Do We Really Need Temporal Convolutions in Action Segmentation?

Dazhao Du  
Institute of Software  
University of Chinese Academy of Sciences  
dudazhao20@mails.ucas.ac.cn

Bing Su*  
Gaoling School of Artificial Intelligence  
Renmin University of China  
bingsu@ruc.edu.cn

Yu Li  
IDEA Vistring Lab  
International Digital Economy Academy  
liyu@idea.edu.cn

Zhongang Qi  
ARC Lab  
Tencent PCG  
zhongangqi@tencent.com

Lingyu Si  
Institute of Software  
University of Chinese Academy of Sciences  
lingyu@iscas.ac.cn

Ying Shan  
ARC Lab  
Tencent PCG  
yingshan@tencent.com

Abstract—Recognizing and segmenting actions from long videos is a challenging problem. Most existing methods focus on designing temporal convolutional models. However, these models are limited in their flexibility and ability to model long-term dependencies. Transformers have recently been used in various tasks. But the lack of inductive bias and the inefficiency of handling long video sequences limit the application of Transformers in action segmentation. In this paper, we present a pure Transformer-based model without temporal convolutions in action segmentation, called Temporal U-Transformer. The U-Transformer architecture not only reduces complexity but also introduces an inductive bias that neighboring frames are more likely to belong to the same class. Besides, we further propose a boundary-aware loss based on the distribution of similarity scores between frames from attention modules to improve the ability to recognize boundaries. Extensive experiments show the effectiveness of our method.

Index Terms—video action segmentation, video understanding, Transformer, computer vision

I. INTRODUCTION

Video action segmentation aims to recognize and segment actions in long videos. Most of the previous deep learning methods adapt temporal convolutional networks (TCNs) as their backbones [1]–[3], which utilize temporal convolutions to capture the temporal relationships between frames. To capture long-term dependencies, the most popular TCN-based model, MS-TCN [2], adopts the strategy of doubling the dilation factor in 1D dilated convolution than the previous layer so that the receptive field grows exponentially, but Global2Local [4] has proven that there exist more effective receptive field combinations than this hand-designed pattern. Even different data distributions will result in different optimal receptive field combinations. Therefore, we need more flexible models that extract the dependency between frames from the data itself, instead of dilated convolutional structures with fixed weights and hand-designed patterns.

Transformers [5] outperform other deep models in many fields due to flexible modeling capabilities [6], [7]. However, there are few works utilizing Transformer to tackle the action segmentation task except ASFormer [8]. There exist two issues when applying Transformer to action segmentation. On the one hand, Transformer has few inductive biases and thus requires larger amounts of data for training. However, limited by the difficulty of labeling, most well-annotated datasets in this task are much smaller than the datasets in other fields [9], [10]. On the other hand, the complexities of attention module increase quadratically with the length of inputs. The untrimmed videos consisting of thousands of frames are too long to be directly processed by Transformer. ASFormer [8] combines a sparse attention mechanism and temporal convolutions to tackle these two issues, but it is more like incorporating additional attention modules into MS-TCN. Therefore, it is still an open problem whether a pure Transformer-based model without temporal convolutions is suitable for action segmentation and how to make it work.

To handle long videos, we first replace full attention in vanilla Transformer with local attention [11], where each frame only attends to frames within the local window. But local attention will reduce the receptive field so that the model still can not capture long-term dependencies. To this end, we introduce temporal sampling in each layer of the local-attended Transformer, resulting in a pure Transformer-based model without temporal convolutions. We call it Temporal U-Transformer (TUT) because temporal downsampling in the encoder and upsampling in the decoder are exploited to construct the temporal U-Net-like architecture [12]. Temporal sampling not only increases the receptive field exponentially with the number of layers but also further reduces the complexity. Moreover, we find that the U-Transformer architecture is well suited for dense prediction tasks because it introduces multi-scale information and priors that adjacent frames are likely to belong to the same class, which compensates for the lack of sufficient training data on action segmentation.

However, because coarse-grained features are supplied into the decoder, the U-Transformer architecture exacerbates the misclassification of boundaries. When decoded, the frames
close to the boundaries that are improperly encoded as coarse-grained features will be misclassified. To better perceive boundary information, we categorize the boundary frame in the video into two types: the start and the end frame, which represent the start and end of an action segment respectively. Intuitively, the start frame should be more similar to the neighboring frames after it, while the end frame should be more similar to those before it, which corresponds to two different similarity distributions. We define the similarity distribution of a frame with its neighbors as the local-attention distribution of the frame, which can be obtained from the local-attention module. We thereby introduce a boundary-aware loss by minimizing the distance between the local-attention distribution of the boundary frame with pre-defined prior distributions, which serves as a regularization to enforce the model to pay more attention to boundaries.

In summary, our main contributions are threefold: (1) For the first time, we propose a pure Transformer-based model without temporal convolutions for action segmentation, which breaks away from the constraints of temporal convolutions; (2) Based on the distribution of inter-frame similarity scores from the attention module and boundary labels, we propose a distribution-based boundary-aware loss to enable our model to classify boundaries more accurately. (3) Extensive experiments demonstrate the superiority of our TUT model and the effectiveness of the boundary-aware loss.

II. RELATED WORK

Action segmentation. Motivated by WaveNet [13], many recent works are devoted to exploring multi-scale information with TCNs for action segmentation [1]. In TDRN [14], deformable convolution is used to replace conventional convolution and residual connections are added to improve TCNs. MS-TCN [2] designs a multi-stage architecture and proposes the truncated MSE loss to penalize over-segmentations. Some researchers add additional modules to the MS-TCN, e.g., the boundary prediction module [15], the bilinear pooling operation [16] and the graph-based temporal reasoning module [17]. MS-TCN++ [3] replaces dilated temporal convolution layers in MS-TCN with dual dilated layers. Global2Local [4] proposes a global-local search scheme to search for effective receptive field combinations instead of hand-designed patterns. C2F-TCN [18] combines U-Net-like architecture and temporal convolutions, resulting in a coarse-to-fine structure. ASFormer [8] firstly introduces attention modules to action segmentation. In this paper, Transformer is integrated into the U-Net-like architecture, resulting in a pure Transformer model without temporal convolutions.

Transformer. Transformers [5] have succeeded in many fields due to flexibility and powerful modeling capabilities [19], [20]. However, to our best knowledge, the only Transformer-based model for the action segmentation task is ASFormer [8]. ASFormer utilizes 1D dilated convolution to bring in strong inductive priors, which does not break through the limitations of convolution. Inspired by Transformer-based models in semantic segmentation [21]–[23], we design a pure Transformer model which combines the U-Net-like architecture and Transformer. Informer [24] selects dominant queries by measuring the distance between their attention probability distributions and uniform distribution to reduce the complexity of attention. Differently, we minimize the distance between the prior distribution for boundary frames and the distribution of local attention to enhance the ability to discriminate boundaries.

III. PRELIMINARY

In this section, we recap the preliminaries in Transformer and the problem formulation of action segmentation.

Transformer. Each layer in Transformer consists of two main components: a multi-head self-attention module and a feed-forward network (FFN). We denote the input of the self-attention module as \( H \in \mathbb{R}^{T \times d} \), where \( T \) and \( d \) are the length and dimension of the input, respectively. The input is projected by three matrices \( W_Q \in \mathbb{R}^{d \times d}, W_K \in \mathbb{R}^{d \times d}, \) and \( W_V \in \mathbb{R}^{d \times d} \) to obtain the query, key and value matrices: \( Q = \{q_1, \ldots, q_T\}, K = \{k_1, \ldots, k_T\} \) and \( V = \{v_1, \ldots, v_T\} \). Then the attention calculation is given by:

\[
\text{Attn}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d_K}})V = \text{Softmax}(A)V
\]

where \( A \) is the attention matrix consisting of all the similarity scores between any query-key pair.

Action Segmentation. The input video is represented by a sequence of frame-wise features: \( x = [x_1, \ldots, x_T] \in \mathbb{R}^{T \times d} \), where \( x_t \) is the feature of the \( t \)-th frame, \( d \) is the dimensionality of \( x_t \), and \( T \) is the length of the video. Our goal is to predict the action class label \( c_t \in [1, \ldots, C] \) for each frame \( x_t \), resulting in the prediction \( \hat{c}_1, \ldots, \hat{c}_T \), where \( C \) is the number of classes.

IV. METHODOLOGY

A. Temporal U-Transformer Architecture

In this section, we present our model in detail. TUT contains a prediction generation stage and \( M \) refinement stages. The generation stage generates initial segmentation predictions while each refinement stage refines the predictions of the
previous stage. These stages have the same architecture. As shown in Fig. 1, each stage can be separated into four components: the input projection, the encoder, the decoder, and the output classifier. Both the input projector and the output classifier are fully connected layers, which reduce the input dimension to feed the encoder and classify the output of the decoder, respectively.

**Local Attention.** In the original self-attention module of Transformer, any query \( q_i \) needs to calculate the similarity scores with all the keys \( \{k_1, \ldots, k_T\} \) to generate the attention matrix \( A \), which leads to quadratic complexity, i.e., the complexity is \( O(T^2) \). Restricting the attention computation to a local window with a fixed size \( w \) can reduce the operation to linear complexity \( O(wT) \), which is called local attention [11]. At this point, each query \( q_i \) only needs to calculate the similarity with those keys in the window centered on its position, i.e., \( \{k_s, \ldots, k_{i-1}, k_{i+1}, \ldots, k_e\} \), where \( s = \max\{i - \lfloor \frac{w}{2}\rfloor, 0\} \) and \( e = \min\{i + \lfloor \frac{w}{2}\rfloor, T\} \) represent the start and end position respectively.

The local attention does not narrow down the overall receptive field of the model. Due to temporal sampling between layers, the receptive field increases exponentially with the number of layers, which is sufficient to cover the entire video sequence to capture global and local dependencies.

**Scale-Shared Positional Encoding.** We employ the learnable relative positional encoding [25] to introduce positional information, the basic idea of which is to embed the relative distances of all query-key pairs as scalars and add them to the attention matrix. Considering the distance between any two frames within a local window does not exceed the window size \( w \), we can get the relative position encoding \( R_{ij} \) between \( q_i \) and \( k_j \) by a learnable embedding matrix \( W_{rpe} \in \mathbb{R}^{w \times h} \), where \( i, j \) represent the position index and \( h \) is the number of heads. The resulting positional encoding \( R \) will be added to the corresponding positions of the attention matrices in different heads. Layers with the same layer index in different stages process inputs with the same temporal resolution, and their RPEs should be the same. Therefore, we adopt a scale-shared strategy, i.e., corresponding layers with the same scale in different stages share the same \( W_{rpe} \).

**Encoder and Decoder.** The encoder is composed of \( N \) identical encoder layers. As shown in Fig. 1, it is similar to the encoder in the vanilla Transformer but there are three differences. Firstly, there exists a nearest-neighbor downsampling process at the beginning of each layer, which halves the input temporal dimension. Secondly, the full attention is replaced by the local attention with scale-shared relative position encoding. Thirdly, we utilize the instance normalization [26] instead of the layer normalization [27].

The decoder is symmetric to the encoder. In each decoder layer, the temporal upsampling is utilized to gradually restore the original temporal resolution of input frames. The upsampling process is implemented by nearest interpolation. We do not concatenate the encoder layer input and the previous layer input as the decoder layer input like the original U-Net [12]. Instead, we modify the cross-attention in the original Transformer to leverage the information from the encoder. Specifically, in our local cross-attention, the query is generated by the output of the previous decoder layer, while the value and key both come from the output of the corresponding encoder layer having the same temporal dimension as the query.

**B. Boundary-aware Loss**

During the training phase, we combine three different losses: the frame-wise classification loss \( L_{CE} \), the smoothing loss \( L_{TMSE} \) in MS-TCN [2], and our proposed boundary-aware loss \( L_{BA} \). Since the loss function of each stage is exactly the same, we only analyze the loss of the \( s \)-th stage \( L_s \).

In the \( s \)-th stage, the frame-wise classification loss \( L_{CE}^s \) and the smoothing loss \( L_{TMSE}^s \) are formulated as:

\[
L_{CE}^s = \frac{1}{T} \sum_{t=1}^{T} -\log(y_t^s(c_t)),
\]

\[
L_{TMSE}^s = \frac{1}{|C|} \sum_{t=1}^{T} \sum_{c=1}^{C} \left[ \max(\log(y_t^s(c)) - \log(y_{t-1}^s(c)), \theta) \right]^2,
\]

where \( y_t^s(c_t) \) is the predicted probability that \( x_t \) belongs to the \( c_t \)-th class, and \( \theta = 4 \) is a pre-set threshold. In \( L_{TMSE}^s \), gradients are not calculated w.r.t. \( y_{t-1}^s(c) \).

Action boundaries are vital for action segmentation, but frame-wise classification treats boundary frames and intermediate frames equally. We propose a novel boundary-aware loss to enhance the ability to discriminate boundaries. This loss regularizes feature learning by imposing additional constraints on the attention matrix in the local attention module.

**Prior Distribution.** Intuitively, the start frame of an action should be more similar to the neighboring frames after it, while the end frame should be more similar to those before it. Therefore, the similarity distribution between a boundary frame \( i \) (anchor) and its neighbors should exhibit two different patterns, depending on whether the anchor is the start frame or the end frame. We use the adapted sign function as the two prior distributions corresponding to the above two patterns:

\[
P_i = \begin{cases} \text{Rescale}(\text{Sgn}(j - i) + 1) & i \in \{\text{start frames}\}, \\ \text{Rescale}(-\text{Sgn}(j - i) + 1) & i \in \{\text{end frames}\}, \end{cases}
\]
where \( j \in [i - \left\lfloor \frac{w}{2} \right\rfloor, i + \left\lfloor \frac{w}{2} \right\rfloor] \) is a frame within the local window, and \( (j - i) \) means the distance between frame \( j \) and anchor \( i \). \{start frames\} and \{end frames\} represent the set of start frames and the set of end frames, respectively. \( \text{Sgn}(x) \) is 1 when \( x \) is greater than or equal to 0 and -1 when \( x \) is less than 0. Further, we use \( \text{Rescale}(\cdot) \) to transform the sum of probabilities to 1.

**Local-Attention Distribution.** The attention matrix in the local attention module consists of all the similarity scores between query-key pairs. We can extract the similarity distribution of the anchor and its neighboring frames from the local attention module.

\[
D_i = A \left[ \lfloor i - \frac{w}{2} \rfloor : \lfloor i + \frac{w}{2} \rfloor \right].
\]

As shown in Fig. 2, we introduce a boundary-aware loss to approximate the local-attention distribution of the boundary to the corresponding prior distribution, which can be implemented by minimizing the symmetrized KL divergence between two distributions of the \( t \)-th frame:

\[
L^t_{BA} = \frac{1}{T} \sum_{t'=0}^{T-1} \text{KL}(P^t_{t'}||D^t_i).
\]

We can freely obtain boundary labels from class labels and the similarity distribution of each boundary from the local attention module. Therefore, we can calculate the boundary-aware loss without additional modules and annotations. We only compute it in the first layer of the encoder and the last layer of the decoder since temporal downsampling blurs high-level boundaries. The final loss for the \( s \)-th stage is the weighted sum of the three losses:

\[
L^s = L^s_{CE} + \lambda L^s_{TMSE} + \beta L^s_{BA}.
\]

**V. EXPERIMENTS**

**A. Datasets, Metrics and Experimental Details**

We empirically perform experiments on two public benchmark datasets: 50Salads [28] and Breakfast [29]. **50Salads** consists of 50 top-view videos with 17 action classes. On average, each video lasts for about 6.4 minutes and contains about 20 action instances. **Breakfast** consists of 1,712 third-person view videos with 48 action classes. On average, each video contains about 6 action instances.

We use the segmental F1 score at overlapping thresholds 10\%, 25\%, 50\% \((F1@\{10,25,50\})\), segmental edit distance (Edit), and frame-wise accuracy (Acc) as evaluation metrics. Following the general settings [2], we perform five-fold cross-validation on 50Salads and four-fold cross-validation on Breakfast and report the average performances.

We represent each video as a sequence of I3D [30] visual features provided in MS-TCN++ [3]. We use the ADAM optimizer [31] for a maximum number of 150 epochs to train our model. The learning rate is 0.0002 for Breakfast and 0.0005 for 50Salads. \( \beta \) is set to 0.02 on 50Salads and 0.005 on Breakfast. The batch size is set to 1.

**B. Comparison with the State-of-the-art**

Table I and Table II show our proposed model with the state-of-the-art methods. For a fair comparison of all models, we list the results of training our model without the additional boundary-aware loss, which corresponds to TUT\(^†\). We also report the results of our model jointly trained with the additional boundary-aware loss, called TUT.

TUT significantly outperforms the TCNs on all metrics by a large margin. Even though the best TCN-based model (C2F-TCN) utilizes data augmentation and additional video-level action loss that we do not employ, TUT\(^†\) outperforms it on both datasets. On 50Salads, our backbone TUT\(^†\) beats all previous methods including ASFormer on all metrics. On Breakfast, our model achieves the best performance on the F1 metrics, while the performance on Edit and Acc ranks in the top two of all methods. We also visualize some prediction results in Fig. 3. Besides, compared with TUT\(^†\), TUT is better able to recognize boundary frames, which leads to improvements in all metrics.

**C. Ablation Study**

**Effect of training data size.** To explore how the TCNs and our pure Transformer model are affected by the size of the training data, we train and evaluate the TUT model and two state-of-the-art TCN-based models under different training data sizes. The superiority of our pure Transformer-based model TUT is shown in Fig. 4. The performance of TUT gradually increases as the number of training samples

| Model   | F1\@\{10,25,50\} | Edit | Acc  |
|---------|-----------------|------|------|
| ED-TCN  | 66.0            | 52.6 | 64.7 |
| TDRN    | 72.9            | 68.5 | 68.1 |
| MS-TCN  | 76.3            | 74.0 | 80.7 |
| MS-TCN++| 80.7            | 75.7 | 83.7 |
| BCN     | 82.3            | 81.3 | 84.4 |
| Global2Local | 80.3 | 78.0 | 82.2 |
| ASRF    | 84.9            | 83.5 | 84.5 |
| C2F-TCN | 84.3            | 81.8 | 84.9 |
| ASFormer | 85.1        | 83.4 | 85.6 |

**TABLE I**

**Comparison with the State-of-the-art on 50Salads.**

| Model   | F1\@\{10,25,50\} | Edit | Acc  |
|---------|-----------------|------|------|
| TUT\(^†\) | 87.7            | 79.9 | 85.9 |
| TUT    | 89.3            | 81.7 | 87.2 |

**TABLE II**

**Comparison with the State-of-the-art on Breakfast.**

**References:**

[1] ED-TCN [1]
[2] TDRN [14]
[3] MS-TCN [2]
[4] MS-TCN++ [3]
[5] BCN [15]
[6] Global2Local [4]
[7] ASRF [32]
[8] C2F-TCN [18]
[9] ASFormer [8]

**1017**
Fig. 3. Visualization of segmentation results on (left) 50Salads and (right) Breakfast datasets. The same color represents the same action class. Our TUT model has more accurate action classification results compared to ASFormer and MS-TCN++, and the TUT model trained with boundary-aware loss can further reduce over-segmentation errors.

Effect of the boundary-aware loss. The quantitative impact of the boundary-aware loss on performance can already be observed in Table I and Table II. Besides, in Fig. 3, we observe that TUT trained without the boundary-aware loss does not accurately locate the boundaries of some action segments and sometimes incorrectly misclassifies intermediate frames as boundary frames, resulting in over-segmentation. The TUT jointly trained with the boundary-aware loss largely improves these problems.

Comparison of different positional encoding. We compare different positional encoding (PE) methods in Table III. Since the lengths of video samples vary over a large span, we observe that inflexible absolute positional encoding (APE) leads to performance degradation. The overall performance of RPE is better than that without PE, which illustrates the importance of position information. We also compare three different sharing ways of applying RPE. Stage-Shared has the worst performance since it applies the same RPE to layers that deal with inputs of different resolutions. Scale-Shared performs best, which means that layers with the same scale need similar positional encoding, even if they belong to different stages.

Ablations of the architecture and attention patterns. We compare two model architectures and three attention patterns, resulting in a total of six combinations in Table IV. We remove all the temporal sampling in TUT to get the standard architecture. Attention patterns include full attention in the original Transformer [5], local attention, and LogSparse attention proposed in [33], where each cell only attends to those cells whose distance from it increases exponentially. In the standard architecture, long video inputs will cause out-of-memory. For a fair comparison, we downsample video samples below 5000 frames and control all hyperparameters to be consistent. The U-Transformer architecture achieves better performance than the standard architecture with less GPU memory consumption. Full attention fails regardless of the architecture, showing that training on small data requires sparser attention patterns. Since adjacent frames usually have a stronger correlation in action segmentation, local attention performs better than LogSparse attention.
In action segmentation tasks, most popular deep learning methods use temporal convolutional networks as their backbones. However, temporal convolutions limit the performance of these methods. For the first time, we propose a model without temporal convolutions which combines the temporal sampling and Transformer to construct a temporal U-Transformer architecture. The temporal downsampling and local attention modules together enable our model to very long videos. Furthermore, we propose a novel boundary-aware loss based on the local-attention distributions of boundary frames, which serves as a regularization term to train the model and can further enhance the ability of discriminating boundaries.

**Table IV**

| Architecture | Attention | F1@{10,25,50} | Edit Acc | GPU |
|--------------|-----------|----------------|----------|-----|
| Full         |           | 4.6 2.8 1.4 3.3 62.8 18.7G |          |     |
| Log          |           | 56.2 51.7 41.2 45.3 69.0 18.7G |          |     |
| U-Trans      |           | 74.6 72.2 63.1 64.8 81.0 4.6G |          |     |
| Local        |           | 35.1 25.4 9.8 31.9 43.8 9.9G |          |     |
|              | Log       | 73.3 71.9 63.7 65.1 80.3 9.9G |          |     |
|              | Local     | 86.5 85.3 76.9 80.6 84.4 2.8G |          |     |

VI. CONCLUSION

This work was supported in part by the National Natural Science Foundation of China No. 61976206 and No. 61832017, Beijing Outstanding Young Scientist Program NO. BJJWZYJH01201910002098, Beijing Academy of Artificial Intelligence (BAAI), the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China 21XNLG05, and Public Computing Cloud, Renmin University of China.

REFERENCES

[1] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, “Temporal convolutional networks for action segmentation and detection,” in CVPR, 2017, pp. 156–165.

[2] Y. A. Farha and J. Gall, “Ms-icn: Multi-stage temporal convolutional network for action segmentation,” in CVPR, 2019, pp. 3575–3584.

[3] S.-J. Li, Y. AbuFarha, Y. Liu, M.-M. Cheng, and J. Gall, “Ms-icn++: Multi-stage temporal convolutional network for action segmentation,” TPAMI, 2020.

[4] H.-H. Gao, Q. Han, Z.-Y. Li, P. Peng, L. Wang, and M.-M. Cheng, “Global2local: Efficient structure search for video action segmentation,” in CVPR, 2021, pp. 16 805–16 814.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in NIPS, 2017.

[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in ICLR, 2020.

[7] G. Li, D. Xu, X. Cheng, L. Si, and C. Zheng, “Simvit: Exploring a simple vision transformer with sliding windows,” in ICME. IEEE, 2022.

[8] F. Yi, H. Wen, and T. Jiang, “Asformer: Transformer for action segmentation,” in BMVC, 2021.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in CVPR. Ieee, 2009, pp. 248–255.

[10] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev et al., “The kinetics human video action dataset,” arXiv preprint arXiv:1705.06950, 2017.

[11] I. Beltagy, M. E. Peters, and A. Cohan, “Longformer: The long-document transformer,” arXiv preprint arXiv:2004.05150, 2020.

[12] D. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention. Springer, 2015, pp. 234–241.

[13] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio.” SSW. 2016.

[14] P. Lei and S. Todorovic, “Temporal deformable residual networks for action segmentation in videos,” in CVPR, 2018, pp. 6742–6751.

[15] Z. Wang, Z. Gao, L. Wang, Z. Li, and G. Wu, “Boundary-aware cascade networks for temporal action segmentation,” in ECCV, 2020.

[16] Y. Zhang, K. Muandet, Q. Ma, H. Neumann, and S. Tang, “Low-rank random tensor for bilinear pooling,” arXiv, pp. arXiv–1906, 2019.

[17] Y. Huang, Y. Sugano, and Y. Sato, “Improving action segmentation via graph-based temporal reasoning,” in CVPR, 2020, pp. 14 024–14 034.

[18] D. Singhania, R. Rahaman, and A. Yao, “Coarse to fine multi-resolution temporal convolutional network,” arXiv preprint arXiv:2105.10859, 2021.

[19] D. Du, B. Su, and Z. Wei, “Preformer: predictive transformer with multi-scale segment-wise correlations for long-term time series forecasting,” arXiv preprint arXiv:2202.11356, 2022.

[20] W. Qiang, J. Li, B. Su, J. Fu, H. Xiong, and W. Jirong, “Meta-attention-generation network for cross-granularity few-shot learning,” International Journal of Computer Vision, 2023.

[21] Q. Pei, N. Thorne, L. Themry, T. Collins, and L. Solet, “U-net transformer: Self and cross attention for medical image segmentation,” in Machine Learning in Medical Imaging: 12th International Workshop, MLMI 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27, 2021, Proceedings 12. Springer, 2021, pp. 267–276.

[22] Z. Wang, X. Cun, J. Bao, W. Zhou, J. Liu, and H. Li, “Uformer: A general u-shaped transformer for image restoration,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 17 683–17 693.

[23] W. Qiang, J. Li, C. Zheng, B. Su, and H. Xiong, “Interventional contrastive learning with meta semantic regularizer,” in International Conference on Machine Learning. PMLR, 2022, pp. 18 018–18 030.

[24] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in AAAI, 2021.

[25] P. Shaw, J. Uszkoreit, and A. Vaswani, “Self-attention with relative position representations,” arXiv preprint arXiv:1803.02135, 2018.

[26] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance normalization: The missing ingredient for fast stylization,” arXiv preprint arXiv:1607.08022, 2016.

[27] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” arXiv preprint arXiv:1607.06450, 2016.

[28] S. Stein and S. 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, 2013, pp. 729–738.

[29] H. Kuehne, A. Arslan, and T. Serre, “The language of actions: Recovering the syntax and semantics of goal-directed human activities,” in CVPR, 2014, pp. 780–787.

[30] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in CVPR, 2017, pp. 6299–6308.

[31] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in ICLR, 2014.

[32] Y. Ishikawa, S. Kasai, Y. Aoki, and H. Kataoka, “Alleviating over-segmentation errors by detecting action boundaries,” in WACV, 2021, pp. 2322–2331.

[33] W. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and Y. Yan, “Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting,” in NIPS, vol. 32, 2019.