with $\boldsymbol{P}_s^{ij}$ corresponding to the probability that the i-th position in $\boldsymbol{T}$ contains the j-th action in $\boldsymbol{A}$) are computed by passing the decoder features $\boldsymbol{D}$ to the prediction layer $g_{\boldsymbol{\chi}}$. In practice, we employ the transformer decoder of [5], which computes $\boldsymbol{P}_s$ in an auto-regressive manner, i.e., a part of $\boldsymbol{T}$ up to the

i-th position is used to predict the (i+1)-th row of $\boldsymbol{P}_s$. In parallel, we convert the transcript $\boldsymbol{T}$ into the segment-level pseudo-label codes $\boldsymbol{Q}_s$. Specifically, we set $\boldsymbol{Q}_s^{ij} = 1$ if the i-th position in $\boldsymbol{T}$ contains the j-th action in $\boldsymbol{A}$, and $\boldsymbol{Q}_s^{ij} = 0$ otherwise. We minimize the following cross-entropy loss between $\boldsymbol{P}_s$ and $\boldsymbol{Q}_s$ with respect to $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, $\boldsymbol{\phi}$, and $\boldsymbol{\chi}$ (note that we do not backpropagate through $\boldsymbol{Q}_s$):

$$L_s = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K}\boldsymbol{Q}_s^{ij}\log \boldsymbol{P}_s^{ij}. \quad (5)$$

In contrast with the supervised method of [5], where framewise labels or timestamp labels are required for supervised training, we estimate the transcript $\boldsymbol{T}$ from the frame-level pseudo-label codes $\boldsymbol{Q}_f$ for unsupervised training. For each j-th action, we find the i-th frame where $\boldsymbol{Q}_f^{ij}$ has the maximum assignment probability along the j-th column, yielding an action-frame pair (j, i). Next, we sort all action-frame pairs by their frame indexes. The resulting temporally sorted list of actions is considered as our estimated transcript $\boldsymbol{T}$. Our motivation is that to predict each action correctly, the method only needs to select a single frame correctly, which is easier than obtaining the correct framewise segmentation result. Note that the above imply that N (the length of the transcript $\boldsymbol{T}$) is equal to K (the length of the action list $\boldsymbol{A}$), and our predicted transcript $\boldsymbol{T}$ shares the same set of unique actions with $\boldsymbol{A}$ despite having different orderings. Fig. 4 illustrates an example of computing $\boldsymbol{T}$ from $\boldsymbol{Q}_f$, and computing $\boldsymbol{Q}_s$ from $\boldsymbol{T}$. Similar to [31], our method tends to assign a small number of frames to the missing actions, leading to minor impacts on the overall segmentation accuracy. Handling repetitive actions is an interesting topic and remains our future work. As we will show later in Sec. 4.2, despite using the above simple heuristic for transcript estimation, our method achieves state-of-the-art results on four public datasets.

3.3. Unsupervised Frame-to-Segment Alignment

To further exploit segment-level cues and improve segmentation results, we employ the frame-to-segment alignment module of [5], which matches frame-level features with segment-level features and models permutations of actions. We pass both the encoder features $\boldsymbol{E}$ and the decoder features $\boldsymbol{D}$ (after positional encoding) to the frame-to-segment alignment module, which performs cross-attention between $\boldsymbol{E}$ and $\boldsymbol{D}$ to predicts the alignment-level predicted codes $\boldsymbol{P}_a$. Here, $\boldsymbol{P}_a^{ij}$ corresponds to the probability that the i-th frame in $\boldsymbol{X}$ is mapped to the j-th action in $\boldsymbol{A}$. We compute $\boldsymbol{P}_a = \text{softmax}\left(\frac{1}{\tau'}\boldsymbol{E}\boldsymbol{D}^\top\right)$ with a temperature τ'.

Unlike with the supervised method of [5], where framewise labels or timestamp labels are required for supervised training, we propose a modified temporal optimal transport module which is capable of handling permutations of actions to compute the alignment-level pseudo-label codes $\boldsymbol{Q}_a$ for unsupervised training. Specifically, instead of using the prior distribution $\boldsymbol{M_A}$ which enforces the fixed order of actions $\boldsymbol{A}$, we utilize the prior distribution $\boldsymbol{M_T}$ which imposes the permutation-aware transcript $\boldsymbol{T}$, yielding the permutation-aware temporal optimal transport problem:

$$\max_{\boldsymbol{Q}\in\mathcal{Q}} \; Tr(\boldsymbol{Q}^\top\boldsymbol{E}\boldsymbol{C}^\top) - \rho KL(\boldsymbol{Q}||\boldsymbol{M_T}). \quad (6)$$

The solution for the permutation-aware temporal optimal transport problem is:

$$\boldsymbol{Q}_a = \text{diag}(\boldsymbol{u})\exp\left(\frac{\boldsymbol{E}\boldsymbol{C}^\top + \rho\log\boldsymbol{M_T}}{\rho}\right)\text{diag}(\boldsymbol{v}). \quad (7)$$

Fig. 5 shows an example of $\boldsymbol{M_T}$ and $\boldsymbol{Q}_a$, where the green boxes highlight the permutations of actions $\{3, 5, 4\}$. This is in contrast with $\boldsymbol{M_A}$ and $\boldsymbol{Q}_f$ in Fig. 3, where the red boxes highlight the fixed order of actions $\{3, 4, 5\}$. As we will show later in Sec. 4.1.2, using the permutation-aware $\boldsymbol{Q}_a$ derived from $\boldsymbol{T}$ yields better performance than using the fixed-order $\boldsymbol{Q}_a$ derived from $\boldsymbol{A}$. We minimize the cross-entropy loss between $\boldsymbol{P}_a$ and $\boldsymbol{Q}_a$ with respect to $\boldsymbol{\theta}$, $\boldsymbol{\psi}$, and $\boldsymbol{\phi}$ (note that we do not backpropagate through $\boldsymbol{Q}_a$):

$$L_a = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{K}\boldsymbol{Q}_a^{ij}\log \boldsymbol{P}_a^{ij}. \quad (8)$$

Our final loss for unsupervised training is a combination of the fixed-order loss L_f (Eq. 4) and the permutation-aware losses L_s (Eq. 5) and L_a (Eq. 8):

$$L = L_f + \alpha L_s + \beta L_a, \quad (9)$$

where α and β are the balancing parameters for L_s and L_a respectively. Following [5], we set $\alpha = \beta = 1$.

4. Experiments

Implementation Details. We train our model in two stages. In the first stage, we train only the frame-level prediction module with the loss in Eq. 4 for 30 epochs, which is then used for initialization in the second stage, where we train the entire model with the loss in Eq. 9 for 70 epochs. Note that we reduce the transformer encoder and transformer decoder of [5] to two layers to avoid overfitting. We implement our approach in pyTorch [42]. We use ADAM optimization [26] with a learning rate of 10^{-3} and a weight decay of 10^{-5}. For inference, we follow [31] to compute cluster assignment probabilities for all frames and then pass them to a Viterbi decoder which smooths out the probabilities given the action order $\boldsymbol{T}$ (instead of $\boldsymbol{A}$ in [31]). More details are provided in the supplementary material.

Competing Methods. We compare our approach, namely *UFSA* (short for *U*nsupervised *F*rame-to-*S*egment

*A*lignment), against a narration-based method [3], sequential learning and clustering methods [30, 36, 49, 57, 59], and joint learning and clustering methods [31, 54].

Datasets. We evaluate our approach on four public datasets, i.e., 50 Salads [52], YouTube Instructions (YTI) [3], Breakfast [27], and Desktop Assembly [31]:

- *50 Salads* includes 50 videos capturing 25 actors making 2 types of salads. The total duration of all videos is over 4.5 hours with an average of 10k frames per video. We test on 2 granularity levels, i.e., *Eval* with 12 action classes and *Mid* with 19 action classes. Following [30], we use pre-computed features by [58].
- *YouTube Instructions (YTI)* contains 150 videos capturing 5 activities with 47 action classes in total and an average video length of about 2 minutes. These videos contain many background frames. We use pre-computed features provided by [3].
- *Breakfast* includes 70 hours of videos (30 seconds to a few minutes long per video) capturing 10 cooking activities with 48 action classes in total. We follow [49] to use pre-computed features proposed by [28].
- *Desktop Assembly* contains 2 sets of videos. *Orig* contains 76 videos of 4 actors performing desktop assembly in a fixed order. *Extra* includes all *Orig* videos and additionally 52 videos with permuted and missing steps, yielding 128 videos in total. We evaluate on both sets using pre-computed features provided by [31].

Evaluation Metrics. Following [30, 31, 49], we perform Hungarian matching between ground truth and predicted segments, which is conducted at the activity level. This is unlike the Hungarian matching performed at the video level in [1, 15, 48]. Note that video-level segmentation, e.g., ABD [15], (i.e., segmenting just a single video) is a subproblem and in general easier than activity-level segmentation, e.g., our work, (i.e., jointly segmenting and clustering frames across all videos). Due to space limits, we convert video-level segmentation results of ABD [15] to activity-level segmentation results via K-Means and evaluate them in the supplementary material. We compute Mean Over Frames (MOF), i.e., the percentage of frames with correct predictions averaged over all activities, and F1-Score, i.e., the harmonic mean of precision and recall, where only positive detections with more than 50% overlap with ground truth segments are considered. We compute F1-Score for each video and take the average over all videos.

4.1. Ablation Studies

4.1.1 Impacts of Different Model Components

We first study the effects of various network components on the 50 Salads (*Eval* granularity) and YTI datasets. The results are reported in Tab. 1. Firstly, using only the frame-level prediction module presented in Sec. 3.1 yields the lowest overall results. The frame-level prediction module exploits frame-level cues only and utilizes the fixed-order prior which does not account for permutations of actions. Next, we expand the network by adding the segment-level prediction module described in Sec. 3.2 to exploit segment-level cues. For 50 Salads, MOF is not changed much, while F1-score is improved by 3.7%. For YTI, MOF is increased by 2.2%, while F1-Score is slightly improved by 0.6%. Although the segment-level prediction module estimates the permutation-aware transcript, the framewise predictions are still suffered from over-segmentation. To address that, the frame-to-segment alignment module proposed in Sec. 3.3 is appended to the network to simultaneously leverage frame-level cues and segment-level cues and refine the framewise predictions, leading to significant performance gains. On 50 Salads, the results are boosted to 55.8% and 50.3% for MOF and F1-Score respectively, while on YTI, MOF is increased to 49.6% and F1-Score to 32.4%.

	Method	MOF	F1
Eval	Frame	43.1	34.4
	Frame+Segment	*43.2*	*38.1*
	Frame+Segment+Alignment	**55.8**	**50.3**
YTI	Frame	42.8	30.2
	Frame+Segment	*45.0*	*30.8*
	Frame+Segment+Alignment	**49.6**	**32.4**

Table 1. Impacts of different model components on 50 Salads with the Eval granularity (*Eval*) and YouTube Instructions (*YTI*). Best results are in **bold**, while second best ones are *underlined*.

4.1.2 Impacts of Different Pseudo Labels

Here, we conduct an ablation study on the 50 Salads (*Eval* granularity) and YTI datasets by using various versions of pseudo labels $\boldsymbol{Q}_s$ and $\boldsymbol{Q}_a$ computed from either the fixed order of actions $\boldsymbol{A}$ or the permutation-aware transcript $\boldsymbol{T}$. Tab. 2 presents the results. Firstly, using the fixed order of actions $\boldsymbol{A}$ for computing both $\boldsymbol{Q}_s$ and $\boldsymbol{Q}_a$ (i.e., we use $\boldsymbol{T} = \boldsymbol{A}$ in both Secs. 3.2 and 3.3) yields the lowest overall numbers on both datasets, i.e., on 50 Salads, 46.1% and 45.2% for MOF and F1-Score respectively, and on YTI, 44.3% and 29.4% for MOF and F1-Score respectively. Next, we experiment with using the permutation-aware transcript $\boldsymbol{T}$ for computing either $\boldsymbol{Q}_s$ or $\boldsymbol{Q}_a$, resulting in performance gains, e.g., for the former ($\boldsymbol{T}$ for $\boldsymbol{Q}_s$, $\boldsymbol{A}$ for $\boldsymbol{Q}_a$), we achieve 50.8% for MOF and 46.9% for F1-Score on 50 Salads, while for the latter ($\boldsymbol{A}$ for $\boldsymbol{Q}_s$, $\boldsymbol{T}$ for $\boldsymbol{Q}_a$), we obtain 54.0% for MOF and 48.7% for F1-Score on 50 Salads. Finally, we employ the permutation-aware transcript $\boldsymbol{T}$ for computing both $\boldsymbol{Q}_s$ and $\boldsymbol{Q}_a$, leading to the best performance on both datasets, i.e., 55.8% for MOF and 50.3%

	$\boldsymbol{Q}_s$	$\boldsymbol{Q}_a$	MOF	F1
Eval	$\boldsymbol{A}$	$\boldsymbol{A}$	46.1	45.2
	$\boldsymbol{T}$	$\boldsymbol{A}$	50.8	46.9
	$\boldsymbol{A}$	$\boldsymbol{T}$	*54.0*	*48.7*
	$\boldsymbol{T}$	$\boldsymbol{T}$	**55.8**	**50.3**
YTI	$\boldsymbol{A}$	$\boldsymbol{A}$	44.3	29.4
	$\boldsymbol{T}$	$\boldsymbol{A}$	45.7	29.7
	$\boldsymbol{A}$	$\boldsymbol{T}$	*46.5*	*29.8*
	$\boldsymbol{T}$	$\boldsymbol{T}$	**49.6**	**32.4**

Table 2. Impacts of different pseudo labels on 50 Salads with the Eval granularity (*Eval*) and YouTube Instructions (*YTI*). Best results are in **bold**, while second best ones are *underlined*.

for F1-Score on 50 Salads, and 49.6% for MOF and 32.4% for F1-Score on YTI. The above results confirm the benefits of using the permutation-aware transcript $\boldsymbol{T}$ for computing both pseudo labels $\boldsymbol{Q}_s$ and $\boldsymbol{Q}_a$.

4.2. Comparisons with the State-of-the-Art

4.2.1 Results on 50 Salads

We now compare the performance of our approach with state-of-the-art unsupervised activity segmentation methods on the 50 Salads dataset for both granularities, i.e., *Eval* and *Mid*. Tab. 3 illustrates the results. It is evident from Tab. 3 that our approach obtains the best MOF and F1-Score numbers on both granularities, outperforming all competing methods. In particular, UFSA outperforms TOT [31] by 8.4% and 4.9% on MOF on the *Eval* and *Mid* granularities respectively, and UDE [54] by 15.9% on F1-Score on the *Eval* granularity. Although TOT [31] and UDE [54] conduct joint representation learning and online clustering as our approach, they only exploit frame-level cues, whereas UFSA leverages segment-level cues as well. Moreover, our approach achieves better results than SSCAP [59], which uses recent self-supervised learning features [16], and ASAL [36], which exploits segment-level cues via action shuffling, e.g., on the *Eval* granularity, UFSA achieves 55.8% MOF, whereas SSCAP [59] and ASAL [36] obtain 41.4% MOF and 39.2% MOF respectively. The substantial improvements of UFSA over previous methods demonstrate the effectiveness of our approach.

4.2.2 Results on YouTube Instructions

Tab. 4 presents the quantitative results of our approach along with previous unsupervised activity segmentation methods on the YTI dataset. We follow the protocol of prior works and report the accuracy excluding the background frames. It is clear from Tab. 4 that our approach achieves the best MOF, outperforming all previous methods, and the second best F1-Score, slightly worse than TOT+TCL [31] (note that our approach currently relies on TOT only, and can further include TCL for potential improvements). Specifically, UFSA has an improvement of 9.0% MOF and 2.4% F1-Score over TOT [31], and an improvement of 5.8% MOF and 2.8% F1-Score over UDE [54]. In addition, our approach obtains a noticeable gain of 4.7% MOF and a slight gain of 0.3% F1-Score over ASAL [36]. Fig. 6 plots the qualitative results of UFSA, TOT [31], and CTE [30] on a YTI video. Our approach demonstrates significant advantages over CTE [30] and TOT [31] in terms of capturing the temporal order of actions and aligning them closely with the ground truth. Due to space constraints, please refer to the supplementary material for more qualitative examples, especially with permuted, missing, and repetitive actions.

Method	Eval		Mid	
	MOF	F1	MOF	F1
CTE [30]	35.5	36.3	30.2	25.6
VTE [57]	30.6	-	24.2	-
ASAL [36]	39.2	-	*34.4*	-
UDE [54]	42.2	34.4	-	-
SSCAP [59]	41.4	30.3	-	-
TOT [31]	*47.4*	42.8	31.8	22.5
TOT+TCL [31]	44.5	*48.2*	34.3	*28.9*
Ours (UFSA)	**55.8**	**50.3**	**36.7**	**30.4**

Table 3. Results on 50 Salads. *Eval* denotes the Eval granularity, while *Mid* denotes the Mid granularity. Best results are in **bold**, while second best ones are *underlined*.

Method	MOF	F1
Frank-Wolfe [3]	-	24.4
Mallow [49]	27.8	27.0
CTE [30]	39.0	28.3
VTE [57]	-	29.9
ASAL [36]	44.9	32.1
UDE [54]	43.8	29.6
TOT [31]	40.6	30.0
TOT+TCL [31]	*45.3*	**32.9**
Ours (UFSA)	**49.6**	*32.4*

Table 4. Results on YouTube Instructions. Best results are in **bold**, while second best ones are *underlined*.

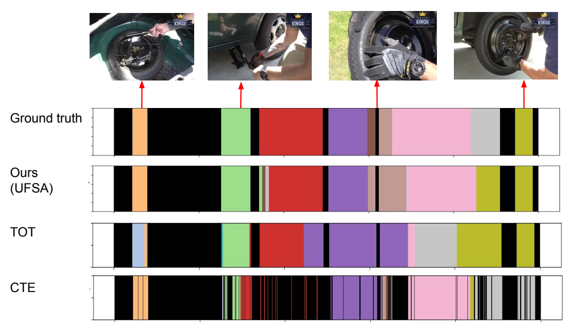


Figure 6. Segmentation results on a YouTube Instructions video (*changing_tire_0005*). Black color indicates background frames.

Method	MOF	F1
Mallow [49]	34.6	-
CTE [30]	41.8	26.4
VTE [57]	48.1	-
ASAL [36]	**52.5**	37.9
UDE [54]	47.4	31.9
SSCAP [59]	51.1	**39.2**
TOT [31]	47.5	31.0
TOT+TCL [31]	39.0	30.3
Ours (UFSA)	<u>*52.1*</u>	<u>*38.0*</u>

Table 5. Results on Breakfast. Best results are in **bold**, while second best ones are <u>*underlined*</u>.

	Method	MOF	F1
Orig	CTE [30]	47.6	44.9
	TOT [31]	56.3	51.7
	TOT+TCL [31]	<u>*58.1*</u>	<u>*53.4*</u>
	Ours (UFSA)	**65.4**	**63.0**
Extra	CTE [30]	40.8	35.6
	TOT [31]	51.0	40.4
	TOT+TCL [31]	<u>*57.9*</u>	<u>*54.0*</u>
	Ours (UFSA)	**58.6**	**55.9**

Table 6. Results on Desktop Assembly. *Orig* includes original fixed-order videos only, while *Extra* further includes additional permuted-step and missing-step videos. Best results are in **bold**, while second best ones are <u>*underlined*</u>.

4.2.3 Results on Breakfast

Tab. 5 includes the performance of different methods on the Breakfast dataset. From Tab. 5, our results are on par with ASAL [36], which leverages segment-level information via action shuffling, and SSCAP [59], which employs more sophisticated self-supervised features [16]. Particularly, ASAL [36] and SSCAP [59] yield the best MOF number (i.e., 52.5%) and the best F1-Score number (i.e., 39.2%) respectively, while UFSA achieves the second best results for both metrics (i.e., 52.1% and 38.0%). In addition, our approach outperforms a number of competing methods, namely Mallow [49], CTE [30], VTE [57], UDE [54], and TOT [31], which exploit frame-level cues only.

4.2.4 Results on Desktop Assembly

We test the performance of our approach on the Desktop Assembly dataset for both *Orig* and *Extra* sets. The results are reported in Tab. 6, which shows superior performance of our approach over CTE [30], TOT [31], and TOT+TCL [31]. For example, UFSA achieves an improvement of 14.9% MOF and 20.5% F1-Score over TOT [31] on the *Orig* set, and a gain of 7.6% MOF and 15.5% F1-Score over TOT [31] on the *Extra* set. Results on the *Orig* set indicate the effectiveness of our approach in preserving the fixed order of actions, while results on the *Extra* set show the ability of our method in handling permuted actions.

	Method	MOF	F1
Eval	CTE [30]	28.6	26.4
	TOT [31]	39.8	37.0
	TOT+TCL [31]	<u>*42.8*</u>	**44.9**
	Ours (UFSA)	**47.6**	<u>*41.8*</u>
YTI	CTE [30]	38.4	25.5
	TOT [31]	40.4	<u>*28.0*</u>
	TOT+TCL [31]	<u>*40.6*</u>	26.7
	Ours (UFSA)	**46.8**	**28.2**
Breakfast	CTE [30]	39.8	25.5
	TOT [31]	<u>*40.6*</u>	<u>*27.6*</u>
	TOT+TCL [31]	37.4	23.2
	Ours (UFSA)	**44.0**	**36.7**
Orig	CTE [30]	35.6	31.8
	TOT [31]	<u>*55.3*</u>	<u>*50.2*</u>
	TOT+TCL [31]	49.2	44.6
	Ours (UFSA)	**63.9**	**63.7**
Extra	CTE [30]	35.7	30.4
	TOT [31]	43.6	35.0
	TOT+TCL [31]	<u>*45.9*</u>	<u>*40.0*</u>
	Ours (UFSA)	**57.9**	**54.0**

Table 7. Generalization results. Best results are in **bold**, while second best ones are underlined

.

4.2.5 Generalization Results

We follow [31] to evaluate the generalization ability of our approach. We divide the datasets, i.e., 50 Salads (*Eval*), YTI, Breakfast, Desktop Assembly (*Orig*, *Extra*) into 80% for training and 20% for testing. For instance, for 50 Salads with 50 videos, 40 videos are used for training and 10 for testing. Tab. 7 shows the results. UFSA continues to outperform CTE [30], TOT [31], and TOT+TCL [31] in this experiment setting. Note the results of CTE [30], TOT [31], and TOT+TCL [31] in Tab. 7 differ from those reported in [31] since different training/testing splits are used (we could not acquire the splits from the authors of [31]). Our splits are available at `https://tinyurl.com/57ya6653`.

5. Conclusion

We propose a novel combination of modules and unsupervised losses to exploit both frame-level cues and segment-level cues for permutation-aware activity segmentation. Our approach includes a frame-level prediction module which uses a transformer encoder for obtaining frame-wise action classes and is trained in unsupervised manner via temporal optimal transport. To leverage segment-level cues, we utilize a segment-level prediction model based on a transformer decoder for predicting video transcripts and a frame-to-segment alignment module for corresponding frame-level features with segment-level features, resulting in permutation-aware segmentation results. For unsupervised training of the above modules, we introduce simple-yet-effective pseudo labels. We show comparable or superior results over prior methods on four public datasets.

References

[1] Sathyanarayanan N Aakur and Sudeep Sarkar. A perceptual prediction framework for self supervised event segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1197–1206, 2019. 6

[2] Oguz Akkas, Cheng Hsien Lee, Yu Hen Hu, Carisa Harris Adamson, David Rempel, and Robert G Radwin. Measuring exertion time, duty cycle and hand activity level for industrial tasks using computer vision. *Ergonomics*, 60(12):1730–1738, 2017. 1

[3] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4575–4583, 2016. 2, 6, 7

[4] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 2

[5] Nadine Behrmann, S Alireza Golestaneh, Zico Kolter, Jürgen 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, 3, 4, 5

[6] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, volume 2, page 4, 2021. 2

[7] Jonas Blattgerste, Benjamin Strenge, Patrick Renner, Thies Pfeiffer, and Kai Essig. Comparing conventional and augmented reality instructions for manual assembly tasks. In *Proceedings of the 10th international conference on pervasive technologies related to assistive environments*, pages 75–82, 2017. 1

[8] Chien-Yi Chang, De-An Huang, Yanan Sui, Li Fei-Fei, and Juan Carlos Niebles. D3tw: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3546–3555, 2019. 1, 2

[9] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 357–366, 2021. 2

[10] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013. 3

[11] Guodong Ding, Fadime Sener, and Angela Yao. Temporal action segmentation: An analysis of modern technique. *arXiv preprint*, 2022. 1

[12] Li Ding and Chenliang Xu. Tricornet: A hybrid temporal convolutional and recurrent network for video action segmentation. *arXiv preprint*, 2017. 1, 2

[13] Li Ding and Chenliang Xu. Weakly-supervised action segmentation with iterative soft boundary assignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6508–6516, 2018. 1, 2

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*. 2

[15] 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. 6

[16] Dave Epstein, Boyuan Chen, and Carl Vondrick. Oops! predicting unintentional action in video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 919–929, 2020. 7, 8

[17] Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3575–3584, 2019. 1, 2

[18] Mohsen Fayyaz and Jurgen Gall. Sct: Set constrained temporal transformer for set supervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 501–510, 2020. 2

[19] Markus Funk, Thomas Kosch, Scott W Greenwald, and Albrecht Schmidt. A benchmark for interactive augmented reality instructions for assembly tasks. In *Proceedings of the 14th international conference on mobile and ubiquitous multimedia*, pages 253–257, 2015. 1

[20] Yifei Huang, Yusuke Sugano, and Yoichi Sato. Improving action segmentation via graph-based temporal reasoning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14024–14034, 2020. 2

[21] Yuchi Ishikawa, Seito Kasai, Yoshimitsu Aoki, and Hirokatsu Kataoka. Alleviating over-segmentation errors by detecting action boundaries. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2322–2331, 2021. 2

[22] Jirasak Ji, Warut Pannakkong, and Jirachai Buddhakulsomsiri. A computer vision-based model for automatic motion time study. *CMC-COMPUTERS MATERIALS & CONTINUA*, 73(2):3557–3574, 2022. 1

[23] Jirasak Ji, Warut Pannakkong, Pham Duc Tai, Chawalit Jeenanunta, and Jirachai Buddhakulsomsiri. Motion time study with convolutional neural network. In *Integrated Uncertainty in Knowledge Modelling and Decision Making: 8th International Symposium, IUKM 2020, Phuket, Thailand, November 11–13, 2020, Proceedings 8*, pages 249–258. Springer, 2020. 1

[24] Svebor Karaman, Lorenzo Seidenari, and Alberto Del Bimbo. Fast saliency based pooling of fisher encoded dense trajectories. In *ECCV THUMOS Workshop*, volume 1, page 5, 2014. 2

[25] Hamza Khan, Sanjay Haresh, Awais Ahmed, Shakeeb Siddiqui, Andrey Konin, M Zeeshan Zia, and Quoc-Huy Tran. Timestamp-supervised action segmentation with graph con-

volutional networks. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10619–10626. IEEE, 2022. 1, 2

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint*, 2014. 5

[27] 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. 6

[28] Hilde Kuehne, Juergen Gall, and Thomas Serre. An end-to-end generative framework for video segmentation and recognition. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8. IEEE, 2016. 2, 6

[29] Hilde Kuehne, Alexander Richard, and Juergen Gall. Weakly supervised learning of actions from transcripts. *Computer Vision and Image Understanding*, 163:78–89, 2017. 1, 2

[30] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12066–12074, 2019. 1, 2, 6, 7, 8

[31] 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, 2, 3, 5, 6, 7, 8

[32] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017. 1, 2

[33] Peng Lei and Sinisa Todorovic. Temporal deformable residual networks for action segmentation in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6742–6751, 2018. 1, 2

[34] 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, 2

[35] 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. 2

[36] 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. 1, 2, 6, 7, 8

[37] 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*, 2020. 1, 2

[38] 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

[39] Lai Xing Ng, Jamie Ng, Keith TW Tang, Liyuan Li, Mark Rice, and Marcus Wan. Using visual intelligence to automate maintenance task guidance and monitoring on a head-mounted display. In *Proceedings of the 2019 5th International Conference on Robotics and Artificial Intelligence*, pages 70–75, 2019. 1

[40] Behnoosh Parsa and Ashis G Banerjee. A multi-task learning approach for human activity segmentation and ergonomics risk assessment. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2352–2362, 2021. 1

[41] Behnoosh Parsa, Behzad Dariush, et al. Spatio-temporal pyramid graph convolutions for human action recognition and postural assessment. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1080–1090, 2020. 1

[42] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[43] Rahul Rahaman, Dipika Singhania, Alexandre Thiery, and Angela Yao. A generalized and robust framework for timestamp supervision in temporal action segmentation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, pages 279–296. Springer, 2022. 1, 2

[44] Alexander Richard, Hilde Kuehne, and Juergen Gall. Weakly supervised action learning with rnn based fine-to-coarse modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 754–763, 2017. 1, 2

[45] Alexander Richard, Hilde Kuehne, and Juergen Gall. Action sets: Weakly supervised action segmentation without ordering constraints. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5987–5996, 2018. 2

[46] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7386–7395, 2018. 1, 2

[47] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *2012 IEEE conference on computer vision and pattern recognition*, pages 1194–1201. IEEE, 2012. 2

[48] Saquib Sarfraz, Naila Murray, Vivek Sharma, Ali Diba, Luc Van Gool, and Rainer Stiefelhagen. Temporally-weighted hierarchical clustering for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11225–11234, 2021. 6

[49] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8368–8376, 2018. 1, 2, 6, 7, 8

[50] Ozan Sener, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collec-

tions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4480–4488, 2015. 2

[51] Yaser Souri, Yazan Abu Farha, Emad Bahrami, Gianpiero Francesca, and Juergen Gall. Robust action segmentation from timestamp supervision. *arXiv preprint*, 2022. 1, 2

[52] 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. 6

[53] Bing Su and Gang Hua. Order-preserving wasserstein distance for sequence matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1049–1057, 2017. 3

[54] Sirnam Swetha, Hilde Kuehne, Yogesh S Rawat, and Mubarak Shah. Unsupervised discriminative embedding for sub-action learning in complex activities. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 2588–2592. IEEE, 2021. 1, 2, 6, 7, 8

[55] Kevin Tang, Li Fei-Fei, and Daphne Koller. Learning latent temporal structure for complex event detection. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1250–1257. IEEE, 2012. 2

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[57] Rosaura G VidalMata, Walter J Scheirer, Anna Kukleva, David Cox, and Hilde Kuehne. Joint visual-temporal embedding for unsupervised learning of actions in untrimmed sequences. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1238–1247, 2021. 1, 2, 6, 7, 8

[58] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013. 6

[59] Zhe Wang, Hao Chen, Xinyu Li, Chunhui Liu, Yuanjun Xiong, Joseph Tighe, and Charless Fowlkes. Sscap: Self-supervised co-occurrence action parsing for unsupervised temporal action segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1819–1828, 2022. 1, 2, 6, 7, 8

[60] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. *arXiv preprint*, 2021. 2, 3