The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Exploring Temporal Preservation Networks for Precise Temporal Action Localization

Ke Yang, Peng Qiao, Dongsheng Li, Shaohe Lv, Yong Dou
National Laboratory for Parallel and Distributed Processing,
National University of Defense Technology
Changsha, China
{yangke13,pengqiao,dongshengli,shaohelv,yongdou}@nudt.edu.cn

Abstract

Temporal action localization is an important task of computer vision. Though a variety of methods have been proposed, it still remains an open question how to predict the temporal boundaries of action segments precisely. Most works use segment-level classifiers to select video segments predetermined by action proposal or dense sliding windows. However, in order to achieve more precise action boundaries, a temporal localization system should make dense predictions at a fine granularity. A newly proposed work exploits Convolutional-Deconvolutional-Convolutional (CDC) filters to upsample the predictions of 3D ConvNets, making it possible to perform per-frame action predictions and achieving promising performance in terms of temporal action localization. However, CDC network loses temporal information partially due to the temporal downsampling operation. In this paper, we propose an elegant and powerful Temporal Preservation Convolutional (TPC) Network that equips 3D ConvNets with TPC filters. TPC network can fully preserve temporal resolution and downsample the spatial resolution simultaneously, enabling frame-level granularity action localization with minimal loss of time information. TPC network can be trained in an end-to-end manner. Experiment results on public datasets show that TPC network achieves significant improvement in both per-frame action prediction and segment-level temporal action localization.

In recent years, temporal action localization has became a very important part of computer vision applications. Many works have been proposed to solve this problem (Escorcia et al. 2016; Jiang et al. 2014; Idrees et al. 2017; Caba Heilbron, Carlos Niebles, and Ghanem 2016; Rohrbach et al. 2012; Oneata, Verbeek, and Schmid 2014; Richard and Gall 2016; Shou, Wang, and Chang 2016; Singh and Cuzzolin 2016; Wang, Qiao, and Tang 2014; Wang and Tao 2016; Yeung et al. 2016; Shou et al. 2017; Qin and Shelton 2017), but how to perform temporal action localization precisely is still an open question.The purpose of temporal action localization is to determine the boundaries and classes of action segments in untrimmed videos. Most works extract various features on action segments pre-determined by action proposals or sliding windows and use them to train segment-level action classifiers.

Recently, it is claimed that action prediction at a fine granularity is important for achieving precise action localization (Shou et al. 2017). There exist some techniques can be adapted to achieve this goal: (1) classifying each frame using 2D CNN without consideration of temporal information; (2) using Recurrent Neural Network to model the temporal structure; (3) 3D CNN. 3D CNN is preferred because it can explicitly model the spatio-temporal information in raw videos. 3D CNN crushes the video in order to classify it. To perform frame-level predictions, one needs to upsample the output of 3D CNN in temporal domain. In (Shou et al. 2017), a Convolutional-De-Convolutional (CDC) layer is proposed to upsample the temporal resolution. By stacking CDC layers on top of 3D ConvNets, the resulting networks are able to make frame-level predictions. CDC network achieves promising performance in both action predictions at the frame granularity and segment-level action localization. However, directly upsampling the output of the classification networks cannot recover the degraded temporal information by downsampling, which harms precise temporal localization.

Therefore, we believe no-downsampling architectures are better than downsampling-upsampling architectures. The most intuitive solution to omit the downsampling is reducing the temporal pooling stride to 1. However, this operation changes the temporal receptive field of convolutional filters after the modified pooling layers. This reduces the amount of temporal context that can inform the prediction produced by each unit and also prevents us from using pre-trained models. In order to preserve the temporal receptive field of subsequent layers and take advantage of pre-trained weights rather than train networks from scratch, we replace standard 3D convolutional filters with Temporal Preservation Convolutional (TPC) filters. TPC filters can easily and explicitly control the temporal receptive field of convolutional filters when using the same kernel size as original convolutional filters. Therefore, TPC can enlarge the temporal resolution of neural network feature responses to cooperate with pooling layers with a stride of 1 to preserve temporal length of videos and make use of pre-trained weights. With TPC, C3D is upgraded to form our TPC network, which can model spatio-temporal information with minimal temporal information loss to make fine-grained action predictions that can be used to refine boundaries of action proposals to precisely

localize action segments. Refinement process is shown in Fig. 2.

It is worth noting that C3D is designed to label video clips, and needs careful design to conduct frame-level action classification which we believe is important for action localization. The design of temporal preservation architecture, which enables C3D to provide per-frame classification, is non-trivial and needs innovative idea and insight on this task. Although TPC is conceptually simple in form, it is able to preserve temporal resolution without exploiting upsampling explicitly as CDC. Our contributions can be concluded as follows: (1) To the best of our knowledge, in computer vision area, this is the first work to apply TPC filters, which can fully preserve temporal resolution and downsample spatial resolution simultaneously, allowing network to infer high-level action semantics with *minimal temporal information loss*. (2) We apply TPC filters to 3D ConvNets to form TPC networks. Our TPC network can be trained in an end-to-end manner to generate frame-level action predictions which can be used to refine action segments. (3) TPC network achieves competitive results in both per-frame action localization and segment-level action localization.

Related Work

Action recognition: Improved Dense Trajectory Feature (iDTF) (Wang et al. 2011; Wang and Schmid 2013) consisting of HOG, HOF, MBH features extracted along dense trajectories has been in a dominant position in the field of action recognition. Recently, 2D Convolutional Neural Networks (2DCNN) trained on ImageNet (Krizhevsky, Sutskever, and Hinton 2012) to perform RGB image classification such as AlexNet (Krizhevsky, Sutskever, and Hinton 2012), VGG (Simonyan and Zisserman 2015), ResNet (He et al. 2016) have gradually shown their power, but their performance is limited since they can only capture appearance information. In order to model motion, two-stream ConvNets taking both RBG and optical flow as input have significantly boost the performance (Feichtenhofer, Pinz, and Zisserman 2016; Wang et al. 2016; Simonyan and Zisserman 2014). To model spatio-temporal feature better, 3D CNN architecture called C3D is proposed to extract spatio-temporal abstraction of high-level semantics directly from raw videos (Tran et al. 2015).

Temporal action localization: A typical framework used in many state-of-the-art systems (Oneata, Verbeek, and Schmid 2014; Singh and Cuzzolin 2016; Wang, Qiao, and Tang 2014; Wang and Tao 2016) extracts various features and train a classifier such as Support Vector Machine (SVM) to classify action segments pre-determined by action proposals or densely sliding windows.

In recent years, deep networks improved performance of temporal localization through end-to-end learning from raw video clips directly to localize action segments. A Long Short Term Memory (LSTM)-based agent is trained using REINFORCE to learn both which frame to look next and when to emit an action segment prediction in (Yeung et al. 2016). A temporal action proposal framework is designed based on LSTM that takes pre-extracted CNN features in (Escorcia et al. 2016). In (Yeung et al. 2015), a LSTM network equipped with attention mechanism proposed to model these temporal relations via multiple input and output connections. In (Yuan et al. 2016), a Pyramid of Score Distribution Feature (PSDF) capturing the motion information at multiple resolutions centered at each sliding window is proposed and incorporated into the RNN to improve temporal consistency. Sun *et al.* (Sun et al. 2015) uses web images as prior to train LSTM model to improve action localization performance with only video-level annotations. Although RNN can make use of temporal information to make frame-level prediction, they are usually placed on top of CNN which take a single frame as input rather than directly modeling spatio-temporal abstraction of high-level semantics directly from from raw videos. In addition, RNN based model produces frame-level smoothing that is actually harmful, not beneficial to the task of precise action localization as (Yeung et al. 2016) claimed.

Based on C3D (Tran et al. 2015), an end-to-end Segment-CNN (S-CNN) action localization framework is proposed to improve action localization performance. S-CNN achieves promising results by capturing spatio-temporal information simultaneously. In (Shou et al. 2017), a fine-grained action localization framework called Convolutional-De-Convolutional (CDC) is designed to detect actions in every frame. Then frame-level action predictions are used to refine the action segment boundaries generated by S-CNN.

Semantic segmentation, audio application and atrous convolution: (Chen et al. 2014; 2016) apply the atrous convolution with upsampled filters to dense feature extraction for image segmentation. Atrous convolution allows to explicitly control the resolution at which feature responses are computed within convolutional neural networks. It also allows to effectively enlarge the field of view of filters to incorporate larger context without increasing the number of parameters or the amount of computation. Considering atrous convolution as a powerful tool in dense predict tasks, it shall have the potential to be adapted for making dense predictions in time for our precise temporal action localization task. However, unlike the image segmentation task in which keeping spatial resolution is import, our precise action localization task needs to preserve temporal resolution and downsample spatial resolution simultaneously. To this end, we propose TPC which allows us to preserve temporal resolution when downsampling spatial resolution at the same time. Our TPC filter can be be regarded as a special case of 3D atrous convolution in the temporal domain.

1D temporal atrous convolution is also applied to speech recognition (Sercu and Goel 2016) and audio generation (van den Oord et al. 2016). However, TPC filter is somewhat different from the 1D temporal dilated convolutions (TDC) used in audio area both in form and motivation. TDC is a 1D convolution filter that used in audio application while TPC is a 3D convolution filter used in video application which can preserve temporal resolution and downsample spatial resolution simultaneously, allowing network to infer high-level semantics (spatial dimension) with minimal temporal information loss simultaneously. TPC also allows us to use pre-trained model weights.

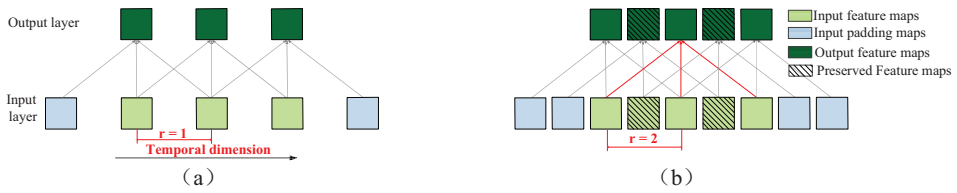

Figure 1: Illustration of temporal preservation convolution. We only show their temporal dimension since spatial dimension is the same. Each box represents the feature maps corresponding to one frame. Bottom line represents input layer while top line represents output layer. (a) Standard temporal convolution on a low resolution feature map that downsampled by pooling layer by a factor of 2. (b) Temporal preservation convolution on a high resolution feature map that is not downsampled. To have the same temporal receptive field size, we need a temporal sample $rate = r$, here $r = 2$.

Temporal preservation networks

C3D architecture which consists of five stages 3D ConvNets and three Fully Connected (FC) layers, has been shown that it can learn spatio-temporal patterns from raw video and has promising performance in action recognition (Tran et al. 2015). However, C3D architecture loses temporal information due to temporal downsampling from conv1a to pool5 layer, and the temporal length of output results in $L/16$ given an input video segment of temporal length L, which prevents us from frame-level predictions. In order to predict actions at frame-level, CDC network (Shou et al. 2017) stacks three CDC layers on top of 3D ConvNets part of C3D (3D ConvNets + 3 FCs $\longrightarrow$ 3D ConvNets + 3 CDCs). A CDC filter makes two copies of the fully connected (FC) layers of C3D [1] to upsample the temporal length by a factor of 2. After temporal upsampling by three times, the temporal length is upsampled to L from $L/8$ [2] ($L/8{\times}2{\times}2{\times}2 \longrightarrow L$). However, CDC network still crushes the temporal resolution during the temporal downsampling-upsampling process ($L \rightarrow 8/L \rightarrow L$), which is harmful to precise temporal localization. In addition, each CDC layer's parameter number is twice that of the corresponding FC layer in C3D, resulting in a higher possibility of overfitting.

In order to make frame-level action predictions with minimal temporal information loss, we had better *preserve temporal resolution throughout the whole forward propagation process rather than using the downsampling-upsampling framework.* To this end, we propose TPC filter and use it to construct a TPC network to make frame-level action predictions.

Temporal preservation convolution

In this section, we will introduce TPC filter and explain how we build a TPC network with the TPC filters. Why is temporal resolution reduced in C3D? It has direct relationship with pooling filters whose temporal stride is bigger than 1. To preserve the resolution from beginning to end, we need to reduce all pooling layers' pooling stride to 1. We will modify the structure inside 3D ConvNets rather than modify three FC layers as CDC network does. TPC network's operations in spatial dimension are the same as that of C3D, so we mainly consider the temporal dimension.

The modified network can preserve temporal length from beginning to end. However, we can notice that the temporal receptive field [3] of the convolutional filters after modified pooling layers is smaller than that of standard filters. However, contextual information is very important in disambiguating local cues (Galleguillos and Belongie 2010). And this also means we can not use the pre-trained model from C3D, but training a network with a small data set from scratch is very difficult. For these two reasons, we need to increase the convolutional filters' temporal receptive field size to match that of the original convolutional filters. To this end, we replace the standard 3D convolutional filters in C3D with our TPC filters which can enlarge the temporal receptive field of filters to incorporate larger context without increasing the number of parameters.

Considering only temporal dimension, temporal preservation convolution can be defined as Equation 1, where $x[t]$ [4] is the feature map corresponding to the $t\text{-}th$ frame, $w[k]$ is convolutional filter, K is the size of filter, r stands for the stride with which filters sample input. Standard convolution is a special case for stride $r = 1$. We illustrate TPC in Fig. 1, the convolutional filter samples in previous layer's feature maps' temporal dimension at a stride of 2. TPC filter can also be treated as a bigger filter with fixed zero-value which not updated when network parameters are adjusted. The other parameters are initialized with the pre-trained model and are trainable.

$$y[t] = \sum_{k=1}^{K} x[t + r \cdot k]w[k] \tag{1}$$

The idea of our TPC is similar to that of atrous convolution used in 2D image segmentation (Chen et al. 2014; 2016), but TPC is performed on temporal dimension rather than spatial dimension. In temporal dimension, TPC is very similar to the 1D temporal atrous convolution used in (van den Oord et al. 2016; Sercu and Goel 2016). In order to be consistent with (Chen et al. 2014; 2016), we assign the

[1] FC layers in C3D have been transformed to convolutional layers following (Long, Shelhamer, and Darrell 2015)

[2] CDC network keeps temporal length by set pooling stride to 1 in pool5 layer, so its temporal length after pool5 is twice that of C3D

[3] We name 3D convolutional filters' receptive field's temporal dimension as *temporal receptive field* for convenience

[4] The shape of $x[t]$ is (number of channels, height, width).

Table 1: Networks architecture comparison. Illustration of output shape and filter size of each layer. We denote layer-wise output shape using the form of (number of channels × temporal length × height × width). Filter shape using (temporal length× height × width, temporal atrous rate) for convolutional layers, and (temporal length× height × width, stride (temporal stride, height stride, width stride)) for pooling layers.

Networks architecture						
Layers	C3D		CDC		Our TPN	
	Blocks	Output size	Blocks	Output size	Blocks	Output size
input	raw input video $3 \times L \times 112 \times 112$					
conv1	$3 \times 3 \times 3, 1$	$64 \times L \times 112 \times 112$	$3 \times 3 \times 3, 1$	$64 \times L \times 112 \times 112$	$3 \times 3 \times 3, 1$	$64 \times L \times 112 \times 112$
pool1	$3 \times 2 \times 2$ stride (1, 2, 2)	$64 \times L \times 56 \times 56$	$3 \times 2 \times 2$ stride (1, 2, 2)	$64 \times L \times 56 \times 56$	$3 \times 2 \times 2$ stride (1, 2, 2)	$64 \times L \times 56 \times 56$
conv2	$3 \times 3 \times 3, 1$	$128 \times L \times 56 \times 56$	$3 \times 3 \times 3, 1$	$128 \times L \times 56 \times 56$	$3 \times 3 \times 3, 1$	$128 \times L \times 56 \times 56$
pool2	$3 \times 2 \times 2$ stride (2, 2, 2)	$128 \times L/2 \times 28 \times 28$	$3 \times 2 \times 2$ stride (2, 2, 2)	$128 \times L/2 \times 28 \times 28$	$3 \times 2 \times 2$ stride (**1**, 2, 2)	$128 \times L \times 28 \times 28$
conv3_x	$\begin{bmatrix} 3 \times 3 \times 3, 1 \\ 3 \times 3 \times 3, 1 \end{bmatrix}$	$256 \times L/2 \times 28 \times 28$	$\begin{bmatrix} 3 \times 3 \times 3, 1 \\ 3 \times 3 \times 3, 1 \end{bmatrix}$	$256 \times L/2 \times 28 \times 28$	$\begin{bmatrix} 3 \times 3 \times 3, \mathbf{2} \\ 3 \times 3 \times 3, \mathbf{2} \end{bmatrix}$	$256 \times \mathbf{L} \times 28 \times 28$
pool3	$3 \times 2 \times 2$ stride (2, 2, 2)	$256 \times L/4 \times 14 \times 14$	$3 \times 2 \times 2$ stride (2, 2, 2)	$256 \times L/4 \times 14 \times 14$	$3 \times 2 \times 2$ stride (**1**, 2, 2)	$256 \times \mathbf{L} \times 14 \times 14$
conv4_x	$\begin{bmatrix} 3 \times 3 \times 3, 1 \\ 3 \times 3 \times 3, 1 \end{bmatrix}$	$512 \times L/4 \times 14 \times 14$	$\begin{bmatrix} 3 \times 3 \times 3, 1 \\ 3 \times 3 \times 3, 1 \end{bmatrix}$	$512 \times L/4 \times 14 \times 14$	$\begin{bmatrix} 3 \times 3 \times 3, \mathbf{4} \\ 3 \times 3 \times 3, \mathbf{4} \end{bmatrix}$	$512 \times \mathbf{L} \times 14 \times 14$
pool4	$3 \times 2 \times 2$ stride (2, 2, 2)	$512 \times L/8 \times 7 \times 7$	$3 \times 2 \times 2$ stride (2, 2, 2)	$512 \times L/8 \times 7 \times 7$	$3 \times 2 \times 2$ stride (**1**, 2, 2)	$512 \times \mathbf{L} \times 7 \times 7$
conv5_x	$\begin{bmatrix} 3 \times 3 \times 3, 1 \\ 3 \times 3 \times 3, 1 \end{bmatrix}$	$512 \times L/8 \times 7 \times 7$	$\begin{bmatrix} 3 \times 3 \times 3, 1 \\ 3 \times 3 \times 3, 1 \end{bmatrix}$	$512 \times L/8 \times 7 \times 7$	$\begin{bmatrix} 3 \times 3 \times 3, \mathbf{8} \\ 3 \times 3 \times 3, \mathbf{8} \end{bmatrix}$	$512 \times \mathbf{L} \times 7 \times 7$
pool5	$3 \times 2 \times 2$ stride (2, 2, 2)	$512 \times L/16 \times 4 \times 4$	$3 \times 2 \times 2$ stride (1, 2, 2)	$512 \times L/8 \times 4 \times 4$	$3 \times 2 \times 2$ stride (**1**, 2, 2)	$512 \times \mathbf{L} \times 4 \times 4$
fc6/cdc6/conv6	$1 \times 4 \times 4, 1$	$4096 \times L/16 \times 1 \times 1$	$1 \times 4 \times 4$ (2 copies)	$4096 \times L/4 \times 1 \times 1$	$1 \times 4 \times 4, 1$	$4096 \times \mathbf{L} \times 1 \times 1$
fc7/cdc7/conv7	$1 \times 1 \times 1, 1$	$4096 \times L/16 \times 1 \times 1$	$1 \times 1 \times 1, 1$ (2 copies)	$4096 \times L/2 \times 1 \times 1$	$1 \times 1 \times 1, 1$	$4096 \times \mathbf{L} \times 1 \times 1$
fc8/cdc8/conv8	$1 \times 1 \times 1, 1$	$(K+1) \times L/16 \times 1 \times 1$	$1 \times 1 \times 1, 1$ (2 copies)	$(K+1) \times L \times 1 \times 1$	$1 \times 1 \times 1, 1$	$(K+1) \times \mathbf{L} \times 1 \times 1$

sampling stride as Temporal Atrous Sampling Rate (TASR). Comparisons of architecture of C3D (Tran et al. 2015), CDC (Shou et al. 2017) and our TPC network are shown in Table 1. For C3D, temporal length is downsampled in $pool_i$ layers($i = 2, 3, 4, 5$) by a factor of 2 and eventually reduced to $L/16$. CDC network first downsamples temporal resolution to $L/8$ and then stacks three CDC layers to upsample to L. Based on C3D, TPC network reduces the pooling stride to 1 in $pool_i$ layers($i = 2, 3, 4, 5$), and set $TASR = 2$ for conv3a and conv3b (same as Fig. 1), $TASR = 4$ for conv4a and conv4b, and $TASR = 8$ for conv5a and conv5b to keep the temporal length be L from beginning to end. In this way, TPC network can preserve as much temporal precision as possible.

More details to construct TPC newtork. To make it easier to align the output and the input in the temporal dimension, we modify the temporal dimension of all pooling layers' kernel size from 2 to 3. In our descriptions above, details of the convolutional and pooling layers have been clarified. As explained in (Long, Shelhamer, and Darrell 2015), the FC layer is a special case of convolutional layer, and we can transform FC6 (weights shape: 4096×8192), FC7 (weights shape: 4096×4096) to conv6 (filter shape: $4096 \times 512 \times 4 \times 4$), conv7 (filter shape: $4096 \times 4096 \times 1 \times 1$) respectively. Now conv6 can slide on L feature maps of size $512 \times 4 \times 4$ stacked in time and output L feature maps of size $4096 \times 1 \times 1$. Conv6, conv7 layers can be initialized with FC6, FC7, but conv8 can not be adapted from FC8 since output classes are not same in conv8 and FC8, so we randomly initialize conv8. We perform softmax operation and compute softmax loss for each frame separately. Given a mini-batch with N training segments, batch output O and label y, the total loss $\mathcal{L}$ is defined as Equation 2. $\mathcal{L}$ can be optimized by standard backpropagation (BP) algorithm.

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{L} \sum_{c=1}^{K+1} - y_n^{(c)}[t] \log \left(\frac{\exp\left(O_n^{(c)}[t]\right)}{\sum_{j=1}^{K+1} \exp\left(O_n^{(j)}[t]\right)} \right) \tag{2}$$

Model training and prediction

Training data construction. Training data consists of video segments with length L. L can be an arbitrary value because TPC network is a fully convolutional network. We chose $L = 64$ frames in practical due to the Graphics Processing Unit (GPU) memory limitation. Following (Shou et al. 2017), we slide temporal window of size L on untrimmed videos and only keep segments include at least one frame belongs to actions to prevent including too many background frames. To construct a balanced training dataset, we resample the segments belong to minority classes to ensure each action class has about $80K$ frames.

Model training. We implement TPC network based on Keras (Chollet and others 2015) and C3D (Tran et al. 2015).

Table 2: Frame-level action localization mAP on THUMOS'14.

Method	mAP	Method	mAP
Single-frame CNN(Simonyan and Zisserman 2015)	34.7	TPC-2	45.5
Two-stream CNN(Simonyan and Zisserman 2014)	36.2	TPC-3	45.1
LSTM(Donahue et al. 2015)	39.3	TPC-4	45.0
MultiLSTM(Yeung et al. 2015)	41.3	TPC-2,3	46.4
Conv & De-conv(Shou et al. 2017)	41.7	TPC-3,4	45.7
CDC(Shou et al. 2017)	44.4	**TPC**	**49.5**

Codes and models will be shared online. We use Stochastic Gradient Descent (SGD) to train TPC network. We first freeze the layers before conv8 and train conv8 with learning rate set to 0.0001, then train all the layer with learning rate set to 0.00001. We set momentum to 0.9 and weight decay to 0.0005. We use C3D (Tran et al. 2015) pre-trained on Sports-1M (Karpathy et al. 2014) to initialize TPC network from conv1 to conv7. We randomly initialize weights for conv8.

Frame-level action predictions. During testing, we slide TPC network on the whole video without overlapping. Then, we get the action predictions for all the frames of the whole video. The difference between TPC network frame-level features and 2D CNN frame-level features is that ours are calculated taking into account whole video segment information, so our features are more robust to noise. Compared to 2D CNN+LSTM framework, our frame-level features align more precisely with input since LSTM smooths temporal information (Yeung et al. 2016).

Segment-level action predictions. In order to further verify the effectiveness of TPC network, we carry out segment-level action localization with TPC network's frame-level action predictions. Here we use two different methods.

For a direct and fair comparison, we first follow (Shou et al. 2017) and apply TPC network on proposal segments generated by (Shou, Wang, and Chang 2016). We apply the same strategy that using frame-level predictions to refine segment proposals as (Shou et al. 2017). We set the category of one segment to the maximum average confidence score over all frames in the video segment. Only the segments not assigned to background class are kept for further boundary refinement. We start from boundaries of each side and move to the middle of the segment, and shrink the temporal boundaries until reach a frame with confidence score lower than the threshold. For more details about the refinement process and the confidence score threshold selecting method please refer to (Shou et al. 2017).

In order to make better use of frame-level prediction results, we design a new frame grouping method that gets action segments from untrimmed videos by thresholding on confidence scores and group adjacent frames. First, we take threshold processing on classification scores of all frames in the test video. As a result, we got a string of "0" and "1" (0 indicates below the threshold, and 1 inversely). Second, we group the adjacent "1" to get the segment-level outputs. Then we use NMS to post-process these segments. For threshold value selection, we set multiple different threshold values (uniformly selected from 0 to 1) instead of dataset-dependent. We denote the frame grouping method as **FGM**.

Evaluation

We evaluate TPC network on the challenging dataset THUMOS'14 (Jiang et al. 2014; Idrees et al. 2017). Temporal action detection task in THUMOS'14 challenge is dedicated to localize the action instances in untrimmed video and involves 20 action classes. Training set consists of 2755 well trimmed videos of these 20 action classes from UCF101 dataset (Soomro, Zamir, and Shah 2012). Validation set consists of 1010 untrimmed videos with temporal annotations in form of (video name, action segment start time, action segment ending time, action category). Test set consists of 1574 untrimmed videos. Same as (Shou, Wang, and Chang 2016; Shou et al. 2017), we only keep the videos that contain action instances of interest for testing. We evaluate TPC network on frame-level action localization and segment-level action localization tasks.

Frame-level action localization

First, we evaluate TPC network in predicting action labels for every frame in the whole video. This task can take multiple frames as input to take into account temporal information.Following (Yeung et al. 2015), we evaluate frame-level prediction as a retrieval problem. For each action class, we rank all the images in the test set by their confidence scores and compute Average Precision (AP) for this class. And mean AP (mAP) is computed by average the AP of 20 action classes.

In Table 2, we compare our TPC network with state-of-the-art methods. All the results are quoted from (Yeung et al. 2015; Shou et al. 2017). Single-frame CNN stands for frame-level VGG-16 2D CNN model in (Simonyan and Zisserman 2015). Two-stream CNN is the frame-level CNN model proposed in (Simonyan and Zisserman 2014) using optical flow and RGB images to perform action recognition. LSTM represents the basic 2D CNN + LSTM model proposed in (Donahue et al. 2015). MultiLSTM stands for an extended LSTM using temporal attention mechanism proposed in (Yeung et al. 2015). MultiLSTM uses THUMOS'14 extended version dataset MultiTHUMOS with much more annotations (Yeung et al. 2015) to train their network. Conv & De-conv stands for the baseline method in (Shou et al. 2017) replacing CDC layers with de-convolutional layers. CDC stands for the convolutional-de-convolutional network proposed in (Shou et al. 2017). We denote our TPC network as **TPC**. Among these methods, Single-frame CNN only takes into account appearance information in a single frame, Two-stream CNN uses appearance information in a single frame and motion information

from two adjacent frames. LSTM and MultiLSTM can make use of temporal information to make frame-level predictions but LSTM based model produces frame-level class probabilities smoothing what is actually harmful, not beneficial to the task of precise action localization as (Yeung et al. 2016) claimed. Conv & De-conv, CDC and our TPC are all based on 3D CNN, can model appearance information and temporal information simultaneously. However, Conv & De-conv, CDC network both lose temporal information, which leds to inferior results. Our TPC network equipped with TPC filters can perform frame-level predictions with minimal temporal information loss, achieving promising performance.

In addition, in order to verify the effectiveness of TPC on temporal information preservation and support our claim that preserving temporal resolution is important for precise localization, we compare TPC with TPC's variants that only use TPC filters on one or two layers. (1) TPC-2: we only use TPC in conv2. (2) TPC-3: we only use TPC in conv3. (3) TPC-4: we only use TPC in conv4. (4) TPC-2,3: we use TPC in conv2 and conv3. (5) TPC-3,4: we use TPC in conv3 and conv4. Complete TPC network use TPC filters on conv2, conv3 and conv4 (i.e., TPC-2,3,4). For the five variants, we apply linear interpolation to upsample predictions to output frame-level predictions for both training and testing. We train them using the same training data as TPC.

Ablation experiment results suggest that preserving temporal information at early stage helps preserve more details and brings better result, but not that much. Preserving more time information with more TPC layers, we get better localizaiton results. TPC-2,3,4 brings notable performance improvement, suggesting that preserving the temporal resolution in all layers brings minimal temporal information loss and better performance.

Temporal action localization

Given frame-level action predictions, we can get segment-level action localization results using various strategies. For more direct comparison, we first use the same strategy as CDC (Shou et al. 2017). First, we generate action segment proposals using the S-CNN(Shou, Wang, and Chang 2016); second, each segment is set to an action category; then, non-background segments' boundaries are refined with frame-level action predictions and confidence scores are calculated by averaging confidence scores of all the frame in refined segments; finally, we perform post-processing steps such as non-maximus suppression. We evaluate our model on THUMOS'14 dataset.

We perform evaluation using mAP as frame-level action localization evaluation. For each action class, we rank all the predicted segments by their confidence results and calculate the AP using official evaluation code. One prediction is correct when its temporal overlap intersection-over-union (IoU) with a ground truth action segment is higher than the threshold, so evaluation under various IoU threshold is necessary. We evaluate our model under IoU threshold from 0.3 to 0.7. Results are shown in Table 3, our model denoted as **TPC** achieves better results than other methods.

As shown in Table 2 and Table 3, TPC achieves clearly improvement over other baselines on frame-level task but the improvement is far less significant on segment-level task. The reason might be that proposals by S-CNN(Shou, Wang, and Chang 2016) help CDC(Shou et al. 2017) much more. Proposals from (Shou, Wang, and Chang 2016) help CDC or TPC filter video segments which might be background frames. TPC performs much better than CDC on frame-level task, which means that TPC also does much better on the filtered frames. So proposals do not improve TPCs performance that much as CDC. To verify this idea, we perform FGM on both TPC and CDC frame-level classification results to get segment-level detections. Results are shown in Table 3, TPCs performance improves significantly after using the new frame grouping method. The reason for the significant improvement is that proposals from (Shou, Wang, and Chang 2016) have false negatives, and TPC can handle these false negative frames. CDCs(Shou et al. 2017) performance decrease (when IoU = 0.3, 0.4, 0.5) because their inferior performance outside the proposals. Overall,results suggest that frame-level results indeed contributes to precise segment-level localization.

Qualitative experiment results are shown in Fig. 2. This results suggest that TPC perform better on frame-level classification, and this better results lead to better segment-level results. We also can clearly observe that CDC suffered from checkerboard artifacts brought by the deconvolution operations (Odena, Dumoulin, and Olah 2016). Our TPC is not affected by this problem because TPC can preserve temporal length and does not need to use deconvolution to upsample in time.

Discussion

TPC network allows us to compute feature responses at the original video temporal resolution, but it indeed increases computational overhead. In order to give a fair comparison, we implemented CDC network (Shou et al. 2017) in our experiment environments. On a NVIDIA Titan X GPU with 12GB memory, our TPC can predict around 250 frames per second (FPS) while CDC network predicts around 390 FPS. Although our method is not as fast as CDC network, it is enough for real-time application. After all, our TPC network can process 10 seconds video clip of 25 FPS within one second.

Conclusion

In this paper, we propose a TPC filter to replace the standard convolutional filters in 3D ConvNets. Then we use TPC filters to construct our TPC network. Our TPC network can make more precise frame-level action predictions since it preserve all the temporal information. We also evaluate our model on segment-level action localization task. Experiments on frame-level and segment-level action localization tasks both suggest that our model achieves superior results compared with previous works. TPC network can predict around 250 frames per second which is good news for real-time applications. In addition, our TPC filter can be adapted for other applications, such as combined with the spatial atrous convolutional filter to perform video segmentation.

Table 3: Segment-level action localization mAP on THUMOS'14. IoU threshold values are ranged from 0.3 to 0.7. '-' in the table indicates that results of that IoU value are not available in the corresponding papers.

IoU threshold	0.3	0.4	0.5	0.6	0.7
Wang et al.(Wang, Qiao, and Tang 2014)	14.6	12.1	8.5	4.7	1.5
Heilbron et al.(Caba Heilbron, Carlos Niebles, and Ghanem 2016)	-	-	13.5	-	-
Escorcia et al.(Escorcia et al. 2016)	-	-	13.9		-
Oneata et al.(Oneata, Verbeek, and Schmid 2014)	28.8	21.8	15.0	8.5	3.2
Richard and Gall(Richard and Gall 2016)	30.0	23.2	15.2	-	-
Yeung et al.(Yeung et al. 2016)	36.0	26.4	17.1	-	-
Yuan et al.(Yuan et al. 2016)	33.6	26.1	18.8	-	-
S-CNN(Shou, Wang, and Chang 2016)	36.3	28.7	19.0	10.3	5.3
Conv & De-conv(Shou et al. 2017) + S-CNN(Shou, Wang, and Chang 2016)	38.6	28.2	22.4	12.0	7.5
CDC(Shou et al. 2017) + S-CNN(Shou, Wang, and Chang 2016)	40.1	29.4	23.3	13.1	7.9
TPC-2 + S-CNN(Shou, Wang, and Chang 2016)	37.8	28.9	22.6	13.7	7.8
TPC-3 + S-CNN(Shou, Wang, and Chang 2016)	37.6	29.0	22.3	13.3	7.4
TPC-4 + S-CNN(Shou, Wang, and Chang 2016)	37.6	28.7	22.1	12.7	6.9
TPC-2,3 + S-CNN(Shou, Wang, and Chang 2016)	39.8	30.7	24.1	13.9	7.8
TPC-3,4 + S-CNN(Shou, Wang, and Chang 2016)	38.5	29.3	22.9	13.5	7.6
TPC + S-CNN(Shou, Wang, and Chang 2016)	**41.9**	**32.5**	**25.3**	**14.7**	**9.0**
CDC(Shou et al. 2017) + FGM	36.1	28.2	20.9	14.9	8.1
TPC + FGM	**44.1**	**37.1**	**28.2**	**20.6**	**12.7**

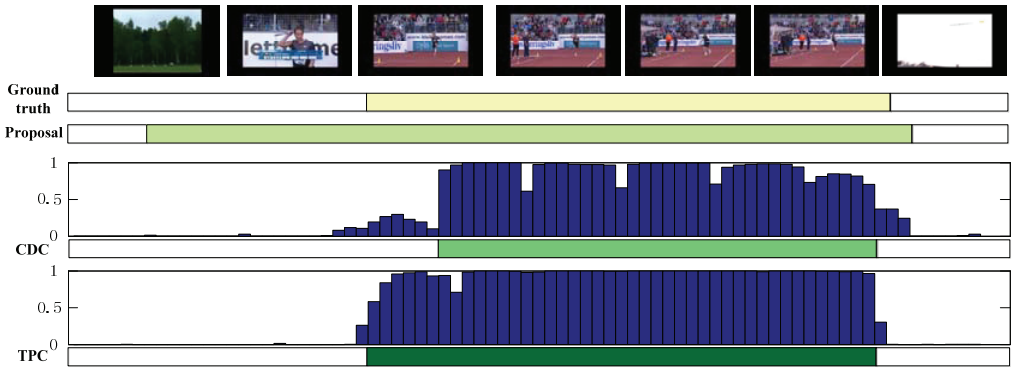

Figure 2: Illustration of the process of temporal boundaries refinement using frame-level predictions. Horizontal axis stands for time and vertical axis stands for confidence score. From the top to the bottom: (1) frame-level ground truth for a JavelinThrow instance in an input video; (2) corresponding proposal generated from (Shou, Wang, and Chang 2016); (3) frame-level predictions of CDC (Shou et al. 2017) and refined action instance using CDC; (4) frame-level predictions of TPC and refined action instance using TPC.

Acknowledgements

This work was supported by the National Basic Research Program of China (973) under Grant No.2014CB340303 and the National Natural Science Foundation of China under Grants U1435219, 61402507 and 61572515.

References

Caba Heilbron, F.; Carlos Niebles, J.; and Ghanem, B. 2016. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1914–1923.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2014. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2016. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*.

Chollet, F., et al. 2015. Keras. https://github.com/fchollet/keras.

Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; and Darrell, T. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2625–2634.

Escorcia, V.; Heilbron, F. C.; Niebles, J. C.; and Ghanem, B. 2016. Daps: Deep action proposals for action understanding. In *European Conference on Computer Vision*, 768–784. Springer.

Feichtenhofer, C.; Pinz, A.; and Zisserman, A. 2016. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1933–1941.

Galleguillos, C., and Belongie, S. 2010. Context based object categorization: A critical survey. *Computer vision and image understanding* 114(6):712–722.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Idrees, H.; Zamir, A. R.; Jiang, Y.-G.; Gorban, A.; Laptev, I.; Sukthankar, R.; and Shah, M. 2017. The thumos challenge on action recognition for videos in the wild. *Computer Vision and Image Understanding* 155:1–23.

Jiang, Y.-G.; Liu, J.; Roshan Zamir, A.; Toderici, G.; Laptev, I.; Shah, M.; and Sukthankar, R. 2014. THUMOS challenge: Action recognition with a large number of classes. http://crcv.ucf.edu/THUMOS14/.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 1725–1732.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.

Odena, A.; Dumoulin, V.; and Olah, C. 2016. Deconvolution and checkerboard artifacts. *Distill*.

Oneata, D.; Verbeek, J.; and Schmid, C. 2014. The lear submission at thumos 2014.

Qin, Z., and Shelton, C. R. 2017. Event detection in continuous video: An inference in point process approach. *IEEE Transactions on Image Processing* 26(12):5680–5691.

Richard, A., and Gall, J. 2016. Temporal action detection using a statistical language model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3131–3140.

Rohrbach, M.; Amin, S.; Andriluka, M.; and Schiele, B. 2012. A database for fine grained activity detection of cooking activities. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 1194–1201. IEEE.

Sercu, T., and Goel, V. 2016. Dense prediction on sequences with time-dilated convolutions for speech recognition. *CoRR* abs/1611.09288.

Shou, Z.; Chan, J.; Zareian, A.; Miyazawa, K.; and Chang, S.-F. 2017. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *CVPR*.

Shou, Z.; Wang, D.; and Chang, S.-F. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1049–1058.

Simonyan, K., and Zisserman, A. 2014. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, 568–576.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations*.

Singh, G., and Cuzzolin, F. 2016. Untrimmed video classification for activity detection: submission to activitynet challenge. *arXiv preprint arXiv:1607.01979*.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

Sun, C.; Shetty, S.; Sukthankar, R.; and Nevatia, R. 2015. Temporal localization of fine-grained actions in videos by domain transfer from web images. In *Proceedings of the 23rd ACM international conference on Multimedia*, 371–380. ACM.

Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; and Paluri, M. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 4489–4497.

van den Oord, A.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A. W.; and Kavukcuoglu, K. 2016. Wavenet: A generative model for raw audio. *CoRR* abs/1609.03499.

Wang, H., and Schmid, C. 2013. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, 3551–3558.

Wang, R., and Tao, D. 2016. Uts at activitynet 2016. *AcitivityNet Large Scale Activity Recognition Challenge* 2016:8.

Wang, H.; Kläser, A.; Schmid, C.; and Liu, C.-L. 2011. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 3169–3176. IEEE.

Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; and Van Gool, L. 2016. Temporal segment networks: towards good practices for deep action recognition. In *European Conference on Computer Vision*, 20–36. Springer.

Wang, L.; Qiao, Y.; and Tang, X. 2014. Action recognition and detection by combining motion and appearance features. *THUMOS14 Action Recognition Challenge* 1:2.

Yeung, S.; Russakovsky, O.; Jin, N.; Andriluka, M.; Mori, G.; and Fei-Fei, L. 2015. Every moment counts: Dense detailed labeling of actions in complex videos. *arXiv preprint arXiv:1507.05738*.

Yeung, S.; Russakovsky, O.; Mori, G.; and Fei-Fei, L. 2016. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2678–2687.

Yuan, J.; Ni, B.; Yang, X.; and Kassim, A. A. 2016. Temporal action localization with pyramid of score distribution features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3093–3102.