Supplementary Material
Graph Convolutional Networks for Temporal Action Localization

1. Proposal Features

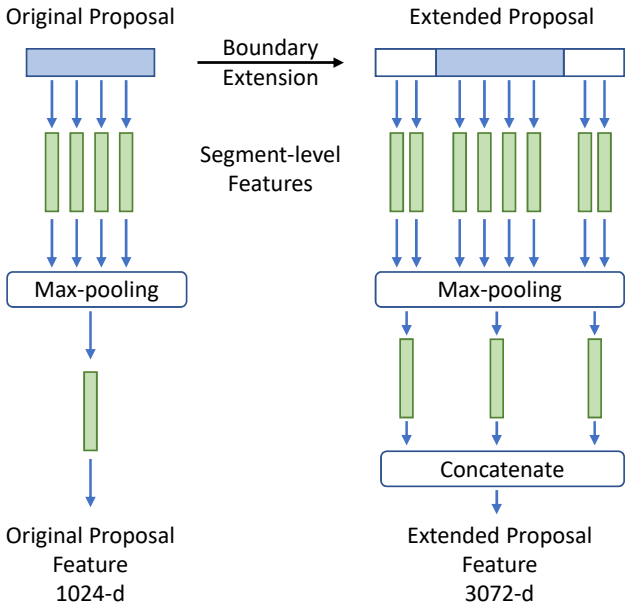


Figure 1. The illustration of (extended) proposal feature extraction.

We have two types of proposal features and the process of feature extraction is shown in Figure 1.

Proposal features. For the original proposal, we first obtain a set of segment-level features within the proposal and then apply max-pooling across segments to obtain one 1024-dimensional feature vector.

Extended proposal features. The boundary of the original proposal is extended with $\frac{1}{2}$ of its length on both the left and right sides, resulting in the extended proposal. Thus, the extended proposal has three portions: *start*, *center* and *end*. For each portion, we follow the same feature extraction process as the original proposal. Finally, the extended proposal feature is obtained by concatenating the feature of three portions.

2. Network Architectures

P-GCN. The network architecture of our P-GCN model is shown in Figure 2. Let N and N_{class} be the number of proposals in one video and the total number of action categories, respectively. On the top of GCN, we have three fully-connected (FC) layers for different purposes. The one with $N_{class} \times 2$ outputs is for boundary regression and the other two with N_{class} outputs are designed for action classification and completeness classification, respectively.

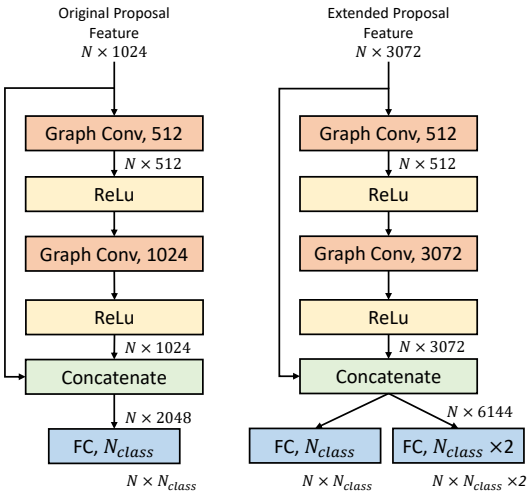


Figure 2. The network architecture of P-GCN model.

MLP baseline. The network architecture of MLP baseline is shown in Figure 3. We replace each of GCNs with a 2-layer multilayer perceptron (MLP). We set the number of parameters in MLP the same as GCN's for a fair comparison. Note that MLP processes each proposal independently without exploiting the relations between proposals.

Mean-Pooling baseline. As shown in Figure 4, the network architecture of Mean-Pooling baseline is the same as the MLP baseline's except that we conduct mean-pooling on the output of MLP over the adjacent proposals.

3. Training Details

We have three types of training samples chosen by two criteria, *i.e.*, the best tIoU and best overlap. For each proposal, we calculate its tIoU with all the ground truth in that video and choose the largest tIoU as the best tIoU (similarly for best overlap). For simplicity, we denote the best tIoU

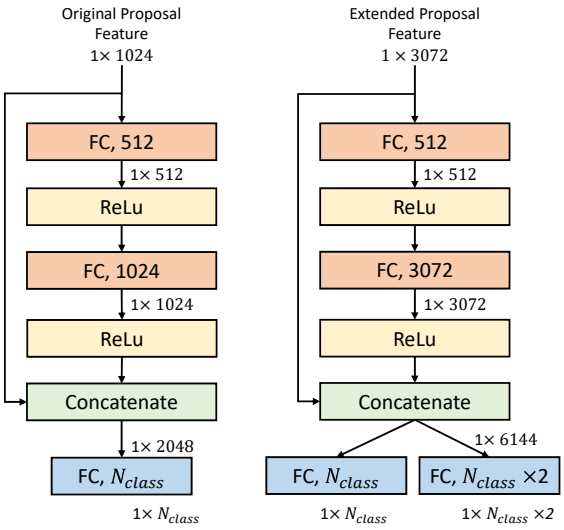


Figure 3. The network architecture of the MLP baseline.

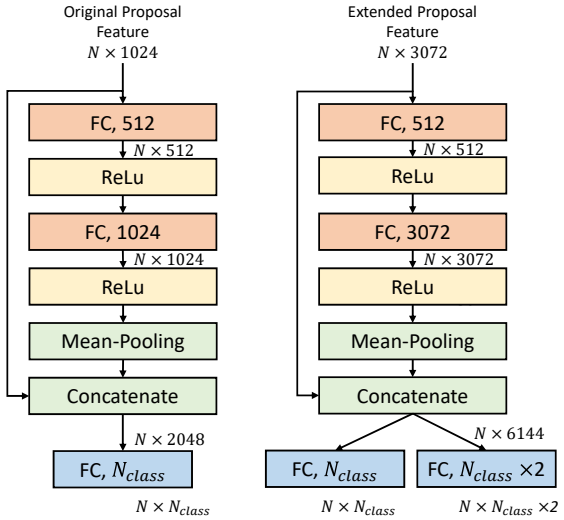


Figure 4. The network architecture of the Mean-Pooling baseline.

and best overlap as tIoU and OL. Then, three types of training samples can be described as: (1) Foreground sample: $tIoU \geq \theta_1$; (2) Incomplete sample: $OL \geq \theta_2, tIoU \leq \theta_3$; (3) Background sample: $tIoU \leq \theta_4$. These certain thresholds are slightly different on two datasets as shown in Table 1. We consider all foreground proposals as the complete proposals.

Table 1. The thresholds on different datasets.

Dataset	θ_1	θ_2	θ_3	θ_4
THUMOS14	0.7	0.7	0.3	0
ActivityNet v1.3	0.7	0.7	0.6	0.1

Each mini-batch contains examples sampled from a single video. The ratio of three types of samples is fixed to (1):(2):(3)=1:6:1. We set the mini-batch size to 32 on THUMOS14 and 64 on ActivityNet v1.3.

For more efficiency, we fix the number of neighborhoods for each node to be 10 by selecting contextual edges with the largest relevance scores and surrounding edges with the smallest distances, where the ratio of contextual and surrounding edges is set to 4:1.

In addition, we empirically found that setting $A_{i,j}$ to 0 (when $A_{i,j} < 0$) leads to better results.

4. Loss function

Multi-task Loss. Our P-GCN model can not only predict action category but also refine the proposal's temporal boundary by location regression. With the action classifier, completeness classifier and location regressors, we define a multi-task loss by:

$$\mathcal{L} = \sum_i \mathcal{L}_{cls}(y_i, \hat{y}_i) + \lambda_1 \sum_i [y_i \geq 1, e_i = 1]\mathcal{L}_{reg}(o_i, \hat{o}_i) + \lambda_2 \sum_i [y_i \geq 1]\mathcal{L}_{com}(e_i, \hat{c}_i), \tag{1}$$

where $\hat{y}_i$ and $y_i \in \{0, \ldots, N_{class}\}$ is the predicted probability and ground truth action label of the i-th proposal in a mini-batch, respectively. Here, 0 represents the background class. e_i is the completeness label. $\hat{o}_i$ and o_i are the predicted and ground truth offset, which will be detailed below. In all experiments, we set $\lambda_1 = \lambda_2 = 0.5$.

Completeness Loss. Here, the completeness term $\mathcal{L}_{com}$ is used only when $y_i \geq 1$, *i.e.*, the proposal is not considered as part of the background.

Regression Loss. We devise a set of location regressors $\{R_m\}_{m=1}^{N_{class}}$, each for an activity category. For a proposal, we regress the boundary using the closest ground truth instance as the target. Our P-GCN model does not predict the start time and end time of each proposal directly. Instead, it predicts the offset $\hat{o}_i = (\hat{o}_{i,c}, \hat{o}_{i,l})$ relative to the proposal, where $\hat{o}_{i,c}$ and $\hat{o}_{i,l}$ are the offset of center coordinate and length, respectively. The ground truth offset is denoted as $o_i = (o_{i,c}, o_{i,l})$ and parameterized by:

$$\begin{aligned} o_{i,c} &= (c_i - c_{gt})/l_i, \\ o_{i,l} &= log(l_i/l_{gt}), \end{aligned} \tag{2}$$

where c_i and l_i denote the original center coordinate and length of the proposal, respectively. c_{gt} and l_{gt} account for the center coordinate and length of the closest ground truth, respectively. $\mathcal{L}_{reg}$ is the smooth L1 loss and used when $y_i \geq 1$ and $e_i = 1$, *i.e.*, the proposal is a foreground sample.

5. Details of Augmentation Experiments on ActivityNet

Our P-GCN model can be further augmented by taking the external video-level labels into account. To achieve this,

we replace the predicted action classes in Eq. (6) with the external action labels. Specifically, given an input video, we use UntrimmedNet to predict the top-2 video-level classes and assign these classes to all the proposals in this video. In this way, each proposal has two action classes.

To further compute mAP, the score of each proposal is required. In our implementation, we follow the settings in BSN by calculating

$$s_{prop} = s_{act} * s_{com} * s_{bsn} * s_{unet}, \tag{3}$$

where s_{act} and s_{com} are the action score and completeness score associated with the action class. s_{bsn} represents the confidence score produced by BSN and s_{unet} denotes for the action score predicted by UntrimmedNet.

6. Explanation and ablation study of θ_{ctx}

The parameter θ_{ctx} is a threshold value for constructing contextual edges, *i.e.* $r(\boldsymbol{p}_i, \boldsymbol{p}_i) > \theta_{ctx}$. Since $r(\boldsymbol{p}_i, \boldsymbol{p}_i) \in [0, 1]$, θ_{ctx} can be chosen from $[0, 1)$. An ablation study is shown in Table 2. Our method performs well when $\theta_{ctx} = 0.7, 0.8, 0.9$.

Table 2. Results on THUMOS14 (Flow) with different θ_{ctx}.

θ_{ctx}	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
mAP(tIoU=0.5)	45.31	45.29	45.37	45.61	45.65	45.82	45.79	46.53	46.64	46.45

7. Ablation study of boundary regression

We conducted an ablation study on boundary regression in Table 3, whose results validate the necessity of using boundary regression.

Table 3. Ablation results of boundary regression on THUMOS14.

mAP@tIoU=0.5	RGB	Flow
without boundary regression	36.4	45.4
with boundary regression	**37.3**	**46.5**

8. Additional runtime compared to [52]

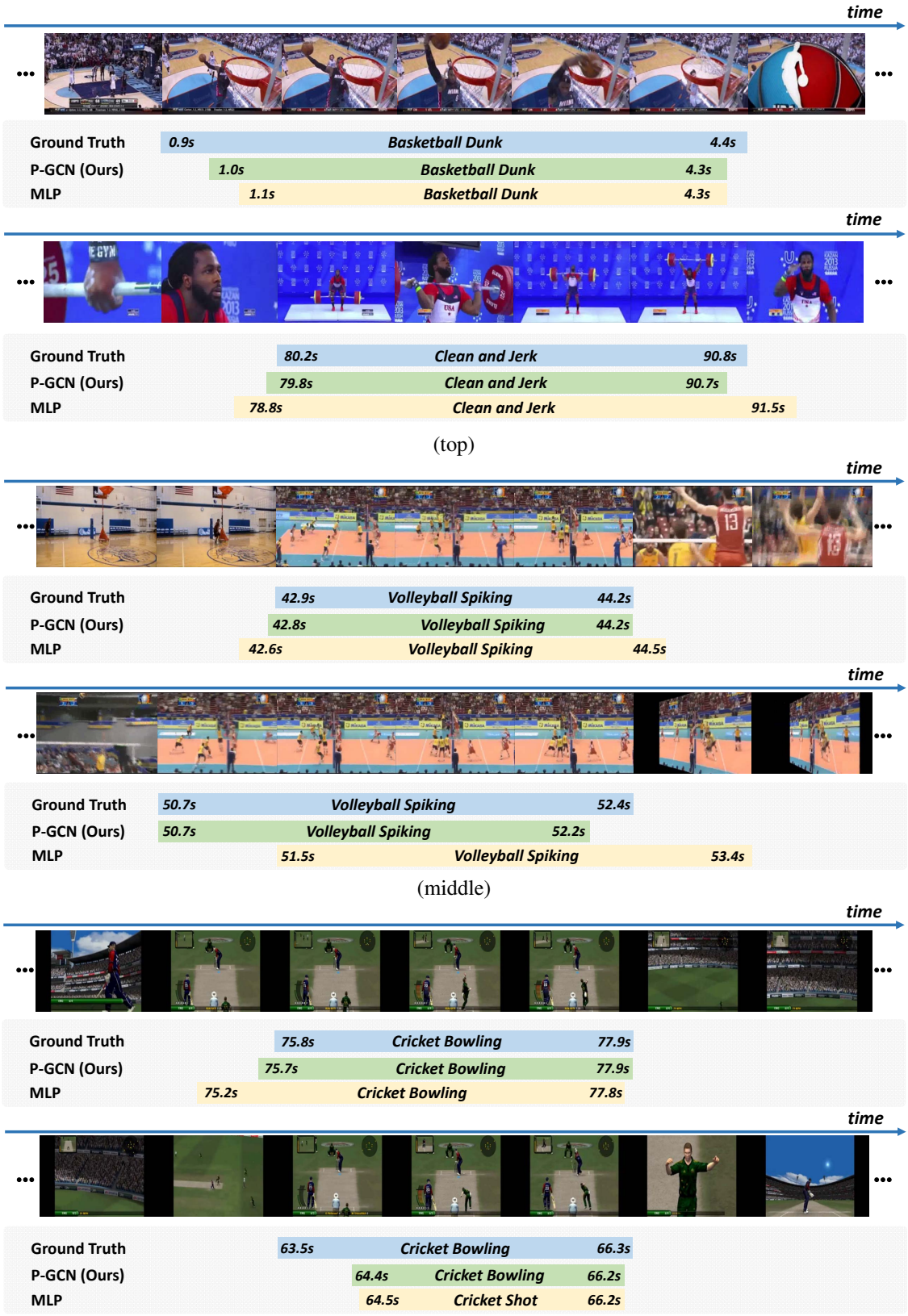
The MLP baseline is indeed a particular implementation of [52], and it shares the same amount of parameters with our P-GCN. We compare the runtime between P-GCN and MLP in Table 4. It reads that GCN only incurs a relatively small additional runtime compared to MLP but is able to boost the performance significantly.

Table 4. Runtime/computation complexity in FLOPs/action localization mAP on THUMOS14. We train each model with 200 iterations on a Titan X GPU and report the average processing time per video per iteration (note that proposal generation and feature extraction are excluded for each model). For P-GCN, we choose the number of sampling neighbourhoods as $N_s = 4$.

Method	Runtime	FLOPs	mAP@tIoU=0.5	
			RGB	Flow
MLP baseline	0.376s	16.57M	34.8	43.7
P-GCN	0.404s	17.70M	37.3	46.5

9. Additional Qualitative Results

Here, we show additional qualitative examples on THUMOS14 in Figure 5. The top and middle examples show that our P-GCN model is able to predict boundary more accurately. The example at the bottom suggests that our method classifies the action correctly despite the similar context between "Cricket Bowling" and "Cricket Shot".



(top)

(middle)

(bottom)

Figure 5. Qualitative results on THUMOS14 dataset.