A PIPELINE OF ENERGY-EFFICIENT ACTION DETECTION

A Thesis

by

XIN HU

Submitted to the Graduate and Professional School of
TexasA&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Anxiao(Andrew) Jiang
Committee Members,	Jiang Hu
	Tie Liu
	Ruihong Huang
Head of Department,	Miroslav M. Begovic

December 2021

Major Subject: Computer Engineering

Copyright 2021 Xin Hu

ABSTRACT

Action detection has been an essential topic in computer vision tasks for the last decade. There is lots of research done to get high accuracy in action detection based on image features. However, image features consume heavy computation and energy. For those models deployed on edge devices, image features are not very suitable and applicable. A less computational method based on the skeleton is proposed by [1]. Compared to image features, skeleton features require significantly fewer flops. Nonetheless, most skeleton-based action detection research focuses on existing datasets. Little research has been put into running action detection on mobile devices. In this thesis, a pipeline in energy-efficient action detection is proposed to achieve action detection on unmanned aerial vehicles (UAVs). The environmental platform is Raspberry Pi. This pipeline could detect person real-time on Raspberry Pi and classify action through a skeleton-based action classification network. There is lots of research in this area for object detection, including deploying models on edge devices. However, most research considers multi-classes and generalize, and the network structure is not light enough in this situation. This thesis proposes a specific detection network with higher FPS but lower flops. Besides, a skeleton-based spatial-temporal transformer network is also proposed in this thesis. Action classification network consists of a graph convolution network and a multi-head transformer module. Gaussian distribution is introduced as weak supervision in action classification. The pipeline in this thesis consists of three parts: proposed object detection network, existing light pose estimation network, and skeleton-based action classification network. This pipeline achieves an end-to-end structure on mobile devices. To speed up inference, this thesis also prunes and quantizes each model. This pipeline has been tested with deployment on a Raspberry Pi and videos recorded by UAV under a public environment. An energy-measuring system is established to measure energy consumption.

DEDICATION

To my mother and my father.

ACKNOWLEDGMENTS

I want to express my deep gratitude to my research supervisor, Dr. Anxiao(Andrew) Jiang, for his patient support through my master study and research, for his encouragement and enthusiastic motivation.

I want to extend my deep appreciation to my committee members, Dr. Hu, Dr. Liu, and Dr. Huang. Without their support, this thesis could not have been completed. I also would like to say thanks to Dr. Shi. I got a lot of precious suggestions from him for my research.

Thanks also to the members of the LPCV team at Texas A&M University. They provided a lot of help and support for this project.

Finally, I would like to thank my parents for their continuous help and support throughout my master's study.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Professor Anxiao Jiang [chair] and Professor Ruihong Huang of the Department of Computer Science and Engineering and Professor Jiang Hu and Professor Tie Liu of the Department of Electrical and Computer Engineering.

Object detection and person ReID parts were conducted with the LPCV team at Texas A&M University.

All other works conducted for the thesis were completed by the student independently.

Funding Sources

There are no funding contributions to acknowledge related to the research and compilation of this thesis.

NOMENCLATURE

CNN	Convolutional Neural Network
DNN	Deep Neural Network
mAP	Mean Average Precision
FLOPs	Floating-point Operations
GT	Ground Truth
ResNet	Residual Neural Network
PANet	Path Aggregation Network
FPS	Frames Per Second
MSE	Mean Squared Error

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iii

ACKNOWLEDGMENTS iv

CONTRIBUTORS AND FUNDING SOURCES v

NOMENCLATURE vi

TABLE OF CONTENTS vii

LIST OF FIGURES ix

LIST OF TABLES xi

1. INTRODUCTION 1

2. LITERATURE REVIEW 3

    2.1 Object Detection 3
    2.2 Pose Estimation 3
    2.3 Skeleton Based Action Classification 4
    2.4 Temporal Action Localization 4

3. PROPOSED METHODOLOGY 5

    3.1 Energy-Efficient Object Detection 5
        3.1.1 YOLO-Mobile v1 5
        3.1.2 YOLO-Mobile v2 6
        3.1.3 Person ReID 7
    3.2 Action Classification 8
    3.3 Weakly Supervised Loss Function 11

4. EXPERIMENTS AND DATASETS 13

    4.1 Datasets 13
    4.2 Experiments 14
        4.2.1 Energy-Efficient Object Detection and ReID 14
            4.2.1.1 Evaluation Metrics 14

4.2.1.2 Training on Different Datasets 14
4.2.1.3 Experiments and Results 16
4.2.2 Action Classification 23
4.2.2.1 Evaluation Metrics 23
4.2.2.2 Experiments and Results 23

5. CONCLUSION 30

REFERENCES 31

LIST OF FIGURES

FIGURE Page

3.1 Structure of YOLOv5 6

3.2 Structure of YOLO-Mobile v1 6

3.3 Structure of YOLOv3 7

3.4 Structure of YOLO-Mobile v2 7

3.5 Overview of ReID Model. Query image and gallery images are input into ResNet-18 to extract image features. With cosine distance between query feature and gallery features, distance score is sorted to determine the best gallery image. 8

3.6 Overview of adjacency matrix. **(a)** is joint position for human pose. Red edge represents 1 walk distance from head. Blue edges represents 2 walk distance from left hand to right hand. **(b)** is original adjacency matrix with 1 walk distance. **(c)** is adjacency matrix with 2 walk distance. **(d)** is proposed weighted matrix based on real world distance. 9

3.7 Calculation of weighted multi-scale adjacency matrix 10

3.8 Overview of GCN. Multi-scale adjacency matrix is a concatenated matrix with different search distance matrix shown as Fig 3.7. Weighted matrix does a split multiplication with each search distance matrix in adjacency matrix. 10

3.9 Proposed Action Classification Network 11

4.1 Training Datasets for Object Detection 15

4.2 Training Datasets for Person ReID 16

4.3 Labeled Pose Orientation for Person ReID 17

4.4 Results for different thresholds 19

4.5 Difference of cropped images and no cropped images 20

4.6 Different Resized Images 22

4.7 Action Score Distribution 24

4.8 True-positive sample segment. The plot is actioness score distribution for each frame. The peak score will be the key frame for action detection. The plot shows that frame 13-17 will be key frame for this action. 25

4.9 True-negative sample segment. This sample is labeled as background. There is not obvIoUs distribution for the score while each score is lower than 0 which will be filtered out. 26

4.10 False-positive sample segment. There are positive value and negative value for these scores. For this situation, all scores will be sum up to determine whether it is an action or not. Here this segment whose label is background is classified as an action 27

4.11 False-negative sample segment. The action in this sample is similar to Fig 4.10. Fig 4.10 is counter clockwise rotation, this is clockwise rotation. The score distribution is not Gaussian Distribution..... 28

LIST OF TABLES

TABLE Page

4.1 Comparison of Different Datasets 15

4.2 Effect of Fine-tune 15

4.3 Energy Consumption Comparison 17

4.4 Comparison of Detection Model 18

4.5 Effect of Cropping Methods 20

4.6 Comparison of Different Pruners 21

4.7 Comparison of Different Image Sizes 23

4.8 Comparison of Action Classification 29

1. INTRODUCTION

Human action detection is a challenging task in computer vision area. Detecting human action could be applied in many situations, such as surveillance for illegal behavior in public, smart homes like controlling light with hand gestures, and VR/AR environments. However, these scenes are all deployed with edge devices that require less energy. To achieve energy efficiency, each part of the detection system should fully exploit allocated energy. An essential measurement of energy consumption is time-consuming. Thus less time each part inferences, the better the system will be.

Standard deep action detection is based on a sequence of video images. The famous algorithm includes I3D, C3D, and Two Stream. However, these methods need extensive computational resources and a long time for training. For example, the video dataset UCF-101 includes hundreds of actions performed by different subjects. Each action consists of tens of videos, and each video has numerous frames. Though interval sampling strategy could be applied to reduce the number of training images, there is still a long time to get a good result. Besides, the network could be too complex and difficult to analyze and optimize. It generally needs several RTX 3090 GPUs to inference one video with the proposed model. If only CPU is available, there will be an unbearable delay. Thus, for those devices which do not have enough computation resources, like unmanned aerial vehicles (UAVs), image features are not applicable in this situation. Skeleton is a salient feature that consists of most information in a target action. Skeleton node is more precise featurewise and less computational than image features. To reach energy efficiency, a skeleton-based action classification network is proposed in this thesis. This thesis proposes a pipeline of energy-efficient action detection, which consists of three parts: a proposed object detection network, an existing pose estimation network, and the proposed skeleton-based action classification network.

The proposed object detection network is a variant of YOLO[2]. This network analyzes the most critical parts in the newest version of YOLOv5, introduces parts of YOLOv3, finally does modification on YOLOv5 to realize real-time detection on Raspberry Pi. Compared to YOLOX[3], the proposed object detection network accomplishes fewer flops but higher accuracy in person de-

tection. The action detection network is a spatial graph convolution network same as [4, 1]. Instead of using TCN, a transformer module replaces TCN to process temporal information. TCN is inefficient in extracting temporal features, and classical methods like RNN, LSTM show weak performance on long temporal information. Transformer is excellent on long sequence information, which could be more precise on action classification. Testing on SHREC 2017 dataset, proposed action classification network performs more robust than ST-GCN[1] with a random interval sample. For top 1 validation accuracy, the proposed network is 91.5%, ST-GCN[1] is 90.7%. This thesis also proposes a new loss function weakly supervised by Gaussian distribution to localize specific timestamps on action detection.

These proposed models are just raw models, which only perform well on those high computational devices. Deployment directly on Raspberry Pi still takes a long time to infer and is not energy-efficient. Traditional model compression methods such as pruning and quantization compress proposed models to smaller sizes and fewer parameters.

This thesis consists of four parts: section 2 is related to literature for human action detection. Some networks serve as a foundation for this thesis. Section 3 is the detailed method instruction of this pipeline. This part will depict details of modifications and derivation of loss functions. Section 4 shows experimental results and comparison of different experiments. Section 5 concludes the whole pipeline and future improvement.

2. LITERATURE REVIEW

This section will introduce related works of the proposed pipeline.

2.1 Object Detection

Object detection is always an important area in computer vision area. There has been numerous research done on this area in several decades. Famous algorithms have greatly influenced the artificial intelligence industry. [2] proposed the fastest object detection network, which plays the most important role in the industry nowadays. The network proposed by [2] was an anchor-based method that detected objects in different scales. Advanced versions based on [2] have been proposed annually. [5] firstly tried to abandon fully connected layers and introduced SVM to predict objectness. [6] proposed a full convolution network to detect objects without anchors. FCOS[6] achieves better AP score comparing to YOLOv2[2] and SSD[5] with fewer flops. [3] combined YOLO[2] and FCOS[6], introduced depthwise convolution, and split bounding box prediction and confidence prediction. [3] achieved SOTA effects with fewer flops. [2, 7, 8, 5, 6, 3] are all one stage detection network. [9, 10, 11, 12] proposed a two-stage detection method with higher AP but more flops.

2.2 Pose Estimation

Pose estimation has been witnessing a rapid focus on the action area. [13, 14, 15, 16, 17] researched on real-time pose estimation. The first[13] proposed a 200fps both pose estimation and bounding box detection. [14] proposed a different view estimation method which was a base of OpenPose. [18] proposed a stack residual hourglass network and captured different receptive field features without missing original features. Inspired by [18], [19] explored a new network on different resolution features and implemented a represent learning to generate heatmaps. [19] achieved the highest mAP on MPII, MSCOCO.

2.3 Skeleton Based Action Classification

Skeleton information is a salient feature for action recognition. Due to its fewer and precise parameters, excellent papers research on this area has been posted annually. [20] utilized LSTM to extract temporal features for skeleton features. [1] firstly pushed skeleton-based action recognition into a new stage. [1] applied graph convolution network on spatial skeleton graph learning and merged temporal sequence with temporal convolution network to classify action. It also proposed a subset split strategy to multi-scale node features. [21] proposed an optimization method on ST-GCN spatial adjacency matrix and a two-stream network. [22] encoded each joint with a one-hot vector and achieved efficient learning. [4] proposed a new spatial-temporal training strategy and implemented a 3D convolution on skeleton data. [23, 24, 25] did some improvements on [1]. [26] attempted to explore directed adjacency matrix. However, these papers do not pay attention to the nodes' distance effect.

2.4 Temporal Action Localization

Temporal action localization(TAL) is different from action recognition. Given an untrimmed video instance, TAL is to find the specific segment for labeled action. There are two localization strategies: fully-supervision[27, 28, 29, 30, 31] and weakly-supervision[32, 33, 34, 35]. Fully-supervision is following a proposal-classification paradigm[28, 36], where proposals are generated firstly and then classified. Given full temporal boundary annotations, the proposal classification methods usually filter out the background frames at the proposal stage via a binary classifier. Some researchers also put a study on action completeness when localizing action segments. [37] explored a structured pyramid pooling followed by an explicit classifier to discriminate if an action is completed. Weak supervision is to solve resource-intensive problems in fully-supervision. A widely used strategy in weakly-supervision temporal action localization is multiple instance learning. [38] proposed a temporal action score sequence to generate proposals. [39] randomly removed frames when training to force the network to capture multiple relevant parts. Despite the improvement on weakly-supervision, the performance is still inferior to fully-supervision.

3. PROPOSED METHODOLOGY

This section introduces three components: object detection, action classification, weakly supervised loss function. Firstly, object detection network detects and localizes the bounding boxes of persons in the video. The detection network outputs the position of persons. Then, the pipeline crops people areas from original images and inputs them to the pose estimation network to generate joint coordinates. Third, these joint coordinates will be input into action classification network to classify the actions. Action localization score will be summed into action classification score as an extra parameter.

3.1 Energy-Efficient Object Detection

Traditional YOLO network performs excellently on object detection. However, if deployed directly on the mobile device, it consumes much energy and super slowly inferences one image. Fig 3.1 shows the structure of YOLOv5. Generally, YOLO needs to detect multi-classes objects. Thus three detection layers are designed for different sizes. Each detection layer has a unique predefined anchor. However, in this thesis, the target is only one object detection. There is no need to use all three detection layers. With K means methods, the size of a person is at a medium scale, then the detection layer for medium-size objects should be used while the other two detection layers will be abandoned in this network. The proposed network is a one-scale detector. Two variants are proposed: one is based on the new version of YOLO - YOLOv5; another is from traditional YOLOv3, reference to the design of the structure, some advanced modules from YOLOv5 are introduced in YOLOv3.

3.1.1 YOLO-Mobile v1

Since a medium-size detection layer is only needed for person detection, the other two detection layers will be abandoned. Fig 3.1 shows the detailed structure of YOLOv5. All three detection layers are to detect small-size objects, medium-size objects, and large-size objects separately. The red mark in Fig 3.1 is one branch of PANet, which is marked by the yellow rectangle. Red detection

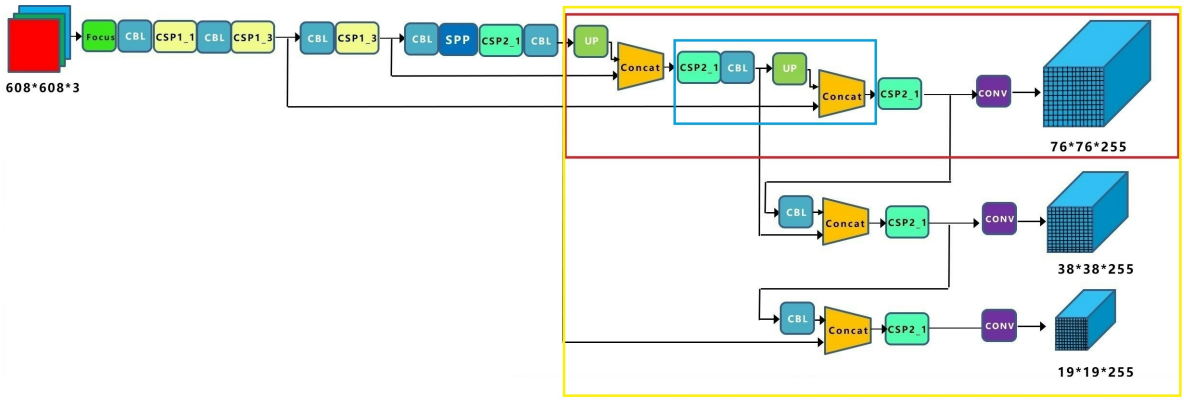


Figure 3.1: Structure of YOLOv5

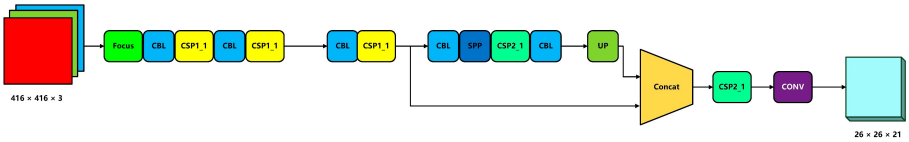


Figure 3.2: Structure of YOLO-Mobile v1

layer is to inference small-size objects. Compared to the other two branches, the marked red branch is more reasonable to be kept as the input features are directly from the backbone of YOLOv5. However, this branch is designed for small size. A further cutting off of the marked blue block is done to detect medium-sized objects. After all pruning operations, the structure of proposed YOLO-Mobile v1 is shown as Fig 3.2.

3.1.2 YOLO-Mobile v2

Though YOLO-Mobile v1 reduces vast FLOPs compared to the original YOLOv5, a lighter variant could be achieved with YOLOv3. Fig 3.3 shows the structure of YOLOv3. The inference speed of YOLOv3 is quicker than YOLOv5, but YOLOv3's mAP score is lower than YOLOv5. A combination with YOLOv3 and YOLOv5 is shown as Fig 3.4. This combination adopts the backbone part of YOLOv3 and detects objects with just one detection layer marked by the red rectangle in YOLOv3. Some high-level modules such as FOCUS, SPP, and CSP from YOLOv5

are added into the backbone to reduce FLOPs and improve performance.

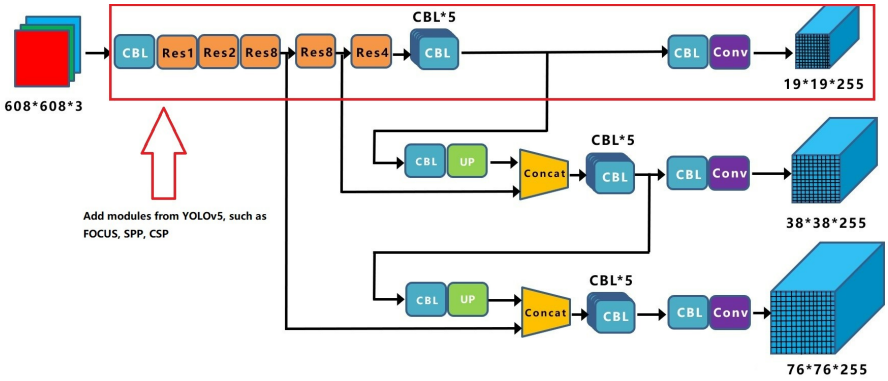


Figure 3.3: Structure of YOLOv3

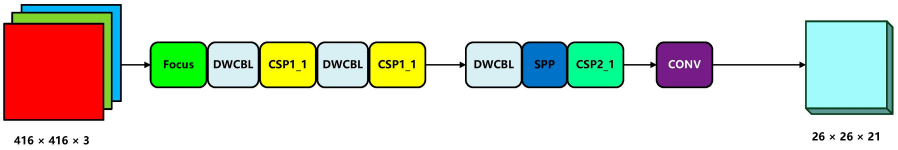


Figure 3.4: Structure of YOLO-Mobile v2

3.1.3 Person ReID

Generally, multiple persons will be detected in one image. To prevent confusion for action classification when detecting multiple persons, there needs to track each person individually. Person ReID assigns each person a unique ID. The network will track each person in different frames. Based on ResNet, this thesis applies a dynamic change strategy for input feature size. The final input size is determined with the best performance through the K-means method in YOLOv5 as prior knowledge and random downsample strategy. The detailed structure is shown as Fig 3.5.

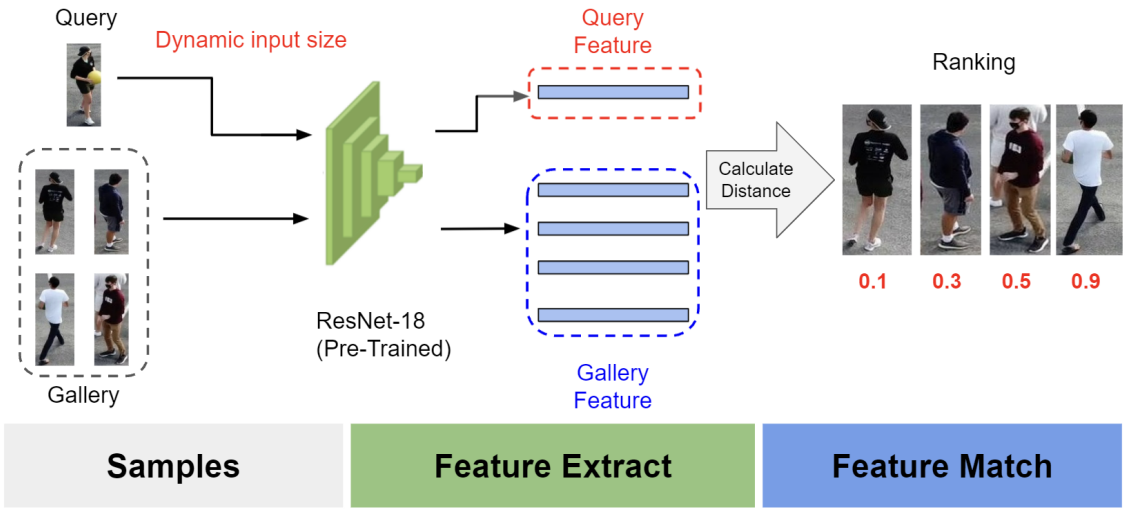


Figure 3.5: Overview of ReID Model. Query image and gallery images are input into ResNet-18 to extract image features. With cosine distance between query feature and gallery features, distance score is sorted to determine the best gallery image.

3.2 Action Classification

This section proposes a new transformer-based graph convolution network for action classification, Most paper on GCN action classification aggregates input features with adjacency matrix's high order polynomials. [4] introduces a new multi-scale way for that basic scale strategy will amply bias on local region nodes and further nodes receive weak weights. However, this strategy still has a problem because it sees the same weight as the root node and target node after k walks. To overcome this drawback, this thesis proposes a new strategy based on real-world distance. An N×N weight matrix W(N is node number) firstly stores distance between row node and column node, then W is normalized by equation 3.1.

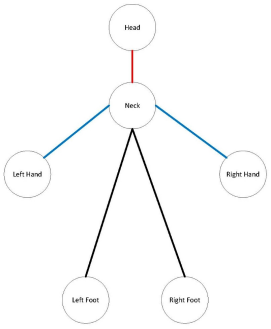
$$|\hat{A}_{(k)}|_{(i,j)} = \begin{cases} \exp^{normalize(-dis(v_i,v_j))}, & if\ d(v_i,v_j) = k \\ 1 & if\ i = j \\ 0 & others \end{cases} \tag{3.1}$$

where $dis(v_i, v_j)$ is real distance, $d(v_i, v_j)$ is the shortest walk distance and k is scale number. $\hat{A}_{(k)}$ is generalization of $\hat{A}$ to further neighborhood nodes, as $\hat{A}_{(0)} = I$ and $\hat{A}_{(1)} = \hat{A} \times W$. $\hat{A}$ is original

adjacency matrix with 1 walk node. $\hat{A}_{(k)}$ is a multiplication between adjacency matrix on root node and k walks' node and W. Given that node number is small, $\hat{A}_{(k)} = (\hat{A}^k - \hat{A}^{k-1} + I) \times W$. Fig 3.6 and Fig 3.7 show details of $\hat{A}_{(k)}$. Node features will be updated by:

$$X_t^{l+1} = \sigma(D_k^{-\frac{1}{2}} \hat{A}_k D_k^{-\frac{1}{2}} X_t^l \theta_k^l) \tag{3.2}$$

where l is layer number, θ is learning weight, D is degree matrix of A, X is relative node features on each layer. $D_k^{-\frac{1}{2}} \hat{A}_k D_k^{-\frac{1}{2}}$ is normalized k adjacency derived from Laplacian and Fourier transformer. Fig 3.8 is an overview of GCN module.



(a) Node distribution of human body

	Head	Neck	Left Hand	Right Hand	Left Foot	Right Foot
Head	1	1	0	0	0	0
Neck	1	1	1	1	1	1
Left Hand	0	1	1	0	0	0
Right Hand	0	1	0	1	0	0
Left Foot	0	1	0	0	1	0
Right Foot	0	1	0	0	0	1

(b) Adjacency matrix(distance = 1)

	Head	Neck	Left Hand	Right Hand	Left Foot	Right Foot
Head	1	0	1	1	1	1
Neck	0	1	0	0	0	0
Left Hand	1	0	1	1	1	1
Right Hand	1	0	1	1	1	1
Left Foot	1	0	1	1	1	1
Right Foot	1	0	1	1	1	1

(c) Adjacency matrix(distance = 2)

	Head	Neck	Left Hand	Right Hand	Left Foot	Right Foot
Head	1	0.976	0.672	0.672	0.495	0.495
Neck	0.976	1	0.781	0.781	0.543	0.543
Left Hand	0.672	0.781	1	0.602	0.426	0.426
Right Hand	0.672	0.781	0.602	1	0.426	0.426
Left Foot	0.495	0.543	0.426	0.426	1	0.412
Right Foot	0.495	0.543	0.426	0.426	0.412	1

(d) Weight matrix

Figure 3.6: Overview of adjacency matrix. **(a)** is joint position for human pose. Red edge represents 1 walk distance from head. Blue edges represents 2 walk distance from left hand to right hand. **(b)** is original adjacency matrix with 1 walk distance. **(c)** is adjacency matrix with 2 walk distance. **(d)** is proposed weighted matrix based on real world distance.

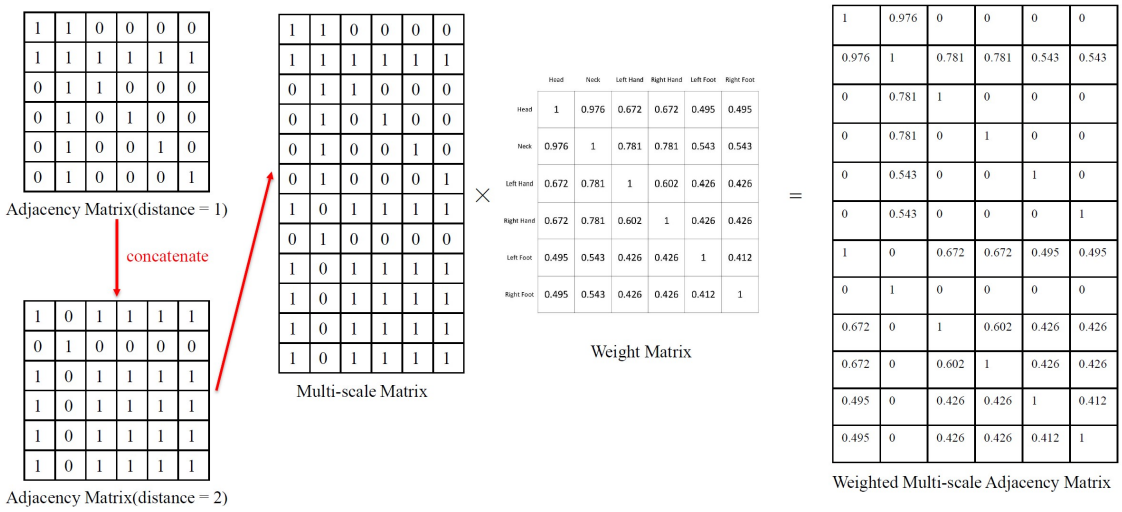


Figure 3.7: Calculation of weighted multi-scale adjacency matrix

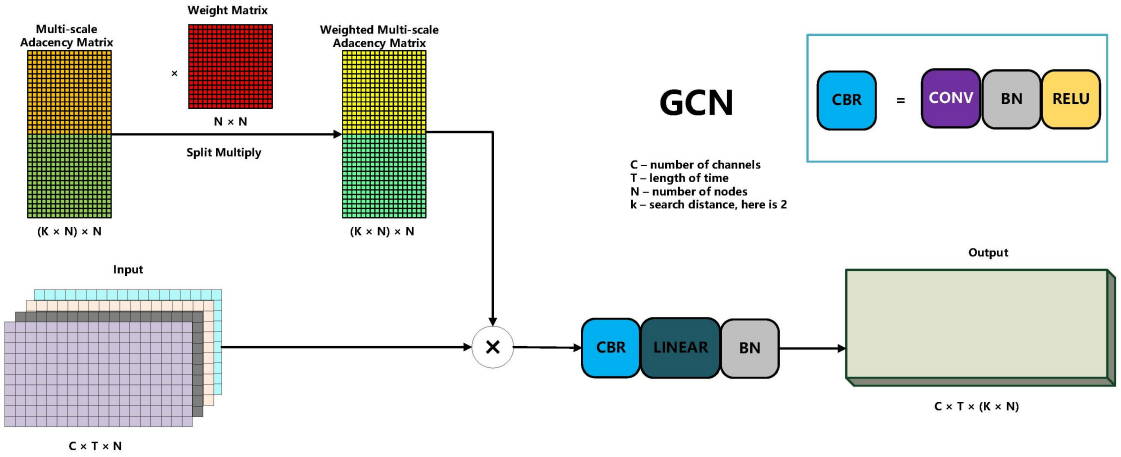


Figure 3.8: Overview of GCN. Multi-scale adjacency matrix is a concatenated matrix with different search distance matrix shown as Fig 3.7. Weighted matrix does a split multiplication with each search distance matrix in adjacency matrix.

Most existing work treats temporal sequence with TCN which operates temporal convolution on neighbor frames. This method will cause weak weights on longer frames in one computation though some papers propose that TCN could operate convolution with time dilation. Transformer could exploit the correlation between different frames, which achieves the same effects as [39]. This thesis replaces TCN with Transformer for longer temporal information, shown as Fig 3.9.

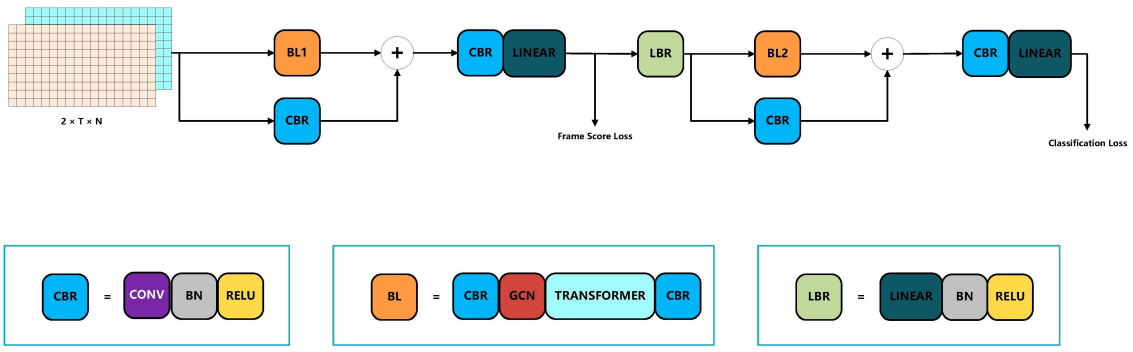


Figure 3.9: Proposed Action Classification Network

Two loss functions serve as criterion in this network. The first loss is frame score distribution function, and the second is classification loss. Details are in Section 3.3.

3.3 Weakly Supervised Loss Function

To localize the action segment, it is useful to know which part is most important in an action. Extracting from pose estimation network, skeleton features are input into action classification network to predict the probability of positive and negative action as P_p and P_n for each frame. P_p should be higher than P_n in real action videos while P_n is higher than P_p in fake action videos. A loss function is proposed by this knowledge.

$$l(B_p, B_n) = \max(0, 1 - \max_{i \in B_p} P_p^i + \max_{j \in B_n} P_n^j) \tag{3.3}$$

Equation 3.3 needs to store lots of information that consumes memory. To solve this problem, some changes are made from lemma 3.4

$$|\max_x g(x) - \max_y h(y)| \leq \max_a |g(a) - h(a)| \tag{3.4}$$

Final loss function is derived as Equation 3.5

$$l(B_p, B_n) = \max(0, 1 - \max_{a \in B_p, B_n} |P_p^a - P_n^a|) \tag{3.5}$$

The key detected action segment is sparse in video segments and the actioness score should be under a smooth distribution since the neighbor of the key segment could not get a very low score. Actioness scores should satisfy Gaussian distribution which is implicitly weak supervision in the positive segment.

$$l_{positive}(B_p, B_n) = \frac{1}{T} \sum_{t=0}^{t=T} ((P_p^t - P_n^t) - \exp^{\frac{(t-i)^2}{2var^2}})^2 \tag{3.6}$$

where i is argmax actioness score frame index in positive segment.

Negative segment does not have key frames for specific actions, to avoid false positive(anomalous positive value in score) and satisfy consistency, the loss function is different from positive segment.

$$l_{negative}(B_p, B_n) = \frac{1}{T} \sum_{t=0}^{t=T-1} ((P_p^{t+1} - P_n^{t+1}) - (P_p^t - P_n^t))^2 + Penalty(t) \tag{3.7}$$

$$Penalty(t) = \begin{cases} \beta, & if\ (P_p^t - P_n^t) > 0 \\ 0, & else \end{cases} \tag{3.8}$$

Concluding from above equations, actioness loss function is as follows.

$$l(B_p, B_n) = \begin{cases} \frac{1}{T} \sum_{t=0}^{t=T} ((P_p^t - P_n^t) - \exp^{\frac{(t-i)^2}{2var^2}})^2, & positive \\ \frac{1}{T} \sum_{t=0}^{t=T-1} ((P_p^{t+1} - P_n^{t+1}) - (P_p^t - P_n^t))^2 + Penalty(t), & negative \end{cases} \tag{3.9}$$

4. EXPERIMENTS AND DATASETS

4.1 Datasets

There are several datasets for object detection, ReID, and action classification. 2021 LPCV organizers provide sample videos as training and validation datasets. COCO dataset is another public dataset for training and validation of object detection. To relieve huge domain gap and imbalance of training datasets, additional synthetic dataset is constructed with excellent image composition and harmonization methods. A private dataset held by the LPCV team in Texas A&M University is set as the test dataset. CUHK and Market 2015 are common public datasets for person ReID. 2021 LPCV dataset serves as test dataset for ReID performance measurement. There are 12936 training images, including 751 identities, and 19732 testing images, including 750 identities in Market 2015 dataset. The size of each image in Market 2015 dataset is 64×128. CUHK person ReID dataset consists of 7365 training images with 767 identities and 5332 testing images with 700 identities. Each image's resolution is 60×160. [40, 41] are skeleton action classification datasets. LPCV dataset includes around 20000 frames. The resolution of each frame is 1920×1080. There are 5 to 7 persons in each frame. COCO dataset has 80000 images labeled as person. There are 1 to 2 persons in each image. The private dataset has 2000 images, and there are 6 to 9 persons in each image. SHREC 2017 dataset[40] is a third-person view hand gesture dataset. This dataset contains coordinates of 22 joints in 2d image space and 3d world space. There are 14 gestures to classify, and each gesture contains 28 subjects to perform. Each image's resolution is 640×480. UTD-MHAD dataset[41] is a third-person view video dataset. Each video consists of around 50 frames whose resolution is 640×480. Four different subjects perform twenty actions.

4.2 Experiments

4.2.1 Energy-Efficient Object Detection and ReID

4.2.1.1 Evaluation Metrics

To evaluate the performance of object detection networks in this thesis, it is different from the typical object detection network. Besides of mAP score, FLOPs, parameters, and FPS are also needed to be considered. The purpose is to achieve an energy-efficient model, which means FLOPs and FPS are more critical than mAP scores. FLOPs calculation is not defined in PyTorch and is usually done by hand. To prevent unexpected mistakes, an implemented library, **thop**, is utilized to measure FLOPs in this thesis. Meaning average precision, also mAP, is standard metrics in object detection, which has been implemented in the source code. FPS is special here since that Model need to run on Raspberry Pi normally. It is the most direct measurement in performance comparison. FPS will be decided by 1/T, where T is inference time for detection on one image. There is only one criterion for ReID. Each id should be the same on the assigned objects in the whole video. The mAP of ReID is calculated as P = N_p/N_{total}, where P is accuracy of ReID model. N_p is the number of positive frames, N_{total} is total frames.

4.2.1.2 Training on Different Datasets

Training on different datasets impacts the performance of proposed Model. The datasets for object detection consist of COCO dataset, synthetic dataset and LPCV dataset. COCO dataset includes 80000+ person instances and 4000+ ball instances. 20000+ images are selected from original COCO dataset as training dataset for object detection network in this thesis. The synthetic dataset is a COCO-based dataset with different image composition methods. Each image in synthetic dataset consists of at least one person and similar ball as LPCV dataset. LPCV dataset involves the official LPCV dataset and LPCV-TAMU dataset by TAMU. Fig 4.1 is the details of the above datasets. The different combination among these datasets shows different results. Table 4.1 is the comparison of different datasets.

Table 4.1 shows remarkable contrast with different datasets. Though the COCO+LPCV dataset

(a) COCO

(b) Synthetic

(c) LPCV

Figure 4.1: Training Datasets for Object Detection

Model	Dataset	Precision	Recall	mAP_0.5	mAP_0.5:0.95
YOLO-small	COCO	0.6842	0.7174	0.6664	0.4827
YOLO-small	COCO+LPCV	**0.9696**	**0.9009**	**0.9369**	0.5346
YOLO-small	COCO+LPCV+Synthetic	0.9611	0.8522	0.9003	**0.5416**
YOLO-Mobile v1(ours)	COCO+LPCV+Synthetic	0.9468	0.8436	0.8752	0.4556

Table 4.1: Comparison of Different Datasets

Model	Dataset	Fine-tune	Precision	Recall	mAP_0.5	mAP_0.5:0.95
YOLO-Mobile v1	COCO+Synthetic+LPCV	N	0.8595	0.6766	0.7329	0.3372
YOLO-Mobile v1	COCO+Synthetic+LPCV	Y	**0.9468**	**0.8436**	**0.8752**	**0.4556**

Table 4.2: Effect of Fine-tune

reaches superior precision, recall, and mAP_0.5, one reason is that the test image is similar to LPCV dataset, which means the model training on the COCO+LPCV dataset is a trim fit for the test dataset, mAP_0.5:0.95 should be more important under this situation. COCO+LPCV+Synthetic datasets increase mAP_0.5:0.95 a lot than COCO+LPCV. This model is more robust and learns more shape features from training images. Traditional YOLOv5 model is more precise than proposed YOLO-mobile v1 in this thesis as table 4.1 shows.

Finetune is also important when training. YOLO-Mobile v1 has been trained on the merged dataset with COCO, Synthetic and LPCV. Two experiments, no fine-tune and fine-tune are conducted with same data augment scheme. The fine-tuned model achieves more excellent performance than the no fine-tuned model as table 4.2 shows.

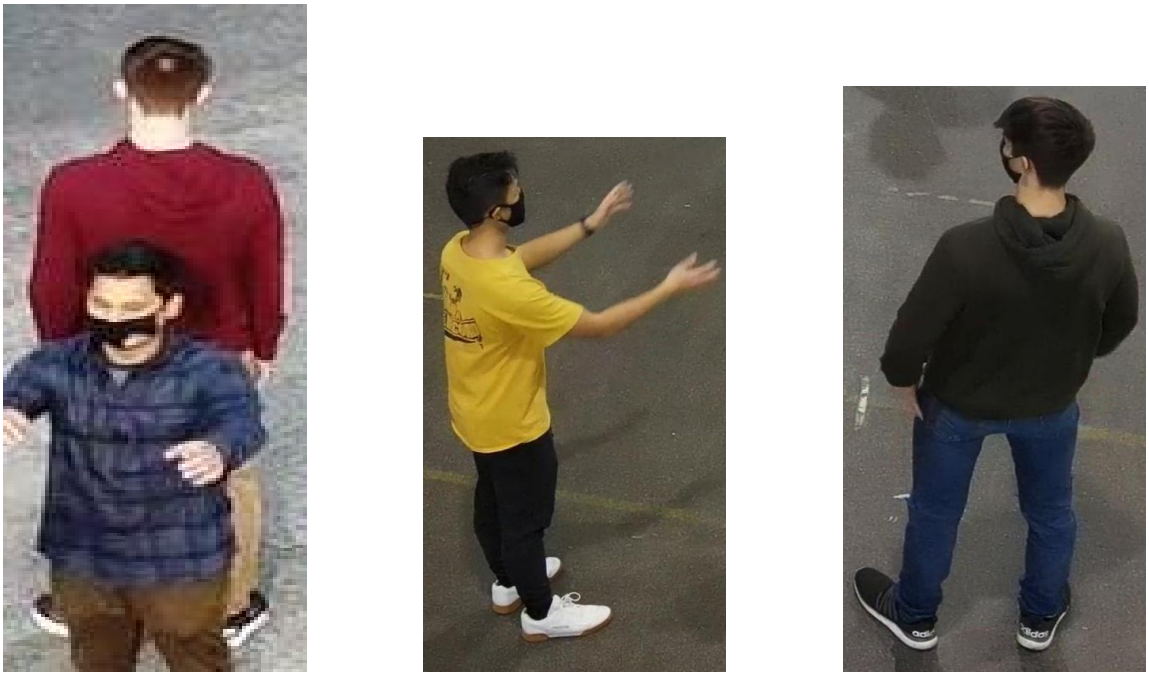
There are three datasets for the Person ReID model's training. Section 4.1 has introduced basic information. Fig 4.2 presents the detailed images for the three datasets. To increase the difficulty of identification, pose orientation is labeled for each test image. Fig 4.3 introduces three labeled orientations.

(a) Market 2015 Dataset (b) CUHK Dataset (c) LPCV

Figure 4.2: Training Datasets for Person ReID

4.2.1.3 Experiments and Results

An energy measurement module is implemented to measure initial energy consumption for each part in YOLOv5. For these parts which consume the most energy, the best way is to cut off from the sequential. There are two methods for prior knowledge pruning: one is manually reducing the channel number for both input channel and output channel, the other is to replace



(a) Front (b) Side (c) Back

Figure 4.3: Labeled Pose Orientation for Person ReID

some modules with similar components. Table 4.3 experiments with different pruning strategies.

Module	Channel Reduction(times)	Parameters	Time(ms)	Final Action
Conv	2	18560	1.507	-
Conv	4	4672	0.715733	+
DWConv	4	752	2.6531	-
CSP(3)	2	156928	4.7974	-
CSP(1)	2	74496	2.9840	-
CSP(3)	4	39552	3.6280	-
CSP(1)	4	18816	2.1760	+
Detect(origin)	2	18879	0.377655	-
Detect(medium)	4	2709	0.30756	+
Detect(small)	4	1365	0.342846	-
Upsample	4	31232	2.254963	-

Table 4.3: Energy Consumption Comparison

Table 4.3 shows the detail for different modules. Decreasing channel numbers is more efficient

than other compress methods. Generally, FLOPs and parameters should be a quarter of the original part with a one-half reduction on channel numbers. Surprisingly, depth-wise convolution consumes more time than standard convolution though parameters are much fewer than standard convolution. One reason is that the operations on reading and saving parameters take the most time. Detect module reduces the least time while the detection layer for small objects needs one more upsample block than the medium detection layer. Since an upsample detection block consumes 2ms for each image, abandoning upsample block is a energy-efficient method. In table 4.3, "-" means this module should be abandoned for the final model, and "+" means the module should be saved. After testing different compressed components, YOLO-Mobile v1 and YOLO-Mobile v2 are proposed as Section 3.1 introduces.

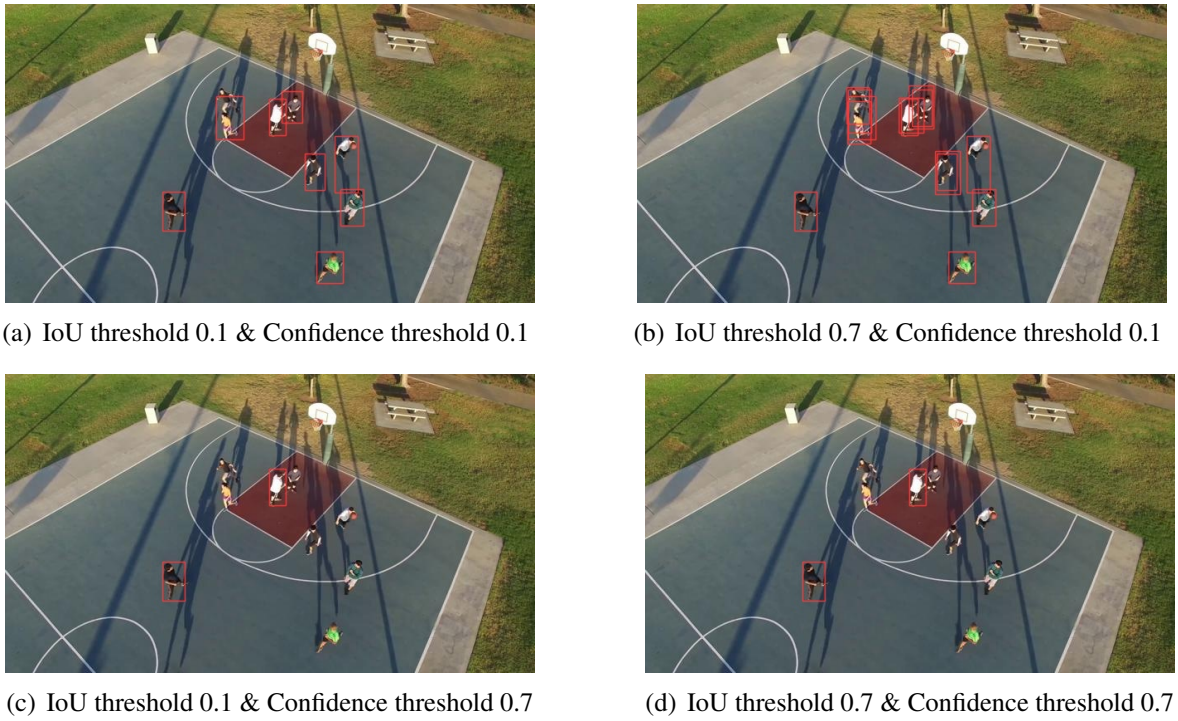
One excellently-compressed model could not ensure good performance on accuracy. Proposed Model, YOLO-mobile v1 and YOLO-mobile v2 are compared with the original YOLOv5 and the best YOLO model, YOLOX, which achieves the highest mAP. To keep conditions controlled, all components in training should be the same except Model. The following are the comparison of YOLOv5-tiny, YOLOX-tiny, YOLO-Mobile v1, and YOLO-Mobile v2. Results are shown in Table 4.4. All Model are trained with COCO dataset and tested with LPCV dataset. Input feature size is 640 $\times$ 640. Besides, all Model are pruned and quantized under the same strategy.

Model	FLOPs(G)	Parameters(M)	FPS	mAP_0.5:0.95
YOLO-small	16.4	7.07	0.03	0.48
YOLOX-tiny	1.8	1.08	5	**0.495**
YOLO-Mobile v1	1.3	0.53	10	0.471
YOLO-Mobile v2	**0.9**	**0.11**	**14**	0.43

Table 4.4: Comparison of Detection Model

Good hyperparameters are also critical for object detection. There are two crucial hyperparameters in object detection model: IoU threshold and confidence threshold. A suitable threshold is helpful to filter out noise while keeping right bounding boxes. Fig 4.4 shows results with different

IoU thresholds and confidence thresholds.



(a) IoU threshold 0.1 & Confidence threshold 0.1

(b) IoU threshold 0.7 & Confidence threshold 0.1

(c) IoU threshold 0.1 & Confidence threshold 0.7

(d) IoU threshold 0.7 & Confidence threshold 0.7

Figure 4.4: Results for different thresholds

Fig 4.4 is detection results with different thresholds. In this sample, the best choice is that both the IoU threshold and confidence threshold are 0.1. However, a lower confidence threshold means false positive rate will be higher. For the situation in Fig 4.4, person is not easy to be detected, then low confidence threshold and low IoU threshold are the most practicable here. Mostly, the confidence threshold should not be less than 0.5, and the IoU threshold should not be greater than 0.45.

Another trick is explored to improve detection accuracy without increasing FLOPs. Since frames in one segmentation share the same background, it is unnecessary to include background into detection, reducing the probability of detecting natural objects. The background would be seen as noise here, while increasing SNR(Signal-to-noise ratio) could be helpful to improve the

performance. A cropping mechanism is introduced to increase SNR. Fig 4.5 is the difference between cropped images and no cropped images. Experiments with cropping mechanism are shown in Table 4.5.

(a) No Cropped Images (b) Cropped Images

Figure 4.5: Difference of cropped images and no cropped images

Method	Dataset	Precison	Recall	mAP_0.5:0.95
No crop	LPCV	0.9513	0.8321	0.4526
Crop	LPCV	**0.9531**	**0.8448**	**0.4694**

Table 4.5: Effect of Cropping Methods

Cropped images increase the recall rate of detection Model. If one image is cropped, all objects are enlarged, and those not detected before could be captured. The false-negative rate will be minimized; thus mAP score is higher than no cropped images.

Two pruners are applied on pruning ReID model: L1 pruner and FPGM pruner. These two pruners are both one-short pruners which means the model will be pruned without any training process. L1 pruner only prunes the convolution layers. This pruner will firstly calculate L1 norm for each convolution layer. Once the norm is big enough, this layer will keep most parameters,

vice versa. FPGM pruner prunes with the smallest geometric median. A smaller geometric median means this layer is more replaceable. Two experiments are conducted with L1 norm pruner and FPGM pruner.

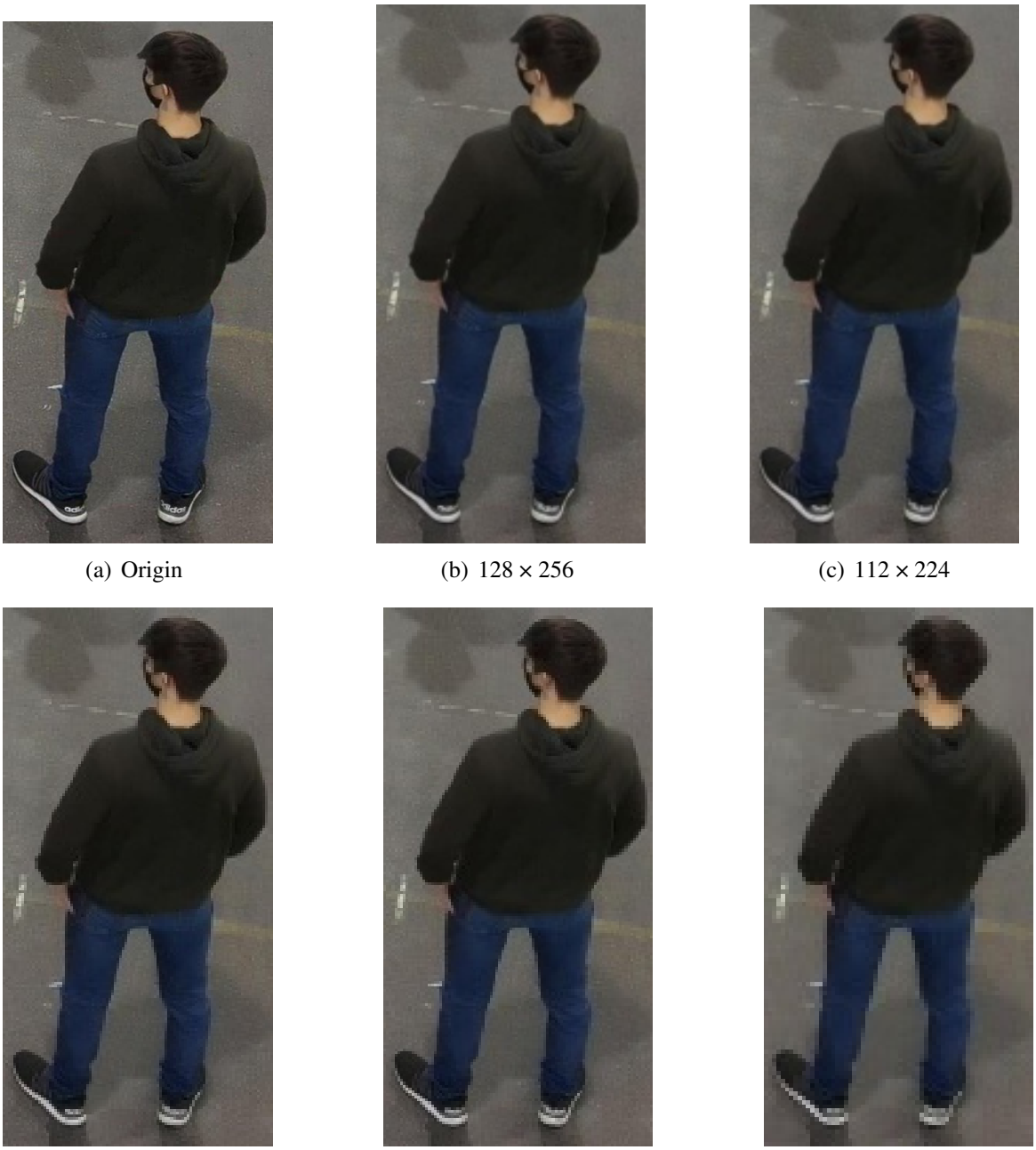
Model	Sparsity Ratio	Rank@1	Rank@5	Rank@10	mAP	Pruner
Resnet	0.4	0.844715	0.940915	0.960808	0.638490	L1
Resnet	0.4	0.847387	0.938242	0.962589	0.650325	FPGM
Resnet	0.5	0.839074	0.937945	0.954572	0.627968	L1
Resnet	0.5	0.842340	0.940321	0.964371	0.631602	FPGM
Resnet	0.6	0.787411	0.904988	0.937352	0.551677	L1
Resnet	0.6	0.799584	0.912114	0.940915	0.573035	FPGM
Resnet	0.7	0.729810	0.883314	0.919240	0.478091	L1
Resnet	0.7	0.740202	0.886876	0.923397	0.492581	FPGM
Resnet	0.8	0.709620	0.868765	0.911520	0.457538	L1
Resnet	0.8	0.708729	0.869359	0.911520	0.463868	FPGM
Resnet	0.9	0.636876	0.830760	0.886876	0.384856	L1
Resnet	0.9	0.671021	0.850653	0.901128	0.415192	FPGM

Table 4.6: Comparison of Different Pruners

Table 4.6 is ReID results with different sparsity ratio and different pruners. FPGM pruner is better than L1 norm pruner in each experiment. The reason is that L1 norm is easily impacted by outliers. Once one weight in the convolution filter is much larger than others, L1 norm of this convolution layer is still larger than other convolution layers, which means noisy convolution layers would be saved while critical convolution layers are filtered; thus, the performance of pruned Model will drop heavily.

The size of input image features is also important in FLOPs reduction. Half of the input size means three-quarters of FLOPs will be reduced. Fig 4.6 shows different resized input images. Table 4.7 is model results of different size images.

Different sizes of images display apparent similarities in both shape and color information. Table 4.7 tells that resized images only cause minor mAP loss. The final person ReID model is a compressed model pruned by FPGM pruner with 0.875 sparsity and quantized by quantization



(a) Origin (b) 128×256 (c) 112×224

(d) 100×200 (e) 96×192 (f) 64×128

Figure 4.6: Different Resized Images

aware training. The input images are resized to shape 64×128. FLOPs are reduced $4 \times$ with only 20% loss on mAP.

Image Size	Rank@1	Rank@5	Rank@10	mAP
128×256	0.703385	0.864608	0.910926	0.438980
112×224	0.690024	0.862233	0.908848	0.435229
100×200	0.677257	0.853325	0.906770	0.418173
96×192	0.668052	0.856591	0.902910	0.418813
64×128	0.611936	0.817102	0.876781	0.354877

Table 4.7: Comparison of Different Image Sizes

4.2.2 Action Classification

4.2.2.1 Evaluation Metrics

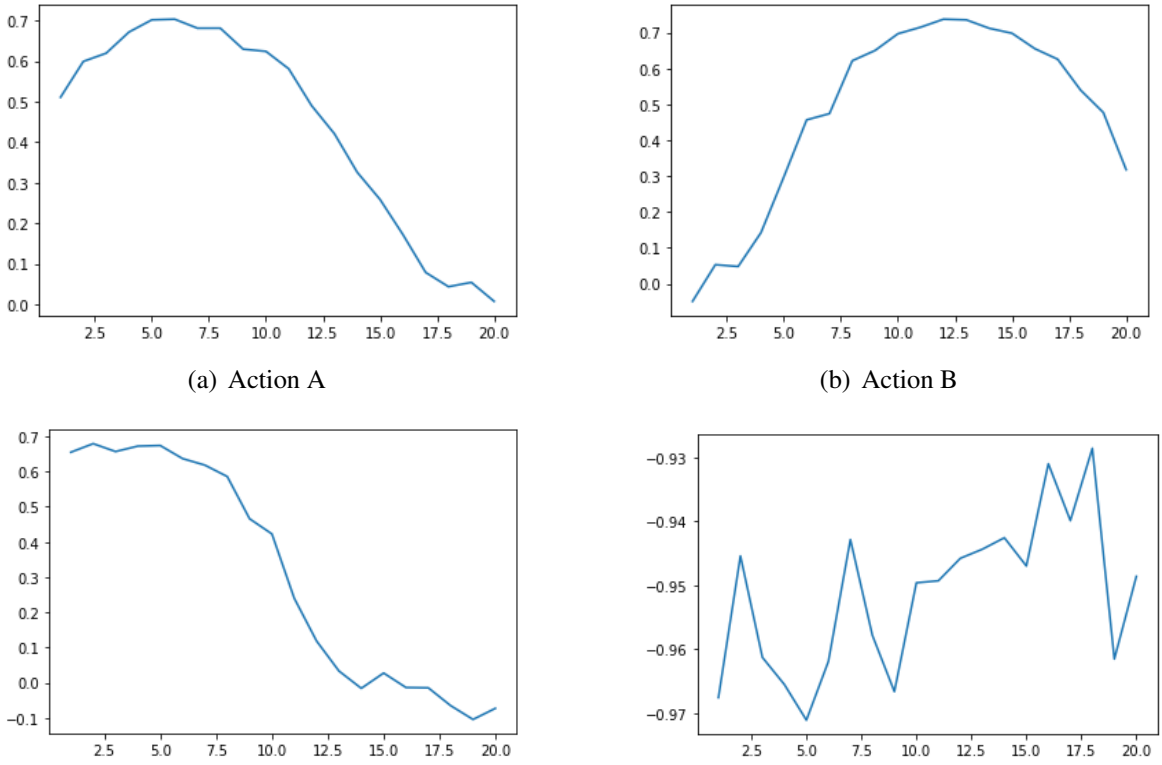
ST-GCN[1] firstly designed skeleton-based network for action recognition. SHREC 2017 dataset[40] and UTD-MHAD dataset[41] are both action recognition dataset. Metrics for action recognition should be considered as classification accuracy. Actioness score by the proposed network will be an extra parameter when classifying an action. If the negative score is higher than the positive score, the input video segment would be considered as background. Unlike object detection, action detection is an offline network to classify action classes. Thus FPS, FLOPs, and parameters are less important than object detection, but accuracy is more critical.

4.2.2.2 Experiments and Results

Original labels are split into two parts to discriminate background and actual actions: some are labeled as existing actions while others are labeled as background.

Actioness score determines whether a frame belongs to a specific action even though it is in the segment. The common method is decreasing expanding radius to minimize the length of input as coarse to fine. Here, a new Gaussian error is introduced to help to find the keyframe. The distribution of outputs would be the key component for the final prediction. Fig 4.7 is the frame score distributions for different actions, where x-axis is indices of different frames, and y-axis is frame score for each frame.

Table 4.7 shows same score distribution for the same action class even though input features are different. For different action classes, the distribution is different. However, if one segment



(a) Action A

(b) Action B

(c) Action A with different input features

(d) Background

Figure 4.7: Action Score Distribution

belongs to real action, its score distribution must follows Gaussian distribution. For those segments labeled as background, the score distribution is not regular Gaussian distribution, each score must be less than 0 in the distribution of background segments.

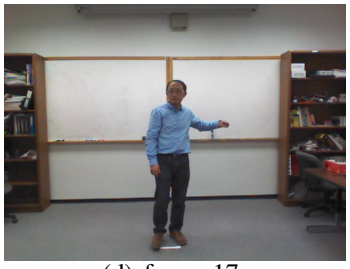
Figures 4.8, 4.9, 4.10, 4.11 are true positive, true negative, false positive and false negative sample segments separately. For true positive samples, the score distribution is strictly subject to Gaussian distribution. All scores are negative values with irregular distribution in true negative samples. False-positive and false-negative samples both have positive values and negative values. The number of positive values is more than negative values in false-positive while converse in false-negative. Samples in false positive and false negative both do not follow Gaussian distribution.

Table 4.8 is comparison of proposed action classification model and ST-GCN[1]. The proposed action classification network not only reaches much fewer FLOPs than ST-GCN[1] but also

(a) frame 11

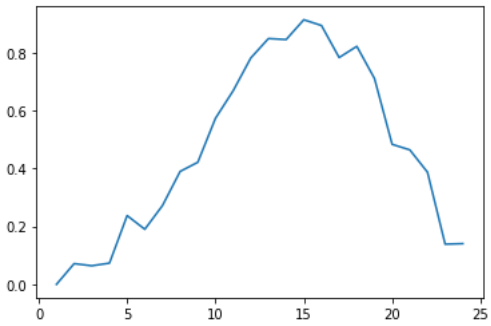
(b) frame 13

(c) frame 15



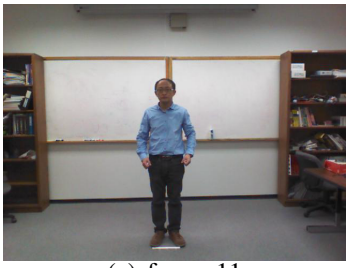
(d) frame 17

(e) frame 19

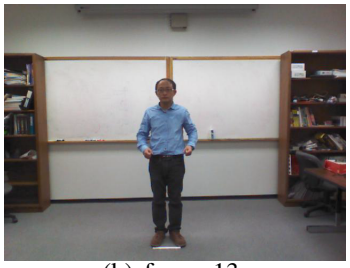


(f) score distribution

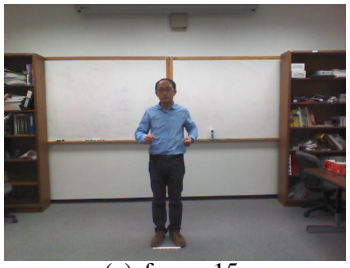
Figure 4.8: True-positive sample segment. The plot is actioness score distribution for each frame. The peak score will be the key frame for action detection. The plot shows that frame 13-17 will be key frame for this action.



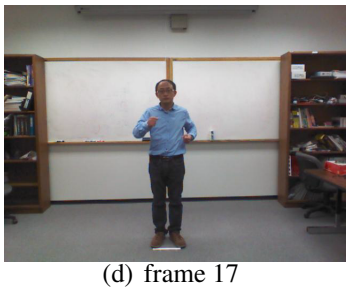
(a) frame 11



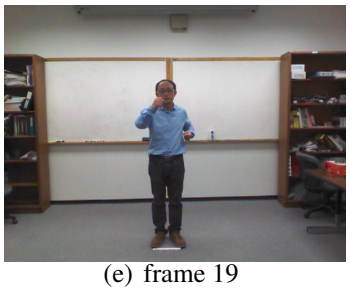
(b) frame 13



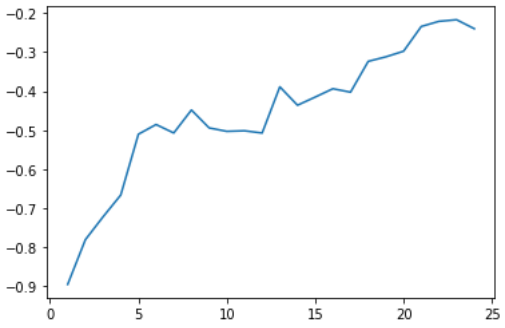
(c) frame 15



(d) frame 17

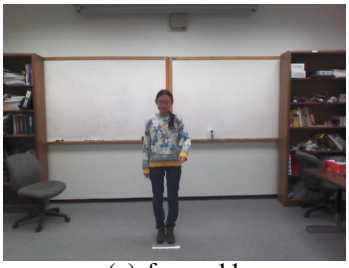


(e) frame 19

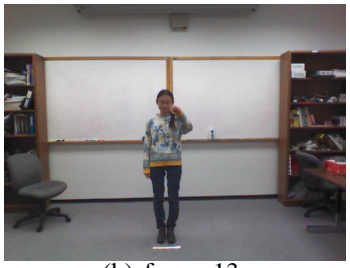


(f) score distribution

Figure 4.9: True-negative sample segment. This sample is labeled as background. There is not obvIoUs distribution for the score while each score is lower than 0 which will be filtered out.

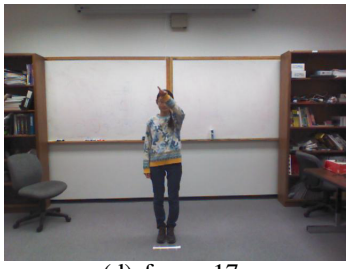


(a) frame 11

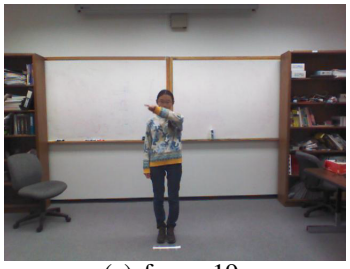


(b) frame 13

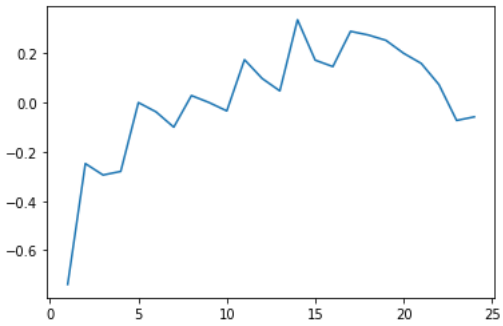
(c) frame 15



(d) frame 17

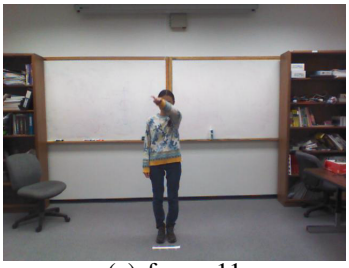


(e) frame 19

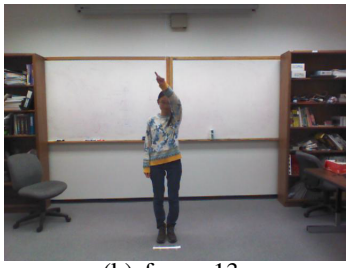


(f) score distribution

Figure 4.10: False-positive sample segment. There are positive value and negative value for these scores. For this situation, all scores will be sum up to determine whether it is an action or not. Here this segment whose label is background is classified as an action

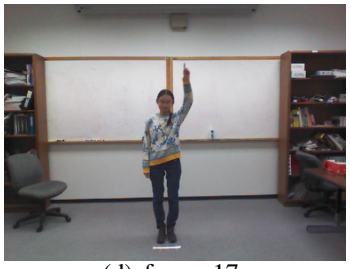


(a) frame 11

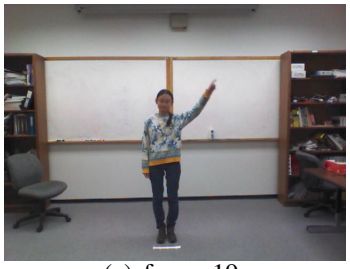


(b) frame 13

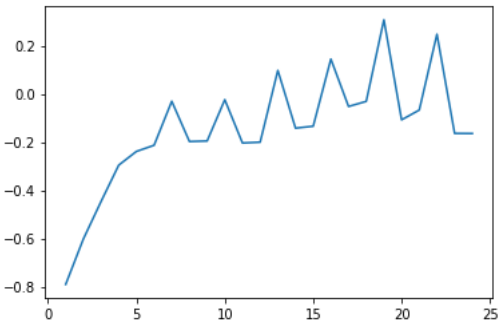
(c) frame 15



(d) frame 17



(e) frame 19



(f) score distribution

Figure 4.11: False-negative sample segment. The action in this sample is similar to Fig 4.10. Fig 4.10 is counter clockwise rotation, this is clockwise rotation. The score distribution is not Gaussian Distribution.

increases a little on accuracy. Through noisy training on both ST-GCN[1] and proposed action classification network, the accuracy of proposed network drops more heavily than ST-GCN[1], ST-GCN[1] is more robust than proposed network for noisy inputs.

Model	Accuracy	FLOPs(G)
ST-GCN	90.4%	16.4
Ours(without Gaussian error)	89.4%	0.368
Ours(with Gaussian error)	**91.5**%	**0.368**

Table 4.8: Comparison of Action Classification

5. CONCLUSION

This thesis proposes a primary pipeline of energy-efficient human action detection in the wild environment. The proposed YOLO-Mobile v1 reduces 16 $\times$ FLOPs with just 24% loss on mAP compared to traditional YOLOv5-small. The synthetic dataset contributes a lot to increase the robustness of detectors. Person ReID model employs an efficient dynamic strategy for input features size to decrease FLOPs with slight loss on mAP. The proposed weighted matrix promotes GCN to learn more from neighbor nodes, which increases the learning efficiency of conventional methods. Tranformer module expands sensitive temporal areas and improves temporal learning. Gaussian loss function weakly supervises the action classification network and enhances the robustness of the action classification network.

REFERENCES

[1] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[3] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[4] Z. Liu, H. Zhang, Z. Chen, Z. Wang, and W. Ouyang, "Disentangling and unifying graph convolutions for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 143–152, 2020.

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[6] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9627–9636, 2019.

[7] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[8] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[11] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[12] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[13] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7291–7299, 2017.

[14] Y. Wang, B. Zhang, and C. Peng, "Srhandnet: Real-time 2d hand pose estimation with simultaneous region localization," *IEEE transactions on image processing*, vol. 29, pp. 2977–2986, 2019.

[15] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Consumer depth cameras for computer vision*, pp. 119–137, Springer, 2013.

[16] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[17] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt, "Vnect: Real-time 3d human pose estimation with a single rgb camera," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–14, 2017.

[18] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European conference on computer vision*, pp. 483–499, Springer, 2016.

[19] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[20] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1110–1118, 2015.

[21] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12026–12035, 2019.

[22] P. Zhang, C. Lan, W. Zeng, J. Xing, J. Xue, and N. Zheng, "Semantics-guided neural networks for efficient skeleton-based human action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1112–1121, 2020.

[23] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Actional-structural graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3595–3603, 2019.

[24] O. Keskes and R. Noumeir, "Vision-based fall detection using st-gcn," *IEEE Access*, vol. 9, pp. 28224–28236, 2021.

[25] S. Gadgil, Q. Zhao, A. Pfefferbaum, E. V. Sullivan, E. Adeli, and K. M. Pohl, "Spatio-temporal graph convolution for resting-state fmri analysis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 528–538, Springer, 2020.

[26] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with directed graph neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7912–7921, 2019.

[27] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, and L. Wang, "Tea: Temporal excitation and aggregation for action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 909–918, 2020.

[28] Y.-W. Chao, S. Vijayanarasimhan, B. Seybold, D. A. Ross, J. Deng, and R. Sukthankar, "Rethinking the faster r-cnn architecture for temporal action localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1130–1139, 2018.

[29] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage cnns," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1049–1058, 2016.

[30] Z. Shou, J. Chan, A. Zareian, K. Miyazawa, and S.-F. Chang, "Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5734–5743, 2017.

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

[32] F. Ma, L. Zhu, Y. Yang, S. Zha, G. Kundu, M. Feiszli, and Z. Shou, "Sf-net: Single-frame supervision for temporal action localization," in *European conference on computer vision*, pp. 420–437, Springer, 2020.

[33] Z. Shou, H. Gao, L. Zhang, K. Miyazawa, and S.-F. Chang, "Autoloc: Weakly-supervised temporal action localization in untrimmed videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 154–171, 2018.

[34] P. Lee, Y. Uh, and H. Byun, "Background suppression network for weakly-supervised temporal action localization," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 11320–11327, 2020.

[35] Z. Liu, L. Wang, Q. Zhang, Z. Gao, Z. Niu, N. Zheng, and G. Hua, "Weakly supervised temporal action localization through contrast based evaluation networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3899–3908, 2019.

[36] X. Dai, B. Singh, G. Zhang, L. S. Davis, and Y. Qiu Chen, "Temporal context network for activity localization in videos," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5793–5802, 2017.

[37] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2914–2923, 2017.

[38] L. Wang, Y. Xiong, D. Lin, and L. Van Gool, "Untrimmednets for weakly supervised action recognition and detection," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4325–4334, 2017.

[39] K. K. Singh and Y. J. Lee, "Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization," in *2017 IEEE international conference on computer vision (ICCV)*, pp. 3544–3553, IEEE, 2017.

[40] Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. Le Saux, and D. Filliat, "Shrec'17 track: 3d hand gesture recognition using a depth and skeletal dataset," in *3DOR-10th Eurographics Workshop on 3D Object Retrieval*, pp. 1–6, 2017.

[41] C. Chen, R. Jafari, and N. Kehtarnavaz, "Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor," in *2015 IEEE International conference on image processing (ICIP)*, pp. 168–172, IEEE, 2015.