Neural Processing Letters
https://doi.org/10.1007/s11063-022-11133-9

Dilated Transformer with Feature Aggregation Module for Action Segmentation

Zexing Du[1] · Qing Wang[1]

Accepted: 16 December 2022

Abstract
Segmenting human actions in long untrimmed videos is challenging due to the complicated temporal correlations between actions and over-segmentation errors. Although Transformer architectures have advanced correlations exploration for action recognition, they are not designed for action segmentation, which would face heavy computational cost and temporal redundancy. In this paper, we propose a Multi-Stage Dilated Transformer Network (MSDTN) to deal with these challenges. Specifically, we construct Transformer between frames of different time spans to capture short- and long-term relationships in videos. Furthermore, to alleviate over-segmentation errors in action segmentation, we propose to generate more stable and distinguishable features via temporal context aggregation at local scales. Especially, our method, termed as Feature Aggregation Module (FAM), is a general module, and can be integrated into existing architectures seamlessly with negligible overheads for action segmentation. We evaluate our proposed MSDTN and FAM on three challenging datasets (GTEA, 50Salads and Breakfast). Experimental results validate the effectiveness of our method on all three datasets.

Keywords Temporal action segmentation · Dilated Transformer Network · Over-segmentation · Feature Aggregation Module

1 Introduction

Video understanding plays an important role in various applications ranging from surveillance to robotics. Regarding this, one task is to classify actions from well-trimmed videos, which has made a great process recently [1–6]. On the other hand, action segmentation aims at simultaneously segmenting untrimmed videos and predicting the action label for each frame, which has attracted great attention in video surveillance, summarization, and retrieval

✉ Qing Wang
qwang@nwpu.edu.cn

Zexing Du
duzexing@mail.nwpu.edu.cn

1 School of Computer Science, Northwestern Polytechnical University, No. 127, You Yi Xi Road, Xi'an 710072, Shaanxi, China

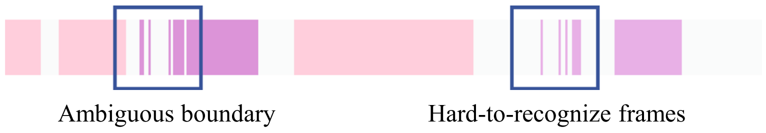


Fig. 1 The over-segmentation errors. Different colors represent different segmented actions. We highlight the over-segmentation errors by solid rectangles. (Color figure online)

[7–10]. Extant researches on action segmentation mainly contain two phases: extracting spatio-temporal features and establishing correlations between these extracted features and classifying them. The first one usually utilizes action recognition networks and has been well explored [1, 2, 4, 6, 11, 12]. For temporal relationship exploration, most approaches [13–24] adapt recurrent, convolutional or graph neural networks (RNN, CNN, GNN) to cope with this problem. Although these methods have achieved good performance, there are several critical challenges in correlations exploration: (1) for RNN, video frames are processed sequentially. (2) For CNN, relationships between frames are content-agnostic. (3) For GNN, graphs are fixed or noisy.

Self-attention-based architectures, in particular Transformers [25], have become the model of choice in natural language processing (NLP), which seems to be an appropriate solution for these challenges. Recently, some works [2, 26] adopted Transformer architectures for video classification. However, they are designed for trimmed videos which only contain several sampled frames. In addition, unlike highly semantic and information-dense languages, videos, on the contrary, are natural signals with *heavy spatial and temporal redundancy*. Applying Transformers directly for long-term untrimmed videos, which usually contain tens of thousands frames, would face challenges of heavy computation, making it hard to build relationships between frames. Recently, ASFormer [27] utilize a pre-defined hierarchical representation pattern for long-term video sequences. However, it is more like incorporating additional attention models into a convolution-based architecture (i.e., MSTCN [20]), which does not break through the limitations of convolution. And the pre-defined pattern restricts its generalization ability to different actions.

Therefore, in this paper, instead of constructing self-attention based on global Transformers without any modification, we propose a Dilated Transformer Network (DTN) to exploit temporal relationships in videos. In detail, we apply multi-layer dilated Transformers between frames of different time spans to efficiently and effectively explore the local and global temporal dependencies. Unlike RNN, CNN and GNN, the proposed DTN can better handle aforementioned challenges: (1) parallelly processing all frames, instead of sequentially; (2) judiciously dealing with every frame, rather than content-agnostic; (3) building the relationship flexibly. And DTN can also resolve the computational challenges and information redundancy faced by global Transformers. Additionally, inspired by [20], we further refine the predictions in a multi-stage way, which we call Multi-Stage DTN (MSDTN).

Another major problem for action segmentation lies in how to alleviate over-segmentation errors. This problem turns out to be very challenging for heretofore state-of-the-art action segmentation models. Generally, as illustrated in Fig. 1, the over-segmentation errors are mainly caused by: hard-to-recognize frames within actions, and ambiguous boundaries. To address these challenges, the MSTCN [20] is proposed to capture the correlations between frames and refine the detection results stage-by-stage. The loss introduced in [20] penalizes over-segmentation errors by forcing the consecutive frames to have the same label. Recently, there are also some attempts trying to cope with this problem by detecting action boundaries [22, 28]. Although excellent performance has been achieved by these methods, there seems to be a tendency in current research to overly focus on pursuing state-of-the-art performance by stacking complicated networks but overlooking the inherent factors that cause over-segmentation.

As analyzed above, previous approaches address the over-segmentation issue based on complicated refining strategies. Instead, in this paper, we depart from classical detect-and-refine strategies by proposing a Feature Aggregation Module (FAM) to aggregate features. In contrast to refining-based approaches, which constrain the mapping between features and labels by *deeper* networks, we propose to generate smoother features via temporal context aggregation at *local* scales. Especially, our method is a general module, and can be integrated into existing architectures seamlessly with negligible overheads for action segmentation. We hope the systematic analysis and experiments could provide insights for the researchers into the intrinsic reasons for over-segmentation errors in action segmentation.

The contributions of this paper are summarized as follows:

1. The MSDTN is proposed for action segmentation to efficiently model the temporal relationships in various timescales.
2. Instead of refining predictions at the later layer of networks, we propose the FAM to generate smoother features via temporal context aggregation at local scales. This allows recovery from disturbances of features unaware temporal context and differentiates the true action boundaries.
3. Our method achieves comparable performance compared to state-of-the-art methods on the task of temporal action segmentation on three challenging datasets, namely, 50Salads [29], Breakfast [17], and Georgia Tech Egocentric Activities (GTEA) [30] datasets.

2 Related Work

2.1 Action Segmentation

Action segmentation has received a lot of attention from computer vision community. In earlier approaches, different scale sliding windows were applied with non-maximum suppression to localize action segments and assign labels to video frames [13–15]. Fathi et al. [31] modeled activities based on the change of objects and materials in the environment. Cheng et al. [32] studied the temporal correlations by employing a Bayesian non-parametric model to jointly classify and segment video sequence. Some methods applied a coarse temporal modeling on top of frame-level classifiers [16–19, 33–38]. Singh et al. [19] used a two-stream network and bi-directional LSTM to capture the dependencies of different chunks. However, such approaches are computationally expensive and cannot be applied to long videos.

Inspired by the success of temporal convolution in speech synthesis [39], researchers have tried to use similar ideas for action segmentation. Lea et al. [40] proposed the TCN for action segmentation and detection. Their approach utilized two encoder-decoder and dilated architectures. Lei et al. [41] used the deformable convolution on the top of TCN and added a residual stream to the encoder-decoder model. Farha et al. [20] and Li et al. [21] further enhanced the dilated TCN in a multi-stage manner, to better catch the long-range temporal dependencies in videos. Wang et al. [22] introduced the BCN to cope with the boundary ambiguity and over-segmentation issues in action segmentation. Gao et al. [42] designed a global-to-local search scheme to find better receptive field combinations. Ishikawa et al. [43] used the ASRF to classify video frames and regress action boundary probabilities. Ahn et al. [44] refined temporal action segmentation results from various models by understanding the overall context of a given video in a hierarchical way. Although these approaches have achieved great success in action segmentation, they face challenges of content-agnostic relationships, which ignore the difference between frames.

Our work is also related to previous efforts such as GTRM [23] and DTGRM [24] that explore relationships using graph convolutions. These methods adopt a procedure: applying a GNN to capture interactions, which is analogous to our Transformer. In detail, Huang et al. [23] built the GNN based on action segments to learn the boundary regression and classification tasks. However, the pre-computed segments are mostly inaccurate and the constructed graphs are noisy. Wang et al. [24] captured temporal relationships via constructing similarity and learned graphs. Nevertheless, wrong predictions make the weights of similarity graph inaccurate and the learned convolutions are fixed during inference, which is not flexible enough to model the complicated correlations.

2.2 Self-Attention Transformers

Self-attention-based architecture, in particular Transformers [25] has revolutionized NLP. Recently, with the development of Transformers in computer vision [45, 46], it has made a great process in video understanding [2, 26]. However, these methods are designed for action recognition and only sample several frames from trimmed videos. For action segmentation, which needs to predict action labels for every frame, directly applying these methods would result in high computational costs. There are also some attempts [47–49] trying to handle long sequences and relationship patterns in the sequences with efficiency considered. Nevertheless, these approaches are designed for highly semantic and information-dense languages. Videos, on the contrary, are signals with heavy spatial and temporal redundancy, and applying Transformers for action segmentation in a frame-wise manner is information redundant. More recently, Yi et al. [27] proposed ASFormer for action segmentation, which applied a per-defined hierarchical representation pattern to deal with long video sequences. Nevertheless, it is more like incorporating additional attention models into MSTCN [20], which does not break through the limitations of convolution. As a result, to correctly grasp the temporal relationships, we propose the MSDTN network to model the temporal dependencies.

3 Multi-stage Dilated Transformer Network

We introduce the multi-stage DTN for temporal action segmentation. Given the frames of a video $x_{1:T} = (x_1, \ldots, x_T)$, the goal is to infer the class label for each frame $c_{1:T} = (c_1, \ldots, c_T)$, where T is the video length. In the rest of this section, we will first give an overview of the proposed architecture. Then details of MSDTN will be discussed. At last, the loss function of the network will be explained.

3.1 Overall Architecture

An overview of the proposed architecture is presented in Fig. 2. A linear embedding layer is applied on the input features to project it to an arbitrary dimension (denote as C), which is the input to our FAM. FAM would generate stable features for MSDTN, and more details of FAM will be introduced in Sect. 4. Inspired by the dilated convolution operation in CNN, in this paper, we propose a Dilated Transformer Block (DTB) to replace the global attention operation. Dilated operation can efficiently increase the receptive field without expensive

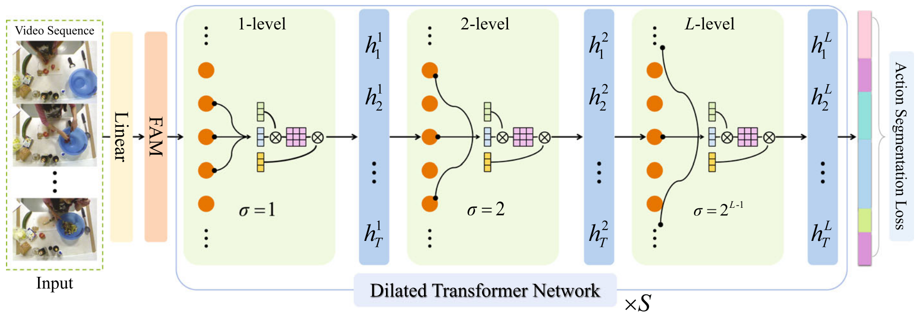


Fig. 2 The pipeline of our proposed MSDTN and FAM. In DTN, we stack L DTBs with dilated rate doubled at each level. The outputs of the L-th level are used to calculate the action segmentation loss. Then the predicted results are refined in a multi-stage way. FAM represents the proposed Feature Aggregation Module and more details are provided in Sect. 4

burden of computation. To explore short- and long-term relationships between actions, we stack L DTBs with dilation rate doubled at each layer. All these layers have the same number of feature dimension. At the L-th layer, the outputs are used to calculate losses. In addition, thanks to the success of multi-stage operation in action segmentation, we also build the DTN S times to iteratively correct the prediction results.

3.2 Dilated Transformer Network

3.2.1 Global Transformers

Global Transformers can be denoted as [25]

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{C}}\right)V, \tag{1}$$

where query, key and value tensors $(Q, K, V) \in \mathbb{R}^{T\times C}$ are generated by inputs. Specifically, the output at time t is

$$\text{Attention}(q_t, K, V) = \sum_{i=1}^{T} \frac{1}{Z}\exp\left(\frac{q_t k_i^\top}{\sqrt{C}}\right) v_i, \tag{2}$$

where T is the length of sequence. Z is the normalization factor, representing the summation over the exponentials of the different products between the query and the key embeddings, and q_t is the query at time t. Global Transformers build the attention based on the entire sequence, which is inefficient and computationally redundant, especially when the sequence is long. Therefore, we propose the DTB to tackle this problem.

3.2.2 Dilated Transformer Block

We present the DTB to explore the temporal relationships of different time spans. Specifically, for the frame at time t, the query, key and value tensors are obtained by

$$\begin{aligned} Q_t' &= W_q \cdot (h_{t-n\sigma}, \ \cdots, h_t, \ \cdots, h_{t+n\sigma}), \\ K_t' &= W_k \cdot (h_{t-n\sigma}, \ \cdots, h_t, \ \cdots, h_{t+n\sigma}), \\ V_t' &= W_v \cdot (h_{t-n\sigma}, \ \cdots, h_t, \ \cdots, h_{t+n\sigma}), \end{aligned} \tag{3}$$

where $(W_q, W_k, W_v) \in \mathbb{R}^{(2n+1)\times(2n+1)}$ are the learnable weights for three different linear projections, $(Q_t', K_t', V_t') \in \mathbb{R}^{(2n+1)\times C}$, σ is the dilated rate, h_t is the feature at time t. The dilated kernel size is $(2n+1)$, which means only $(2n+1)$ timestamps are considered when building the attention. Then the operation of the DTB at time t is

$$\text{DTB}_t = \sum_{i=t-n\sigma}^{t+n\sigma} \frac{1}{Z'} \exp\left(\frac{q_t' k_i'^{\top}}{\sqrt{C}}\right) v_i'. \tag{4}$$

We can see that the DTB has larger receptive field without increasing the computation. Then, InstanceNorm and ReLU activation are applied after each DTB and we further use residual connections to facilitate gradients flow. We stack L DTBs to exploit the temporal relationships of different time spans with the dilation rate doubled at each layer, i.e., $\{1, 2, 4, 8, \ldots\}$. In addition, we also follow [27] to apply a dilated convolution layer as the feed-forward layer before the self-attention operation. The output of the l-th layer is presented as $h_{1:T}^{(l)}$, which is also the input of $(l+1)$-th layer. Therefore, the set of operations at each layer can be formally described as

$$\begin{aligned} \text{out} &= \text{FeedForward}(h^{(l)}), \\ h^{(l+1)} &= \text{InstanceNorm}(\text{DTB}(\text{out})) + \text{out}. \end{aligned} \tag{5}$$

It is noted that we omit the subscript $(1:T)$ for simplicity. $h^{(0)}$ is the input tensor at the first layer. After stacking DTB for L times, our DTN can efficiently capture short- and long-term temporal relationships in videos.

At L-th level, to get the action class likelihoods, we construct a convolutional layer followed by a softmax activation over $h^{(L)}$, i.e.,

$$y = \text{softmax}(Wh^{(L)} + b), \tag{6}$$

where $W \in \mathbb{R}^{C\times K}$ and $b \in \mathbb{R}^K$ are learned weights and bias, K is the number of classes.

3.3 Multi-stage Architecture

Furthermore, stacking several predictors sequentially has shown significant boosts in action segmentation. In this paper, we stack DTN for S times to improve the output results. The structure of MSDTN is illustrated in Fig. 3. The frame-wise probabilities of s stage are the input of $s+1$ stage. At the first stage, the query, key and value vector tensors are calculated by Eq. (3). After the first stage, the value V' is computed by the output of previous stage, i.e., the value vector at the stage s is based on the output of stage $s-1$. Under this circumstance, the feature V' is completely transformed from the output of previous stage, and will not disturbed with the influence of previous layers [27]. Such design results in more stable predictions, because the V' would not be disturbed by the varied receptive field of DTB. Prediction results are iteratively refined by MSDTN.

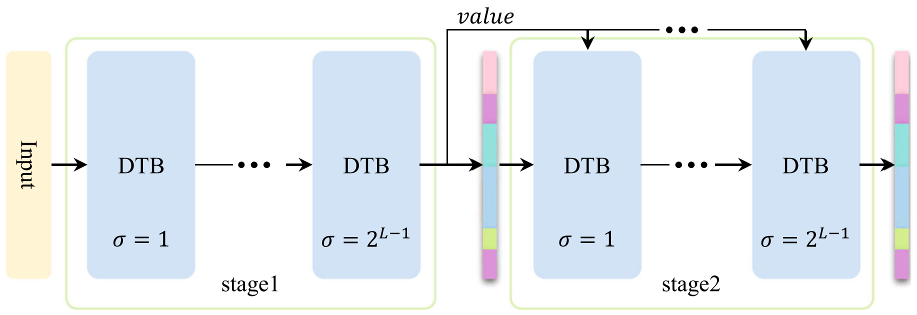


Fig. 3 The structure of MSDTN. The predicted results are refined by the proposed DTN in a multi-stage way. After the first stage, the value vector at stage s is calculated based on the output of stage $s-1$. We only show two stages for simplicity

3.4 Computational Complexity

Supposing the kernel size is $(2n+1)$, at each layer, the computational complexity of global Transformers and DTB on a video of T frames is

$$\begin{aligned} \Omega(\text{global}) &= 3TC^2 + 2T^2C, \\ \Omega(\text{DTB}) &= 3TC^2 + 2T(2n+1)^2C, \end{aligned} \tag{7}$$

where the global Transformers are quadratic to sequence length T, whereas DTB is linear to T. Note that $T \gg (2n+1)^2$ and we omit SoftMax computation in determining complexity. Therefore, global self-attention computation is generally unaffordable for large T, while our DTB is much more affordable. It is worth noting that the convolution computation is also linear to T, which is the same as our DTB.

3.5 Loss Function

We train our MSDTN in an end-to-end manner. The loss function in training is a combination of multiple parts, denoted as

$$\begin{aligned} \mathcal{L}_s &= \mathcal{L}_{seg} + \lambda \mathcal{L}_{t-mse}, \\ \mathcal{L}_{seg} &= \frac{1}{T}\sum_{t=1}^{T} -log(y_{t,c}), \end{aligned} \tag{8}$$

where $y_{t,c}$ is the predicted probability for the ground truth label c_t, $\mathcal{L}_{seg}$ is the action segmentation loss, $\mathcal{L}_{t-mse}$ is the truncated mean squared error, which is proposed in [20]. λ is the hyper-parameter to determine the contributions of different losses. Following [20], we set $\lambda = 0.15$. Moreover, since our DTN has S stages, the total loss $\mathcal{L}_{total}$ of MSDTN is

$$\mathcal{L}_{total} = \sum_{s=1}^{S} \mathcal{L}_s. \tag{9}$$

4 Feature Aggregation Module

Besides establishing complicated temporal correlations between actions, another major challenge for action segmentation lies in the over-segmentation errors. To alleviate this problem, an examination of the factors that cause over-segmentation errors is described in this section. Figuring out the inherent reasons on the feature level plays an important role in guiding the design of our approach. Then, we would show the details of our proposed FAM.

4.1 Challenges of Over-Segmentation

There usually exists a distinct difference between a transition event from action A to action B occurring at action boundaries, along with the change of frame-wise action label. However, because of the local consecutiveness of videos, the features would not always show significant changes immediately near boundaries. Therefore, sudden labels changes with gradual transitions of features makes the network hard to correctly classify the frames near boundaries. Fluctuating predictions will be continually generated when convolutions slide over ambiguous boundaries.

Another reason for over-segmentation arises from the hard-to-recognize frames occurring within actions, which lead to inconsistency between features and action labels. For example, networks may be confounded by many reasons such as changing viewpoint, illumination and action pause.

4.2 FAM

To deal with aforementioned challenges, we propose the FAM to dilute the abnormality of these frames. Formally, given an input video $x_{1:T} = (x_1, \dots, x_T)$, FAM can be formulated as

$$\hat{x}_t = \frac{1}{2\theta + 1} \sum_{i=t-\theta}^{t+\theta} x_i, \tag{10}$$

where $(2\theta + 1)$ is the size of the local window. The networks in original approaches [20] are convolved with individual frames, i.e., $\{x_{t-1}, x_t, x_{t+1}\}$. Nevertheless, in our method, the weights are convolved by groups of frames. Then, the weights would be trained using rich context features instead of individual ones, which would mitigate the issue of ambiguous boundaries. In addition, We can obtain context-aware and consistent features by FAM, diluting the effect of hard-to-recognize frames. FAM is an extremely light-weight plug-and-play block, and can be directly applied into action segmentation networks.

5 Datasets and Evaluation Metrics

5.1 Datasets

To evaluate the performance of the proposed method, we conduct experiments on three challenging datasets, including 50Salads [29], Georgia Tech Egocentric Activities (GTEA) [30] and the Breakfast dataset [17]. Some sample images are shown in Fig. 4. Details of these three datasets can be seen in Table 1. We can see that Breakfast is the dataset that contains

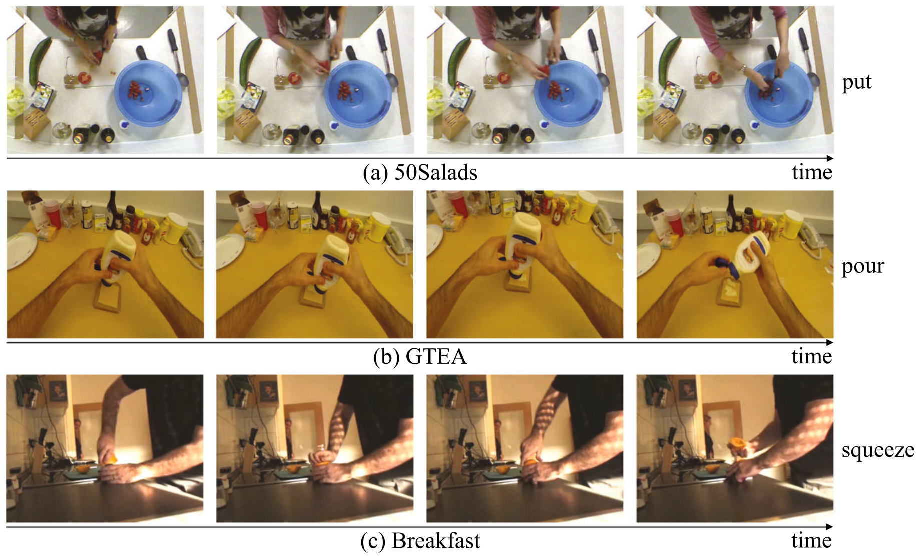


Fig. 4 Sample image categories from the 50Salads, GTEA, and Breakfast datasets

Table 1 Details of three dataset

	#Vid	#Cls	#Ins	#Frame	Cross-validation
50Salads [29]	50	17	20	11,552	5
GTEA [30]	28	11	20	1115	4
Breakfast [17]	1712	48	6	2097	4

#Vid and #Cls are the total number of videos and classes respectively. #Ins is the average instances of videos. #Frame is the average frames of videos

the most videos. Therefore, we perform our ablations mainly on Breakfast if not otherwise stated. For evaluation, we perform fivefold cross-validation for 50Salads and fourfold cross-validation for GTEA and Breakfast datasets. The features we used are extracted by I3D [1] as previous methods.

5.2 Evaluation Metrics

To report the performance of our method, we employ the same evaluation metrics as [20–22, 24, 50], which include frame-wise accuracy (Acc), segmental edit score (Edit) and segmental F1 score at overlapping thresholds 10%, 25% and 50%. Frame-wise accuracy is the most widely used metric for action segmentation, which reports the average accuracy of every frame. However, it has little effect on the over-segmentation errors. Therefore, edit and F1 scores are also used as measures of the prediction quality.

Table 2 The F1@10 score and frame-wise accuracy comparison between MSTCN, ASFormer and our method

Method	F1@10	Acc	Params (M)	FLOPs (G)	Run-time (h)
MSTCN [20]	52.6	66.3	0.799	1.671	2.7
ASFormer [27]	76.0	73.5	1.134	2.410	57.3
MSDTN	78.2	74.4	1.048	2.198	3.3

We also show the number of parameters, FLOPs and run-time. Experiments are performed on Breakfast dataset. The run-time is measured on a single RTX 2080 Ti GPU with batch size = 1 in PyTorch evaluation mode. The FLOPs is for one sample rate with frames = 2097, which is the average number of frames for Breakfast dataset. The run-time is the time of training for 50 epochs on Breakfast of split 1

6 Experiments

6.1 Study on MSDTN

In this section, we validate the ability of MSDTN by comparing it to MSTCN [20] and ASFormer [27], where the former is the most widely used convolution-based architecture, and the latter is based on the Transformer. Experimental results are shown in Table 2. It is easily observed that MSDTN achieves the best performance compared to MSTCN and ASFormer. Especially, MSDTN improves the MSTCN from 52.6 to 78.2% on F1@10 score, and also achieves 8.1% improvements on frame-wise accuracy. This is because capturing both local and global temporal features makes it easier to recognize simple-to-hard frames of the action segments. Importantly, MSDTN gets these improvement with affordable overheads, including model parameters, FLOPs and run-time. Compared to ASFormer, MSDTN achieves better performance regardless of evaluation metrics (e.g., 2.2% on F1@10 and 0.9% on frame-wise accuracy). Moreover, MSDTN greatly reduces the training and inference time compared to ASFormer. We conjecture it is because ASFormer pays too much time on the pre-defined representation pattern, which is replaced by the efficient dilated operation in MSDTN.

6.2 Study on FAM

In this section, we compare the performance of several networks (i.e., MSTCN [20], MSTCN++ [21], DTGRM [24] and our MSDTN) with and without FAM applied in Table 3 . Thanks to the model-agnostic of our proposed method, the training recipes, i.e., optimizer, learning rate, batch size, feature dimension, the number of stages and layers, and any other hyper-parameters all keep the same as its duplicates for a fair comparison. Experiments are performed on Breakfast dataset.

From the experimental results, we can see these methods yield significant improvements over their baselines. Especially, FAM achieves 6.9% gains on edit and 16.7% boosts on F1@10 when employed on MSTCN. DTGRM gets a higher edit score by 2.9%. Moreover, MSTCN++ achieves 6.0% improvements the F1@10 score. All these gains on F1 and edit demonstrate that over-segmentation errors are further alleviated when applied with FAM. In addition, superior results on frame-wise accuracy indicate that our method can model the video correlations more comprehensively and correctly. These results verify that our FAM does work when cooperating with refining-based approaches.

Table 3 Cooperating with networks on Breakfast dataset

Method	F1@{10,25,50}			Edit	Acc
MSTCN [20]	52.6	48.1	37.9	61.7	66.3
FAM-MSTCN	**69.3**	**63.3**	**50.3**	**68.6**	**69.0**
MSTCN++ [21]	64.1	58.6	45.9	65.6	67.6
FAM-MSTCN++	**70.1**	**63.8**	**50.5**	**69.3**	**68.9**
DTGRM [24]	68.7	61.9	46.6	68.9	68.3
FAM-DTGRM	**72.8**	**65.7**	**50.3**	**71.8**	**69.9**
MSDTN	78.2	72.6	59.9	76.8	74.4
FAM-MSDTN	**78.5**	**72.9**	**60.2**	**77.5**	**74.8**

FAM- means applying FAM on the original networks
Bold values indicate the best performance

Table 4 Ablation studies of MSDTN, including the number of stages (S), the number of layers (L), and the kernel size ($2n + 1$)

		F1@{10,25,50}			Edit	Acc
Effect of S	2	70.4	65.1	52.1	70.0	73.8
	3	77.6	72.2	58.8	76.3	74.5
	4	78.2	72.6	59.9	76.8	74.4
	5	78.8	73.2	59.5	77.6	73.8
Effect of L	7	77.8	72.6	59.0	76.0	74.0
	8	77.9	72.8	59.5	76.4	74.0
	9	78.0	72.5	59.8	76.7	74.2
	10	78.2	72.6	59.9	76.8	74.4
	11	78.3	73.0	59.8	77.0	74.1
Effect of $2n + 1$	3 (MSTCN)	52.6	48.1	37.9	61.7	66.3
	3	77.2	71.4	58.2	76.3	73.0
	5	78.4	72.6	59.2	76.9	73.9
	7	78.4	72.6	58.8	77.1	74.2
	9	78.2	72.6	59.9	76.8	74.4
	11	78.0	72.1	58.9	77.0	74.1

All experiments are conducted on Breakfast dataset. We can see that our method has wide tolerance intervals for all hyper-parameters

6.3 Ablation Study

This section will demonstrate that our method has wide tolerance intervals for hyper-parameters, including the number of stages (S), the number of layers (L), the kernel size ($2n + 1$) in MSDTN, and the window size ($2\theta + 1$) in FAM.

6.3.1 Effect of the Number of Stages

We start our ablation analysis by exploring the effect of multi-stage architecture. Table 4 (upper) shows the effect of stages S. As shown in the table, all models can achieve a comparable frame-wise accuracy. However, the F1 and edit scores have some gaps between different stages, indicating over-segmentation errors are alleviated with the increase of stages. Especially, all metrics have significant boosts when the number of stages increases to 4. However,

Table 5 Metrics as functions of window size $(2\theta + 1)$ on Breakfast, where the model we used is FAM-MSDTN

		F1@{10,25,50}			Edit	Acc
Effect of $2\theta + 1$	1	78.2	72.6	59.9	76.8	74.4
	3	78.5	72.9	59.7	77.4	74.2
	5	78.5	72.9	60.2	77.5	74.8
	7	78.4	72.6	59.8	76.9	74.1

$2\theta + 1 = 1$ represents the MSDTN without FAM

adding the fifth stage only improves the F1@10, F1@25 and edit scores, and decrease happens on other metrics. This might be an over-fitting problem as a result of increasing the number of parameters. Therefore, unless pointed out, we set the number of stages as four in this paper.

6.3.2 Effect of the Number of Layers

In Table 4 (middle), we investigate the effect of layers L. Increasing L to 10 can greatly reduce over-segmentation errors and improve frame-wise accuracy. This is mainly because the increase in the receptive field makes the network can explore more global correlations. However, the performance has no more improvement when $L > 10$, which is also caused by over-fitting problem and keeps consistency with the analysis in Sect. 6.3.1. As a result, we set the number of layers as ten.

6.3.3 Effect of the Kernel Size in MSDTN

Furthermore, we investigate the effect of kernel size $(2n + 1)$. From the results shown in Table 4 (lower), we can see that better performance can be obtained with the increase of kernel size, which indicates more information is taken into consideration when constructing temporal relationships. Especially, when compared with MSTCN, which adapts temporal convolution with kernel size as 3, our MSDTN achieves improvements of 24.6% on F1@10 and 6.7% on frame-wise accuracy with the same kernel size, indicating our MSDTN can improve the quality of temporal relationship exploration. However, when the kernel size increases to 11, negligible or no increase on all three metrics suggests that the benefits of kernel size may have plateaued. Accordingly, we set dilation kernel size as nine, which is the sweet point in the table.

6.3.4 Effect of the Window Size in FAM

FAM aims at alleviating over-segmentation errors in action segmentation. In this section, we will explore the effect of window size $(2\theta + 1)$ in Eq. (10). As depicted in Table 5, a larger window size always benefits all metrics, indicating over-segmentation errors are further alleviated when taking more frames for aggregation. A small window can not provide enough context information to recognize hard-to-recognize frames and ambiguous boundaries. Moreover, we also observe that the accuracy drops a little when we stretch the window too large. It may be because the features of short-term actions are diluted when they are involved in a longer window. In addition, too large windows would introduce unnecessary information and then result in unexpected interruptions in long-term actions.

Table 6 Compared with state-of-the-art on 50Salads and GTEA datasets

	50Salads					GTEA				
	F1@{10,25,50}			Edit	Acc	F1@{10,25,50}			Edit	Acc
IDT-LM [33]	44.4	38.9	27.8	45.8	48.7	–	–	–	–	–
Bi-LSTM [19]	62.6	58.3	47.0	55.6	55.7	66.5	59.0	43.6	–	55.5
Dilated TCN [40]	52.2	47.6	37.4	43.1	59.3	58.8	52.2	42.2	–	58.3
ST-CNN [18]	55.9	49.6	37.1	45.9	59.4	58.7	54.4	41.9	–	60.6
ED-TCN [40]	68.0	63.9	52.6	64.7	–	72.2	69.3	56.0	–	64.0
TDRN [41]	72.9	68.5	57.2	66.0	68.1	79.2	74.4	62.7	74.1	70.1
MSTCN [20]	76.3	74.0	64.5	67.9	80.7	85.8	83.4	69.8	79.0	76.3
MSTCN++ [21]	80.7	78.5	70.1	74.3	83.7	88.8	85.7	76.0	83.5	80.1
DTGRM [24]	79.1	75.9	66.1	72.0	80.0	87.8	86.6	72.9	83.0	77.6
BCN [22]	82.3	81.3	74.0	74.3	84.4	88.5	87.1	77.3	84.4	79.8
Global2Local [42]	80.3	78.0	69.8	73.4	82.2	89.9	87.3	75.8	84.6	78.5
ASRF [28]	84.9	83.5	77.3	79.3	84.5	89.4	87.8	79.8	83.7	77.3
ASRF + HASR [51]	86.6	85.7	78.5	81.0	83.9	89.2	87.2	74.8	84.5	76.9
ASFormer [27]	85.1	83.4	76.0	79.6	85.6	90.1	88.8	79.2	84.6	79.7
ASFormer + †Br-Prompt [49]	89.2	87.8	81.3	83.8	88.1	94.1	92.0	83.0	91.6	81.2
ASFormer + ‡MCFM [38]	90.6	89.5	84.2	84.6	90.3	91.8	91.2	80.8	88.0	80.5
MSDTN	85.9	84.1	75.9	80.0	86.2	91.7	90.1	79.2	84.6	80.1
FAM-MSDTN	86.2	84.4	77.9	79.9	86.4	91.6	90.9	80.9	88.3	80.7

The ndash indicates "not reported". †Trained with integrated text prompts for supervision. ‡Utilized additional hand features during training and inference

Table 7 Compared with state-of-the-art on Breakfast dataset

	F1@{10,25,50}			Edit	Acc
MSTCN [20]	52.6	48.1	37.9	61.7	66.3
MSTCN++ [21]	64.1	58.6	45.9	65.6	67.6
DTGRM [24]	68.7	61.9	46.6	68.9	68.3
BCN [22]	68.7	65.5	55.0	66.2	70.4
Global2Local [42]	74.9	69.0	55.2	73.3	70.7
ASRF [28]	74.3	68.9	56.1	72.4	67.6
ASRF + HASR [51]	74.7	69.5	57.0	71.9	69.4
ASFormer [27]	76.0	70.6	57.4	75.0	73.5
MSDTN	<u>78.2</u>	<u>72.6</u>	<u>59.9</u>	<u>76.8</u>	<u>74.4</u>
FAM-MSDTN	**78.5**	**72.9**	**60.2**	**77.5**	**74.8**

Bold values indicate the best performance and underlined ones are the second best

6.4 Compared with State-of-the-Art

In the section, we compare several state-of-the-art approaches with our method on three datasets. The results on three datasets are shown in Tables 6and 7, from which we can see that our model could achieve comparable performance with respect to all three evaluation metrics. Specifically, on Breakfast, which is the largest one, our method achieves 25.9%

Table 8 Experimental results on MPII Cooking 2 dataset

Method	F1@{10,25,50}			Edit	Acc
MSTCN [20]	47.1	40.0	27.1	54.8	52.9
ASFormer [27]	47.3	40.2	27.4	55.2	54.0
FAM+MSDTN	**49.0**	**42.1**	**28.1**	**57.0**	**56.5**

Bold values indicate the best performance

increase on F1@10 and 15.8% gains on edit score compared to MSTCN. On GTEA, when using the same input features and supervision, our method significantly outperforms other methods by a large margin. On 50Salads, our network is superior on all metrics. The MSDTN structure and FAM improve the MSTCN model from 76.3 to 86.2% on F1@10 score, and also achieve 5.7% improvements on frame-wise accuracy. Moreover, when compared to the recently proposed ASFormer, we still get better performance on all three metrics. The results further demonstrate that, compared with previous methods, our MSDTN and FAM have a stronger ability for temporal relationship exploration.

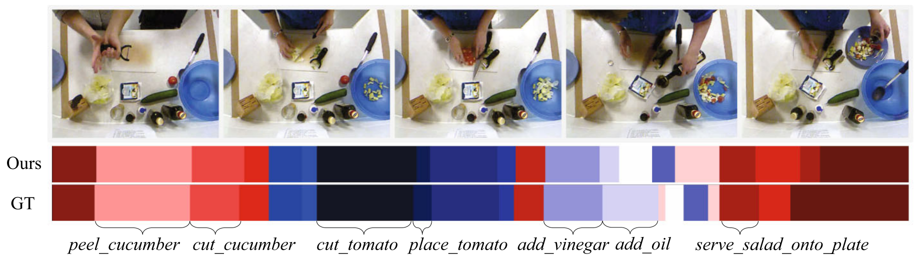
Furthermore, we also conduct more experiments on MPII Cooking 2 dataset [52], which contains more activity classes (67) and longer videos (more than 10,000 frames per video). As no action segmentation approach is conducted on this dataset, we reproduce MSTCN [20] and ASFormer [27] on MPII Cooking 2 dataset with publicly available source code. All training recipes keep the same for a fair comparison. Experimental results are shown in Table 8. It is observed that our method achieves the best performance with respect to all three evaluation metrics.

6.5 Qualitative Results

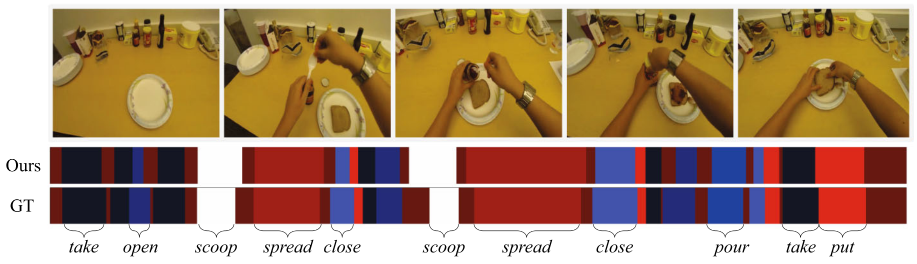
Qualitative results on three datasets are shown in Fig. 5, visualizing that our predictions have high accuracy on action segmentation. Specifically, although there are some mistakes at boundaries, the proposed model can model the correlations of actions correctly, which is benefiting from our MSDTN. Moreover, contribute to the effectiveness of FAM, over-segmentation errors caused by ambiguous boundaries and hard-to-recognize frames in actions are also alleviated. As reflected in the quantitative and qualitative results, our proposed method achieves outstanding performance in action segmentation.

7 Conclusion

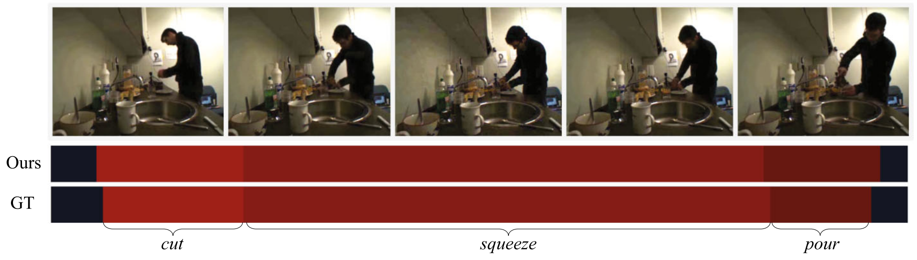
In this paper, we propose the MSDTN to model the temporal relationships for action segmentation. We extend the conventional global Transformers to DTN to excavate temporal relationships in various timescales. Furthermore, an effective and efficient FAM is proposed to generate more stable and distinguishable features via temporal context aggregation at local scales. Evaluation on three datasets demonstrates our method achieves competitive performance compared with state-of-the-art methods. In addition, we hope the insight gleaned from our theoretical analysis and experiments could motivate future algorithm design.



(a) *rgb-24-2* in 50Salads.



(b) *S3_Pealate_C1* in GTEA.



(c) *P29_cam01_P29_juice* in Breakfast.

Fig. 5 Qualitative results of our method with color-coding on 50Salads, GTEA, and Breakfast datasets, where GT represents ground truth. We annotate some key actions in long videos

Acknowledgements This work was supported by NSFC under Grant 62031023.

References

1. Carreira J, Zisserman A (2017) Quo vadis, action recognition? A new model and the kinetics dataset. In: CVPR, pp 6299–6308
2. Arnab A, Dehghani M, Heigold G, Sun C, Lučić M, Schmid C (2021) Vivit: a video vision transformer. In: ICCV, pp 6836–6846
3. Feichtenhofer C, Fan H, Malik J, He K (2019) Slowfast networks for video recognition. In: ICCV, pp 6202–6211
4. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. arXiv:1406.2199
5. Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2016) Temporal segment networks: towards good practices for deep action recognition. In: ECCV, pp 20–36
6. Qiu Z, Yao T, Mei T (2017) Learning spatio-temporal representation with pseudo-3d residual networks. In: ICCV, pp 5533–5541
7. Collins RT, Lipton AJ, Kanade T (2000) Introduction to the special section on video surveillance. TPAMI 22(8):745–746. https://doi.org/10.1109/TPAMI.2000.868676

8. Vishwakarma S, Agrawal A (2013) A survey on activity recognition and behavior understanding in video surveillance. Vis Comput 29(10):983–1009
9. Lee YJ, Ghosh J, Grauman K (2012) Discovering important people and objects for egocentric video summarization. In: CVPR, pp 1346–1353
10. Ma Y-F, Hua X-S, Lu L, Zhang H-J (2005) A generic framework of user attention model and its application in video summarization. TMM 7(5):907–919
11. Feichtenhofer C, Pinz A, Wildes RP (2017) Spatiotemporal multiplier networks for video action recognition. In: CVPR, pp 4768–4777
12. Wang X, Girshick R, Gupta A, He K (2018) Non-local neural networks. In: CVPR, pp 7794–7803
13. Rohrbach M, Amin S, Andriluka M, Schiele B (2012) A database for fine grained activity detection of cooking activities. In: CVPR, pp 1194–1201
14. Karaman S, Seidenari L, Del Bimbo A (2014) Fast saliency based pooling of fisher encoded dense trajectories. In: ECCV THUMOS workshop, p 5
15. Oneata D, Verbeek J, Schmid C (2014) The lear submission at thumos 2014. In: ECCV THUMOS challenge
16. Kuehne H, Gall J, Serre T (2016) An end-to-end generative framework for video segmentation and recognition. In: WACV, pp 1–8
17. Kuehne H, Arslan A, Serre T (2014) The language of actions: recovering the syntax and semantics of goal-directed human activities. In: CVPR, pp 780–787
18. Lea C, Reiter A, Vidal R, Hager GD (2016) Segmental spatiotemporal cnns for fine-grained action segmentation. In: ECCV, pp 36–52
19. Singh B, Marks TK, Jones M, Tuzel O, Shao M (2016) A multi-stream bi-directional recurrent neural network for fine-grained action detection. In: CVPR, pp 1961–1970
20. Farha YA, Gall J (2019) Ms-tcn: multi-stage temporal convolutional network for action segmentation. In: CVPR, pp 3575–3584
21. Li S-J, AbuFarha Y, Liu Y, Cheng M-M, Gall J (2020) Ms-tcn++: multi-stage temporal convolutional network for action segmentation. TPAMI
22. Wang Z, Gao Z, Wang L, Li Z, Wu G (2020) Boundary-aware cascade networks for temporal action segmentation. In: ECCV, pp 34–51
23. Huang Y, Sugano Y, Sato Y (2020) Improving action segmentation via graph-based temporal reasoning. In: CVPR
24. Wang D, Hu D, Li X, Dou D (2021) Temporal relational modeling with self-supervision for action segmentation. In: AAAI, vol 35, pp 2729–2737
25. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: NeurIPS
26. Bertasius G, Wang H, Torresani L (2021) Is space-time attention all you need for video understanding? In: ICML
27. Yi F, Wen H, Jiang T (2021) Asformer: transformer for action segmentation. arXiv:2110.08568
28. Ishikawa Y, Kasai S, Aoki Y, Kataoka H (2021) Alleviating over-segmentation errors by detecting action boundaries. In: WACV, pp 2322–2331
29. Stein S, McKenna SJ (2013) Combining embedded accelerometers with computer vision for recognizing food preparation activities. In: UbiComp, pp 729–738
30. Fathi A, Ren X, Rehg JM (2011) Learning to recognize objects in egocentric activities. In: CVPR, pp 3281–3288
31. Fathi A, Rehg JM (2013) Modeling actions through state changes. In: CVPR, pp 2579–2586
32. Cheng Y, Fan Q, Pankanti S, Choudhary A (2014) Temporal sequence modeling for video event detection. In: CVPR, pp 2227–2234
33. Richard A, Gall J (2016) Temporal action detection using a statistical language model. In: CVPR, pp 3131–3140
34. Ding L, Xu C (2018) Weakly-supervised action segmentation with iterative soft boundary assignment. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6508–6516
35. Richard A, Kuehne H, Gall J (2017) Weakly supervised action learning with RNN based fine-to-coarse modeling. In: CVPR, pp 754–763
36. Fayyaz M, Gall J (2020) Sct: set constrained temporal transformer for set supervised action segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 501–510
37. Zhang J, Cao Y, Wu Q (2021) Vector of locally and adaptively aggregated descriptors for image feature representation. Pattern Recogn 116:107952
38. Ishihara K, Nakano G, Inoshita T (2022) Mcfm: Mutual cross fusion module for intermediate fusion-based action segmentation. In: 2022 IEEE international conference on image processing (ICIP). IEEE, pp 1701–1705

39. van den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu K (2016) Wavenet: A generative model for raw audio. In: 9th ISCA Speech Synthesis Workshop
40. Lea C, Flynn MD, Vidal R, Reiter A, Hager GD (2017) Temporal convolutional networks for action segmentation and detection. In: CVPR, pp 156–165
41. Lei P, Todorovic S (2018) Temporal deformable residual networks for action segmentation in videos. In: CVPR, pp 6742–6751
42. Gao S-H, Han Q, Li Z-Y, Peng P, Wang L, Cheng M-M (2021) Global2local: efficient structure search for video action segmentation. In: CVPR
43. Ishikawa Y, Kasai S, Aoki Y, Kataoka H (2021) Alleviating over-segmentation errors by detecting action boundaries. In: WACV, pp 2322–2331
44. Ahn H, Lee D (2021) Refining action segmentation with hierarchical video representations. In: ICCV, pp 16302–16310
45. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N (2021) An image is worth 16x16 words: transformers for image recognition at scale. In: ICLR
46. Yu J, Li J, Yu Z, Huang Q (2019) Multimodal transformer with multi-view visual representation for image captioning. IEEE Trans Circuits Syst Video Technol 30(12):4467–4480
47. Kitaev N, Kaiser L, Levskaya A (2020) Reformer: the efficient transformer. In: ICLR
48. Ridley J, Coskun H, Tan DJ, Navab N, Tombari F (2022) Transformers in action: weakly supervised action segmentation. arXiv:2201.05675
49. Li M, Chen L, Duan Y, Hu Z, Feng J, Zhou J, Lu J (2022) Bridge-prompt: towards ordinal action understanding in instructional videos. In: CVPR, pp 19880–19889
50. Chen M-H, Li B, Bao Y, AlRegib G, Kira Z (2020) Action segmentation with joint self-supervised temporal domain adaptation. In: CVPR, pp 9454–9463
51. Ahn H, Lee D (2021) Refining action segmentation with hierarchical video representations. In: ICCV, pp 16302–16310
52. Rohrbach M, Rohrbach A, Regneri M, Amin S, Andriluka M, Pinkal M, Schiele B (2016) Recognizing fine-grained and composite activities using hand-centric features and script data. Int J Comput Vis 119(3):346–373