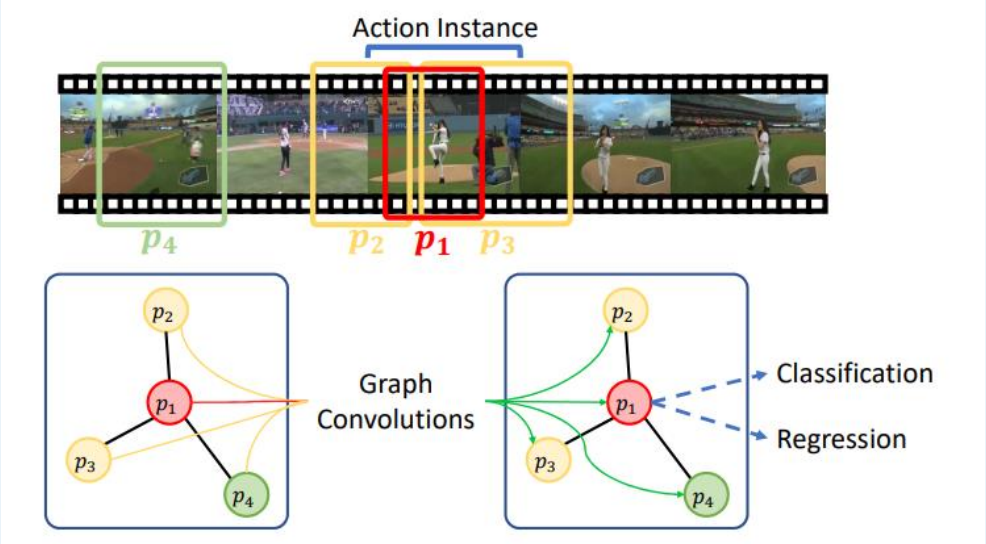
Graph Convolutional Networks for Temporal Action Localization (ICCV 2019)

Seulki Park

Dept. of Electrical and Computer Engineering

Seoul National University



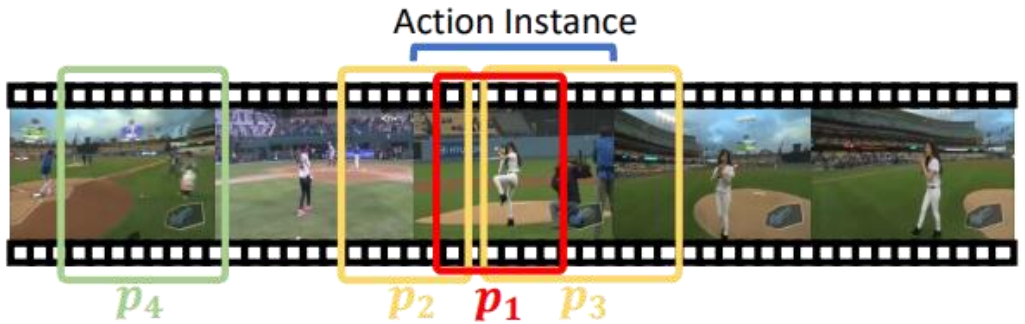
Temporal Action Localization

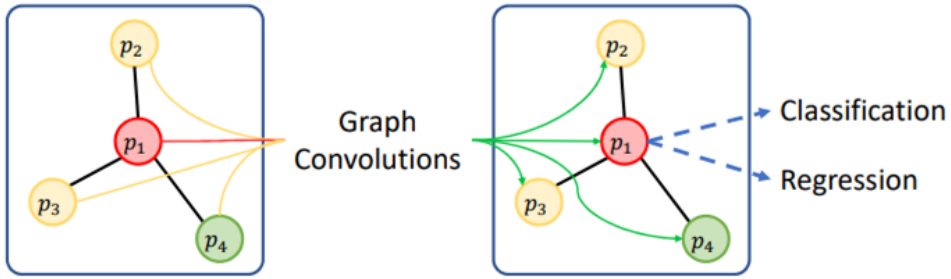
Longboarding

- Given an untrimmed video, the goal is to
  - 1) classify the actions of interest
  - 2) localize the start and end time of every action instance
- Prominent research topic in computer vision with various applications (e.g., security surveillance, human behavior analysis)
- Two-stage framework
  - Proposal generation
  - Classification and boundary regression

Picture from: Deep Learning for Video: Action Detection (UPC 2018)

Intuition



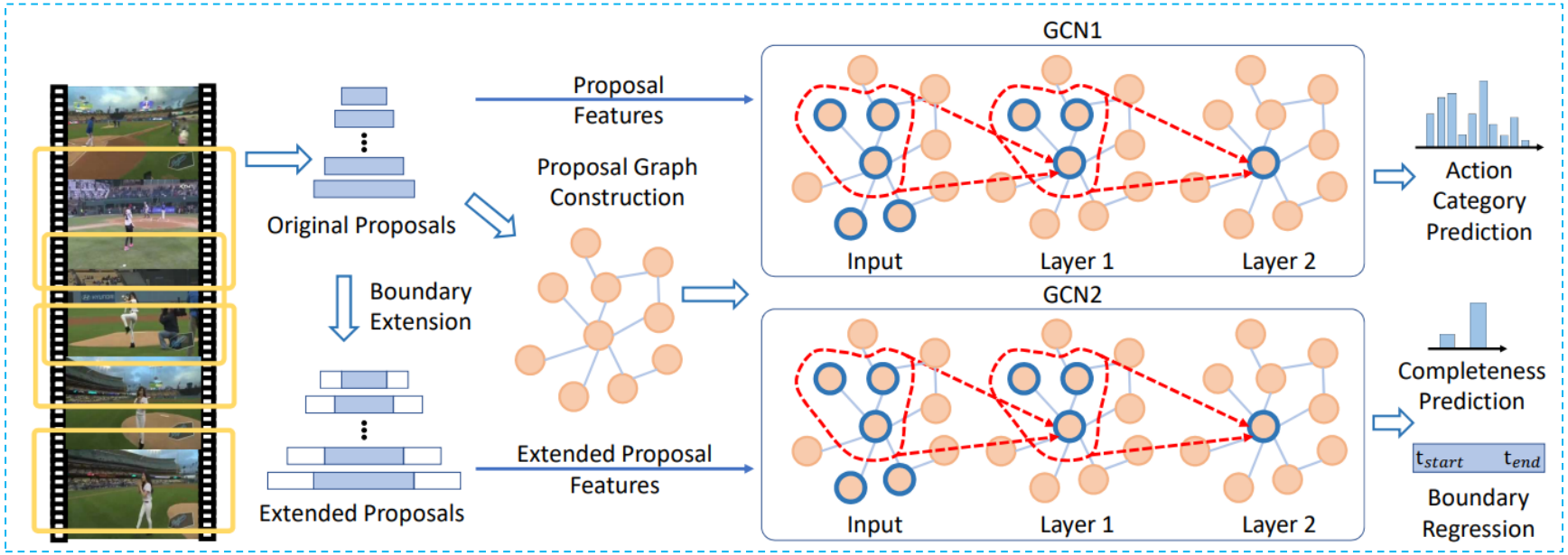


- Limitation of existing methods:
  - process each proposal separately that neglect the context information of proposals.
  - e.g., conventional methods perform prediction on p_1 by using its feature alone.
- Motivation:
  - Taking the features of p_2, p_3, p_4 will give more contextual information!
  - p_2, p_3 → especially for the temporal boundary regression of p_1
  - p_4 → can partly guide the classification of p_1 (e.g. sport field)
  - GCNs can affect each node by aggregating information from the adjacent nodes

→ Very suitable for leveraging the relations between proposals!!

Proposed Method

Schematic of P-GCN model



- Input: an input video V
- Output: the action category $\hat{y}_i$, temporal position $(\hat{t}_{i,s}, \hat{t}_{i,e})$ for each proposal p_i
- Such that: exploiting proposal relations.
- Formulation: $\left\{\left(\hat{y}_i, \left(\hat{t}_{i,s}, \hat{t}_{i,e}\right)\right)\right\}_{i=1}^{N} = F\left(GCN\left(\{x_i\}_{i=1}^{N}, \mathcal{G}(\mathcal{P}, \mathcal{E})\right)\right)$,
- F : mapping function to be learned, $\mathcal{G}(\mathcal{P}, \mathcal{E})$: proposal graph of N nodes
- Assume the action proposals have been obtained beforehand & feature extractor exists

How to construct graph?

- For the graph $\mathcal{G}(\mathcal{P}, \mathcal{E})$ of each video,
  - Node v_i: instantiated as the action proposals $\boldsymbol{p_i}$, where $P = \left\{ \boldsymbol{p_i} \middle| \boldsymbol{p_i} = \left(\boldsymbol{x_i}, \left(t_{i,s}, t_{i,e} \right) \right) \right\}_{i=1}^{N}$
  - Edges $e_{i,j}$: ?
- How to characterize to better model the proposal relations?
  - Linking all proposals with each other → heavily computational & redundant/noisy info.
- Contextual Edges
  - Establish an edge between proposal $\boldsymbol{p_i}$ and $\boldsymbol{p_j}$ if $r(\boldsymbol{p_i}, \boldsymbol{p_j}) > \theta_{ctx}$, θ_{ctx} is a threshold.
  - $r(\boldsymbol{p_i}, \boldsymbol{p_j}) = tIOU(\boldsymbol{p_i}, \boldsymbol{p_j}) = {}^{I(\boldsymbol{p_i}, \boldsymbol{p_j})}/_{U(\boldsymbol{p_i}, \boldsymbol{p_j})}$, $I(.)$: Intersection, U(.): Union of two proposals.
  - Selecting overlapping proposals will provide and share rich contextual information.
- Surrounding Edges
  - As a complement, surrounding edges enable the message to pass across distinct action instances and thereby provides more temporal cues for the detection.
  - First utilize $r(\boldsymbol{p_i}, \boldsymbol{p_j}) = 0$ to query distinct proposals, and add edges if $d(\boldsymbol{p_i}, \boldsymbol{p_j}) < \theta_{sur}$
  - $d(\boldsymbol{p_i}, \boldsymbol{p_j}) = {}^{|c_i - c_j|}/_{U(\boldsymbol{p_i}, \boldsymbol{p_j})}$, c_i: center coordinate of $\boldsymbol{p_i}$.

Graph Convolution for Action Localization

- For the k -th layer in K-layer graph convolution network,
- $X^{(k)} = AX^{(k-1)}W^{(k)}$.
  - A : adjacency matrix.
  - $X^{(k)} \in \mathbb{R}^{N \times d_k}$: the hidden features for all proposals at layer k.
  - $W^{(k)} \in \mathbb{R}^{d_k \times d_k}$: the parameter matrix to be learned.
- Then, the first GCN for action category prediction is formulated as:
  - $\{(\hat{y}_i)\}_{i=1}^{N} = softmax\left(FC_1\left(GCN_1\left(\{x_i\}_{i=1}^{N}, \mathcal{G}(\mathcal{P}, \mathcal{E})\right)\right)\right)$,
- The second GCN for boundary regression is formulated as:
  - $\left\{\left(\hat{t}_{i,s}, \hat{t}_{i,e}\right)\right\}_{i=1}^{N} = FC_2\left(GCN_2\left(\{x'_i\}_{i=1}^{N}, \mathcal{G}(\mathcal{P}, \mathcal{E})\right)\right)$,
  - $\left\{\left(\hat{c}_{i,}\right)\right\}_{i=1}^{N} = FC_3\left(GCN_2\left(\{x'_i\}_{i=1}^{N}, \mathcal{G}(\mathcal{P}, \mathcal{E})\right)\right)$
  - $\hat{c}_{i,}$: predicting the proposal is complete or not.
- Efficient training by SAGE method.

Experimental results (1) Action Localization results compared with SOTA

- Dataset: THUMOS14, ActivityNet
  - THUMOS14 :
    - Training set: 13,320 trimmed videos with 101 action categories.
    - Background data: 2,980 untrimmed videos
    - Validation data: 2,104 untrimmed videos of 20 action categories with one to multiple actions.
- Evaluation Metric: mean Average Precision (mAP):
  - Reached the highest mAP over all thresholds, implying that the method can recognize and localize actions much more accurately.

Table 1. Action localization results on THUMOS14, measured by mAP (%) at different tIoU thresholds α.

tIoU	0.1	0.2	0.3	0.4	0.5
Oneata *et al.* [29]	36.6	33.6	27.0	20.8	14.4
Wang *et al.* [40]	18.2	17.0	14.0	11.7	8.3
Caba *et al.* [3]	-	-	-	-	13.5
Richard *et al.* [31]	39.7	35.7	30.0	23.2	15.2
Shou *et al.* [34]	47.7	43.5	36.3	28.7	19.0
Yeung *et al.* [48]	48.9	44.0	36.0	26.4	17.1
Yuan *et al.* [49]	51.4	42.6	33.6	26.1	18.8
Escorcia *et al.* [11]	-	-	-	-	13.9
Buch *et al.* [2]	-	-	37.8	-	23.0
Shou *et al.* [33]	-	-	40.1	29.4	23.3
Yuan *et al.* [50]	51.0	45.2	36.5	27.8	17.8
Buch *et al.* [1]	-	-	45.7	-	29.2
Gao *et al.* [18]	60.1	56.7	50.1	41.3	31.0
Hou *et al.* [21]	51.3	-	43.7	-	22.0
Dai *et al.* [8]	-	-	-	33.3	25.6
Gao *et al.* [17]	54.0	50.9	44.1	34.9	25.6
Xu *et al.* [46]	54.5	51.5	44.8	35.6	28.9
Zhao *et al.* [52]	66.0	59.4	51.9	41.0	29.8
Lin *et al.* [27]	-	-	53.5	45.0	36.9
Chao *et al.* [6]	59.8	57.1	53.2	48.5	42.8
P-GCN	**69.5**	**67.8**	**63.6**	**57.8**	**49.1**

Experimental results (2)

- Is Proposal-Proposal relation helpful?
- Is Graph Convolution helpful?

Table 3. Comparison between our P-GCN model and MLP on THUMOS14, measured by mAP (%).

mAP@tIoU=0.5	RGB	Gain	Flow	Gain
$MLP_1 + MLP_2$	34.75	-	43.68	-
$MLP_1 + GCN_2$	35.94	1.19	44.59	0.91
$GCN_1 + MLP_2$	35.82	1.07	45.26	1.58
P-GCN ($GCN_1 + GCN_2$)	**37.27**	**2.52**	**46.53**	**2.85**

Table 4. Comparison between our P-GCN model and mean-pooling (MP) on THUMOS14, measured by mAP (%).

mAP@tIoU=0.5	RGB	Gain	Flow	Gain
$MP_1 + MP_2$	35.32	-	43.97	-
$MP_1 + GCN_2$	36.50	1.18	45.78	1.81
$GCN_1 + MP_2$	36.22	0.90	44.42	0.45
P-GCN ($GCN_1 + GCN_2$)	**37.27**	**1.95**	**46.53**	**2.56**

- Qualitative results

Figure 4. Qualitative results on THUMOS14 dataset.

Conclusion

- First to exploit the proposal-proposal relations for temporal action localization in videos.
- Construct a graph of proposals, and then apply GCNs to do message aggregation among proposals.
- The proposed method verified the effectiveness that significantly outperforms the state-of-the-art. (i.e., 49.1% vs. 42.8%)

논문 Reproducing – 1) Data 준비하기

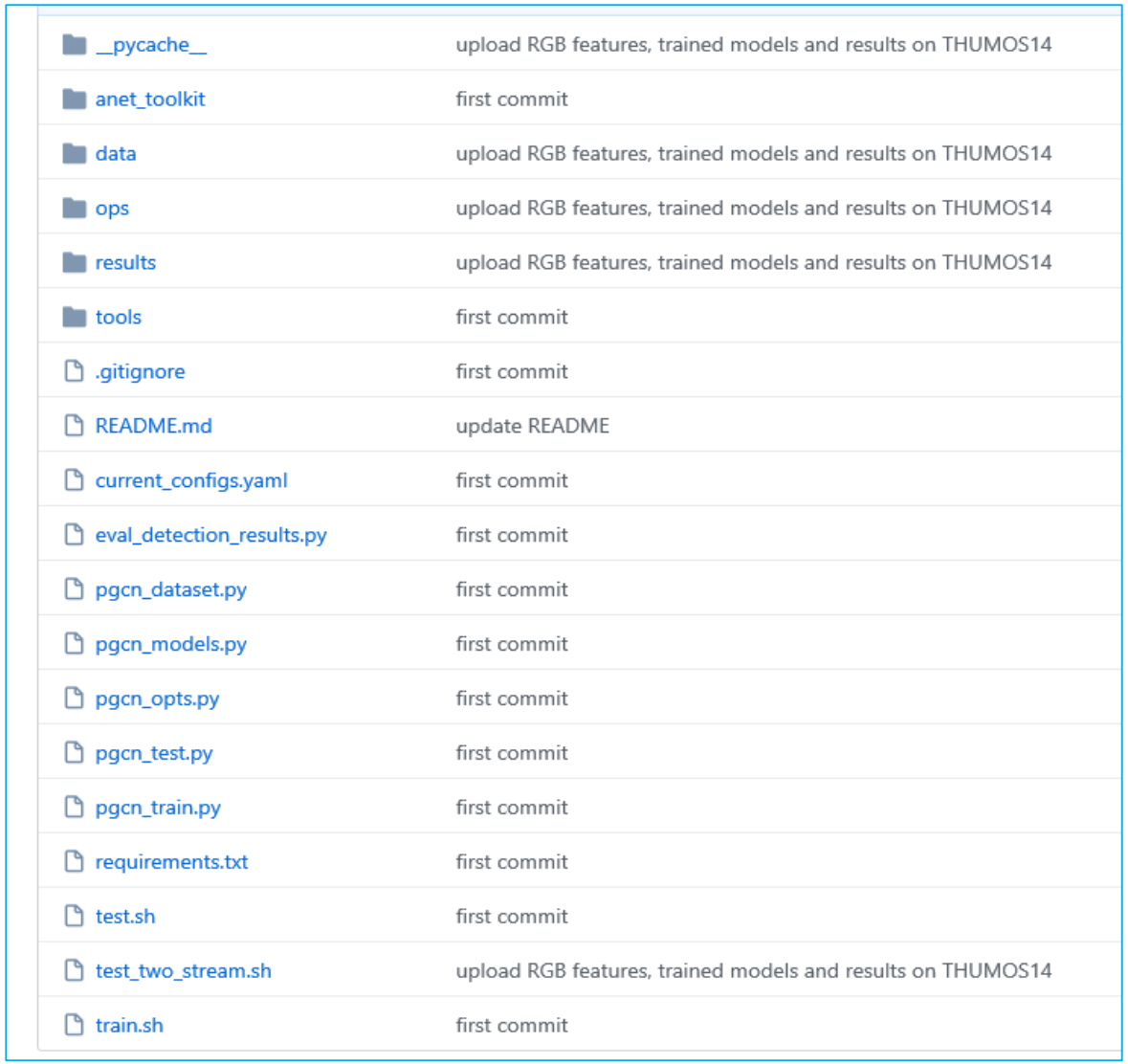
- **Github 주소:** https://github.com/Alvin-Zeng/PGCN
- **코드 사용 방법**:
  - Dataset: 직접 THUMOS14, ActivityNet v1.3데이터로부터 Proposal로 사용할 feature를 추출해도 되지만, 저자가 I3D feature를 제공하고 있으므로 이를 활용할 수 있습니다.
    (다운로드 링크: https://drive.google.com/drive/folders/1C6829qlU_vfuiPdJSqHz3qSqqc0SDCr_)
  - 위에서 다운받은 feature의 위치를 'data/dataset_cfg.yaml' 파일에 아래와 같이 업데이트 합니다.

```
thumos14:
  dataset_configs:

    train_ft_path: /home/pseulki/Experiments/gcn_final/i3d_models/Flow_Train_All
    test_ft_path: /home/pseulki/Experiments/gcn_final/i3d_models/Flow_Test_All

    train_dict_path: data/thumos14_train_prop_dict.pkl
    val_dict_path: data/thumos14_val_prop_dict.pkl
    test_dict_path: data/thumos14_test_prop_dict.pkl
    train_prop_file: data/bsn_train_proposal_list.txt
    test_prop_file: data/bsn_test_proposal_list.txt
```

논문 Reproducing – 2) Structure



__pycache__	upload RGB features, trained models and results on THUMOS14
anet_toolkit	first commit
data	upload RGB features, trained models and results on THUMOS14
ops	upload RGB features, trained models and results on THUMOS14
results	upload RGB features, trained models and results on THUMOS14
tools	first commit
.gitignore	first commit
README.md	update README
current_configs.yaml	first commit
eval_detection_results.py	first commit
pgcn_dataset.py	first commit
pgcn_models.py	first commit
pgcn_opts.py	first commit
pgcn_test.py	first commit
pgcn_train.py	first commit
requirements.txt	first commit
test.sh	first commit
test_two_stream.sh	upload RGB features, trained models and results on THUMOS14
train.sh	first commit

- 코드 실행에 필요한 소스코드는 다음과 같습니다.
- pgcn_dataset.py: Dataset을 읽어와서 training에 필요한 형태로 data를 정제하고 그래프를 만드는 역할을 한다. (proposal 추출, regression 계산, IOU계산을 통한 edge 구성, 노드 샘플링 등의 함수를 포함.)
- pgcn_train.py: graph convolution network 모델을 생성하고, 트레이닝하는 역할을 한다. (pgcn_models.py와 중복이므로 이 파일만 보면 됨.)
- pgcn_test.py & eval_detection_results.py: 학습된 모델을 테스트할 수 있는 코드.
- pgcn_opts: 파일을 실행시키는데 사용되는 hyperparameter를 모아 놓았으므로, 이 파일을 참조하여 원하는 대로 learning rate, epoch 수 등을 변경할 수 있음.
- requirements.txt: 코드 실행을 위해 필요한 package dependency를 확인할 수 있음.

논문 Reproducing – 3) 구현

- 앞에서와 같이 데이터 위치 지정이 끝나면 간단하게 다음과 같이 Training 시킬 수 있습니다.

python pgcn_train.py (필수:데이터) (옵션)

(예) python pgcn_train.py thumos14 –snapshot_pre .

```
(pip36) ➜ ~/Experiments/gcn_final/PGCN git:(master) ✗ python pgcn_train.py thumos14 --snapshot_pref .                [1504/1553]
2020-06-19 09:29:41,764 INFO
creating folder: .
|===>Backups using time: 0.180 s
|===>Backups will be saved at .
2020-06-19 09:29:41,947 INFO
runtime args

Namespace(batch_size=32, clip_gradient=None, comp_loss_weight=0.5, dataset='thumos14', dataset_configs=ordereddict([('train_ft_path', '/home/pseulki/Experiments/gcn_final/i3d_models/Flow_Train_All'), ('test_ft_p
ath', '/home/pseulki/Experiments/gcn_final/i3d_models/Flow_Test_All'), ('train_dict_path', 'data/thumos14_train_prop_dict.pkl'), ('val_dict_path', 'data/thumos14_val_prop_dict.pkl'), ('test_dict_path', 'data/thu
mos14_test_prop_dict.pkl'), ('train_prop_file', 'data/bsn_train_proposal_list.txt'), ('test_prop_file', 'data/bsn_test_proposal_list.txt'), ('training_epoch_multiplier', 20), ('testing_epoch_multiplier', 1), ('f
g_iou_thresh', 0.7), ('bg_iou_thresh', 0.01), ('incomplete_iou_thresh', 0.3), ('bg_coverage_thresh', 0.02), ('incomplete_overlap_thresh', 0.01), ('prop_per_video', 8), ('fg_ratio', 1), ('bg_ratio', 1), ('incompl
ete_ratio', 6), ('iou_threshold', 0.7), ('dis_threshold', 0), ('starting_ratio', 0.5), ('ending_ratio', 0.5)]), dropout=0.8, epochs=70, eval_freq=5, evaluate=False, evaluation=ordereddict([('top_k', 2000), ('nms
_threshold', 0.2)]), gpus=None, graph_configs=ordereddict([('adj_num', 21), ('child_num', 4), ('iou_num', 8), ('dis_num', 2)]), iter_size=1, lr=0.01, lr_steps=[15, 30, 45], model_configs=ordereddict([('num_class
', 20), ('act_feat_dim', 1024), ('comp_feat_dim', 3072), ('dropout', 0.8), ('gcn_dropout', 0.7)]), momentum=0.9, print_freq=20, reg_loss_weight=0.5, resume='', snapshot_pref='.', start_epoch=0, training_epoch_mu
ltiplier=10, weight_decay=0.0005, workers=4)

config

ordereddict([('train_ft_path', '/home/pseulki/Experiments/gcn_final/i3d_models/Flow_Train_All'), ('test_ft_path', '/home/pseulki/Experiments/gcn_final/i3d_models/Flow_Test_All'), ('train_dict_path', 'data/thumos
14_train_prop_dict.pkl'), ('val_dict_path', 'data/thumos14_val_prop_dict.pkl'), ('test_dict_path', 'data/thumos14_test_prop_dict.pkl'), ('train_prop_file', 'data/bsn_train_proposal_list.txt'), ('test_prop_file',
 'data/bsn_test_proposal_list.txt'), ('training_epoch_multiplier', 20), ('testing_epoch_multiplier', 1), ('fg_iou_thresh', 0.7), ('bg_iou_thresh', 0.01), ('incomplete_iou_thresh', 0.3), ('bg_coverage_thresh', 0.
02), ('incomplete_overlap_thresh', 0.01), ('prop_per_video', 8), ('fg_ratio', 1), ('bg_ratio', 1), ('incomplete_ratio', 6), ('iou_threshold', 0.7), ('dis_threshold', 0), ('starting_ratio', 0.5), ('ending_ratio',
 0.5)])
File parsed. Time:5.81
Dict constructed. Time:26.37
File parsed. Time:1.43
Dict constructed. Time:1.64
2020-06-19 09:30:20,038 INFO group: normal_weight has 7 params, lr_mult: 1, decay_mult: 1
2020-06-19 09:30:20,038 INFO group: normal_bias has 7 params, lr_mult: 2, decay_mult: 0
2020-06-19 09:30:22,713 INFO Epoch: [0][0/124], lr: 0.01000     Time 2.674 (2.674)      Data 2.480 (2.480)      Loss 3.8905 (3.8905)    Act. Loss 3.044 ( 3.044)        Comp. Loss 1.006 ( 1.006)       Reg. Loss
0.687 (0.687)
 Act. FG 8.82 (8.82) Act. BG 0.00 (0.00)
2020-06-19 09:30:30,421 INFO Epoch: [0][20/124], lr: 0.01000    Time 1.072 (0.494)      Data 0.942 (0.359)      Loss 2.8851 (3.1872)    Act. Loss 2.019 ( 2.287)        Comp. Loss 0.965 ( 0.990)       Reg. Loss
0.768 (0.810)
 Act. FG 11.11 (7.79) Act. BG 94.21 (94.21)
2020-06-19 09:30:37,705 INFO Epoch: [0][40/124], lr: 0.01000    Time 1.033 (0.431)      Data 0.912 (0.297)      Loss 2.0414 (2.8122)    Act. Loss 1.177 ( 1.916)        Comp. Loss 0.886 ( 0.964)       Reg. Loss
0.843 (0.828)
 Act. FG 50.00 (20.29) Act. BG 95.52 (95.52)
2020-06-19 09:30:45,259 INFO Epoch: [0][60/124], lr: 0.01000    Time 1.101 (0.413)      Data 0.969 (0.280)      Loss 2.1477 (2.5702)    Act. Loss 1.191 ( 1.682)        Comp. Loss 0.898 ( 0.944)       Reg. Loss
1.015 (0.833)
 Act. FG 44.44 (28.79) Act. BG 94.46 (94.46)
2020-06-19 09:30:52,908 INFO Epoch: [0][80/124], lr: 0.01000    Time 1.120 (0.406)      Data 0.991 (0.272)      Loss 2.0085 (2.3995)    Act. Loss 1.208 ( 1.526)        Comp. Loss 0.894 ( 0.922)       Reg. Loss
0.706 (0.825)
```

- 아래와 같은 결과를 확인했습니다.

```
2020-06-19 11:01:36,107 INFO Testing Results: Loss 1.52620       Activity Loss 0.765      Completeness Loss 0.816
Act Accuracy 79.76 FG Acc. 79.60 BG Acc. 80.00    Regression Loss 0.708
```