Graph Attention based Proposal 3D ConvNets for Action Detection

Jun Li[1], **Xianglong Liu**[1,2*], **Zhuofan Zong**[1], **Wanru Zhao**[1], **Mingyuan Zhang**[1], **Jingkuan Song**[3]
[1]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[2]Beijing Advanced Innovation Center for Big Data-Based Precision Medicine, Beihang University, Beijing, China
[3]Innovation Center, University of Electronic Science and Technology of China, Chengdu, China
junmuzi@gmail.com, xlliu@nlsde.buaa.edu.cn

Abstract

The recent advances in 3D Convolutional Neural Networks (3D CNNs) have shown promising performance for untrimmed video action detection, employing the popular detection framework that heavily relies on the temporal action proposal generations as the input of the action detector and localization regressor. In practice the proposals usually contain strong intra and inter relations among them, mainly stemming from the temporal and spatial variations in the video actions. However, most of existing 3D CNNs ignore the relations and thus suffer from the redundant proposals degenerating the detection performance and efficiency. To address this problem, we propose graph attention based proposal 3D ConvNets (AGCN-P-3DCNNs) for video action detection. Specifically, our proposed graph attention is composed of intra attention based GCN and inter attention based GCN. We use intra attention to learn the intra long-range dependencies inside each action proposal and update node matrix of Intra Attention based GCN, and use inter attention to learn the inter dependencies between different action proposals as adjacency matrix of Inter Attention based GCN. Afterwards, we fuse intra and inter attention to model intra long-range dependencies and inter dependencies simultaneously. Another contribution is that we propose a simple and effective framewise classifier, which enhances the feature presentation capabilities of backbone model. Experiments on two proposal 3D ConvNets based models (P-C3D and P-ResNet) and two popular action detection benchmarks (THUMOS 2014, ActivityNet v1.3) demonstrate the state-of-the-art performance achieved by our method. Particularly, P-C3D embedded with our module achieves average mAP 3.7% improvement on THUMOS 2014 dataset compared to original model.

Introduction

In recent years, with the tremendous increase of video data, effective and efficient video analysis is becoming a problem demanding prompt solutions. Specifically, video action detection, not only locating the action temporal boundary but also classifying the action category, is one of the challenging tasks. Many early studies developed hand-craft features to achieve action detection, such as DLSBP (Duchenne et al. 2009), ASM (Gaidon, Harchaoui, and Schmid 2011) and SDPM (Tian, Sukthankar, and Shah 2013). In the past decade, because of the stronger modeling capabilities of convolutional neural networks, more studies have focused on deep features based action detection (Singh et al. 2016; Gao et al. 2019; Lin et al. 2019; Liu et al. 2019).

Among the deep features based solutions, 3D ConvNet methods have shown promising performance in modeling the motion and appearance information, which can alleviate expensive optical flow computation, and have been widely studied recently. Shou, Wang, and Chang exploited the sliding windows and 3D ConvNets to classify and localize the video activities. Inspired by Faster R-CNN (Ren et al. 2015), Xu, Das, and Saenko proposed R-C3D for action detection, which integrated temporal action proposals generation and proposals classification in one end-to-end network. Wang and Gupta chose a graph-based reasoning framework to model human-object and object-object relationships. In (Long et al. 2019) a set of Gaussian kernels were learnt to dynamically model temporal scale of each action proposal in 3D ConvNets.

Despite the successful progress using 3D ConvNets, most of these methods heavily rely on the temporal action proposal generation as the input of the action detector and localization regressor, which plays a very important role on promising the action detection accuracy. In practice, the proposals usually contain strong inherent relations, mainly stemming from the temporal and spatial variations in the video actions. The intra relations inside each proposal are useful for correcting the wrong action proposal towards the ground truth label, while the inter relations across different proposals can help adjusting the imprecise boundary of the temporal proposal. Unfortunately, the existing solutions often ignore such inherent relations and thus largely suffer from deviated action boundaries and inaccurate detection.

To address this issue, we propose graph attention based proposal 3D convolutional networks (AGCN-P-3DCNNs) to simultaneously model the intra and inter dependencies of temporal action proposals for accurate action detection. Specifically, an intra and inter attention mechanism are respectively introduced and fused in the graph convolutional network (GCN) to dynamically represent the proposals and

*Corresponding author

capture their dependencies. The intra attention mechanism enables the 3D CNNs to learn discriminative feature representation for proposals, by modeling long-range dependencies of pixels inside single proposal. The inter attention mechanism learns the adaptive dependencies among the proposals, which tends to adjust the imprecise boundary of the temporal proposal. The overall architecture with our graph attention module is depicted in Figure 1.

As we all know, there is a large performance gap for 3D CNNs between different training methods that are training from scratch and training based pretrained model (Hara, Kataoka, and Satoh 2017). Inspired by the facts above, we propose to use a framewise classifier to train the proposal 3D CNNs firstly, and then use these well-trained parameters to initialize the graph attention based proposal 3D CNNs, as Figure 1 shows. Properly speaking, we define a framewise classifier to constrain the backbone subnet and train a more precise 3D ConvNet model with stronger abilities of feature modeling. Meanwhile, considering that most of the current proposal based action detection frameworks only focus on the proposal level optimization and ignore the frame level optimization, we combine the coarse-grained (proposal level) action detector and fine-grained (frame level) framewise classifier to optimize the training process.

To our best knowledge, we are the first to study the intra and inter relations of the action proposals in video action detection, and devise an attention based GCN module for 3DCNNs that prominently boosts the video action detection performance. Besides, we use framewise classifier to improve the performance of backbone subnet, forming our graph attention based proposal 3D CNNs with framewise classifier constraint (FC-AGCN-P-3DCNNs) for action detection. Our method serves as a generic module that can be applied in many 3D CNNs for untrimmed video action detection. Our proposed FC-AGCN-3DCNNs achieves the state-of-the-art performance on the THUMOS 2014 and ActivityNet v1.3 datasets for the temporal action detection task.

Related work

Attention Attention is a popular mechanism that has been applied in many fields. (Bahdanau, Cho, and Bengio 2014) presented the attention mechanism, a model that is able to automatically (soft-)search for parts of a source sentence related to the prediction of a target word, in the domains of neural machine translation. Later, based on attention mechanisms, (Vaswani et al. 2017) introduced Transformer and dispensed with recurrence and convolutions. As a result of the surprising performance of attention mechanism, in the filed of computer vision, (X. et al. 2015) proposed "soft" and "hard" attention and applied attention in image caption generation. (Wang et al. 2017) presented residual attention network for image classification. (Wang et al. 2018) introduced the non-local neural network and applied it in video classification, static image recognition, object detection/segmentation and pose estimation. (Wang et al. 2019) proposed a graph attention convolution with learnable kernel shapes to dynamically adapt to the structure of the objects.

Graph Convolutional Networks (GCNs) To deal with non Euclidean structure data, graph based machine learning has been developed. Due to the fact that CNNs can not deal with non Euclidean structure data, GCNs have been widely used in recent years. In the early years, (Scarselli et al. 2008) introduced graph neural networks (GNNs), which extended existing neural network methods for processing the data represented in graph domains. Different from GNNs, (Kipf and Welling 2016) introduced a layer-wise propagation rule for neural network models motivated from a first-order approximation of spectral graph convolutions. Meanwhile, since CNNs can not model graph structure data directly, GCNs have been widely used in recent years (Simonovsky and Komodakis 2017; Shi et al. 2019; Jiang et al. 2019; Li et al. 2019). (Simonovsky and Komodakis 2017) were the first to apply graph convolutions to point cloud classification. The graph attention networks presented by (Busbridge et al. 2017) are able to specify different weights to different nodes in a neighborhood. (Jiang et al. 2019) introduced graph learning-convolutional network (GLCN) integrating both graph learning and graph convolution.

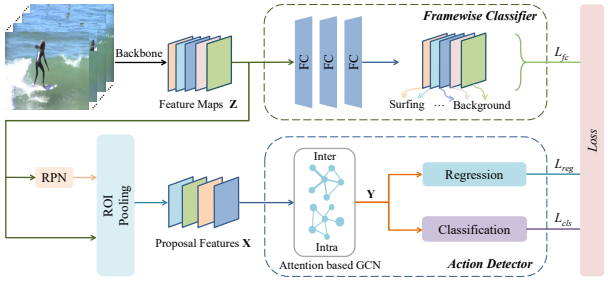

Figure 1: The architecture of P-3DCNNs embedded with our attention based GCN and framewise classifier modules. The model includes backbone subnet, RPN subnet, framewise classifier subnet and action detector subnet.

Method

In this section, we will first introduce the 3DCNNs based video action detection framework equipped with the proposed attention based GCN module, and then elaborate the corresponding technical details.

The Framework

Our proposed temporal action detection framework, called Framewise Classifier Constraint Graph Attention based Proposal 3D ConvNets (FC-AGCN-P-3DCNNs) mainly consists of four parts (Backbone subnet, RPN subnet, Framewise Classifier subnet and Action Detector subnet). The overview of our model is visualized as Figure 1 shown. The input of our FC-AGCN-P-3DCNNs is a long clip of video frames (e.g., 768 frames) and forward them to a 3D ConvNets based backbone. The output of this backbone is the feature maps $\mathbf{Z} = \{\mathbf{z}_i \in \mathbb{R}^{C\times H\times W}, i = 1, \cdots, L\}$, where C, L, H and W denote the number of channels, the length of the feature maps in temporal dimension, height and width in spatial dimension respectively. To generate the anchor temporal action proposals, we use Region Proposal Network (RPN) (Ren et al. 2015; Xu, Das, and Saenko 2017) to predict potential proposal segments with respect to anchor seg-

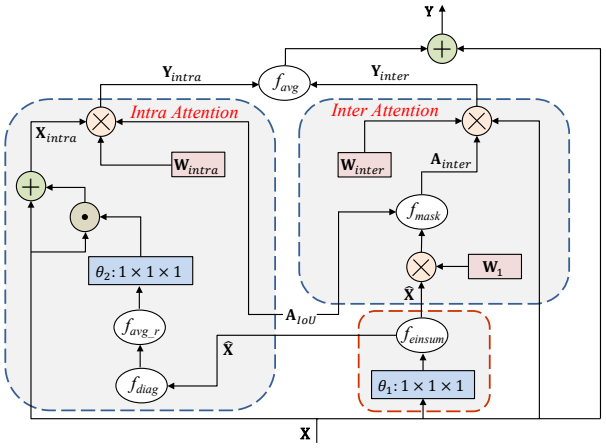

Figure 2: The architecture of our attention based GCN module. This module includes intra attention based GCN and inter attention based GCN submodules.

ments and a binary label indicating whether the predicted proposal contains an action or not. The NMS selected (we omit the NMS in Figure 1 for simplicity) output through RoI Pooling layer to generate the fixed-size proposal features $\mathbf{X} \in \mathbb{R}^{N\times C\times T\times H'\times W'}$ for each variable-length volume proposal, where N is the total number of the temporal action proposals, C, T, H' and W' represent the number of channel, the length in temporal and spatial dimensions for one temporal action proposal, respectively.

The proposal features $\mathbf{X}$ passing through our attention based GCN module will be enhanced into temporal action proposal features $\mathbf{Y} \in \mathbb{R}^{N\times C\times T\times H'\times W'}$. Our attention based GCN module includes inter attention based GCN and intra attention based GCN, which build the corresponding inter and intra dependencies of the temporal action proposals respectively. To be more specific, as shown in Figure 1, intra attention learns the long-range dependencies inside each proposal for node matrix of the graph, while inter attention learns the dependencies between different proposals to form a adaptive adjacency matrix in the graph. And then we regress and classify the feature enhanced proposals $\mathbf{Y}$ to corresponding temporal boundary and activity categories, respectively. As Figure 1 shows, we use loss function L_{cls} and L_{reg} to accomplish the proposal level activity classification and boundary prediction respectively. Besides, for feature maps $\mathbf{Z}$, we use framewise classifier to enhance the backbone feature representation abilities. We will present more detailed design in later section.

Attention based Graph Convolutional Network

In practice, our attention based graph convolutional network modules learn the inter and intra dependencies of the temporal action proposals and enhance the action detection capabilities of the network. As Figure 2 shows, our attention based graph convolutional network is composed of two submodules, that are, intra attention based GCN and inter attention based GCN. In the following paragraphs, we first review the concepts about graph convolutional network. As (Wang and Gupta 2018) shows, a GCN can be written as

$$\mathbf{Y} = \mathbf{AXW}, \tag{1}$$

where $\mathbf{A}$ is the adjacency graph matrix. The node matrix $\mathbf{X}$ represents proposal features, output of the RPN subnet in Figure 1 (here RoI Pooling is omitted for simplicity), and $\mathbf{W}$ is the learnable weight matrix.

As above Equation 1 shows, $\mathbf{A}$ and $\mathbf{X}$ are usually the fixed values for the given graph. Thus, for video action detection, the fixed $\mathbf{A}$ can not express the dynamic dependencies among these temporal action proposals. Motivated by GAT (Busbridge et al. 2017), we propose inter attention to learn the adjacency matrix $\mathbf{A}$ adaptively. At the same time, the pixels inside one temporal action proposal may effect each others. Inspired by non-local operation (Wang et al. 2018), we use intra attention to learn the long-range dependencies in each action proposal and update the node matrix $\mathbf{X}$. In this paper, we learn inter dependences between different temporal action proposals (Inter Attention based GCN) and intra long-range dependences among pixels inside one temporal action proposal (Intra Attention based GCN).

Then we will elaborate our attention based graph convolutional network module, as shown in Figure 2, which includes intra attention based GCN and inter attention based GCN submodules. The inputs and outputs for our graph attention module are the temporal action proposals $\mathbf{X}$ and the enhance proposals $\mathbf{Y}$ respectively. To reduce the expensive computation as well as to learn the dependencies of the temporal action proposals, we use θ_1 and f_{einsum} operations to compute the pairwise similarity or the attention coefficients between every two proposals for both intra and inter attention based GCN, as follows:

$$\widehat{\mathbf{X}} = f_{einsum} \circ \theta_1(\mathbf{X}), \tag{2}$$

where θ_1 is a $1 \times 1 \times 1$ 3D convolution. It simplifies the computation for f_{einsum} by reducing the number of channels, and adds learnable parameters for every temporal action proposals. f_{einsum} is an Einstein summation convention function for many common multi-dimensional, linear algebraic array operations, which compute the relationships between any two parametered action proposals. $\{\widehat{\mathbf{X}}|\hat{\mathbf{x}}_{i,j} \in \mathbb{R}^{TH'W'\times TH'W'}, i,j \in \{1,2,3,...,N\}\}$ is the relation matrix of the temporal action proposals. $\widehat{\mathbf{X}}$ is a $N \times N$ matrix and each of its elements is a $TH'W' \times TH'W'$ matrix. $\hat{\mathbf{x}}_{\mathbf{i},\mathbf{j}}$ denotes the relationship between ith and jth temporal action proposal.

Intra Attention based GCN For intra attention, we want to learn the long-range dependencies among pixels inside each action proposal for each node of node matrix in graph. As we have the relation matrix $\widehat{\mathbf{X}}$ of temporal action proposals, inspired by (Wang et al. 2018), we only need to resize every diagonal elements $\hat{\mathbf{x}}_{i,i}$ in the $\widehat{\mathbf{X}}$ to $C\times T\times H'\times W'$ because the pixels inside each $\hat{\mathbf{x}}_{i,i}$ have relation to each others. To obtain the relations, we use f_{diag} to pick up the diagonal elements, which record the intra relations of every proposals. And then we use f_{avg_r} to calculate the average along

rows or columns for every matrix of these N matrices and resume the channel dimension via function θ_2 ($1 \times 1 \times 1$ 3D convolution), obtaining the response matrix for each temporal action proposal. To utilize the "shortcut connection" to update the nodes $\mathbf{X}$, we use non-linear function $Sigmoid$ to normalize the learned relation matrix as shown in Figure 2. It can be written as

$$\mathbf{X}_{intra} = \mathbf{X} \cdot (1 + Sigmoid \circ \theta_2 \circ f_{avg\_r} \circ f_{diag}(\widehat{\mathrm{X}})). \quad (3)$$

Then, as Figure 2 shows, we can obtain the output $\mathbf{Y}_{intra}$ of our attention based graph convolutional network

$$\mathbf{Y}_{intra} = \mathbf{A}_{IoU}\mathbf{X}_{intra}\mathbf{W}_{intra}, \quad (4)$$

where $\mathbf{W}_{intra}$ is the learnable parameter matrix for graph convolutional network, $\mathbf{A}_{IoU}$ records the Intersection over Union (IoU) for every two temporal action proposals.

Inter Attention based GCN For inter attention, we want to learn the adjacent matrix in the graph, which can reflect the dynamic dependencies among the proposals. The same as intra attention based GCN, we still use $\widehat{\mathbf{X}}$ as the input. Inspired by (Busbridge et al. 2017), to obtain the dynamic adaptive adjacency matrix $\mathbf{A}_{inter}$, we only need to make every elements $\hat{\mathbf{x}}_{\mathbf{i},\mathbf{j}}$ in $\widehat{\mathbf{X}}$ to be a single real valued coefficients. Thus, we use a parametered matrix $\mathbf{W}_1 \in \mathbb{R}^{1 \times TH'W'TH'W'}$ to make $\widehat{\mathbf{X}}$ to be $N \times N$ dimensions matrix with every elements is a real valued coefficient (we omit the matrix reshape operation for simplicity). Then non-linear function $LeakyReLU$ is used to enhance the representation ability. Next transfer to zero the elements whose corresponding proposals have no intersections with each other using the mask function f_{mask}. After that, non-linear function $Softmax$ is used to normalize the adjacency matrix, respectively.

$$\mathbf{A}_{inter} = Softmax \circ f_{mask}(LeakyReLU(\mathbf{W}_1\widehat{\mathrm{X}}), \mathbf{A}_{IoU}).$$

We define the output $\mathbf{Y}_{inter}$ of inter attention based graph convolutional network as follows:

$$\mathbf{Y}_{inter} = \mathbf{A}_{inter}\mathbf{X}\mathbf{W}_{inter}. \quad (5)$$

Intra and Inter Attention based GCN Fusion As Figure 2 shows, we use f_{avg} function to fuse intra attention based GCN Y_{intra} and inter attention based GCN Y_{inter} as follows:

$$\mathbf{Y} = f_{avg}(\mathbf{Y}_{intra}, \mathbf{Y}_{inter}) = \frac{1}{2}(\mathbf{Y}_{intra} + \mathbf{Y}_{inter}). \quad (6)$$

We omit the BN and ReLU operations between Y_{intra} and f_{avg} in Figure 2 and Equation 6 (it is the same with Y_{inter} and f_{avg}), when fusing Y_{intra} and Y_{inter} to form our attention based graph convolutional network. Besides, as we have $\mathbf{A}_{inter}$ and $\mathbf{X}_{intra}$, there is another naive fusion method

$$\mathbf{Y} = \mathbf{A}_{inter}\mathbf{X}_{intra}\mathbf{W}_{other}, \quad (7)$$

where $\mathbf{A}_{inter}$, $\mathbf{X}_{intra}$ and $\mathbf{W}_{other}$ are the learned adjacency matrix, the node matrix and the learnable parameter matrix, respectively. We will discuss the naive fusion method in the experiments section.

Framewise Classifier

In this section, we will narrate our framewise classifier and describe the design of loss function. The goal of our framewise classifier is to constrain the feature maps $\mathbf{Z}$ at frame level and perform frame level classification. Therefore, we need to resume the length of $\mathbf{Z}$ in temporal dimension because the 3D convolutional and 3D pooling layers in backbone subnet have down sampled the size of the feature maps of the input video clip. In general, we can use deconvolutional layers to resume the temporal length. However, up sampling via deconvolution will bring lots of uncertained errors, so it is very difficult to train the up sampling network. As a result of the above considerations, we simply classify feature maps $\mathbf{Z}$ along the temporal axis of $\mathbf{Z}$, rather than the original temporal of the input video clip. In other words, our proposed framewise classifier is aimed at feature maps. The designed framewise classifier is simply and effective.

As Figure 1 shows, our framewise classifier consists of three FC layers. The inputs of the framewise classifier are the feature maps $\mathbf{Z}$. Our framewise classifier classifies the feature maps $\mathbf{Z}$ along temporal dimension. As the temporal length of $\mathbf{Z}$ is L, the batch size of the classifier is L. The first FC layer reduces the feature dimension from $n = C \times H \times W$ into a middle level dimension n', and then the second FC enhances the feature learning. The input and the output number of the second FC are the same, and n' is unchanged. Afterwards, we use the third FC to transfer the feature dimension of $\mathbf{Z}$ from n into the specified class number n'', where n'' is the number of activity categories of the corresponding dataset. Next, we use cross entropy loss to optimize the framewise classifier training. Specifically speaking, for feature maps $\mathbf{Z}$, we define the loss function for the framewise classifier as

$$L_{fc} = \frac{1}{N_{fc}} \sum_{i=1}^{N_{fc}} l_{cls}(\mathbf{a}_i, a_i^*), \quad (8)$$

where $l_{cls}(\mathbf{a}_i, a_i^*)$ is a softmax loss function for feature $\mathbf{z}_i$ classification in feature maps $\mathbf{Z}$. $\mathbf{a}_i$ is the predicted probability vector of the feature $\mathbf{z}_i$ for each category, and a_i^* is the ground truth label. N_{fc} stands for the number of the feature $\mathbf{z}_i$ in $\mathbf{Z}$, equal to L in this paper.

Formulation

Similar to (Xu, Das, and Saenko 2017; Ren et al. 2015), we define the proposals classification and regression loss as follows (since both the classification loss and regression loss of RPN subnet are very similar to those defined below, we omit RPN loss in this paper, however, we reserve the corresponding RPN loss while conducting experiments.)

$$L_{cls} = \frac{1}{N_{cls}} \sum_{i=1}^{N_{cls}} l_{cls}(\mathbf{p}_i, p_i^*), \quad (9)$$

$$L_{reg} = \frac{1}{N_{reg}} \sum_{i=1}^{N_{reg}} \mathbb{I}(p_i^* > 0) l_{reg}(t_i, t_i^*), \quad (10)$$

where $\mathbf{p}_i$ is the predicted probability vector of the ith temporal action proposal for each category, and p_i^* is the ground

truth label, $p_i^* \in \{0, 1, 2, ..., K\}$, K is the number of the categories. $\mathbb{I}(\cdot)$ is the indicator function which takes 1 if the condition holds, or 0 otherwise (here we let the background label be 0 for convenience). l_{reg} is the smooth L1 loss function for temporal action proposals regression task. N_{cls} and N_{reg} are the batch size and the number of the temporal action proposals. $t_i = (t_i^c, t_i^l)$ represents predicted relative offset to temporal action proposal, while $t_i^* = (\hat{t_i^c}, \hat{t_i^l})$ represents the ground truth action region relative offset to temporal action proposal. For the ith temporal action proposal regression, we adopt the parameterizations of the 2 coordinates as follows:

$$\begin{aligned} t_i^c &= (c_i - c_i^a)/l_i, \quad t_i^l = log(l_i/l_i^a), \\ \hat{t_i^c} &= (c_i^* - c_i^a)/l_i^*, \; \hat{t_i^l} = log(l_i^*/l_i^a), \end{aligned} \tag{11}$$

where c_i, c_i^a and c_i^* are for the predicted action region center location, action proposal region center location and ground truth action region center location respectively. l_i, l_i^a and l_i^* are the predicted action region length, action proposal region length and the ground truth action region length respectively.

Thus, the total loss can be written as (we omit RPN subnet classification and regression loss for simplicity.)

$$Loss = \lambda_1 L_{fc} + \lambda_2 L_{cls} + \lambda_3 L_{reg}, \tag{12}$$

where λ_i is used to balance the weights of different modules. We simply set all λ_i to be 1 in our experiment, treating all modules as the same weight in training.

In practice, we first train our framewise classifier module based proposal 3D CNNs without the AGCN module to get the well-trained model parameters. Afterwards, we train the proposal 3D convolutional networks equipped with both AGCN and framewise classifier modules.

Experiments

In this section, we evaluate our proposed FC-AGCN-P-3DCNN model on two popular public datasets (THUMOS 2014, ActivityNet v1.3). Specifically, we use two kinds of proposal 3D ConvNets (C3D or ResNet34 as backbone) to demonstrate the effectiveness of our proposed modules.

THUMOS 2014 (Jiang et al. 2014) It includes two tasks, action recognition and temporal action detection. The temporal action detection task contains 20 activity classes. It includes 2765 trimmed videos of these 20 actions in UCF101 for training, 200 and 213 untrimmed videos with temporal annotations for the validation and the test sets respectively. Most of the videos consist of more than one action instance.

ActivityNet v1.3 (Fabian Caba Heilbron and Niebles 2015) ActivityNet includes two versions (v1.2 and v1.3). In our experiments, we use the latest release 1.3 of ActivityNet, which contains 200 activity categories samples from 203 activity classes and 19994 videos in total. It includes untrimmed video classification and activity detection. It is divided into training, validation and test sets with ratio 2:1:1. It has 10024, 4926 and 5044 videos for training, validation and test sets. However, actually, we have download 9193, 4499 and 4616 videos for training, validation and test sets in that some youtube URLs are now unavailable.

Implementation Details

For both models and datasets, we decompress the videos into frames at 25 frames per second (fps), and create a buffer of 768 frames. The input for the network is the 30.7s video clip.

Backbone with C3D We implement our graph attention module with framewise constraint mainly on R-C3D[1] model, which is written in pytorch. For THUMOS 2014, the learning rate is kept fixed at 10^{-4} for first 3 epochs and is decreased to 10^{-5} for the last 2 epochs. We choose 10 anchor segments with specific scale values [2, 4, 5, 6, 8, 9, 10, 12, 14, 16]. We use Sports-1M pretrained model to initialize the training. For ActivityNet v1.3, the learning rate is still kept fixed at 10^{-4} for first 6 epochs and is decreased to 10^{-5} for the last 2 epochs on ActivityNet v1.3. We choose 37 anchor segments with specific scale values [1, 1.25, 1.5, 1.75, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 7, 8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 28, 32, 36, 40, 44, 52, 60, 68, 76, 84, 92, 100]. We use ActivityNet pretrained model to initialize the training. The learning rate of our module is 10 times larger than basic model for both datasets. The output volume features of RoI Pooling layer with the size $512 \times 4 \times 2 \times 2$

Backbone with ResNet34 We also choose 10 anchor segments for THUMOS 2014, which is the same as C3D backbone. For THUMOS 2014, the learning rate is kept fixed at 10^{-4} for first 4 epochs and is decreased to 10^{-5} for the last 2 epochs. We use UCF-101 pretrained model to initialize the training. Specificially, our AGCN module inserts after layer4 and the input volume features of AGCN module with the size $512 \times 6 \times 2 \times 2$.

Comparison with State-of-the-art Methods

To begin with, we show the comparison results on THUMOS 2014 dataset. We compare FC-AGCN-P-C3D model with state-of-the-art methods, as shown in Table 1, our proposed methods achieve almost superior action detection results and outperforms previous work when IoU larger than 0.2 on the test set of THUMOS 2014. The "Average" column denotes the average value of first 7 columns. Our FC-AGCN-P-C3D model, compared against original R-C3D model, has a 3.7% improvement on average. Simultaneously, our AGCN-P-C3D obtains 2.6% relative improvement, especially better performance improvement for high IoU thresholds. Moreover, we compare with state-of-the-art methods on ActivityNet v1.3 dataset. As Table 2 shows, our FC-AGCN-P-C3D can obtain the best performance.

To further analyze the effectiveness of our AGCN module, we show the every category accuracy at IoU threshold with 0.5 on THUMOS 2014 dataset. From Figure 3, we can see R-C3D equipped with AGCN module obtain better detection accuracy on most categories of THUMOS 2014 dataset, especially, for "Pole Vault", we get 14.1% improvement compared against original R-C3D model.

We also test our proposed module with another backbone (ResNet34) on THUMOS 2014. As Table 3 shows, our proposed model (FC-AGCN-P-ResNet34) obtains improved performance on THUMOS 2014 for all IoU thresholds.

[1] https://github.com/sunnyxiaohu/R-C3D.pytorch

Table 1: mAP comparison of state-of-the-art action detection methods on THUMOS 2014.

Methods	IoU							
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	Average
(Karaman, Seidenari, and D. Bimbo 2014)	4.6	3.4	2.4	1.4	0.9	-	-	-
(Wang, Qiao, and Tang 2014)	18.2	17.0	14.0	11.7	8.3	-	-	-
(Richard and Gall 2016)	39.7	35.7	30.0	23.2	15.2	-	-	-
S-CNN (Shou, Wang, and Chang 2016)	47.7	43.5	36.3	28.7	19.0	-	-	-
(Yeung et al. 2016)	48.9	44.0	36.0	26.4	17.1	-	-	-
(Yuan et al. 2017)	51.0	45.2	36.5	27.8	17.8	-	-	-
(Hou, Sukthankar, and Shah 2017)	51.3	-	43.7	-	22.0	-	-	-
TURN (Gao et al. 2017)	54.0	50.9	44.1	34.9	25.6	-	-	-
R-C3D (Xu, Das, and Saenko 2017)	55.2	55.1	52.8	45.9	34.8	27.3	15.4	40.9
(Alwassel, Caba Heilbron, and Ghanem 2018)	-	-	51.8	42.4	30.8	20.2	11.1	-
TPC (Yang et al. 2018)	-	-	44.1	37.1	28.2	20.6	12.7	-
BSN (Lin et al. 2018)	-	-	53.5	45.0	36.9	28.4	**20.0**	-
(Gleason et al. 2019)	52.1	51.4	49.7	46.1	37.4	26.2	15.2	39.7
DBS (Gao et al. 2019)	56.7	54.7	50.6	43.1	34.3	24.4	14.7	39.8
MGG+SCNN-cls (Liu et al. 2019)	-	-	44.9	37.8	29.9	23.6	15.8	-
BMN+SCNN-cls (Lin et al. 2019)	-	-	45.7	40.2	32.2	24.5	17.0	-
GTAN(C3D) (Long et al. 2019)	**67.2**	**61.1**	56.9	46.5	37.9	-	-	-
AGCN-P-C3D	57.2	57.8	54.4	49.0	38.4	**29.7**	17.7	43.5
FC-AGCN-P-C3D	59.3	59.6	**57.1**	**51.6**	**38.6**	28.9	17.0	**44.6**

Table 2: mAP@0.5 performance comparison of state-of-the-art methods on ActivityNet v1.3 dataset.

Method	validation	test
UPC (Montes et al. 2016)	22.5	22.3
MSB-RNN (Singh et al. 2016)	26.0	17.7
(Li et al. 2017)	19.6	-
R-C3D (Xu, Das, and Saenko 2017)	29.3	29.7
STPN (Nguyen et al. 2018)	29.3	20.1
FC-AGCN-P-C3D	**30.4**	**30.4**

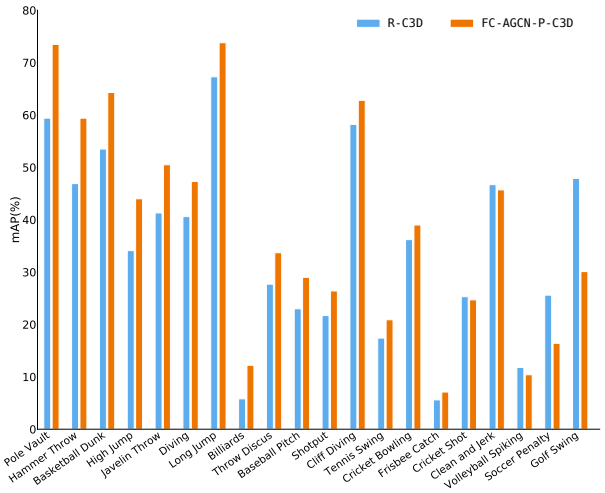

Figure 3: Per-category results for the FC-AGCN-P-C3D model and the R-C3D model with mAP@0.5 on the test set of THUMOS 2014 dataset.

Ablation Study

To demonstrate the reasonableness of our designed module, we analyze the effect of every submodule and some function operations in this subsection.

Inter-attention, Intra-attention, Naive attention fusion based GCN We analyze the performance of only with one submodule (inter-attention based GCN (Inter-AGCN), intra-attention based GCN (Intra-AGCN)) and naive fusion based GCN (Naive-AGCN) on THUMOS 2014 dataset. Based on the Figure 4, we see Intra-AGCN-P-C3D model can obtain improvement performance, compared with original R-C3D module. Naive-AGCN-P-C3D only gets better performance when IoU threshold is larger than 0.3. Although the performance of Inter-AGCN-P-C3D model is worse than original model, fuse inter and intra attention based GCN (AGCN-P-C3D) can get better performance compared against Inter-AGCN-P-C3D, Intra-AGCN-P-C3D and original R-C3D models. At the same time, we study the framewise classifier module embedded in R-C3D model, we can see that FC-P-C3D model can effectively improve the performance, especially at the lower IoU thresholds. At last, we also show results of combining framewise classifier module and our AGCN module (FC-AGCN-P-C3D model). As Figure 4 shows, FC-AGCN-P-C3D model can get the best performance for all the value of IoU threshold.

GCN with and without Attention Mechanism We also compare the performance for regular GCN with and without attention mechanism. As Table 4 shows, the performance of the 1layer-GCN-P-C3D model without attention mechanism only obtains almost the same accuracies with original R-C3D (Xu, Das, and Saenko 2017) model. Inspired by (Wang and Gupta 2018), we also show the 3-layer GCN

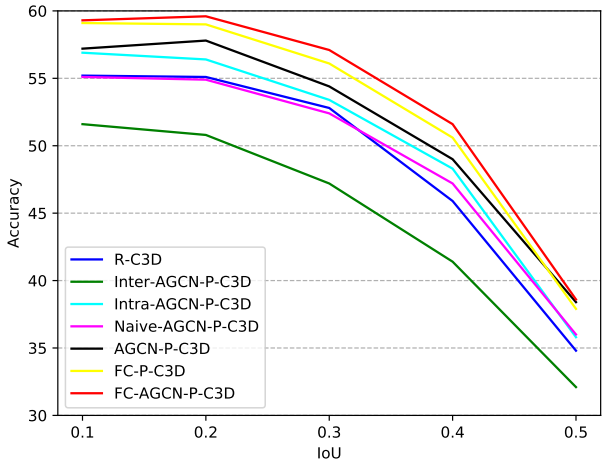

Figure 4: mAP at different overlap IoU thresholds performance comparison of FC (framewise classifier), Inter-AGCN, Intra-AGCN, Naive-AGCN and AGCN modules on THUMOS 2014.

Table 3: mAP comparison of original and our proposed model with ResNet-34 as backbone on THUMOS 2014.

Methods	IoU				
	0.1	0.2	0.3	0.4	0.5
P-ResNet34	54.2	53.1	49.5	43.8	36.9
FC-AGCN-P-ResNet34	**56.7**	**55.4**	**51.9**	**46.7**	**38.2**

Table 4: mAP comparison between GCN with and without attention mechanism on THUMOS 2014.

Methods	IoU				
	0.1	0.2	0.3	0.4	0.5
R-C3D	55.2	55.1	52.8	45.9	34.8
1layer-GCN-P-C3D	55.3	54.8	52.0	46.4	34.7
3layers-GCN-P-C3D	**57.8**	57.6	54.2	47.4	34.8
GAT-P-C3D	55.1	54.8	52.0	46.2	34.4
AGCN-P-C3D	57.2	**57.8**	**54.4**	**49.0**	**38.4**

without attention mechanism (3layers-GCN-P-C3D) model. By contrast, the model (AGCN-P-C3D, 1 layer) with attention mechanism can get almost the best performance. We also show the performance of GAT-P-C3D (Busbridge et al. 2017), which is similar to our inter attention based GCN, but our model achieves less computational complexity.

Choosing Learnable Parameter Matrix or f_{avg_r} Function for Intra Attention based GCN We also compare the results of using learnable parameter matrix with those of using f_{avg_r} function in intra attention based GCN. As Table 5 shows, using learnable parameter matrix (AGCN-P-C3D (lpm)) has the worse temporal action detection accuracy. Thus, simply using average along rows not only reduces the computation complex but also improves performance.

Table 5: mAP comparison between using **l**earnable **p**arameter **m**atrix (lpm) and f_{avg_r} function for intra attention based GCN on THUMOS 2014.

Methods	IoU				
	0.1	0.2	0.3	0.4	0.5
AGCN-P-C3D (lpm)	53.0	53.2	50.2	44.1	33.9
AGCN-P-C3D	**57.2**	**57.8**	**54.4**	**49.0**	**38.4**

Table 6: mAP comparison between using f_{mask} function or not for intra attention based GCN on THUMOS 2014.

Methods	IoU				
	0.1	0.2	0.3	0.4	0.5
AGCN-P-C3D (w/o)	**58.7**	57.2	53.7	46.9	35.8
AGCN-P-C3D	57.2	**57.8**	**54.4**	**49.0**	**38.4**

Using f_{mask} or not We also analysis whether using f_{mask} function or not for inter attention based GCN. As Table 6 shows, without f_{mask} function, AGCN-P-C3D (w/o) model obtains worse performance almost in all overlap IoU thresholds except 0.1. Furthermore, with f_{mask}, the improved performance compared against that without f_{mask} will be larger at the higher value of the overlap IoU threshold. This is because that the temporal action proposals which do not intersect almost have no affects with each other. If we still use vision similarity to create the connected relation for adjacency matrix of GCN, the performance will be worse.

Conclusion

This paper proposes an attention based GCN for action detection in video, solving the problem that the proposal 3D CNNs based video action detection can not utilize the relations of temporal action proposals. Moreover, our AGCN can learn the intra long-range dependencies for every node in graph node matrix and learn the inter dependencies among proposals for adjacency matrix in the graph. Besides, to improve the whole network temporal action detection performance, we introduce the simple and effective framewise classifier module to enhance the backbone presentation capabilities. Compare with state-of-the-art methods, our proposed method can get the best performance.

Acknowledgments

This work was supported by National Natural Science Foundation of China (61690202, 61872021), Fundamental Research Funds for Central Universities (YWF-19-BJ-J-271), Beijing Municipal Science and Technology Commission (Z171100000117022), and State Key Lab of Software Development Environment (SKLSDE-2018ZX-04).

References

Alwassel, H.; Caba Heilbron, F.; and Ghanem, B. 2018. Action search: Spotting actions in videos and its application to temporal action localization. In *ECCV*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Busbridge, D.; Sherburn, D.; Cavallo, P.; and Hammerla, N. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Duchenne, O.; Laptev, I.; Sivic, J.; Bach, F.; and Ponce, J. 2009. Automatic annotation of human actions in video. In *ICCV*.

Fabian Caba Heilbron, Victor Escorcia, B., and Niebles, J. C. 2015. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*.

Gaidon, A.; Harchaoui, Z.; and Schmid, C. 2011. Actom sequence models for efficient action detection. In *CVPR*.

Gao, J.; Yang, Z.; Chen, K.; Sun, C.; and Nevatia, R. 2017. Turn tap: Temporal unit regression network for temporal action proposals. In *ICCV*.

Gao, Z.; Wang, L.; Zhang, Q.; Niu, Z.; Zheng, N.; and Hua, G. 2019. Video imprint segmentation for temporal action detection in untrimmed videos. In *AAAI*.

Gleason, J.; Ranjan, R.; Schwarcz, S.; Castillo, C.; Chen, J.; and Chellappa, R. 2019. A proposal-based solution to spatio-temporal action detection in untrimmed videos. In *WACV*.

Hara, K.; Kataoka, H.; and Satoh, Y. 2017. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? *arXiv preprint arXiv:1711.09577*.

Hou, R.; Sukthankar, R.; and Shah, M. 2017. Real-time temporal action localization in untrimmed videos by sub-action discovery. In *BMVC*.

Jiang, Y.; 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/.

Jiang, B.; Zhang, Z.; Lin, D.; Tang, J.; and Luo, B. 2019. Semi-supervised learning with graph learning-convolutional networks. In *CVPR*.

Karaman, S.; Seidenari, L.; and D. Bimbo, A. 2014. Fast saliency based pooling of fisher encoded dense trajectories. In *ECCV*.

Kipf, T., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Li, W.; Wang, W.; Chen, X.; Wang, J.; and Li, G. 2017. A joint model for action localization and classification in untrimmed video with visual attention. In *ICME*.

Li, G.; Müller, M.; Thabet, A.; and Ghanem, B. 2019. Can gcns go as deep as cnns? *arXiv preprint arXiv:1904.03751*.

Lin, T.; Zhao, X.; Su, H.; Wang, C.; and Yang, M. 2018. Bsn: Boundary sensitive network for temporal action proposal generation. In *ECCV*.

Lin, T.; Liu, X.; Li, X.; Ding, E.; and Wen, S. 2019. Bmn: Boundary-matching network for temporal action proposal generation. *arXiv preprint arXiv:1907.09702*.

Liu, Y.; Ma, L.; Zhang, Y.; Liu, W.; and Chang, S. 2019. Multi-granularity generator for temporal action proposal. In *CVPR*.

Long, F.; Yao, T.; Qiu, Z.; Tian, X.; Luo, J.; and Mei, T. 2019. Gaussian temporal awareness networks for action localization. In *CVPR*.

Montes, A.; Salvador, A.; Pascual, S.; and Giro-i Nieto, X. 2016. Temporal activity detection in untrimmed videos with recurrent neural networks. *arXiv preprint arXiv:1608.08128*.

Nguyen, P.; Liu, T.; Prasad, G.; and Han, B. 2018. Weakly supervised action localization by sparse temporal pooling network. In *CVPR*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*.

Richard, A., and Gall, J. 2016. Temporal action detection using a statistical language model. In *CVPR*.

Scarselli, F.; Gori, M.; Tsoi, A.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *TNN* 20(1):61–80.

Shi, L.; Zhang, Y.; Cheng, J.; and Lu, H. 2019. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*.

Shou, Z.; Wang, D.; and Chang, S. 2016. Temporal action localization in untrimmed videos via multi-stage CNNs. In *CVPR*.

Simonovsky, M., and Komodakis, N. 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *CVPR*.

Singh, B.; Marks, T.; Jones, M.; Tuzel, O.; and Shao, M. 2016. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *CVPR*.

Tian, Y.; Sukthankar, R.; and Shah, M. 2013. Spatiotemporal deformable part models for action detection. In *CVPR*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*.

Wang, X., and Gupta, A. 2018. Videos as space-time region graphs. In *ECCV*.

Wang, F.; Jiang, M.; Qian, C.; Yang, S.; Li, C.; Zhang, H.; Wang, X.; and Tang, X. 2017. Residual attention network for image classification. *arXiv preprint arXiv:1704.06904*.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local neural networks. In *CVPR*.

Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; and Shan, J. 2019. Graph attention convolution for point cloud semantic segmentation. In *CVPR*.

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

X., K.; B., J.; K., R.; C., K.; C., A.; S., R.; S., R.; and B., Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*.

Xu, H.; Das, A.; and Saenko, K. 2017. R-c3d: Region convolutional 3d network for temporal activity detection. In *ICCV*.

Yang, K.; Qiao, P.; Li, D.; Lv, S.; and Dou, Y. 2018. Exploring temporal preservation networks for precise temporal action localization. In *AAAI*.

Yeung, S.; Russakovsky, O.; Mori, G.; and Li, F. 2016. End-to-end learning of action detection from frame glimpses in videos. In *CVPR*.

Yuan, Z.; Stroud, J. C.; Lu, T.; and Deng, J. 2017. Temporal action localization by structured maximal sums. In *CVPR*.