PACE: Predictive and Contrastive Embedding for Unsupervised Action Segmentation

Jiahao Wang[1] , **Jie Qin**[2] , **Yunhong Wang**[1] and **Annan Li**[1*]
[1]State Key Laboratory of Virtual Reality Technology and System, Beihang University
[2]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
{jhwang, yhwang, liannan}@buaa.edu.cn, qinjiebuaa@gmail.com

Abstract

Action segmentation, inferring temporal positions of human actions in an untrimmed video, is an important prerequisite for various video understanding tasks. Recently, unsupervised action segmentation (UAS) has emerged as a more challenging task due to the unavailability of frame-level annotations. Existing clustering- or prediction-based UAS approaches suffer from either over-segmentation or overfitting, leading to unsatisfactory results. To address those problems, we propose **P**redictive **A**nd **C**ontrastive **E**mbedding (PACE), a unified UAS framework leveraging both predictability and similarity information for more accurate action segmentation. On the basis of an auto-regressive transformer encoder, predictive embeddings are learned by exploiting the predictability of video context, while contrastive embeddings are generated by leveraging the similarity of adjacent short video clips. Extensive experiments on three challenging benchmarks demonstrate the superiority of our method, with up to 26.9% improvements in F1-score over the state of the art.

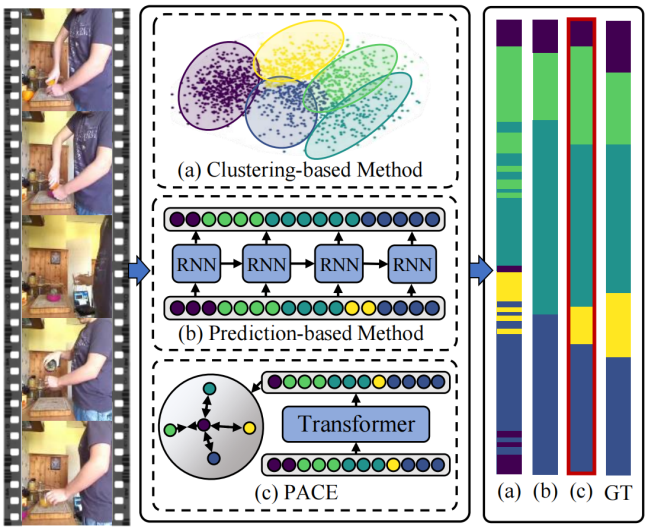

Figure 1: Comparison of different UAS methods. (a) Clustering-based methods suffer from over-segmentation with weak contextual modeling. (b) Prediction-based methods yield incomplete results due to overfitting. (c) Our proposed PACE leverages both predictability and similarity information for more accurate predictions.

1 Introduction

Aiming to classify video frames into predefined action categories, action segmentation is an important step towards fine-grained human action understanding [Lea *et al.*, 2017; Li *et al.*, 2021; Wang *et al.*, 2021]. Since annotating frame-wise action labels is both tedious and laborious, increasing attention has recently been paid to unsupervised action segmentation (UAS), where only the number of action categories is given. UAS is much more challenging than its supervised counterpart, because no supervision is provided to guide the optimization of frame-wise classification models.

To tackle this challenge, there have been several attempts that can be roughly divided into two categories, which respectively exploit the similarity and predictability of spatio-temporal patterns in videos. As shown in Figure 1 (a), clustering-based methods employ unsupervised algorithms like *k*-means [Bhatnagar *et al.*, 2017; Kukleva *et al.*, 2019] and FINCH [Sarfraz *et al.*, 2021] to cluster extracted frame-level features. The resulting centroids are then mapped to different action categories, based on which segmentation results are obtained. Since frame-level representations are discretely clustered, these methods are vulnerable in video context modeling. Although techniques like Viterbi decoding [Kukleva *et al.*, 2019] and temporal weighting [Sarfraz *et al.*, 2021] can partially make up for the limitation, they still have over-segmentation problems with fragmented results.

Prediction-based methods are inspired by the cognitive psychology theory that a different event is perceived by humans when the received information deviates from the predictions [Zacks *et al.*, 2001]. Figure 1 (b) illustrates the perceptual prediction framework proposed in [Aakur and Sarkar, 2019]. Specifically, they first utilize a convolutional neural network (CNN) to encoded frame-level features. A recurrent neural network (RNN) is then employed to predict future representations from the history. Boundaries between different actions are detected by finding peaks on the curve of predic-

*Corresponding Author

tion errors. However, as the RNN propagates errors at every time step, it is prone to overfitting, leading to missing and incorrect detection of action boundaries.

To overcome the above limitations, we propose ***P**redictive **A**nd **C**ontrastive **E**mbedding* (PACE), the first unified UAS framework that simultaneously leverages similarity and predictability information for more accurate action segmentation. Figure 2 illustrates the overall framework of PACE. In the training stage, given extracted frame-level representations, we employ an auto-regressive transformer encoder [Vaswani *et al.*, 2017] to predict future representations from the past ones, generating frame-level predictive embeddings. Meanwhile, we divide the whole video into short clips and form clip-level representations by merging predictive embeddings within each clip. We perform contrastive learning on these representations to generate clip-level contrastive embeddings, by pulling temporally adjacent clips closer and pushing nonadjacent ones farther. In the test stage, prediction errors and embedding similarities are inferred from the above predictive and contrastive embeddings. After fusing the two values, we perform action boundary detection with the fusion values to generate final segmentation results.

PACE shows its unique advantages over the existing UAS methods. Compared to clustering-based methods, it well exploits contextual dependencies within each video by learning predictive embeddings. As a result, smoother segmentation results are generated with less over-segmentation. In comparison with prediction-based methods, PACE alleviates the overfitting problem with additional similarity information from contrastive embeddings. Moreover, long-term temporal information is captured by the transformer encoder, resulting in more temporally consistent segmentation results.

Our main contributions are summarized as follows:

- We are the first to exploit both predictability and similarity information for UAS. By leveraging their complementarity, we overcome the limitations of existing clustering- and prediction-based methods.
- We design a unified framework to simultaneously learn predictive and contrastive embeddings, based on which accurate action boundaries are detected.
- Extensive experiments on three challenging benchmarks demonstrate the superiority of PACE, with up to 26.9% improvements in F1-score over the state of the art.

2 Related Work

Fully Supervised Action Segmentation. With complete temporal annotations available, action segmentation is treated as a frame-wise classification problem. Frame-level features are first extracted by handcrafted [Wang and Schmid, 2013] or deep learning methods [Carreira and Zisserman, 2017]. Models like RNN [Singh *et al.*, 2016], Hidden Markov Model (HMM) [Kuehne *et al.*, 2014] and temporal convolutional network (TCN) [Lea *et al.*, 2017; Wang *et al.*, 2019; Wang *et al.*, 2020] are then utilized to explore temporal context and generate segmentation results.

Unsupervised Action Segmentation. Recent works start to focus on training action segmentation models without any supervision signals. It is a much more challenging problem since no information can be directly utilized to optimize frame-wise classification models. Therefore, existing works usually exploit internal action patterns trough clustering- or prediction-based methods.

Clustering-based methods cluster frame-level features into certain sub-classes, which are then mapped to different actions. In [Kukleva *et al.*, 2019], frame-level features are projected into continuous temporal embeddings with time-stamp supervision. *k*-means algorithm is adopted to cluster the embeddings into semantically meaningful action classes. The TW-FINCH clustering algorithm proposed in [Sarfraz *et al.*, 2021] represents a video with a one-nearest neighbor graph reflecting both spatial and temporal distance of video frames.

Prediction-based methods aim to segment different actions from the perspective of future predictability. In [Aakur and Sarkar, 2019], a CNN-LSTM framework is proposed to predict future video frames in the representation space. Boundaries of different actions are determined by finding local maximal values in prediction errors.

In general, clustering-based methods leverage the similarity information of visual representations, while prediction-based methods utilize the predictability of video context. In this work, we propose PACE, which well exploits the complementarity of both similarity and predictability information for more accurate action segmentation results.

3 Method

Figure 2 illustrates the overall framework of PACE. During training, on the basis of an auto-regressive transformer encoder, frame-level predictive embeddings are learned by predicting future representations from the past ones. Furthermore, contrastive embeddings are generated by contrastive learning on clip-level representations. In the test stage, we fuse prediction errors and embedding similarities for action boundary detection. Below, we first present the detailed architecture of the encoder, and then introduce the procedures of learning predictive and contrastive embeddings. Finally, we elaborate on how to infer action boundaries.

3.1 Encoder Design

Given extracted frame-level features $F \in \mathbb{R}^{n\times d}$, we first project them into hidden representations $R \in \mathbb{R}^{n\times h}$ with a single fully connected layer, where n is the number of video frames, and d, h are feature dimensionalities. We then employ an encoder network for temporal modeling, which is of an auto-regressive transformer architecture [Vaswani *et al.*, 2017]. Let SA, FFN, LN denote multi-head self-attention, position-wise feed-forward network and layer normalization, respectively. The basic encoder layer can be formulated as:

$$\begin{aligned} R_l^{'} &= \mathrm{LN}(\mathrm{SA}(R_{l-1}, M) + R_{l-1}) \\ R_l &= \mathrm{LN}(\mathrm{FFN}(R_l^{'}) + R_l^{'}) \end{aligned}, \tag{1}$$

where R_l is the output features of the l-th layer and $M \in \mathbb{R}^{n\times n}$ is a lower triangular mask. Since the predictive embedding prohibits any leftward information flow, M is used here to make our encoder auto-regressive. We also add positional encodings to R. Compared with the RNN and TCN models used in previous works [Aakur and Sarkar, 2019; Lea *et al.*, 2017], our transformer based encoder captures

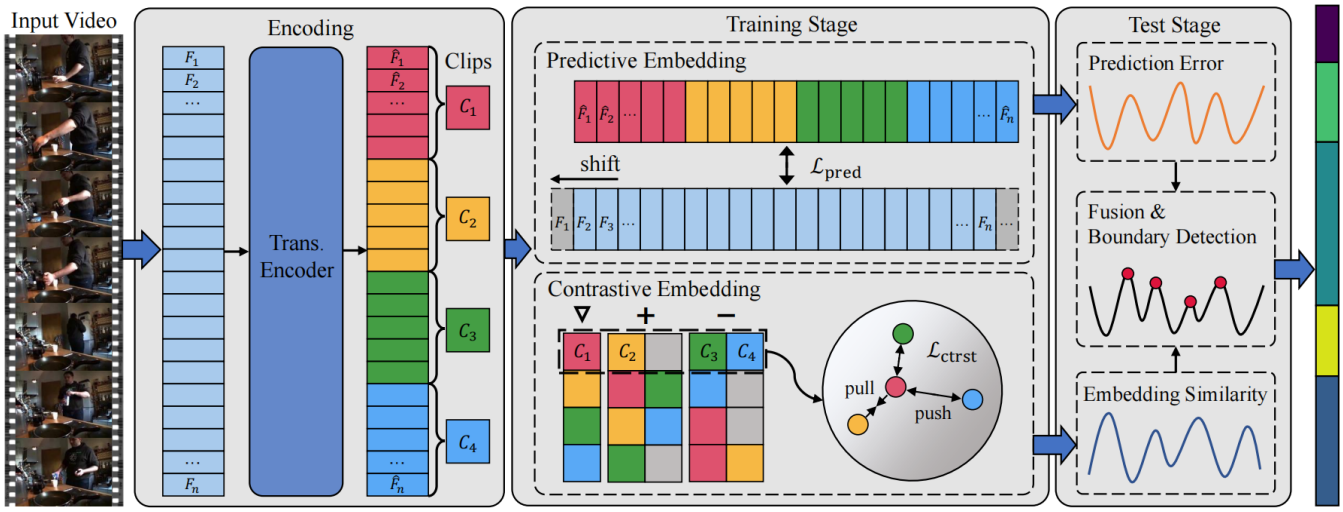

Figure 2: Overall framework of PACE. Given extracted frame-level features, we employ a transformer encoder for temporal modeling. In the training stage, predictive embeddings are learned by predicting future representations from the past ones, while contrastive embeddings are generated with contrastive learning on clip-level representations. In the test stage, we fuse prediction errors and embedding similarities for action boundary detection. $\triangledown$, $+$ and $-$ denote anchors, positives and negatives in contrastive learning, respectively.

temporal information with non-local attention mechanisms, modeling context dependencies in longer temporal spans with better flexibility.

3.2 Embedding Learning

Predictive Embedding. Cognitive studies have shown that humans perceive activities by evaluating the difference between true reality and inner predictions [Zacks *et al.*, 2001]. Therefore, we exploit such predictability information for UAS with predictive embeddings. Rather than directly predict future video frames, we choose to predict high-level representations, since it is more robust and efficient than pixel-level reconstruction [Aakur and Sarkar, 2019].

Specifically, let R_o denote the output representations from the encoder. A predictive projection head P_{pred} is first employed to transform R_o into predictive representations $\hat{F} \in \mathbb{R}^{n\times d}$ as:

$$\hat{F} = \mathrm{P}_{\text{pred}}(R_o). \tag{2}$$

We then learn predictive embeddings by imposing a predictive loss $\mathcal{L}_{\text{pred}}$ between $\hat{F}$ and the original frame-level representations F as:

$$\mathcal{L}_{\text{pred}} = \frac{1}{(n-1)d}\sum_{i=1}^{n-1}\sum_{j=1}^{d}(\hat{F}_{i,j} - F_{i+1,j})^2. \tag{3}$$

More concretely, as shown in Figure 2, we shift F one step left and compute the ℓ_2 distance between prediction $\hat{F}_i$ and ground truth F_{i+1}. In the test stage, action boundary information can be easily inferred from the predictive loss.

Contrastive Embedding. Due to the strong predictive ability of our transformer encoder, solely training predictive embeddings without other constraints is usually prone to overfitting. Fortunately, clustering-based UAS methods have demonstrated that similarity information among visual representations can also be utilized to separate actions with different semantics. Hence, we propose to further exploit similarity information via contrastive learning to circumvent the limitations in predictive embeddings.

As shown in Figure 2, unlike predictive embeddings, contrastive embeddings are based on clip-level representations, which capture more contextual semantics than frame-level representations. Concretely, given $\hat{F}$ containing predictive embeddings of all video frames, we merge features within each consecutive clip of s frames, resulting in clip-level representations $C^{'} \in \mathbb{R}^{t\times d}$, where $t = n/s$ indicates the total number of clips. We propose an adaptive merging method facilitated by a temporal attention block TA as:

$$C_i^{'} = \mathrm{TA}(\hat{F}_{is+1:(i+1)s})\hat{F}_{is+1:(i+1)s}, \tag{4}$$

where TA produces temporal attention weights through a fully connected layer. The i-th clip-level representation $C_i^{'}$ is then generated with a weighted sum of predictive embeddings within the clip, i.e. $\hat{F}_{is+1:(i+1)s} \in \mathbb{R}^{s\times d}$.

Next, we employ a contrastive projection head $\mathrm{P}_{\text{ctrst}}$ to transform $C^{'}$ into contrastive embeddings $C \in \mathbb{R}^{t\times m}$ with m being the new dimensionality, which we formulate as:

$$C = \mathrm{P}_{\text{ctrst}}(C^{'}). \tag{5}$$

The essential of contrastive embedding is to pull temporally adjacent clips closer and push nonadjacent ones farther, since clips closer to each other usually share more semantic similarities. To this end, we introduce a contrastive loss $\mathcal{L}_{\text{ctrst}}$ by adapting the widely adopted InfoNCE objective [Mnih and Kavukcuoglu, 2013] to our framework as:

$$\mathcal{L}_{\text{ctrst}} = \sum_{i=1}^{t}\frac{-1}{|\mathcal{P}(i)|}\sum_{p\in\mathcal{P}(i)} log\frac{\exp(C_i\cdot C_p/\tau)}{\sum_{j=1,j\neq i}^{t}\exp(C_i\cdot C_j/\tau)}, \tag{6}$$

where $\mathcal{P}(i) = \{i-1, i+1\} \cap \{1 \le p \le t\}$ contains the indexes of positive samples for anchor C_i, i.e. its left and right

neighbours, and τ is the temperature parameter. We take each of the t clips in C as anchors and construct positive and negative pairs based on the remaining clips. By calculating the inner product of contrastive embeddings, $\mathcal{L}_{\text{ctrst}}$ enforces cosine similarity between positive pairs. Such similarity values can be conveniently deduced for UAS in the test stage.

Overall Objective. Combining the predictive and contrastive losses, the overall objective of PACE is to optimize the following loss function $\mathcal{L}$:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \alpha \mathcal{L}_{\text{ctrst}}, \tag{7}$$

where α is the weighting factor balancing the two terms.

3.3 Action Boundary Inference

In terms of contextual predictability, a new action is most likely to begin where the prediction error is high. In contrast, adjacent video clips with lower embedding similarity is more possible to contain an action boundary. Below, we introduce how to infer segmentation results by fusing both values.

Prediction Error. As we make one-step-ahead predictions with predictive embeddings, predictability information can be directly inferred from $\mathcal{L}_{pred}$. For a video with n frames, we generate the prediction errors $E \in \mathbb{R}^n$ by computing the ℓ_2 distance between predictions and the ground truth.

Embedding Similarity. To infer the embedding similarities for the entire video, we utilize a sliding window of $2s$ frames. Each time the window slides through $\hat{F}$ by moving 1 frame forward. Representations within each window can form two adjacent contrastive embeddings. We compute the cosine similarities between these adjacent embeddings to generate the similarity values $S \in \mathbb{R}^n$ covering the entire video.

Value Fusion. To exploit the complementarity of predictability and similarity information for UAS, we perform the following fusion strategy:

$$V = \frac{E - \min(E)}{\max(E) - \min(E)} - \frac{S - \min(S)}{\max(S) - \min(S)}, \tag{8}$$

where $\min$ and $\max$ represent the minimum and maximum values of a vector, respectively. The fused value $V \in \mathbb{R}^n$ is practically formed by E and $-S$ after min-max normalization, giving them equal importance.

Action Boundary Detection. After the fusion, V_i indicates the potential of frame i being an action boundary. We then employ an action boundary detection method to generate the final action segmentation results. Specifically, we first convolve V with a smoothing kernel $K \in \mathbb{R}^k$ to filter out the noise. Here $K_i = \frac{1}{k}$, and k indicates the kernel size. Next, we employ the local extrema detection method from Scipy [Virtanen *et al.*, 2020] to predict action boundaries.

4 Experiments

Datasets. We evaluate the performance of PACE on three UAS benchmarks, namely Breakfast [Kuehne *et al.*, 2014], 50Salads [Stein and McKenna, 2013] and YouTube Instructions (YTI) [Alayrac *et al.*, 2016]. Note that there exist normal and finer action granularities for 50Salads, which are called eval and mid. On Breakfast and 50Salads, we utilize the same 2,048-d I3D [Carreira and Zisserman, 2017] extracted features as in [Farha and Gall, 2019]. On YTI, we adopt the 3,000-d features provided by the authors.

Metrics. Following most existing works [Kukleva *et al.*, 2019; Sarfraz *et al.*, 2021; Li and Todorovic, 2021], we use the Hungarian algorithm to map predicted segments to ground-truth action categories. We comprehensively evaluate PACE in all three metrics employed in previous works, i.e. mean over frames accuracy (MoF), F1-score and intersection over union (IoU), which verify action segmentation results from different perspectives. Specifically, MoF simply calculates frame-wise classification accuracy; F1-score is based on precision and recall of segment-level predictions; IoU evaluates the divergence between predicted and ground-truth segments with the Jaccard index.

Implementation Details. Our model is implemented with TensorFlow. The encoder has 3 basic layers in total. There are 4 attention heads in SA, and the inner-layer of FFN is 2,048-d. We set the dimensionalities of both hidden representations (h) and contrastive embeddings (m) to 512. The encoder takes as input video sequences of 100 frames (i.e. $n = 100$) and the sequence is then divided into clips with length $s = 5$. We set the training batch size to 32. To increase the contrastive power in $\mathcal{L}_{\text{ctrst}}$, we expand negative samples with C_j from different sequences in the same batch. We empirically set α to 0.1 in order to maintain comparable scales of the two losses. We utilize the Adam [Kingma and Ba, 2015] optimizer with a learning rate of 0.0001. The total training epochs are 50, 30, 10 on Breakfast, 50Salads and YTI, respectively, according to their scales. All experiments are conducted with two NVIDIA RTX 3090 GPUs.

4.1 Comparison with State-of-the-Art Methods

As shown in Table 1, we thoroughly compare PACE with the state-of-the-art UAS methods on three benchmarks. Results of some representative fully and weakly supervised methods are also included for reference.

Breakfast. PACE outperforms other methods on Breakfast in all three metrics. Concretely, we respectively achieve 26.9%, 1.9%, 1.3% improvements in F1-score, MoF and IoU compared with the previous best results. The significant improvement in F1-score indicates that our method maintains good balance between precision and recall of segment-level predictions, which is not sufficiently evaluated by MoF.

It can be observed that clustering-based methods like TW-FINCH obtain high performance in MoF but show unsatisfactory results in IoU. In contrast, prediction-based method like LSTM+AL shows the opposite trends. This is because that these two paradigms focus on frame-level similarity and video context predictability, respectively. Since PACE leverages the information from both sides, it manages to improve the performance in MoF and IoU at the same time.

50Salads. The videos on 50Salads have an average length of over $10,000$ frames, more than five times longer than the other two benchmarks. Therefore, it is more critical to model long-term temporal dependencies on this dataset. In terms of action granularities, the mid-level is more challenging with 19 actions, while the eval-level only has 10. PACE achieves state-of-the-art results under both granularities. Compared with the previous best-performing TW-FINCH, PACE has

Supervision	Method	Ref.	Breakfast			50Salads (eval)			50Salads (mid)			YTI		
			F1	MoF	IoU	F1	MoF	IoU	F1	MoF	IoU	F1	MoF	IoU
Fully Sup.	ED-TCN [Lea *et al.*, 2017]	CVPR'17	-	-	-	-	72.0	-	-	64.7	-	-	-	-
	MS-TCN [Farha and Gall, 2019]	CVPR'19	-	66.3	-	-	80.7	-	-	-	-	-	-	-
Weakly Sup.	NN-Vit. [Richard *et al.*, 2018]	CVPR'18	-	43.0	-	-	-	-	-	49.4	-	-	-	-
	CDFL [Li and Todorovic, 2020]	CVPR'20	-	50.2	33.7	-	-	-	-	54.7	-	-	-	-
Unsup.	Frank-Wolfe [Alayrac *et al.*, 2016]	CVPR'16	-	-	-	-	-	-	-	-	-	24.4	-	-
	LSTM+KNN [Bhatnagar *et al.*, 2017]	IJCAI'17	-	-	-	-	54.0	-	-	-	-	-	-	-
	Mallow [Sener and Yao, 2018]	CVPR'18	-	34.6	47.1	-	-	-	-	-	-	27.0	27.8	-
	UTE [Kukleva *et al.*, 2019]	CVPR'19	26.4	41.8	-	-	35.5	-	-	30.2	-	28.3	39.0	-
	LSTM+AL [Aakur and Sarkar, 2019]	CVPR'19	-	42.9	46.9	-	60.6	-	-	-	-	39.7	-	-
	TW-FINCH [Sarfraz *et al.*, 2021]	CVPR'21	-	63.8	44.1	-	71.7	-	-	66.8	-	51.9	**58.6**	-
	ASAL [Li and Todorovic, 2021]	CVPR'21	37.9	52.5	-	-	39.2	-	-	34.4	-	32.1	44.9	-
	SSCAP [Wang *et al.*, 2022]	WACV'22	39.2	51.1	-	30.3	41.4	-	-	-	-	-	-	-
	PACE	Ours	**66.1**	**65.7**	**48.4**	**56.4**	**72.2**	**52.3**	**54.5**	**67.4**	**48.7**	**52.1**	58.2	**56.8**

Table 1: Performance comparison on Breakfast, 50Salads and YTI. '-' indicates no results are reported. The best UAS results are in bold.

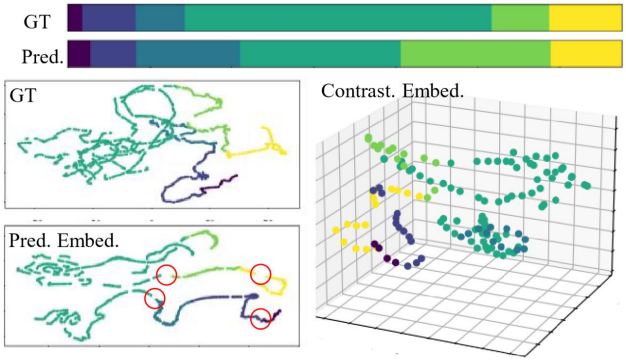

Figure 3: Visualization of predictive and contrastive embeddings for sample *webcam01_P17_juice* from Breakfast. Different colors indicate ground-truth action labels of corresponding embeddings.

0.5% and 0.6% improvements in MoF. This is nontrivial considering that the performance already surpasses some fully and weakly supervised methods such as ED-TCN and CDFL.

Although rarely reported in prior arts, 50Salads is more challenging in segment-level metrics like F1-score and IoU, due to the significantly longer video duration. PACE exceeds SSCAP, i.e. the former best performer in F1-score, by 26.1%. This demonstrates that our transformer encoder combined with temporal embeddings has a distinct advantage handling action segments in long-term video context.

YTI. Since 63.5% of all video frames are background frames, available action instances are very limited on YTI. Nevertheless, PACE achieves the best results in F1-score and IoU. Note that with the small scale of YTI, prediction-based method like LSTM+AL is prone to overfitting, while PACE tackles this issue by combing both predictive and contrastive embeddings. Although TW-FINCH achieves slightly higher performance in MoF, PACE is still competitive with better segment-level action predictions.

4.2 Qualitative Results

Segmentation Results. To qualitatively compare predictions of different models, we visualize segmentation results of representative clustering-based methods (UTE, TW-FINCH) and prediction-based method (LSTM+AL) in Figure 4. More results can be found in the supplementary material.

Figure 4 (a) shows the prediction results on a sample from Breakfast, which is a short video with 345 frames. The two major sub-actions, namely *pour coffee* and *pour milk*, are framed with red lines in the ground truth with sample frames shown at the top. It can be observed that UTE and TW-FINCH suffer from over-segmentation by confusing the ground truth with other sub-actions, like *take cup* and *pour sugar*. This is mainly because the clustering-based methods focus too much on frame-wise similarity and overlook contextual information. LSTM+AL fails to separate the two sub-actions due to overfitting, which makes it insensitive to some action boundaries. Our method overcome above limitations with the most consistent prediction.

Figure 4 (b) shows the prediction results on a sample from 50Salads, which has a long duration of $8,599$ frames. We frame five ground-truth action instances in red, which are *add salt*, *add vinegar*, *add oil*, *mix ingredients* and *add dressing*, respectively. Our method successfully separates the first three consecutive segments while the other competitors fail. Since most frames in the three segments are very similar in appearance, it is difficult to separate them with only similarity information. But there still exist transitions between two different segments, which can be detected in prediction errors. The similar phenomenon can be found in the last two segments.

Embedding Visualization. To shed light on the learned predictive and contrastive embeddings, we visualize them in Figure 3. Frame-level predictive embeddings together with ground-truth features are shown on the bottom left. We project them into a 2D plane with the t-SNE algorithm [Van der Maaten and Hinton, 2008] in ℓ_2 distance. Due to the strong temporal correlation, the ground-truth features exhibit a linear distribution. Our predictive embeddings have a similar distribution but with more disconnections. As circled in red, some disconnections lie in between different action segments, indicating that the predictive embeddings become heterogeneous at boundary positions.

We also depict a 3D projection of the clip-level contrastive embeddings with t-SNE on the bottom right. We can see that embeddings from different action segments gather into different clusters. As the projection is made in cosine distance, this result demonstrates that contrastive embeddings are capable of separating different actions with similarity information.

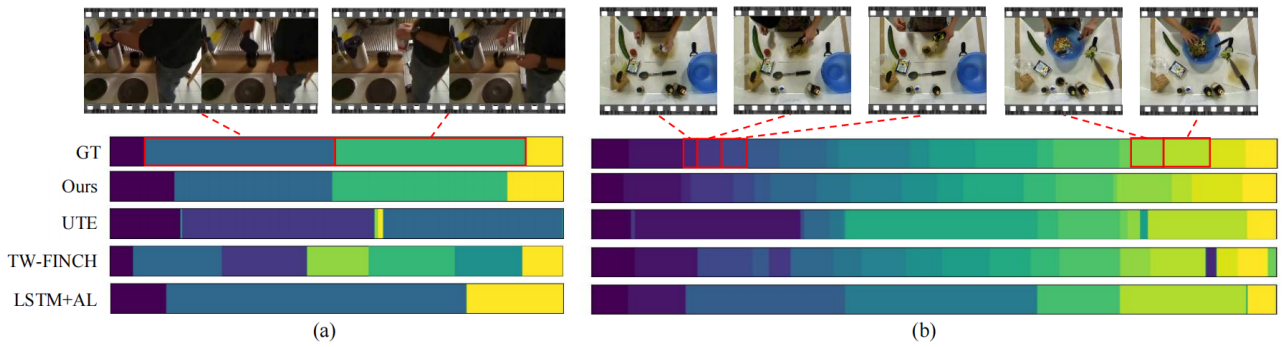

Figure 4: Comparison of segmentation results on two sample videos. (a) *webcam01_P14_coffee* from Breakfast. (b) *rgb-14-1* from 50Salads. PACE alleviates over-segmentation and overfitting problems in other methods, generating accurate results for both short and long videos.

Experiments		F1	MoF	IoU
PACE (original)		66.1	**65.7**	**48.4**
Encoder	LSTM	57.1	58.6	44.3
	TCN	64.5	62.4	46.2
Embedding	Predictive	63.1	62.8	45.8
	Contrastive	63.4	64.2	46.1
Clip Length	$s = 1$	62.7	62.5	45.3
	$s = 3$	64.3	65.1	47.5
	$s = 7$	**66.7**	65.4	48.1
	$s = 9$	65.8	65.2	47.9

Table 2: Results of ablation study on Breakfast.

4.3 Ablation Study

Encoder Architecture. We compare the transformer encoder with LSTM [Aakur and Sarkar, 2019] and TCN [Lea *et al.*, 2017] under the same PACE framework. Both models adopt the same 512-d hidden representations as with our transformer. As shown in Table 2, LSTM achieves the worst performance, which is 9% lower than PACE in F1-score. TCN obtains much better performance, showing the importance of capturing long-term temporal dependencies. However, TCN still underperforms our transformer encoder due to the locality of temporal convolutions.

Embedding Methods. To evaluate the efficacy of predictive and contrastive embeddings, we ablate either of them and show the results in Table 2. The predictive embedding only method shows a performance drop of 3%, 2.7% and 2.6% in F1-score, MoF and IoU, respectively, while only adopting the contrastive embedding has a respective decline of 2.7%, 1.5% and 2.3%. We further show the advantage of combining both embeddings with a sample in Figure 5. Curves of smoothed prediction errors, embedding similarities and fusion values are visualized with their predictions. Although mis-predictions are made by solely using predictability or similarity information, combining both of them yields a better result with correct action orders. This indicates that the complementarity of both information is well exploited by PACE.

Clip Length. We also conduct experiments to evaluate the influence of clip length s for contrastive embeddings. As shown in Table 2, we evaluate shorter and longer clips with 1, 3, 7 and 9 frames. In fact, setting s to 1 is equivalent to taking a single frame as a clip. As s becomes larger, the performance first increases and then decreases. Although employing 7 frames results in higher F1-score, a better balance among all metrics is guaranteed with 5 frames, showing that a moderate length is desired by contrastive embeddings.

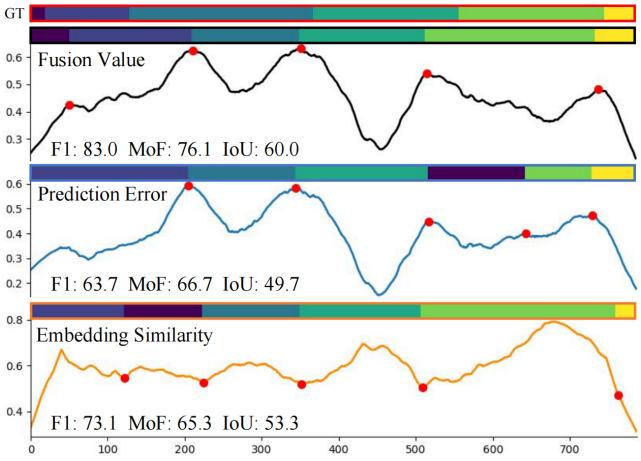

Figure 5: Visualization of prediction errors, embedding similarities and fusion values for sample *cam01_P14_milk* from Breakfast. Red dots denote the predicted boundaries.

5 Conclusion

We propose PACE in this paper, which is the first method leveraging both predictability and similarity information for UAS. On the basis of a transformer encoder, the predictive embeddings are learned by forecasting future frames in the representation space, while the contrastive embeddings are generated by enforcing similarity of neighboring video clips. We evaluate PACE on three challenging benchmarks, where it outperforms the competitors by a large margin. Further experiments show PACE well exploits the complementarity of both embeddings. As future works, we will explore the potential of PACE in other video understanding tasks like summarization and captioning.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant No. U20B2069).

References

[Aakur and Sarkar, 2019] Sathyanarayanan N Aakur and Sudeep Sarkar. A perceptual prediction framework for self supervised event segmentation. In *Proc. CVPR*, pages 1197–1206, 2019.

[Alayrac *et al.*, 2016] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proc. CVPR*, pages 4575–4583, 2016.

[Bhatnagar *et al.*, 2017] Bharat Lal Bhatnagar, Suriya Singh, Chetan Arora, and CV Jawahar. Unsupervised learning of deep feature representation for clustering egocentric actions. In *Proc. IJCAI*, pages 1447–1453, 2017.

[Carreira and Zisserman, 2017] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proc. CVPR*, pages 6299–6308, 2017.

[Farha and Gall, 2019] Yazan Abu Farha and Jurgen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proc. CVPR*, pages 3575–3584, 2019.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2015.

[Kuehne *et al.*, 2014] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proc. CVPR*, pages 780–787, 2014.

[Kukleva *et al.*, 2019] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Jurgen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *Proc. CVPR*, pages 12066–12074, 2019.

[Lea *et al.*, 2017] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *Proc. CVPR*, pages 156–165, 2017.

[Li and Todorovic, 2020] Jun Li and Sinisa Todorovic. Set-constrained viterbi for set-supervised action segmentation. In *Proc. CVPR*, pages 10820–10829, 2020.

[Li and Todorovic, 2021] Jun Li and Sinisa Todorovic. Action shuffle alternating learning for unsupervised action segmentation. In *Proc. CVPR*, pages 12628–12636, 2021.

[Li *et al.*, 2021] Yixuan Li, Lei Chen, Runyu He, Zhenzhi Wang, Gangshan Wu, and Limin Wang. Multisports: A multi-person video dataset of spatio-temporally localized sports actions. In *Proc. ICCV*, pages 13536–13545, 2021.

[Mnih and Kavukcuoglu, 2013] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *Proc. NeurIPS*, pages 2265–2273, 2013.

[Richard *et al.*, 2018] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *Proc. CVPR*, pages 7386–7395, 2018.

[Sarfraz *et al.*, 2021] Saquib Sarfraz, Naila Murray, Vivek Sharma, Ali Diba, Luc Van Gool, and Rainer Stiefelhagen. Temporally-weighted hierarchical clustering for unsupervised action segmentation. In *Proc. CVPR*, pages 11225–11234, 2021.

[Sener and Yao, 2018] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *Proc. CVPR*, pages 8368–8376, 2018.

[Singh *et al.*, 2016] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proc. CVPR*, pages 1961–1970, 2016.

[Stein and McKenna, 2013] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proc. UbiComp*, pages 729–738, 2013.

[Van der Maaten and Hinton, 2008] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[Virtanen *et al.*, 2020] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3):261–272, 2020.

[Wang and Schmid, 2013] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proc. ICCV*, pages 3551–3558, 2013.

[Wang *et al.*, 2019] Jiahao Wang, Zhengyin Du, Annan Li, and Yunhong Wang. Atrous temporal convolutional network for video action segmentation. In *Proc. ICIP*, pages 1585–1589. IEEE, 2019.

[Wang *et al.*, 2020] Zhenzhi Wang, Ziteng Gao, Limin Wang, Zhifeng Li, and Gangshan Wu. Boundary-aware cascade networks for temporal action segmentation. In *Proc. ECCV*, pages 34–51. Springer, 2020.

[Wang *et al.*, 2021] Jiahao Wang, Yunhong Wang, Sheng Liu, and Annan Li. Few-shot fine-grained action recognition via bidirectional attention and contrastive meta-learning. In *Proc. ACM MM*, pages 582–591, 2021.

[Wang *et al.*, 2022] Zhe Wang, Hao Chen, Xinyu Li, Chunhui Liu, Yuanjun Xiong, Joseph Tighe, and Charless C Fowlkes. Sscap: Self-supervised co-occurrence action parsing for unsupervised temporal action segmentation. In *Proc. WACV*, volume 2022, 2022.

[Zacks *et al.*, 2001] Jeffrey M Zacks, Barbara Tversky, and Gowri Iyer. Perceiving, remembering, and communicating structure in events. *Journal of Experimental Psychology: General*, 130(1):29, 2001.