Image-based Learning of Human Body Information for Natural Human-Computer Interaction

Zhao, Ying

(Degree)
博士（工学）

(Date of Degree)
2020-03-25

(Date of Publication)
2021-03-01

(Resource Type)
doctoral thesis

(Report Number)
甲第7786号

(URL)
https://hdl.handle.net/20.500.14094/D1007786

Doctoral Dissertation

Image-based Learning of Human Body Information for Natural Human-Computer Interaction

自然なヒューマンコンピュータインタラクションにおる
イメージベースの人間身体情報学習

ZHAO YING

January 2020

Graduate School of System Informatics
Kobe University

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 40,000 words including bibliography, footnotes, tables and equations and has fewer than 100 figures.

ZHAO YING

January 2020

ACKNOWLEDGEMENTS

Thanks to all of my advisers and mentors from whom I learned a lot: Prof. Luo Zhiwei and Assoc. Prof. Quan Changqin. I want o thank all my thesis committee members: Prof. Kageyama Akira and Prof. Uehara Kuniaki. Many thanks to my mother and father. I want to also thank all my friends and colleagues who helped me at different stages of my education with their letters and support.

TABLE OF CONTENTS

Acknowledgments i

List of Tables vii

List of Figures viii

Chapter 1: Introduction 1

1.1 Background 1

1.2 Previous Research 4

1.2.1 Hand Segmentation 4

1.2.2 Human Pose Estimation 5

1.3 Research Objective 9

1.4 Overview of Our Research 10

1.4.1 Hand Segmentation 10

1.4.2 Human Pose Estimation 11

Chapter 2: Review on Basic Knowledge 13

2.1 Image segmentation 13

2.1.1 Superpixel 14

2.1.2 Saliency 16

2.2 Convolutional Neural Network 17
2.2.1 Foundational Components 17
2.2.2 Evolutionary Components 23
2.3 Human Pose Estimation with CNN 25
2.3.1 DeepPose: Human Pose Estimation via Deep Neural Networks . . . 25
2.3.2 Efficient Object Localization Using Convolutional Networks 27
2.3.3 Stacked Hourglass Networks for Human Pose Estimation 29
2.4 Evaluation for Human Pose Estimation 30
2.4.1 Datasets 30
2.4.2 Metrics 32

Chapter 3: Coarse-to-fine Online Learning for Hand Segmentation in Egocentric Video 36

3.1 Introduction 36
3.2 Related Work 37
3.3 Method 40
3.3.1 Ego-saliency Based Hand Detection 41
3.3.2 Online Training Two-level Hand Classifiers 43
3.3.3 Evaluation to Update Classifier 49
3.4 Results and Discussion 51
3.4.1 Evaluation on Benchmark Dataset 51
3.4.2 Evaluation on Egocentric Application 55
3.5 Conclusion 56

Chapter 4: Cluster-wise Learning for Multi-person Pose Estimation 59

4.1 Introduction 59

4.2 Related Work 63

4.3 Method 67

4.3.1 Network Overview 67

4.3.2 Cluster-wise Keypoint Detection 69

4.3.3 Cluster-wise Tag Embedding 70

4.3.4 Feature Aggregation 74

4.4 Results and Discussion 77

4.4.1 Implementation 77

4.4.2 Performance 78

4.5 Conclusion 80

Chapter 5: Lite Hourglass Network for Multi-person Pose Estimation 86

5.1 Introduction 86

5.2 Related Work 88

5.3 Method 89

5.3.1 Network Overview 89

5.3.2 Lite Multi-context Block 90

5.3.3 Bottom-up Integral Regression 93

5.4 Results and Discussion 96

5.4.1 Performance 96

5.5 Conclusion 100

Chapter 6: Joint Bottom-up and Top-down Learning for Multi-person Pose Estimation 101

6.1 Introduction 101

6.2 Related Works 104

6.3 Method 105

6.3.1 Joint Bottom-up and Top-down Learning 105

6.3.2 Bottom-up Integral Regression 107

6.3.3 Offset Encoding 110

6.4 Results and Discussion 114

6.4.1 Performance 114

6.5 Conclusion 117

Chapter 7: Conclusion 119

References 133

LIST OF TABLES

2.1 Compare DeepPose [66] with prior-art: LSP dataset(PCP@0.5). 26

2.2 Compare Tompson et al. [69] with prior-art: MPII dataset(PCKh@0.5) . . . 28

2.3 Compare SHN [45] with prior-art: MPII dataset(PCKh@0.5) 30

3.1 Comparison with state-of-the-art on F-score 52

3.2 Comparison with state-of-the-art on time 52

4.1 Comparison results on MSCOCO testdev dataset with OKS. 79

4.2 Comparison results on PoseTrack validation dataset with mAP. 80

5.1 Dilation setting on COCO held-out500 with training resolution of 512x512. 96

5.2 Performances of lite blocks on COCO test-dev with training resolution of 512x512. 96

5.3 Comparison on COCO test-dev dataset. 97

6.1 Results on MSCOCO validation500 dataset. 113

6.2 Multi-scale results on MSCOCO validation500 dataset. 113

6.3 Comparison on MSCOCO test-dev dataset. 115

LIST OF FIGURES

1.1 Illustration of the concept of sensor-based and vision-based HCI. 2

1.2 Example of egocentric vision-based HCI. 2

1.3 Example of conventional vision based HCI. 3

1.4 Example of challenge cases for learning human body information. 4

1.5 Example of failure prediction at underexposure illumination. 5

1.6 Examples of human pose estimation for sports video analytics. 6

1.7 Examples of related tasks of human pose estimation. 6

1.8 Illustration of single person pose estimation pipeline. 7

1.9 Illustration of multi-person pose estimation pipelines. 8

1.10 Illustration of challenge cases for existing method. 9

2.1 Illustration of superpixel generation. 15

2.2 Illustration of saliency region detection. 16

2.3 Illustration of convolution. 18

2.4 Illustration of pooling. 19

2.5 Illustration of ReLU. 20

2.6 Illustration of Dropout [63]. 21

2.7 Illustration of fully connection. 22

2.8 Illustration of the Residual block [64]. 23

2.9 Illustraion of the Bottleneck block [64]. 24

2.10 Illustration of the Fire block [65]. 25

2.11 Architecture of DeepPose [66]. 26

2.12 Architecture of Tompson et al. [69]. 27

2.13 Architecture of SHN [45]. 30

2.14 Definition of keypoints. 31

2.15 Illustration of keypoint connections and color coding. 32

3.1 Results of proposed method in challenge cases. 38

3.2 Framework of proposed method. 40

3.3 Initialize hand label based on ego-saliency. 42

3.4 Example of two-level classification. 44

3.5 Select superpixel samples based on energy optimization. 47

3.6 Challenge illumination cases comparison. 53

3.7 The original image has similar color in background and hand regions. . . . 54

3.8 Virtual keyboard interaction. 55

3.9 Performance of hand segmentation in virtual keyboard interaction. 57

4.1 Overview of cluster-wise learning based keypoint detection. 62

4.2 Illustration of our cluster-wise learning network. 66

4.3 Illustration of heatmaps of each stack module. 72

4.4 Demonstration of feature aggregation. 75

4.5 Visualization of tag embeddings of dense keypoint cluster. 76

4.6 Visualization of our results on MSCOCO validation dataset. 81

4.7 Visualization of our results on PoseTrack test dataset. 82

4.8 Visualization of failure cases. 83

4.9 More examples of feature aggregation. 84

5.1 Overview of proposed approach. 87

5.2 Illustration of lite blocks for compacting Hourglass module. 91

5.3 Illustration of bottom-up integral regression. 93

5.4 Analysis of error distributions. 98

5.5 Visualization of pose errors correction comparison. 98

5.6 Illustration the performance improvement according to the size of targets. . 99

5.7 Confidence score and OKS comparison. 99

6.1 Overview of proposed joint bottom-up and top-down learning. 102

6.2 The overview of our model architecture. 106

6.3 Illustration of bottom-up integral regression. 109

6.4 Illustration of associative tagmaps. 110

6.5 Illustration of holistically and separately learnt offsets. 111

6.6 Comparison of error distributions. 115

6.7 Comparison of the performance improvement according to the size of targets. 116

6.8 Qualitative results on MSCOCO validation500 dataset. 117

SUMMARY

This thesis focuses on the image-based learning methods that contribute to the natural Human-Computer Interaction (HCI). Compare to physical equipments, human body is a much more natural medium for many HCI applications in both egocentric (first-person) and conventional (third-person) viewpoint, such as hand segmentation for virtual keyboard interaction of smart wearable glasses and human pose estimation for automatic stage lights control system. Therefore, this thesis targets to analyze images of human body and unconsciously recognize their diversity activities in the wild. This task is challenging due to unpredictable environmental condition in the natural application scene, such as rapid changes in illuminations, background clutter, multiple persons with variant articulation of body limbs, diverse appearance, self-occlusion, different scales, significant overlap between neighbor persons and crowd. To overcome the problems and meet the target, we develop image-based learning methods for hand segmentation and multi-person pose estimation.

In this thesis, we first present an unsupervised on-the-fly hand segmentation approach which consists of top-down classification and bottom-up optimization. From the point of view of egocentric interaction loop, an unsupervised frame-level hand detector is proposed for the purpose of reducing the false positive caused by hand absence. We implement the frame-level detection by setting a non-interactive border based on an assumption that the hand is hardly to enter the view field from the top side for egocentric interaction. Based on the frame-level detection result, the superpixel-level and pixel-level classifiers are trained on-the-fly sequentially aimed at improving reliability of hand segmentation. To get stable samples for superpixel-level training, we select the candidates based on steps of confidence score calculation and energy optimization. In order to be robust to vary environmental conditions, the classifiers are updated from the bottom up based on the proposed performance evaluation method. This online-learning strategy makes our approach robust to varying

illumination conditions and hand appearances.

This thesis also discusses three approaches of multi-person pose estimation from perspectives network head and backbone structure. The cluster-wise learning method illustrates the benefits of using multiple task heads to enforce the network learn richer contextual information. The lite hourglass network presents the potential way of balancing performance and computational cost by using hybrid dilated blocks in the network backbone. The joint bottom-up and top-down learning method demonstrates the effectiveness of integrating instance and keypoint detection heads and the bottom-up integral regression. The overview of these three approaches are presented as following.

The recent popular method for pose estimation is extracting the local maximum response from each heatmap that trained for a specific type of keypoint. Learning each keypoint individually makes the approach difficult to parse from occlusion and variant pose. To target this problem, our cluster-wise feature aggregation method simultaneously learns complementary semantic information to encourage the detected keypoints subject to a certain contextual constraint. Specifically, we adopt stacked hourglass networks to generate paired multi-peak heatmaps for clusters of keypoints and the cluster configuration varies from stack to stack. Our network encodes global and local consistency of entire body and parts by dense and sparse cluster branches. To enhance feature passing from shallow stack to deep stack, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches. Meanwhile, the intra-cluster and inter-cluster relationships are embedded with tag learning to guide the instance grouping and individual keypoint identification. Thus, our network optimizes the features at a specific keypoint by the received information from multi-level context.

From the structure point of view, the previous proposed method relies on sequential downsampling and upsampling procedures to capture multi-scale features and stacking ba-

sic modules to reassess local and global contexts. However, the network parameters become huge and difficult to be trained under limited computational resource. Motived by this observation, we design a lite version of conventional module that uses hybrid convolution blocks to reduce the number of parameters while maintaining performance. Moreover, due to the limitation of heatmap representation, the networks need extra and non-differentiable post-processing to convert heatmaps to keypoint coordinates. Therefore, we also introduce a novel bottom-up integral regression operation to reduce the quantization error of converting heatmaps to keypoint coordinates specifically for bottom-up pipeline multi-person pose estimation methods.

Multi-person pose estimation is mostly challenged by occlusion and variant postures. Existing bottom-up and top-down methods have their own advantages and limitations. The bottom-up approaches detect body keypoints without advance knowledge of person locations which are firstly explored in the top-down pipelines. However, the benefit of unifying the two pipeline is lack of exploration. Motived by this observation, we further propose a joint bottom-up and top-down learning method to better predict multi-person join locations. Our network not only learns the body keypoints but also the centers which indicate different person instances. Meanwhile, the offsets from body keypoints to the centers are encoded for both retrieving and grouping the joints. Based on shared features of keypoints-to-instances and instances-to-keypoints branches, our method can efficiently perform multi-person pose estimation. Moreover, we improve the bottom-up integral regression by adding a local restriction to complement the over-segmentation.

We will introduce the significance of our research topic as well as our contribution separately in each of the following part: In Chapter 1, we introduce our research background, related previous research, research objective and overview of our research. In Chapter 2, we review the basic knowledge related to our research. In Chapter 3, we present an online learning approach that fully exploits the traits of egocentric vision. In Chapter 4, a deep convolution neural network is proposed for estimating multi-human poses in the wild. We

use cluster-wise learning and feature aggregation to achieve this goal. In Chapter 5, we present a lite hourglass network for learning human body keypoints with less parameters and computational cost. In Chapter 6, we introduce a joint learning method which explores more geometric information from the top-down pipeline to complement the appearance features learnt from the bottom-up pipeline. Finally in Chapter 7, we give a conclusion of our research topic and discuss the remaining issues.

CHAPTER 1
INTRODUCTION

1.1 Background

The Human-Computer Interaction (HCI) domain is a field of study that focuses on interpreting human intent to computer systems and feeding computer responses back to human. One of its essential targets is to provide user a natural interaction system that is simple and easy to learn. The most common systems are wearable sensor-based and computer vision-based, as illustrated in Figure 1.1. The wearable sensor-based system infers human activities according to the signals collected from sensors such as accelerometer, magnetometer, and gyroscope that are broadly used in smart phones, watches and bands. By contrast, the computer vision-based system relies much less on multiple devices while captures richer semantic information. It analyses images of the users and unconsciously recognizes their diversity activities. Comparatively speaking, computer vision-based approaches provide more natural interaction between human and computers than the sensor-based methods. To meet the end of natural HCI, this thesis focuses on the image-based methods for learning human body information which is the fundamental medium to the computer vision-based system.

Generally, computer vision enables computers to understand images or videos from the perspective of human visual system. It transforms the visual images into descriptions that can interact with other processes and trigger response actions. Computer vision-based HCI system benefits to a more natural way of interaction by using human body information as inputs rather than physical equipments. The human and scene of the interaction are captured by camera of the HCI system and then inferred by computer vision technology, such as hand segmentation, human detection, pose estimation, activity recognition, etc. Based

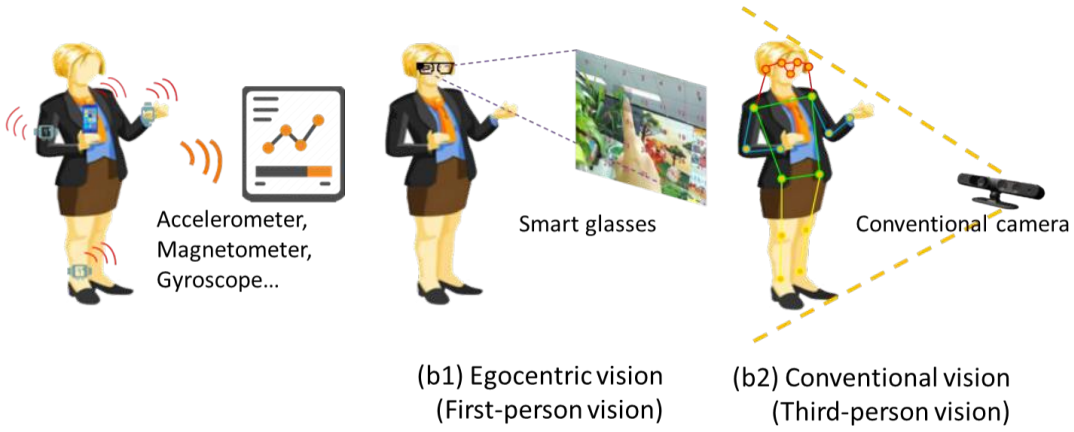


Figure 1.1: Illustration of the concept of sensor-based and vision-based HCI.

Figure 1.2: Example of egocentric vision-based HCI.

on the viewpoint of camera, computer vision-based HCI system can be further catalogued into egocentric vision-based and conventional vision-based, as illustrated in Figure 1.1(b). The conventional vision acquires images of the interaction subjects from the third-person viewpoint, as shown in Figure 1.1(b2). Compare to the conventional vision, the egocentric vision mainly focuses on understanding images from the first-person viewpoint of the interaction subjects, such as the hand image from the viewpoint of the smart camera wearer which is shown in Figure 1.1(b1).

The head mounted display and smart glasses are typical devices for the egocentric vision-based HCI system. As an example of egocentric vision based system shown in Fig-

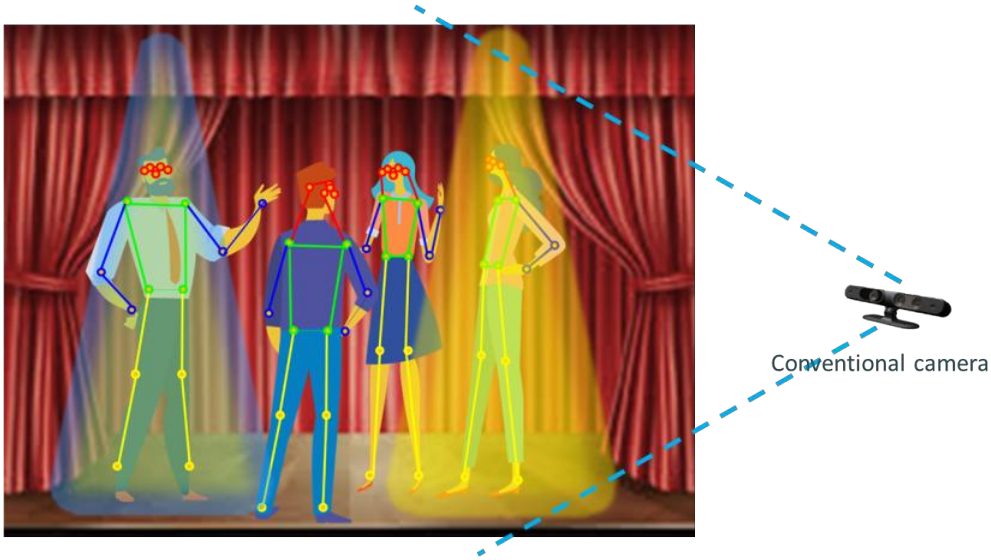


Figure 1.3: Example of conventional vision based HCI.

ure 1.2, the user wearing a head mounted display (Microsoft HoloLens [1]) utilizes hand gesture to browse the immersive content captured by a panoramic camera, such as Ricoh THETA [2]. Beside that, the egocentric vision also benefits the concentration of daily life records based on analysis of hand-object interaction. The essential technology supporting the system is to segment hand region from the captured image. Figure 1.3 illustrates an example of the conventional vision-based HCI system. The system camera captures the images of the stage, extracts body keypoints of the actors, analyzes their body poses and automatically adjusts the stage lights based on some certain kinds of poses, such as stand with arms akimbo. The fundamental process of the system is human pose estimation.

Due to the practical application scene of natural HCI is complex, there are a lot of challenges for computer vision-based system to interpret human body information, as examples shown in Figure 1.4. To segment hand region from natural scene, the HCI system needs to confront the challenges of varying illumination or background having similar skin color. To accurately estimate human pose, the HCI system has to overcome the occlusion between body parts or different people.

(a) Varying illumination (b) Similar skin color in background (c) Multi-person occlusion

Figure 1.4: Example of challenge cases for learning human body information.

1.2 Previous Research

According to a survey of egocentric vision [3], the most commonly explored objective of egocentric vision is object recognition and tracking. Furthermore, hands are among the most common objects in the user's field of view, and a proper detection, localization, and tracking could be a main input for other objectives. G. Serra et al. [4] develop a gesture recognition algorithm to be used in an interactive museum using 5 different hand gestures. Moreover, people in the visual field are also the important objects need to be understood. Lu and Grauman [5] and Lee et al. [6] summarize an egocentric video based on the important objects and faces. Singletary and Starner [7] use face detection to identify when first-person is involved in a social interaction. Ng et al. [8] infers body pose in egocentric video by leveraging interactions with another person whose body pose can be directly observed. Their research also shows that the human pose estimation from third-person viewpoint can be easily transferred and applied to the egocentric vision. The most recent research from G. Hidalgo et al. [9] presents a single-network approach integrating hand, body, face and foot pose estimation.

1.2.1 Hand Segmentation

Great efforts have been made in detecting user's hand from the egocentric video especially in pixel-level detection [10, 11, 4, 12, 13, 14, 15]. Most of these methods are under an implicit assumption that the hand presents in the video all the time. But, the assumption fails in many situations in which the hand is not used, such as before or after the human-

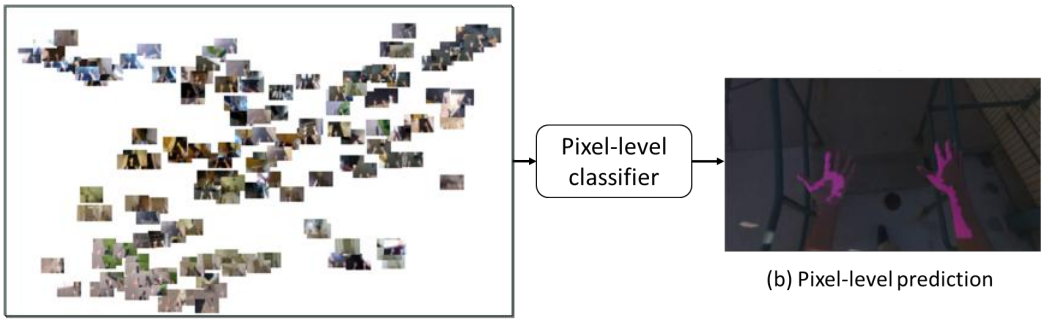


Figure 1.5: Example of failure prediction at underexposure illumination.
(a) is the overview of training dataset [10]. (b) is the prediction result generated by code of [10].

computer interaction. Subsequently, some cascade detection methods are put forwarded to get rid of the assumption by checking out hand presence before performing pixel-by-pixel classification [16, 17, 18]. However, these approaches rely on the existence of a large training set containing a broad variety of data which are collected from multiple users under diverse illumination conditions. Hand appearance varies greatly in diverse users and environmental conditions. Not only does the training set cost a lot of manual effort in data collection and labelling but also it does not guarantee to make the approach adapt to any hand appearance and environmental condition, as shown in Figure 1.5. However, the egocentric view also brings opportunities for hand detection and segmentation. Since the video is recorded from a first-person perspective, the occlusions are less likely to happen at the attention hand and the user prefers to concentrate on region in the center of view field. From this point of view, we devote to robust hand segmentation by utilizing coarse-to-fine online learning.

1.2.2 Human Pose Estimation

Human pose estimation refers to the task of locating and recognizing the body keypoints or joints (such as nose, eyes, ears, shoulders, elbows, wrists, hips, knees, ankles, etc.) from single or multiple persons in a single RGB image or a frame of videos. It is an essential component in many computer vision applications, such as human computer interaction

(a) Frame-i (b) Frame-j (c) Frame-k

Figure 1.6: Examples of human pose estimation for sports video analytics.

(a) Human semantic part segmentation (b) Fashion landmarks

Figure 1.7: Examples of related tasks of human pose estimation.
(a) Human semantic part segmentation [40]. (b) Fashion landmarks detection [41].

system, somatosensory gaming, human action recognition, video surveillance and sports video analytics. Figure 1.6 shows examples of human pose estimation. Beside body joints, keypoints can be extended to refer to the small visual units with semantic information indicating the compositions, shapes and poses of the target objects, such as finger joints or key positions of any other objects. Therefore, accurate keypoint detection in unconstrained environments brings benefit to other more detailed visual understanding tasks, including hand segmentation [19, 20] and pose estimation [21, 22], human action recognition [23, 24, 25, 26], human parsing [27, 28, 29], person re-identification [30, 31], viewpoint estimation [32, 33], salient object detection [34, 35], fashion landmark [36] and 3D reconstruction [37, 38, 39]. Figure 1.7 shows examples of human pose estimation related computer vision tasks.

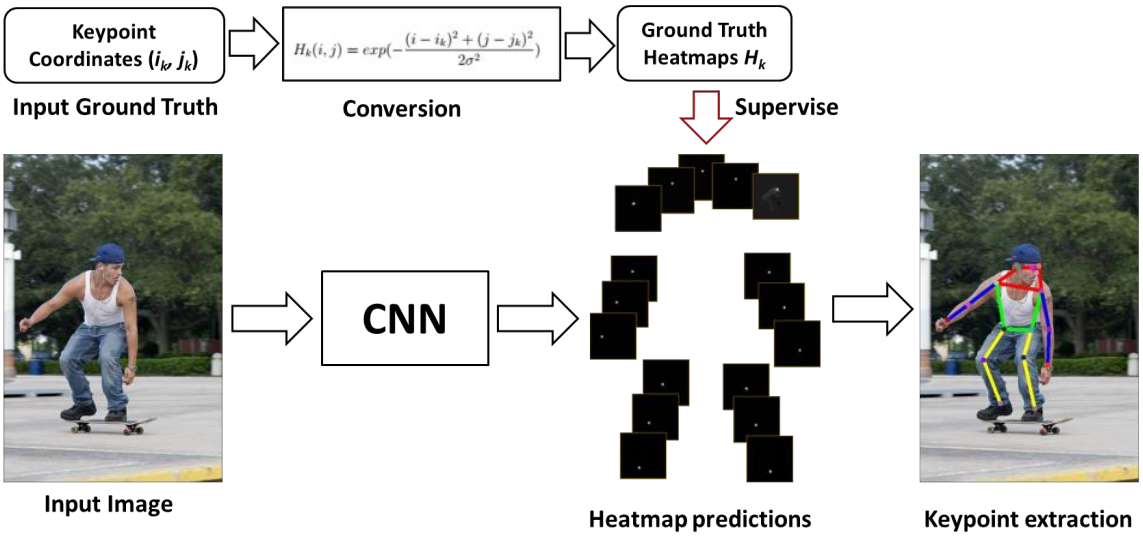


Figure 1.8: Illustration of single person pose estimation pipeline.

Single Person Pose Estimation

Similar as many computer vision tasks, the progress on human pose estimation problem is significantly improved by deep convolutional neural networks, especially by the network with structure of encoding-decoding. Figure 1.8 demonstrates the general pipeline of single person pose estimation. Generally, instead of directly regressing the keypoint coordinates, the network generates a likelihood heatmap for each keypoint and locates the keypoint as the point with the maximum likelihood in the heatmap. Therefore, the supervision of the network is a set of heatmaps which are converted from the input keypoint coordinates. Figure 1.8 also shows the detail of ground truth heatmap generation. The predicted heatmaps are then converted to keypoint coordinates by find the local maximums from the heatmaps. The type of each keypoint is retrieved by the order of the output. The final human pose can be obtained by connecting all detected keypoints. Since the natural HCI scene always contains multiple people, the hot spot has been focused on the much more complex task of multi-person pose estimation.

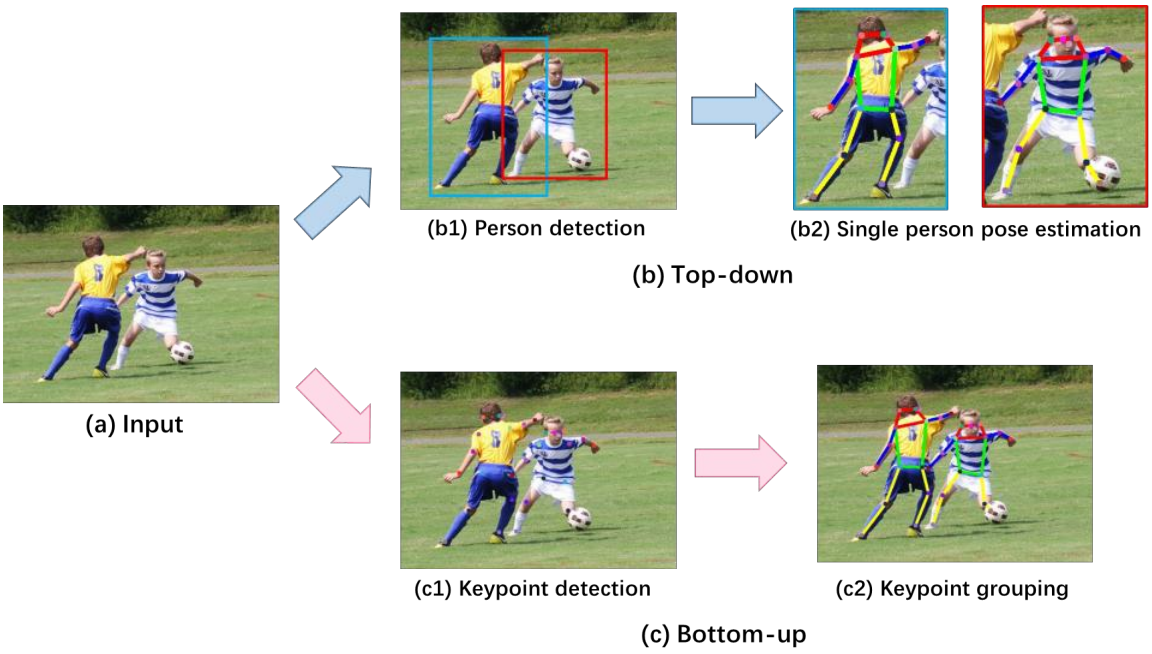


Figure 1.9: Illustration of multi-person pose estimation pipelines.

Multi-person Pose Estimation

In spite the good performance of extensive prior arts of single person pose estimation, it remains challenging to get accurate keypoint detection from variant articulation of body limbs, diverse appearance, self-occlusion, different scales, significant overlap between neighbor persons and crowd. Moreover, for multi-person pose estimation, it is also demanding to allocate the detected keypoints to multiple scales and unknown number of persons. The two major branches of approaches in multi-person pose estimation are structured in top-down and bottom-up. Figure 1.9 illustrates overview of these two pipelines. On one hand, the top-down methods [42, 43, 44, 45, 46, 47, 48] firstly detect persons and then repeatedly estimate pose for each of them. On the other hand, the bottom-up methods [49, 50, 51, 52, 53] detect body keypoints without advance knowledge of person locations and then group them into person instances. Comparatively, the bottom-up methods are the less costly processes, since they do not need to repeat the single person pose estimation for each person detection. However, current methods are still challenged by the occlusion. Figure 1.10 shows predictions generated by a state-of-the-art bottom-up method [52]. Be-

(a) Missing entire person (b) Missing keypoints

Figure 1.10: Illustration of challenge cases for existing method.

sides occlusion, the number of parameters is also another crucial factor effecting the overall performance of the network.

1.3 Research Objective

Compared with sensor-based methods, computer vision-based approaches provide more natural interaction between human and computers. This thesis aims at developing image-based approaches to learn human body information that is the natural medium contributing to computer vision-based HCI system. More specifically, this thesis proposes image-based approaches of hand segmentation that is robust to varying illumination and human pose estimation that is robust to occlusion with low computational cost.

For contributing to the egocentric vision-based HCI system, we target to segment user's hand from varying illumination and appearance conditions. Most previous approaches rely on the existence of a large training set containing a broad variety of data which are collected from multiple users under diverse illumination conditions. Hand appearance varies greatly in diverse users and environmental conditions. Not only does the training set cost a lot of manual effort in data collection and labelling but also it does not guarantee to make the approach adapt to any hand appearance and environmental condition. To meet this target, we explore properties of interaction loop and propose a coarse-to-fine online learning method.

To be helpful to the conventional vision-based HCI system, we aim to estimate body

pose of user's social objects robust to occlusion. In most recent methods, the body keypoints are inferred separately from each other according to their corresponding heatmaps. This separately learning may not fully exploit the relation between keypoints since the approaches are still struggling in overcoming the problem of occlusions and complex multi-person situations. To this end, we exploit relation between body keypoints by using cluster-wise learning. We also integrate bottom-up and top-down learning to explore geometric information that complements the appearance information. Besides that, we devote to make a balance between performance and computation cost. Existing methods are based on stacked modules which bring huge parameters. To solve the problem, we develop a lite version network which reduces the number of parameters while maintain the performance.

1.4 Overview of Our Research

Overall, one hand segmentation and three human pose estimation approaches constitute the basic structure and main content of our research.

1.4.1 Hand Segmentation

Coarse-to-fine Online Learning for Hand Segmentation in Egocentric Video

Egocentric view brings benefit that the video is recorded from a first-person perspective, the occlusions are less likely to happen at the attention hand and the user prefers to concentrate on region in the center of view field. However, the egocentric video also presents new challenges including rapid changes in illuminations, significant camera motion and background clutter. To address this issue, we propose a method for unsupervised hand detection and segmentation in egocentric video. In our approach, the frame-level hand presence or absence is observed based on motion saliency which is particular in the egocentric view. By combining motion and appearance property, we get unsupervised labeling results for the superpixel-level hand classification. Then, the pixel samples of hand are extracted according to confidences of the superpixels and used to train a pixel-level classifier

which produces fine-grained hand segmentation. In order to be robust with varying environmental condition, we constantly update the classifier and detector by using a bottom-up optimization method.

1.4.2 Human Pose Estimation

Cluster-wise Learning for Multi-person Pose Estimation

Similar with most recent methods [42, 45, 54, 55], the number of detection channels are intuitively set as the same of the number of keypoints. The keypoints are inferred separately from each other according to their corresponding heatmaps. This brings us an intuitive thinking that the separately learning may not fully exploit the relation between keypoints since the approaches are still struggling in overcoming the problem of occlusions and complex multi-person situations. Based on this point, we propose a cluster-wise feature aggregation network that exploits multi-level contextual association for multi-person pose estimation. Moreover, due to the limitation of heatmap representation, the networks need extra and non-differentiable post-processing to convert heatmaps to keypoint coordinates. Therefore,we propose a simple and efficient operation based on integral loss to fill this gap specifically for bottom-up pose estimation methods.

Lite Hourglass Network for Multi-person Pose Estimation

Recent multi-person pose estimation networks rely on sequential downsampling and upsampling procedures to capture multi-scale features and stacking basic modules to reassess local and global contexts. However, the network parameters become huge and difficult to be trained under limited computational resource. Motived by this observation, we design a lite version of Hourglass module that uses hybrid convolution blocks to reduce the number of parameters while maintaining performance.

Joint Bottom-up and Top-down Learning for Multi-person Pose Estimation

Multi-person pose estimation is mostly challenged by occlusion and variant postures. Existing bottom-up and top-down methods have their own advantages and limitations. The bottom-up approaches detect body keypoints without advance knowledge of person locations which are firstly explored in the top-down pipelines. However, the benefit of unifying the two pipeline is lack of exploration. Motived by this observation, we propose a joint bottom-up and top-down learning method to better predict multi-person join locations.

CHAPTER 2
REVIEW ON BASIC KNOWLEDGE

In this Chapter, we review the basic knowledge enlightening our research. Image segmentation results are crucial cues for locating objects and boundaries in images. Superpixel generation and saliency detection are important processing related to image segmentation. In following sections, we first briefly review these basic knowledge that inspire our proposed approach for hand segmentation. With development of deep learning, the Convolutional Neural Network (CNN/ConvNet) [56] becomes one of the mainstream tool to solve the problem of human pose estimation. Unlike handcrafted features used in classical approaches, CNN-based methods perform well with regard to both good feature extraction and strong discrimination ability. To design an effective CNN achieving enhanced performance, it is crucial to understand the principle of CNN components, the typical approaches representing the evolution of human pose estimation task and the evaluation metrics. Therefore, we also give a brief introduction of CNN-based methods with regard to human pose estimation.

2.1 Image segmentation

Image segmentation is the process of partitioning an image into perceptually meaningful segments such as superpixels or salient regions. The goal is to represent the image in an meaningful way and make it easier to further analyse. From the classification point of view, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share common characteristics.

2.1.1 Superpixel

Pixels are the foundational components in most computer vision and image processing tasks. However, they are not the natural representation of visual scenes but a set of discrete units. It would be more efficient to work with perceptually meaningful entities called superpixels that consist of pixels sharing common properties, such as color and texture. Compare to pixels, superpixels encode more information and reduce complexity of images to only a few hundred processing units. Superpixels are the results of an image oversegmentation that align better with object edges than rectangular image patches, as illustrated in Figure 2.1(b). They provide a convenient and compact representation of images that can be very useful for computationally demanding tasks, such as image segmentation, semantic labelling, etc.

Simple Linear Iterative Clustering (SLIC) [57] algorithm generates superpixels by clustering pixels having similar color and neighbouring position. This is done in the five-dimensional $labxy$ space, where lab is the pixel color vector in CIELAB color space and (x, y) is the pixel coordinate in image plane. The maximum possible distance between two colors in the CIELAB space is limited whereas the spatial distance in the xy plane depends on the image size. In order to cluster pixels in the five-dimensional $labxy$ space, the algorithm introduces a new distance measure that considers superpixel size. By using this distance measurement, the expected cluster sizes and their spatial extent are approximately equal. The normalized distance measure D_s to be used in the five-dimensional space is defined as follows:

$$D_s = d_{lab} + \frac{m}{S} d_{xy} \tag{2.1}$$

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2} \tag{2.2}$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \tag{2.3}$$

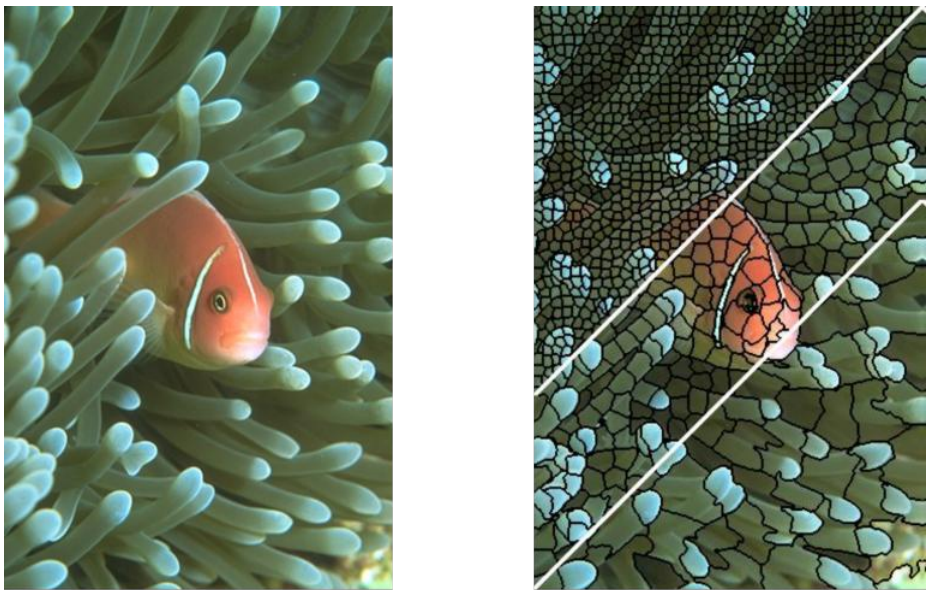


Figure 2.1: Illustration of superpixel generation.
The superpixel map is generated by SLIC [57].

where d_{lab} is the lab distance and d_{xy} is the xy plane distance and normalized by the gird interval S, m controls the compactness of a superpixel. The greater the value of m, the more spatial proximity is emphasized and the more compact the cluster.

SLIC [57] algorithm begins by sampling K regularly spaced cluster centers and moving them to seed locations corresponding to the lowest gradient position in a 3x3 neighborhood. This is done to avoid placing them at an edge and to reduce the chances of choosing a noisy pixel. Image gradients $G(x, y)$ are computed using both color and intensity information.

$$G(x, y) = ||I(x + 1, y) - I(x - 1, y)||^2 + ||I(x, y + 1) - I(x, y - 1)||^2 \tag{2.4}$$

where $I(x, y)$ is the lab vector corresponding to the pixel at position (x, y). Each pixel in the image is associated with the nearest cluster center whose search area overlaps this pixel. A new center is computed as the average $labxy$ vector of all the pixels belonging to the cluster. The process of associating pixels with the nearest cluster center and recomputing the cluster center is repeated until convergence. Connectivity can be enforced in the last step

Figure 2.2: Illustration of saliency region detection.
The saliency map is generated by global contrast based method [58].

of the algorithm by relabeling disjoint segments with the labels of the largest neighboring cluster. Figure 2.1 illustrates an example of superpixel generation with different sizes.

2.1.2 Saliency

Visual saliency indicates the most relevant parts that stand out from the image and attract the viewer's attention. It simulates the attention mechanism in organisms to filter the important regions of what they see. The filtered regions are then perceived and processed in finer details for the extraction of richer high-level information. Saliency detection has been widely used in many computer vision tasks since it helps find the objects or regions that efficiently represent a scene and thus harness complex vision problems such as fine grained segmentation.

A key step in detecting salient regions is to distinguish them from distractors [59]. To this end, some methods [60, 61, 58] firstly segment an input image into regions aligned with intensity edges using region generation methods such as SLIC [57]. Then, it computes a regional saliency map based on the uniqueness in terms of global regional contrast. In the global contrast based method [58], a region-based saliency algorithm is introduced by measuring the global contrast between the target region with respect to all other image regions. The image is first segmented into N regions $\{r_i\}_{i=1}^{N}$ and then saliency of each

region r_i is estimated as:

$$s(r_i) = \sum_{j=1}^{N} w_{ij} D_r(r_i, r_j) \tag{2.5}$$

where $D_r(r_i, r_j)$ captures the appearance contrast between two regions, such as color, texture and structure.The higher saliency scores are assigned to regions with the larger global contrast. w_{ij} is a weight term between r_i and r_j, which incorporates spatial distance and region size. Figure 2.2 shows an example of saliency map generated by the global contrast based method [58].

2.2 Convolutional Neural Network

CNNs can be considered as the special case of ordinary Neural Networks, which extract representative features of the input images and have shown state-of-the-art performance on various computer vision tasks. By using large amount of multiple type of layers, CNNs have powerful learning ability and can automatically learn complex representations from the training data.

2.2.1 Foundational Components

A simple CNN structure for keypoint location regression may consist of a sequence of foundational components, including convolutional layers, activation layers, pooling layers and fully connected layers. To output coordinate predictions, the CNN transforms the input image layer by layer from the original pixel values to the final keypoint coordinates. Each layer transforms an input volume to an output volume with some differentiable function that may or may not have parameters. The convolutional layer extracts locally correlated features by computing dot product between learned weights and a slice that relates to a small region of the input. The activation layer applies an elementwise activation function and embeds non-linearity in the feature space. The pooling layer performs a downsampling operation along the spatial dimensions and summarizes the results. The fully connected

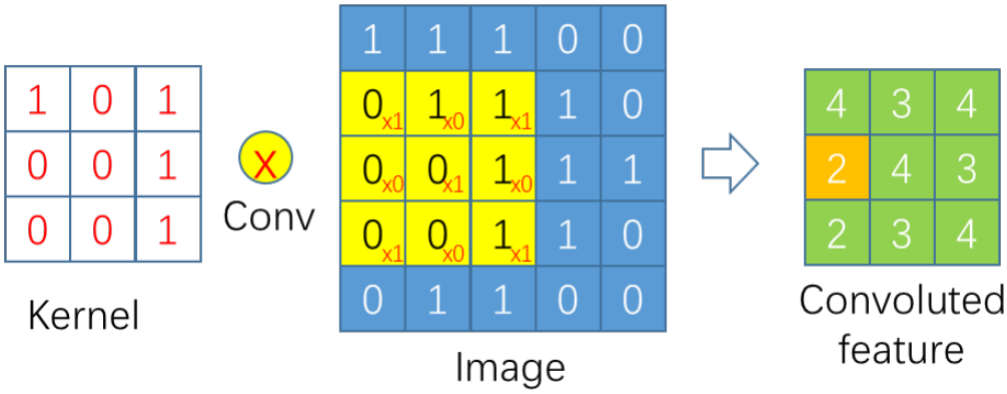


Figure 2.3: Illustration of convolution.

layer connects all outputs of previous layer and computes the keypoint coordinates. The details of each component are presented in following sections.

Convolutional Layer

The convolutional layer is the core component of a CNN. The layer is composed of a set of learnable kernels (filters) which associate each position in the feature map with a small area of the input image. The convolution operation is carried out by sweeping the kernels through out the inputs and calculating dot product between the weights and pixel values. It produces convolved features which encode the relevance between local pixels and develop visual context for analysis the image. Note that the weights are shared within each input which makes CNNs having condensed number of parameters compare to fully connected neural networks. Moreover, the sweeping stride may be different for all kernels and the feature map may be padded to fit the kernel sizes. The more kernels are deployed, the more channels (depth dimension) that the convolved features will contain but with more computational cost. Figure 2.3 illustrates the process of a 3x3 kernel convolution.

Pooling Layer

To reduce the dimensions of the feature maps, CNNs may include pooling layers which streamline the feature representation with a local aggregation approach. Generally, the

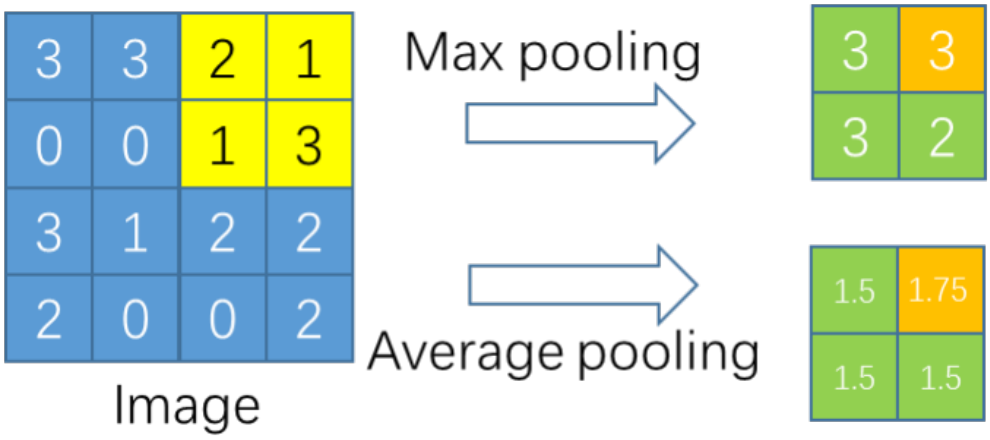


Figure 2.4: Illustration of pooling.

pooling layer is inserted in-between successive convolutional layers to progressively reduce the spatial size of the feature maps. Pooling layers not only decrease the redundant information but also make the features invariant to translational shifts and small distortions. The pooling operation may select out the maximum or average value from a local region to represent the dominant response. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height. In an extreme case, the pooling globally combines the output all the neurons of the convolutional layer into a single neuron in the next layer. Figure 2.4 shows examples of max pooling and average pooling.

Activation Function

Activation functions determine the output of a neural network. The function is attached to each neuron in the network and activates the neuron whose input is relevant for the network prediction. Non-linear activation functions enhance the network representation ability and makes the model to generalize or adapt with variety of data and to differentiate between the output. The Rectified Linear Unit (ReLU) is the most popular activation function used in almost all the convolutional neural networks or deep learning. It is defines as the positive

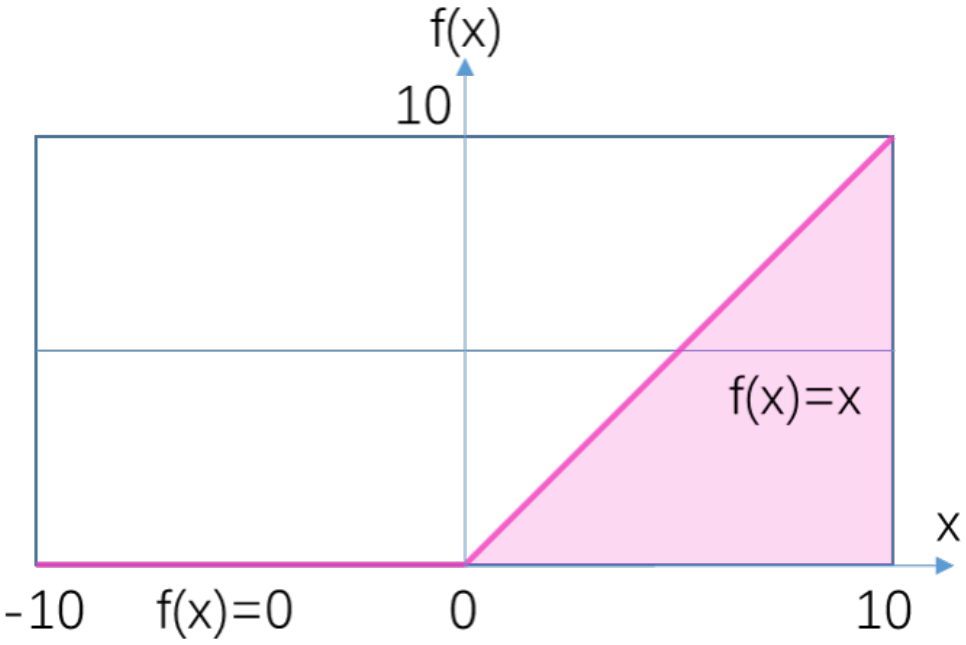


Figure 2.5: Illustration of ReLU.

part of its argument:

$$A(x) = max(0, x) \tag{2.6}$$

where x is the input to a neuron. ReLU is simple in mathematical operations and less computationally expensive than other non-linear functions, such as tanh or sigmoid. Figure 2.5 illustrates the activation of ReLU.

Batch Normalization

The batch normalization [62] layer aims to solve the problems of training caused by varying distribution of each layer's inputs. It normalizes a part of the model architecture and performs the normalization for each training mini-batch. With this regard, the CNN training is carried out under much higher learning rates and relaxing method of the parameter initialization. Generally, the batch normalization transform can be added to a network followed an activation layer. Formally, the batch normalization transforms a feature map F_i

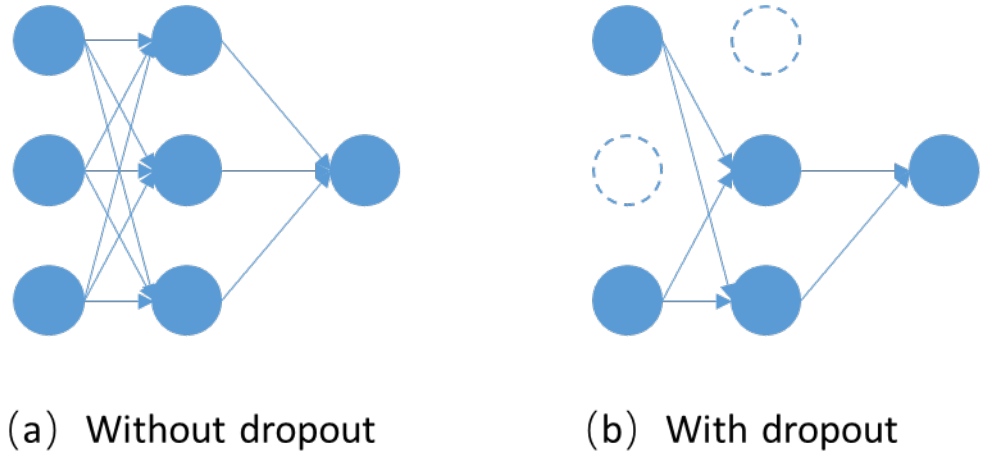


Figure 2.6: Illustration of Dropout [63].

to have zero mean and unit variance by following equations.

$$N_i = \frac{F_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{2.7}$$

$$Y_i = \gamma * N_i + \beta \tag{2.8}$$

where F_i is the input feature map, N_i is the normalized feature map, μ_B denotes the mini-batch mean and σ_B indicates the mini-batch variance, ϵ is a constant added to the mini-batch variance for numerical stability. Y_i is the linear transforms of normalized values with learnable scale and shift parameters γ and β. By doing this, batch normalization preserves the representation ability of the network but not brings too many parameters.

Dropout Layer

The Dropout [63] is an approach to prevent over-fitting in neural networks by reducing interdependent learning amongst the neurons. The key idea of dropout layer is to randomly drop neurons in accordance with a certain probability from the networks during training. The abandoned neurons are not contributed to a particular forward or backward pass. This operation reduces the over-fitting caused by co-dependency amongst neurons and improves

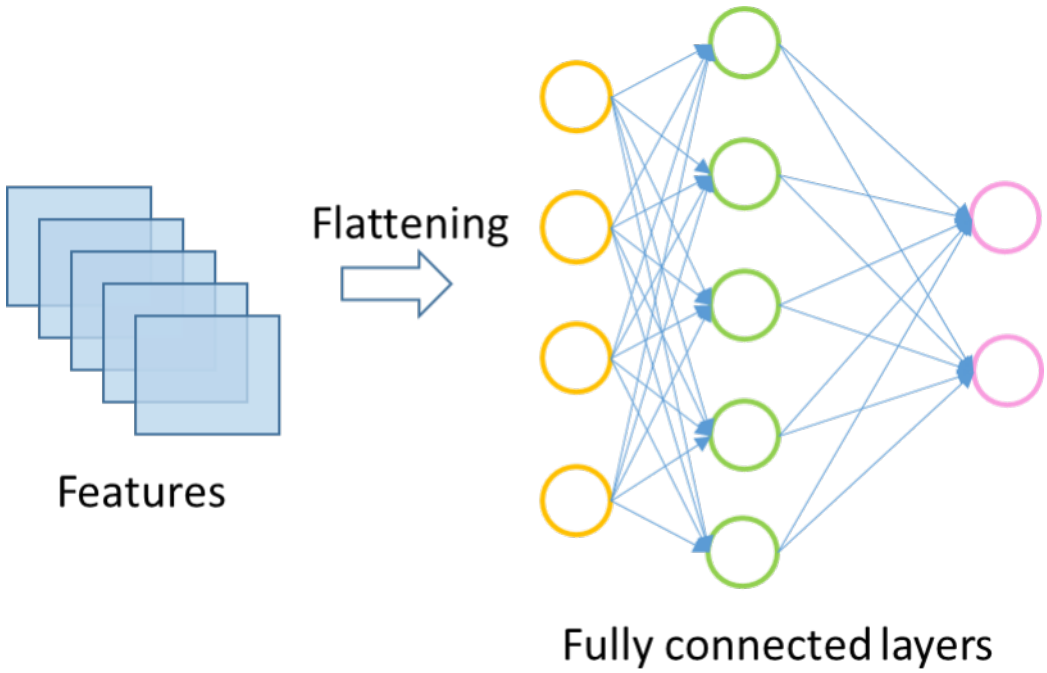


Figure 2.7: Illustration of fully connection.

network generalization. At test time, all dropped neurons are used but with reduced activations based on the dropout proportion. Figure 2.6 demonstrates the process of dropout.

Fully Connected Layer

In fully connected layer, all neurons are connected to learn non-linear combinations of the high-level features as represented by the output of the convolutional or pooling layer. It drives the network to produce a final label to describe the input image. More specifically, the output of convolution or pooling layer is flattened into a single vector of values that represent the probabilities of certain features belongs to a specific label. The final label is decided with the highest probability. However, the fully connected layers have fixed dimensions and throw away spatial coordinates. Recent networks replace the fully connected layers with 1x1 convolutions that make the networks can take input of any size and output classification maps. Figure 2.7 shows an example of fully connection.

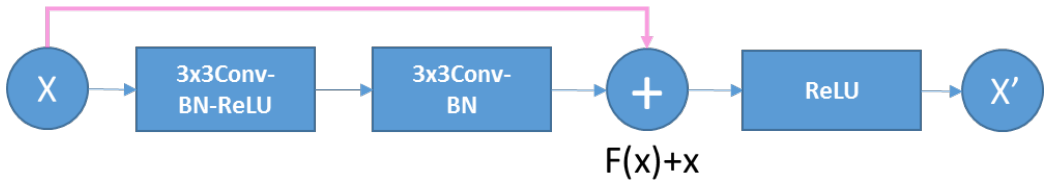


Figure 2.8: Illustration of the Residual block [64].

2.2.2 Evolutionary Components

To improve the CNNs performance, recent designed architectures become deeper and wider and the foundational components are evolved from layers to blocks. By increasing the network depth, more abstractive high level features are combined with the lower and mid-level features and the representational capacity of a CNN can be gradually enhanced. In addition, another major improvement in representational capacity of the CNN is achieved by the evolution of the foundational components. Different ideas in the component design have been explored such as the use of multi-path and different convolutional kernel sizes, residual connection and attention processing. With this regard, the subsequent sections cover the elementary understanding of various CNN components.

Residual Block

As more layers using certain activation functions are added to neural networks, the gradients of the loss function approaches zero, making the network hard to train. Residual block [64] is a popular solution to solve the gradient vanishing problem. It provides residual connections straight to earlier layers. The core idea of residual block is introducing a so-called identity shortcut connection that skips one or more layers. As seen in Figure 2.8, the residual connection (pink path) directly adds the value at the beginning of the block, x, to the end of the block $(F(x) + x)$. This residual connection doesn't go through activation functions that squashes the derivatives, resulting in a higher overall derivative of the block.

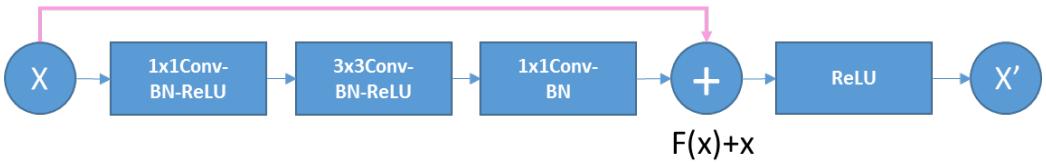


Figure 2.9: Illustraion of the Bottleneck block [64].

The Residual block [64] is defined as:

$$y = F(x) + x \tag{2.9}$$

where x and y are the input and output vectors of the layers. The function $F(x)$ denotes the residual mapping to be learned and can represent multiple convolutioal layers. The element-wise addition is performed on two feature maps, channel by channel. The shortcut connections in Eqn.(2.9) introduce neither extra parameter nor computation complexity.

Bottleneck Block

The Bottleneck block [64] is an improved version of residual block for deeper networks. It preserves the identity shortcut but uses a 1x1 convolutional layers to reduce the number of feature maps before and after the expensive 3x3 convolution. For each residual function F, the Bottleneck block uses a stack of three convolutional layers instead of two. Figure 2.9 illustrates the Bottleneck block. In the variant form of the Bottleneck block, the divide-transform-concatenate is done by pointwise grouped convolutional layer, which divides its input into groups of feature maps and perform normal convolution respectively, their outputs are depth-concatenated and then fed to a 1x1 convolutional layer.

Fire Block

The Fire block from Squeezenet [65] is designed to reduce number of parameters while maintaining competitive accuracy. It also replaces 3x3 convolutional kernels with 1x1 kernels, since a 1x1 kernel has 9X fewer parameters than a 3x3 kernel. Moreover, it decreases

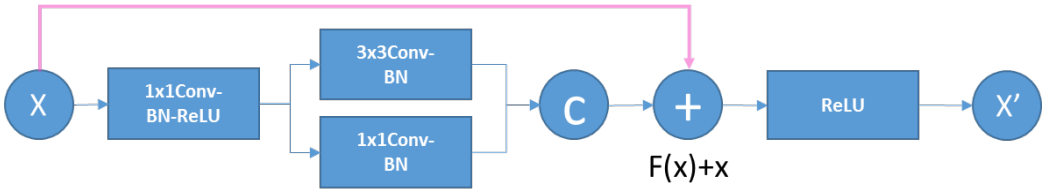


Figure 2.10: Illustration of the Fire block [65].

the number of input channels by 1x1 kernels and feed the low dimension features into 3x3 kernels. Compare to the Bottleneck block [64], the Fire block uses multi-path in the dimension expansion phase, as illustrated in Figure 2.10. The multi-path consists of not only 3x3 but also 1x1 convolitonal kernels which helps increasing the output dimension without introducing too many parameters.

2.3 Human Pose Estimation with CNN

With the introduction of DeepPose by Toshev et al. [66], CNN-based approaches are replacing the classic feature engineering-based methods and become the dominant tools for human pose estimation. By carefully designed learning targets and network architectures, recent pose estimation systems yield drastic improvements on standard benchmarks.

2.3.1 DeepPose: Human Pose Estimation via Deep Neural Networks

The DeepPose [66] was the first major approach that applied deep learning to human pose estimation. By using CNN, there is no need to explicitly design feature representations, detectors for parts and interactions between joints. This approach formulates the pose estimation as a CNN-based regression problem towards body joints.

As shown in Figure 2.11, it utilizes a 7-layered convolutional neural network to regress the location of each body joint from the input image. The network outputs $2k$ joint coordinates $(x_i, y_i) * 2$ for $i \in 1, 2, ..., k$, where k is the number of joints. Since each joint regressor uses the full image as a signal, the CNN is capable of capturing the full context of each body joint. Moreover, this approach uses a cascade of such regressors to refine

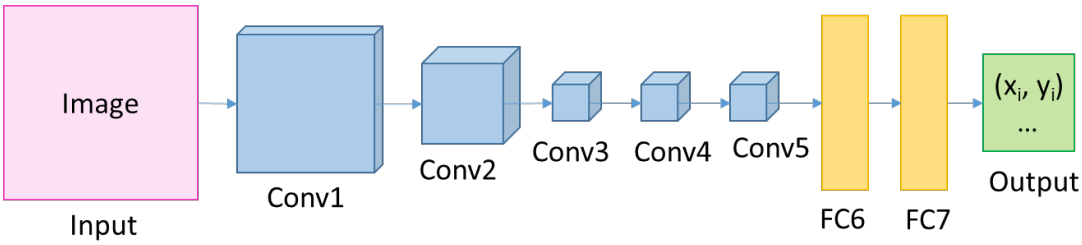


Figure 2.11: Architecture of DeepPose [66].

Table 2.1: Compare DeepPose [66] with prior-art: LSP dataset(PCP@0.5).

Method	Upper Arm	Lower Arm	Upper Leg	Lower Leg	Ave.
Johnson et al. [67]	0.54	**0.38**	0.75	0.66	0.58
Wang et al. [68]	**0.565**	0.37	0.76	0.68	0.59
DeepPose [66]	**0.56**	**0.38**	**0.77**	**0.71**	**0.61**

the pose estimates and get better predictions. Images are cropped around the initially predicted joint and fed to the next stage, in this way the subsequent pose regressors see higher resolution images and thus learn features for finer scales which ultimately leads to higher precision. It trains a linear regression on top of the last network layer to predict a pose vector by minimizing $L2$ distance between the prediction and the true pose vector. Then the $L2$ loss for obtaining optimal network parameters reads:

$$L = \sum_{(x_i,y_i)\in(D_N)} \sum_{i=1}^{k} ||y_i - \psi_i(x;\theta)||_2^2 \tag{2.10}$$

where D_N is the normalized training set, k is the number of keypoints, y_i is the ground truth, $\psi_i(x;\theta)$ is the prediction under parameter θ. The network performance is shown in Table 2.1.

DeepPose [66] innovatively applied CNN to human pose estimation and provoked an research boom in this direction. However, directly regressing to keypoint coordinate is difficult and adds learning complexity which weakens generalization and hence performs

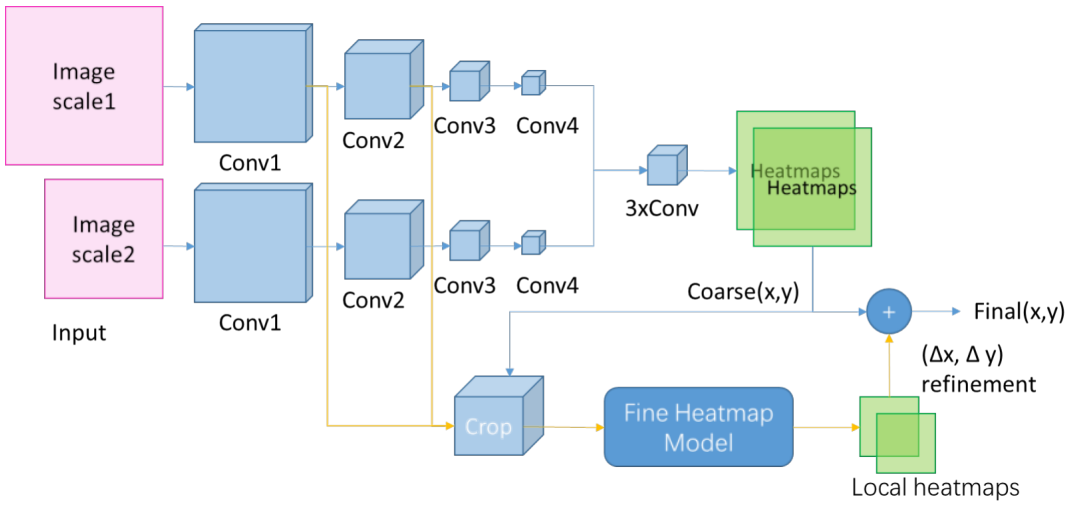


Figure 2.12: Architecture of Tompson et al. [69].

poorly in certain regions. Recent state-of-the-art methods transform the problem to estimating K heatmaps to indicate the probability of keypoint locations. The next approach was fundamental in introducing this idea.

2.3.2 Efficient Object Localization Using Convolutional Networks

Rather than directly regressing body joint coordinates, Tompson et al. [69] propose a method that outputs heatmaps that describe the likelihood of a joint occurring in each spatial location. The input image is fed into multiple resolution banks in parallel to simultaneously capture features at a variety of scales. The outputs are a set of discrete heatmaps instead of continuous regressions. Each heatmap predicts the probability of a specific type of joint occurring at each pixel. Inspired by this network, many following approaches predict heatmaps instead of direct coordinates regression.

As shown in Figure 2.12, a multi-resolution CNN architecture is used to implement a sliding window detector to produce a coarse heatmap output. The model is trained by minimizing the Mean-Squared-Error (MSE) distance between the predicted and target heatmaps. The target is a 2D Gaussain of constant variance centered at the ground truth

Table 2.2: Compare Tompson et al. [69] with prior-art: MPII dataset(PCKh@0.5)

Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	UpperBody	FullBody
Yang et al. [70]	73.2	56.2	41.3	32.1	36.2	33.2	34.5	43.2	44.5
Pischulin et al. [71]	74.2	49.0	40.8	34.1	36.5	34.4	35.1	41.3	44.0
Tompson et al. [69]	**96.1**	**91.9**	**83.9**	**77.8**	**80.9**	**72.3**	**64.8**	**84.5**	**82.0**

coordinate (x, y). The objective function is:

$$E = \frac{1}{N} \sum_{k=1}^{N} \sum_{x,y} ||H_k^*(x, y) - H_k(x, y)||^2 \tag{2.11}$$

where H^* and H are the predicted and ground truth heatmaps for the the $k - th$ keypoint.

To produce fine predictions, this approach uses a graphical model to learn typical spatial relationships between joints from the coarse heatmap model. In essence, the model consists of the heat-map-based parts model for coarse localization, a module to sample and crop the convolution features at a specified (x, y) location for each joint, as well as an additional convolutional model for fine-tuning. The network performance is shown in Table 2.2.

The method of Tompson et al. [69] verified that heatmaps work better than direct joint regression. It jointly uses CNN and graphical model to produce fine results. However, these methods lack structure modelling. The space of 2D human poses is highly structured because of body part proportions, left-right symmetries, interpenetration constraints, joint limits (e.g. elbows do not bend back) and physical connectivity (e.g. wrists are rigidly related to elbows), among others. Modelling this structure should make it easier to pinpoint the visible keypoints and make it possible to estimate the occluded ones. The next approach tackles this by an innovative network structure.

2.3.3 Stacked Hourglass Networks for Human Pose Estimation

The Stacked Hourglass Network (SHN) [45] introduces a remarkable convolutional network architecture for the task of human pose estimation. Its main principle is to utilize repeated bottom-up (from high resolutions to low resolutions) and top-down (from low resolutions to high resolutions) feature processing to capture the various spatial relationships associated with the body. The basic module is called Hourglass based on the successive steps of pooling and upsampling to capture information at multiple scales. Moreover, each Hourglass module uses skip connections to preserve spatial information at each resolution and passes it along for upsampling until reach the largest scale. Further, the network stacks multiple hourglass modules with intermediate supervisions to improving the performance.

Figure 2.13 illustrates the architecture of the stacked hourglass network. The network pools down the feature maps to a very low resolution, and then upsamples and combines features across multiple resolutions. The most significant attribute of the network is the symmetric topology that consists of a series of convolution, pooling and upsampling layers. This architecture provides the benefit of capturing both global and local features for the pixel-wise prediction tasks. Moreover, the network is extended by repeating multiple Hourglass modules. Each layer in the Hourglass makes extensive use of the residual bottleneck block with special setting for the number of feature channels. The MSE loss is applied comparing the predicted heatmap to a ground-truth heatmap consisting of a 2D Gaussian (with standard deviation of 1 pixel) centered on the joint location. The network performance is shown in Table 2.3.

This innovative design shows the importance of multi-scale feature extraction and intermediate supervision for human pose estimation. Its idea enlightens the research afterwards with the thoughts on fully usage of global and local information and coarse-to-fine strategy to enforce the network produce better predictions.

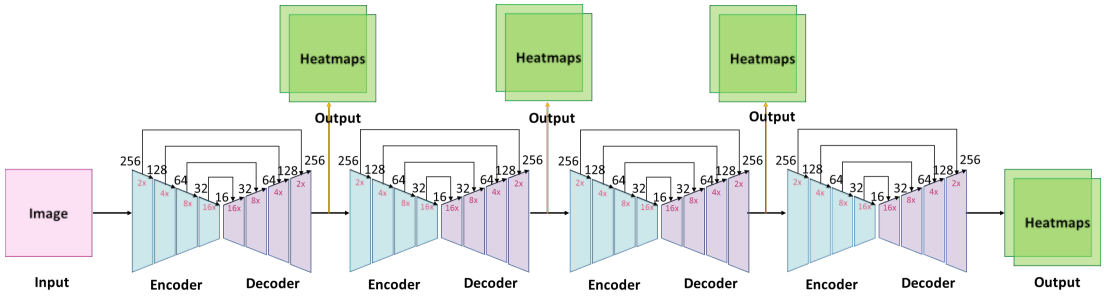


Figure 2.13: Architecture of SHN [45].

Table 2.3: Compare SHN [45] with prior-art: MPII dataset(PCKh@0.5)

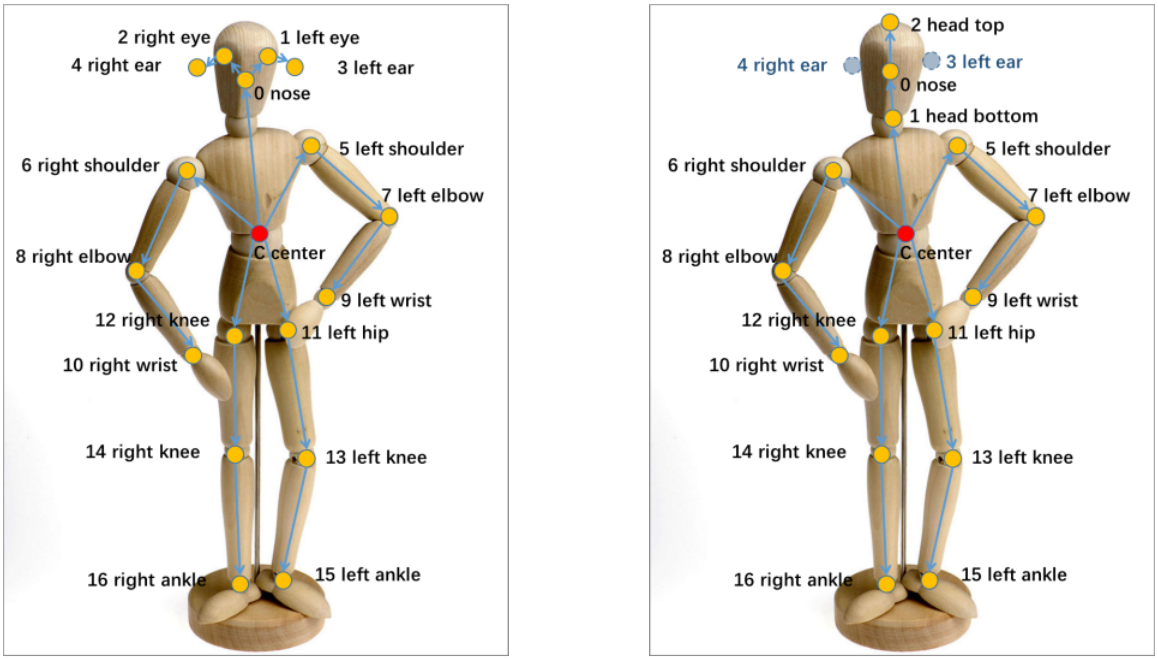
Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
Hu et al. [72]	95.0	91.6	83.0	76.6	81.9	74.5	69.5	82.4
Wei et al. [73]	97.8	95.0	88.7	84.0	88.4	82.8	79.4	88.5
SHN [45]	**98.2**	**96.3**	**91.2**	**87.1**	**90.1**	**87.4**	**83.6**	**90.9**

2.4 Evaluation for Human Pose Estimation

In this section, we introduce the common evaluation datasets and metrics which are used to measure the performance of human pose estimation models.

2.4.1 Datasets

The two main datasets for human pose estimation task are MSCOCO [74] and PoseTrack [75]. Figure 2.14 gives intuitive introduction of the annotations of these two datasets. There are 17 keypoints from body parts of face and limbs. The annotation for each people is a sequence of 17 keypoints with a certain order. Therefore, we can retrieve the type of each keypoint from a predefined dictionary (e.g., the second keypoint is the left eye). Moreover, the body keypoints are highly related with their adjacent keypoints based on the physical structure. Figure 2.14 uses blue arrows to indicate the adjacent relation between keypoints and the front point is more stable than the gear point. Besides the difference in



(a) MSCOCO (b) PoseTrack

Figure 2.14: Definition of keypoints.

(a) MSCOCO [74] datasets. (b) PoseTrack [75] datasets. The center point is the average of the all keypoints. The arrows show the adjacent relation between keypoints. The ear keypoints of PoseTrack have placeholders rather than real labels.

keypoint types from one to four, the annotation format of PoseTrack [75] is compatible with MSCOCO [74]. The ear keypoints of PoseTrack have placeholders rather than real labels. By averaging locations of all keypoints, we can obtain the center point which is a useful cue to identify instance. More specifically, the MSCOCO [74] dataset consists of 118,287 training, 5000 validation and 40,670 testing images (including 20,880 test-dev and 19,790 test-challenge). And over 150K labeled person instances from 56,599 training and 2346 validation images are publicly available. The PoseTrack [75] dataset is a large-scale benchmark for human pose estimation and tracking in video and based on the diverse real-world activity videos from the MPII Human Pose dataset [76]. It annotates 15 body keypoints from 550 video sequences (292 training, 50 validation and 208 testing) with 66, 374 frames. The length of the most videos ranges between 41 and 151 frames, and 30 frames from the middle of the sequence have annotations. The validation and test sequences are densely annotated with a step of four frames. In total, PoseTrack2018 provides around

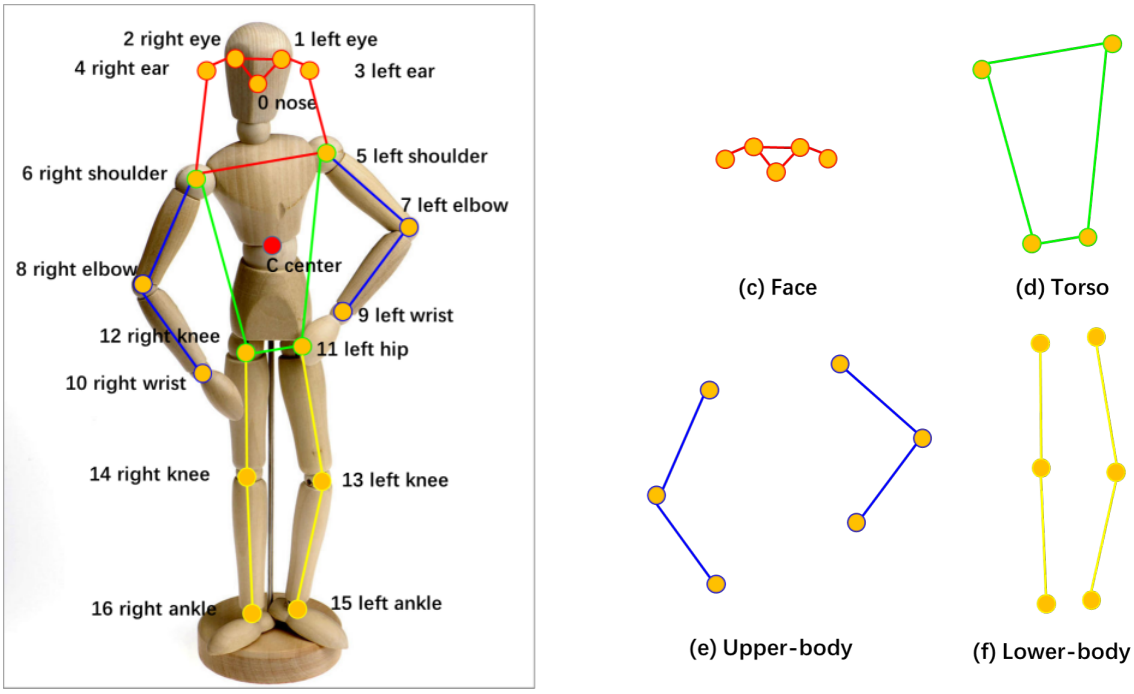


Figure 2.15: Illustration of keypoint connections and color coding.

23,000 labeled frames and 153,615 pose annotations. To further semantic understanding, we usually connect the detected keypoints based on the body part structure with predefined colors, as shown in Figure 2.15.

2.4.2 Metrics

Percentage of Correct Parts (PCP)

The metric named Percentage of Correct Parts (PCP) [77] measures the detection rate of limbs (body parts). A candidate body part is labelled as correct if its segment endpoints lie within 50% of the length of the ground-truth annotated endpoints. This is commonly denoted as $PCP@0.5$. The PCP penalizes shorter limbs more since shorter limbs have smaller thresholds. For a specific part ,

$$PCP_i = \frac{C_i}{T_i} \tag{2.12}$$

where C_i is the number of correct parts for entire dataset and T_i is the number of total parts of entire dataset. The higher the PCP, the better the model performance. However, PCP has the drawback of penalizing shorter limbs, such as lower arms, which are usually harder to detect.

Percentage of Correct Keypoints (PCK)

The metric named Percentage of Correct Keypoints (PCK) [28] defines a candidate keypoint to be correct if the distance between the predicted and the true joint is within a certain threshold. If the candidate keypoint falls within $\alpha * max(h, w)$ pixels of the ground truth keypoint, where h and w are the height and width of the bounding box respectively, and α controls the relative threshold for considering correctness. Commonly, the PCK is calculated refer to the head bounding box with $\alpha = 0.5$ and denoted as PCKh@0.5. This metric alleviates the drawback of PCP since the detection criteria for all joints are based on the same distance threshold.

Object Keypoint Similarity (OKS)

The Object Keypoint Similarity (OKS) [74] measures the similarity between ground truth keypoints and predicted keypoints based on distance rather than the bounding box overlap. Given the OKS, we can compute average precision (AP) and average recall (AR) of the keypoint detection just as the IoU (Intersection Over Union) allows us to compute these metrics for object detection. The OKS is defined as

$$OKS = \frac{\sum_i [exp(-d^2/2s^2{k_i}^2\delta(v_i > 0))]}{\sum_i [\delta(v_i > 0)]} \tag{2.13}$$

Where, d_i is the Euclidean distance between detected keypoint and ground truth, the v_i indicates visibility of the ground truth, sk_i is the standard deviation of an unnormalized Guassian, where s is the object scale and k_i is a per-keypiont constant that controls falloff.

For each keypoint, this yields a keypoint similarity that ranges between 0 and 1. To put it simply, OKS plays the same role that IoU plays in object detection. It is calculated from the distance between predicted points and ground truth points normalized by the scale of the person. Typically, the following standard average precision and recall scores are reported in papers: AP/AR (the mean of AP and AR scores at 10 positions, OKS = 0.50, 0.55, . . . , 0.90, 0.95), AP50/AR50 (AP and AR at OKS = 0.50), AP75/AR75 (AP and AR at OKS = 0.75), APM/ARM for medium objects, and APL/ARL for large objects.

Error Diagnosis

Images of natural HCI scene may contain multiple instances of people, and a variable number of body parts (or keypoints) are visible. However, the evaluation metrics like PCK [28] or OKS [74] are not sufficient for analysis the underlying causes of error that may affect performance. Ronchi et al. [78] propose an error diagnosis system for fully understanding the behaviour of algorithms.

The error diagnosis system [78] defines four types of localization errors, including *Jitter*, *Miss*, *Inversion* and *Swap*, as a function of the keypoint similarity $ks(.,.)$ between the detected keypoint $\hat{\theta}_i^{(p)}$ and the ground truth annotation $\theta_i^{(p)}$ of a person p. Every keypoint detection having a keypoint similarity with its ground truth that exceeds 0.85 is considered *Good*.

$$ks(\hat{\theta}_i^{(p)}, \theta_i^{(p)}) = \exp^{-\frac{||\hat{\theta}_i^{(p)} - \theta_i^{(p)}||_2^2}{2s^2k_i^2}} \tag{2.14}$$

The *Jitter* is defined as a small error around the correct keypoint location.

$$0.5 \leq ks(\hat{\theta}_i^{(p)}, \theta_i^{(p)}) < 0.85 \tag{2.15}$$

The *Miss* denotes large localization error which means that the detected keypoint is not

within the proximity of any body part.

$$ks(\hat{\theta}_i^{(p)}, \theta_i^{(q)}) < 0.5, \forall q \in \mathcal{P} \quad and \quad \forall j \in \mathcal{J} \tag{2.16}$$

where $\mathcal{P}$ is the set of people, $\mathcal{J}$ is the set of keypoints. The *Inversion* measures confusion between semantically similar parts belonging to the same instance. The detection is in the proximity of the true keypoint location of the wrong body part.

$$ks(\hat{\theta}_i^{(p)}, \theta_i^{(p)}) < 0.5 \tag{2.17}$$

$$\exists j \in \mathcal{J} \quad | \quad ks(\hat{\theta}_i^{(p)}, \theta_i^{(p)}) \geq 0.5 \tag{2.18}$$

The *Swap* indicates confusion between semantically similar parts of different instances. The detection is within the proximity of a body part belonging to a different person.

$$ks(\hat{\theta}_i^{(p)}, \theta_i^{(p)}) < 0.5 \tag{2.19}$$

$$\exists j \in \mathcal{J} \quad and \quad \exists q \in \mathcal{P} \quad | \quad ks(\hat{\theta}_i^{(p)}, \theta_i^{(q)}) \geq 0.5 \tag{2.20}$$

The false positive keypoint localizations are typically caused by the self occlusion of body parts or occlusion by other objects. In these circumstances, the OKS scores of detected keypoints are lower than the evaluation thresholds.

CHAPTER 3
COARSE-TO-FINE ONLINE LEARNING FOR HAND SEGMENTATION IN EGOCENTRIC VIDEO

3.1 Introduction

Hand segmentation is one of the most fundamental and crucial steps for egocentric human-computer interaction. The special egocentric view brings new challenges to hand segmentation task, such as the unpredictable environmental conditions. The performance of traditional hand segmentation methods depend on abundant manually labeled training data. However, these approaches do not appropriately capture the whole properties of egocentric human-computer interaction for neglecting the user specific context. It is only necessary to build a personalized hand model of the active user. Based on this observation, we propose an online-learning hand segmentation approach without using manually labeled data for training. Our approach consists of top-down classifications and bottom-up optimizations. More specifically, we divide the segmentation task into three parts, a frame-level hand detection which detects the presence of the interactive hand using motion saliency and initializes hand masks for online learning, a superpixel-level hand classification which coarsely segments hand regions from which stable samples are selected for next level, and a pixel-level hand classification which produces a fine-grained hand segmentation. Based on the pixel level classification result, we update the hand appearance model and optimize the upper layer classifier and detector. This online-learning strategy makes our approach robust to varying illumination conditions and hand appearances. Experimental results demonstrate the robustness of our approach.

Egocentric view brings benefit that the video is recorded from a first-person perspective, the occlusions are less likely to happen at the attention hand and the user prefers to concen-

trate on region in the center of view field. However, the egocentric video also presents new challenges including rapid changes in illuminations, significant camera motion and background clutter. To address this issue, we propose a method for unsupervised hand detection and segmentation in egocentric video. In our approach, the frame-level hand presence or absence is observed based on motion saliency which is particular in the egocentric view. By combining motion and appearance property, we get unsupervised labeling results for the superpixel-level hand classification. Then, the pixel samples of hand are extracted according to confidences of the superpixels and used to train a pixel-level classifier which produces fine-grained hand segmentation. In order to be robust with varying environmental condition, we constantly update the classifier and detector by using a bottom-up optimization method. We test our method on challenging datasets and the experimental results show that our method robustly produces precise segmentation, as illustrated in Figure 3.1.

In summary, our method makes three main contributions:

- We propose a frame-level hand presence detection method that utilizes hand motion saliency in the egocentric human-computer interaction, which reduces the false positive rate for the final target of pixel-level hand segmentation.
- We present a top-down cascaded classification method which segments hand hierarchically in levels of frame, superpixel and pixel so as to reduce computational cost, in which the classifiers are trained on-the-fly so as to be robust to diverse users.
- We analyze and optimize the online trained classifiers by a bottom-up method which makes the hand segmentation robust to varying environmental conditions.

3.2 Related Work

Egocentric vision is an emerging area in computer vision. According to survey of [3] that the most commonly explored objective of egocentric vision is object recognition and tracking. Furthermore, hands are among the most common objects in the user's field of

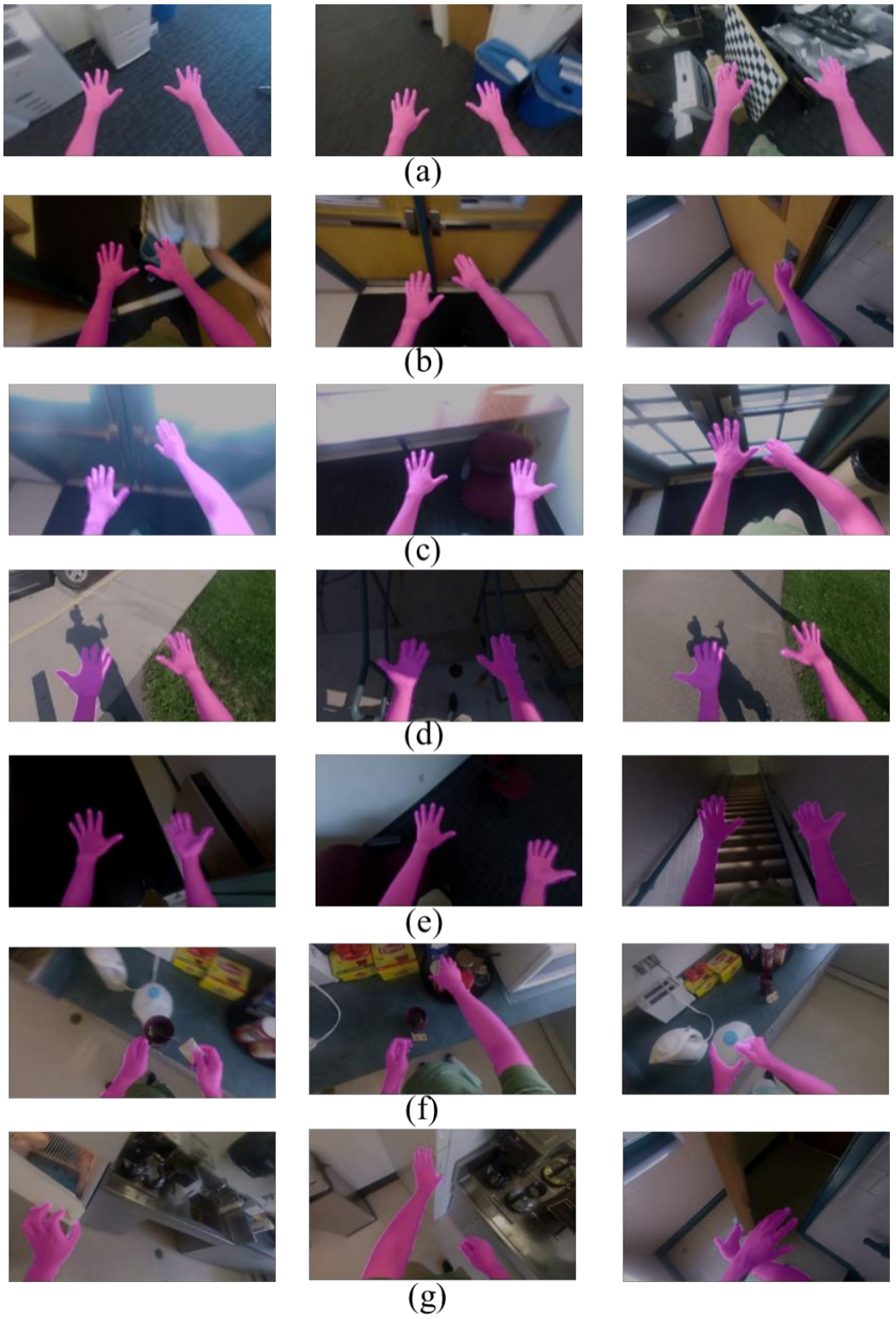


Figure 3.1: Results of proposed method in challenge cases.

From (a) to (g) are cases of hands are motion blur, background having skin-color, frames are overexposed, hands in contrast shadow, frames are underexposed, hands interacting with objects, and hands in varies poses.

view, and a proper detection, localization, and tracking could be a main input for other objectives, such as gesture recognition, understanding hand-object interactions and activity recognition [13, 79, 80, 81, 82, 5]. Recently, egocentric pixel-level hand detection has attracted more and more attention.

Most of the proposed methods are based on pre-training classifiers using abundant manually labeled data. Li et al. [10, 12] propose a pixel-level hand detection method using color and texture based features. Zhu et al. [11] propose a method which use local hand shape information in the training data and enforces shape constraints in the estimation. Serra et al. [4] integrate temporal and spatial consistency to complement the appearance features. Betancourt et al. [83] identify the left and right hands and models hand occlusions to improve the accuracy of hand segmentation. These methods improve the precision of pixel-level hand detection but still under the implicit assumption of hand presence in all frames. This assumption is not always true since that the hand may be absence before or after the egocentric human-computer interaction.

Some of the proposed methods conquer the hand segmentation task sequentially. Betancourt [16] proposes a sequential classifier consists of a hand-detector and a hand-segmentator. Betancourt et al. [17] extend SVM-based hand detector with a dynamic bayesian network. These methods reduce false positive rate of hand segmentation but also needs the offline training which requires manual labeled data. Kumar et al. [18] illustrate an on-the-fly hand detection training method which is initialized by a calibration gesture performed by the user. This simple preprocessing step saves a great deal of manual labeling but may not be friendly to the user. Zhu et al. [84] propose a two-stage detector which firstly generates bounding box proposals and secondly evaluates the proposals by a convolutional neural network. Moreover, all of these methods are still challenged with varying environment conditions since they don't have any model updating strategy.

In this chapter, we are going to illustrate our fine-grained hand segmentation method which leverages unsupervised online learning pattern to robustly segment the hand in pixel-

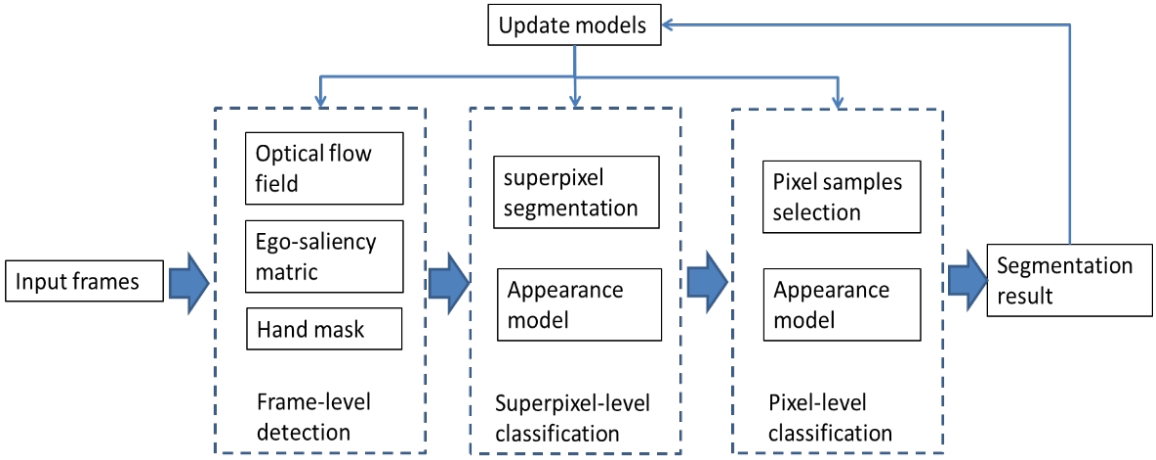


Figure 3.2: Framework of proposed method.

level from egocentric video.

3.3 Method

In this section, we discuss an unsupervised online learning method for fine-grained hand segmentation based on top-down classification and bottom-up optimization. By learning hand appearance and motion features on-the-fly, we segment out the hand with precise boundary from the egocentric video which is captured in varying illumination condition. From the point view of top-down strategy, we divide the classification task into three parts: frame-level detection, superpixel-level and pixel-level classifications. Before scanning pixel by pixel, we firstly estimate whether a frame contains a hand and whether a region of the frame contains hand pixels. By doing this, we reduce the false positive and initialize samples for further online training. After that, we learn feature from the labeled region and train two-level classifiers. To make sure the classifiers adapt to varying hand appearance, we update the hand appearance model and optimize the upper layer classifier and detector. Figure 3.2 shows the framework of our method.

3.3.1 Ego-saliency Based Hand Detection

Before scanning the frame pixel-by-pixel, the first task is detecting presence of hand from a frame-level perspective and then automatically initialize hand masks for subsequent classifications. Motion-based methods [85, 86, 87] are proposed for background subtraction for freely moving camera. In general, it is difficult to determine whether the hand is presence or not without prior information about the environment or appearance of the hand. Fortunately, the egocentric interaction scenario provides many constraints that are suggestive of the hand's presence.

From the point view of an interaction cycle, the motion of hand in egocentric view has periodical specialty. In the interaction preparatory phase, the whole hand and part of the arm together gradually enter into the view field. During the interaction, the whole hand moves around the center of view field and the fingers are likely to make more vigorous motion than the palm and arm, such as making a gesture. When the interaction is finished, the whole hand and part of the arm together gradually move out of the view field. We observe that the preparatory phase is a natural bootstrap since the hand motion is more salient than other regions and the hand is hardly to enter into the view field from the top side.

Based on this observation, we define an ego-saliency metric E_f consists of spatial and temporal terms to estimate how likely the hand is presence in the frame f. The higher the ego-saliency value, the more likely the hand is presence.

$$E_f^{IN} = \sum_{i=1}^{W}\sum_{j=1}^{H}\left(\frac{1}{1+e^{\lambda(h-j)}-0.5} * M_f(i,j)\right) + \sum_{t=f-n}^{f} sgn(N_t - N_{t-1}) \tag{3.1}$$

Where, the first term is the spatial cue that restricts the hand motion should be salient and happened in the right position. The second term is the temporal cue that restricts the hand motion should be consequently increased. W and H denote width and height

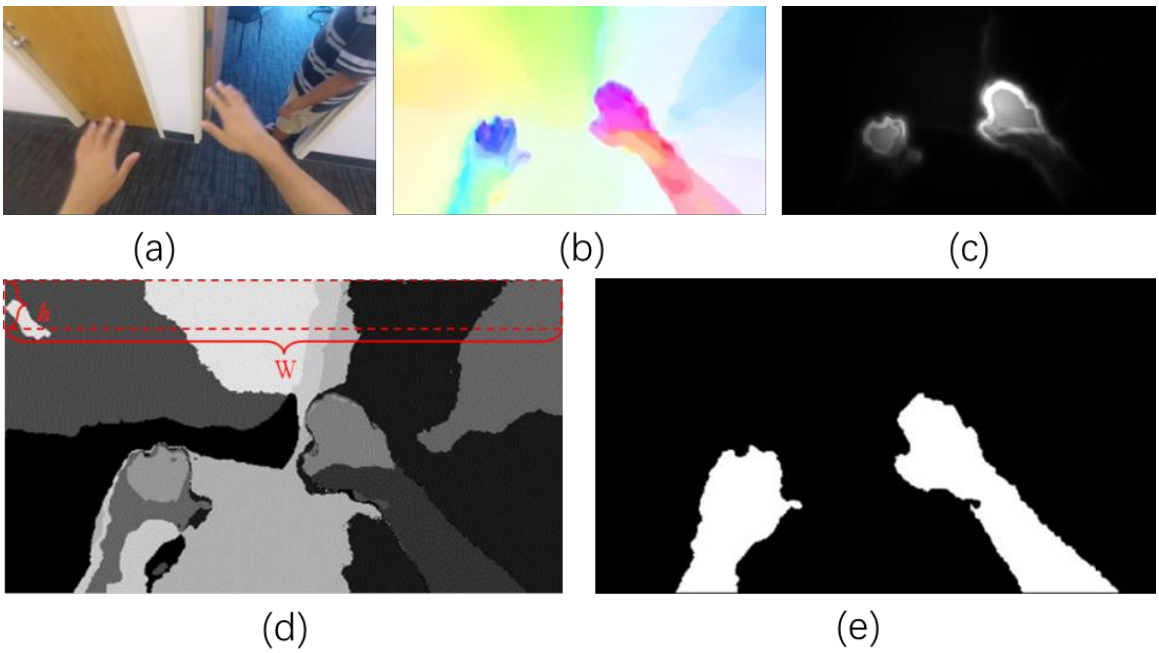


Figure 3.3: Initialize hand label based on ego-saliency.
(a)-(e) are original image, optical flow to the next frame, motion saliency of (b), clustering result of (b) with non-interactive border labeled in red, and estimate hand regions.

of the frame respectively. $M_f(i,j)$ is the motion saliency of a pixel at position (i, j) and calculated based on optical flow map using method [26]. As shown in Figure 3.3 d, we set a non-interactive border with width W and height h from the top of the frame. We set h as one tenth of the frame height in experiments. And we use a distance-based exponential weight to restrict that hand motion should happen away from the non-interactive border. λ is the weight response control factor. The farther a pixel is away from the non-interactive border, the greater its weight is assigned. N_t is the number of non-zero values in the motion saliency map M_t. The consequent motion increment is observed by a sign function $sgn(\cdot)$ based on the number of pixels having salient motion in adjacent n frames.

After detecting the presence of hands, we initially segment moving hand regions based on motion and appearance clustering. By using dual TV-L1 optical flow [88], we extract dense motion flow fields and get a motion map. We cluster the motion map into k groups of regions using K-means and we set k as 10 in the experiments. The motion clustering naturally divides foreground and skin-colored background into different regions since they

usually move differently. Figure 3.3 (d) shows the regions got from motion clustering and the non-interactive border. With the help of non-interactive border, we easily select out a set of background regions R_{BG} which intersect with the border. The rest unknown regions are further determined based on appearance clustering. According to equation 3.2, we calculate the likelihood $H_f(R_i)$ of an unknown region R_i belonging to hand region based on the similarity between the unknown region R_i and background regions R_{BG}. $S(\cdot,\cdot)$ is a function calculating color histograms similarity of two regions. Then, we find out the hand regions which have low color similarity with all the background regions. Figure 3.3 (d) shows the initialized hand mask which is generated by using motion and color clustering.

$$H_f(R_i) = \sum_{j \in BG} S(R_i, R_j) \tag{3.2}$$

3.3.2 Online Training Two-level Hand Classifiers

With the ending of the interaction preparatory phase, hands motion is attenuating and may eventually become much less salient, such as only the fingers move to make a gesture while the palm holds still. Moreover, motion-based segmentation usually produces the result with blurry and noise boundaries around objects. Therefore, the appearance feature is more discriminative than motion cue for fine-grained hand segmentation during the interaction phase.

Here we address a coarse-to-fine strategy based hand segmentation method that learns appearance feature of hand and background on-the-fly. Based on the initial set of hand masks B_t got from the frame-level detection, we firstly train a superpixel-level hand classifier so as to segment frames into superpixel regions from which the stable pixel samples are selected out. Then, we utilize the selected pixel samples to train a pixel-level classifier which produces fine-grained hand segmentations.

In the frame-level detection step, we obtain a coarse segmentation of the hands using motion and ego-saliency cues. It initially provides the ground truth labels of hand regions

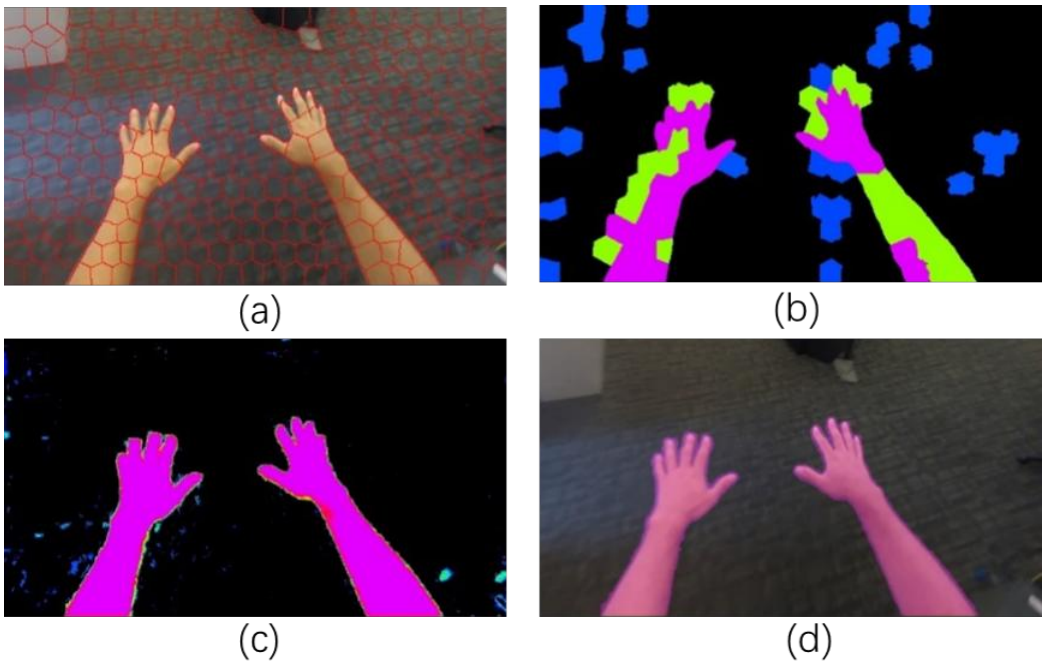


Figure 3.4: Example of two-level classification.
(a)-(d) are Superpixel segmentation overlaid on original image, probability map of superpixel-level classification, probability map of pixel-level classification, final segmentation overlaid on original image.

for superpixel-level training. We overly segment the recent n consecutive frames F_t into superpixels by using a modification of a state-of-the-art algorithm termed simple linear iterative clustering (SLIC) [57].

The K-means clustering of motion map derives a binary segmentation separating the foreground from the background. However, the K-means segmentation has coarse boundaries which are sometimes inconsistent with the superpixels'. To select good samples for superpixel-level training, we initialize a label map based on the portion of positive pixels in each superpixel and refine it by energy optimization. Figure 3.5 illustrates the process of superpixel sample selection. Given a binary mask of the K-means segmentation, we assign the superpixels having 80 percent positive pixels as foreground candidates and their dilated superpixels as background candidates. The candidates are further selected based on confidence score calculation and energy optimization.

We define a confidence score to describe how much the superpixel is more similar to its homogeneous neighbors than the heterogeneous neighbors. For a candidate superpixel, we

calculate its confidence score as equation 3.3. After normalization, we get a score map as shown in Figure 3.5 (d).

$$Score_i = \frac{1}{Z} \sum_{i \in \Omega_i^-} \sum_{k \in \Omega_i^+} \frac{D(h^i, h^j)}{D(h^k, h^j)} \tag{3.3}$$

$$D(h^i, h^j) = \frac{1}{\sum_k c_k} c_1 D(h^i_{SIFT}, h^j_{SIFT}) + c_2 D(h^i_{RGB}, h^j_{RGB}) \tag{3.4}$$

where Ω_i^- and Ω_i^+ are sets containing samples collected from the neighborhood of superpixel i, the superscript "-" indicates that the samples have different class label with superpixel i while "+" stands for the contrary situation, and Z is a normalization factor. And, h_{SIFT} and h_{RGB} denote the SIFT and RGB histograms respectively, $D(h^i, h^j)$ is the Chi-square distance between the histograms h^i and h^j, c_k is a constant to normalize the *k-th* descriptor.

We take the score as a label and optimize it for each superpixel by using Ising model [89][90]. The foreground and background candidates constitute a foreground system and a background system respectively. The energy of each system consists of the affinities and consistencies of superpixels to their neighborhood within the system. Color and texture are useful cues since foreground tends to have a difference appearance than the background behind it. Therefore, the affinity between a superpixel and its neighbor is computed as the Chi-square distance between their color and texture histograms. Higher affinity indicates stronger consistency for belonging to the same class. Therefore, we optimize the label based on an energy which encourages coherence in superpixels of similar appearance. For a superpixel, we inverse its label and calculate the energy change caused by the inversion. This label inversion is directly accepted if the system energy is increased. On the contrary, the process is further judged by an acceptance function. This routine is repeatedly executed until the system reaches equilibrium. Then, the superpixel labels are optimized.

Given a labelled region, we calculate the energy of each superpixel within it and ac-

cumulate them together to describe its system energy. For a superpixel, we first compute an affinity score and a label consistency score for each pair of adjacent superpixels. After normalizing the scores, we calculate their correspondence which is proportional to the superpixel energy. Based on the exponential correspondence, we obtain the superpixel energy. After that, we compute the system energy as

$$E = \sum_{i} \sum_{j \in \Omega_i^o} e^{-|S(i,j)-L(i,j)|} \tag{3.5}$$

where Ω_i^o is the neighborhood of superpixel i within the system, $S(i,j)$ is the affinity and $L(i,j)$ is the label consistency between two adjacent superpixels.

To describe the appearance of a superpixel, we compute the histograms of SIFT features and RGB values from the image area of it occupies. Considering the appearance feature is prone to be coherence in a local region, we use the distance between two adjacent superpixels to restrict the contribution of the neighboring superpixel. Larger distance indicates smaller contribution. Moreover, the superpixels nearing to the system boundary are tend to be unstable. Hence, the distance from superpixel to boundary is also a term of the affinity score. Based on these four descriptors, the affinity score $S(i,j)$ is defined for superpixel pair (i,j) as equation 3.6.

$$S(i,j) = 1 - \frac{1}{\sum_k c_k}(c_1 D(h^i_{SIFT}), h^j_{SIFT}) + (c_2 D(h^i_{RGB}), h^j_{RGB}) + c_3 A(i,j) + c_4 B(j) \tag{3.6}$$

where $A(i,j)$ is the Euclidean distance between the adjacent superpixel centers, and $B(j)$ is the Euclidean distance from superpixel j to the system boundary.

We inverse the label of superpixel i and get its updated label consistency score $L^i(i,j)$ with the adjacent superpixel j. The energy of the system is renovated correspondingly. Then, we compute the increment ΔE of the system energy. Based on the increment, we

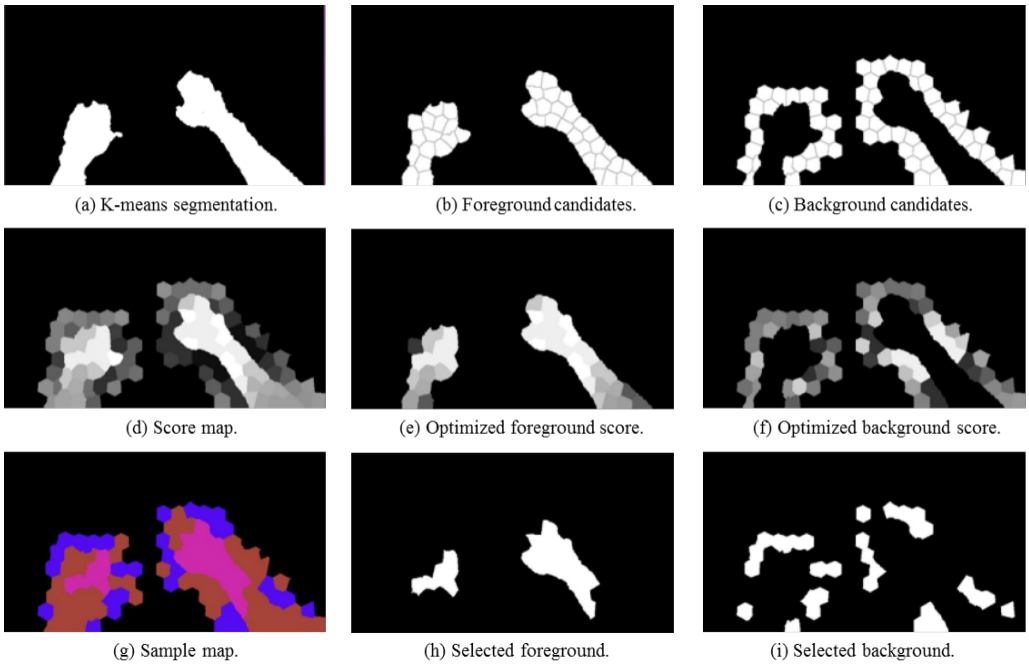


Figure 3.5: Select superpixel samples based on energy optimization.
(a)-(i) show the process of selection. Blue, pink and brick in (g) illustrate the selected background, foreground and abandoned samples respectively.

decide if it should accept the label inversion.

$$label(i) = \begin{cases} label(i), & \Delta E \leq 0 \&\& exp(-\beta \Delta E), \\ lable^{-1}(i), & otherwise. \end{cases} \tag{3.7}$$

where $label^{-1}(i)$ is the inversion value of $label(i)$, $\beta$is a weight factor and R is a pseudo random number from uniform distribution. In a word, the label inversion will be accepted if it increases the correspondence between the superpixel's appearance similarity and classification type.

Given the notations of all superpixels, we initially train the superpixel-level classifier based on appearance features which consist of color and gradient statistic in each superpixel. The classifier is able to select out the superpixels belonging to hands with confidence values. Note that the motion cue may eventually become much less discriminative. Therefore, we apply the SLIC [57] on color frame to get the superpixel segmentation in the subsequent online training. Because of benefiting from the relatively accurate boundaries

produced by the SLIC [57], the segmentation in superpixel-level is improved than the result of frame-level detection. However, the superpixel having low confidence value may partially contain hand region. It will cause misclassification if we take that kinds of superpixels as background and select negative pixel samples from them. Therefore, we proposed a sample selection strategy for pixel-level classifier training.

For pixel-level training, we select samples from the superpixels based on their classification confidence values. The negative samples are selected from the superpixel having confidence smaller than a threshold value T_U. The positive samples are selected from the candidate superpixels which have confidences greater than a threshold value T_L. The unstable superpixels having confidences between T_U and T_L are abandoned as unknown. Moreover, the higher the confidence of a superpixel belongs to the hand region, the more positive samples are extracted from it. Based on the property of superpixel generated by SLIC [57] method, we suppose that pixels nearing to the center of the superpixel are more likely to be in the same class with the superpixel. Therefore, we divide the pixels of candidate superpixels into training and unknown groups based on the distance between the pixel and the superpixel's center. By combining the area A_{sp} and confidence W_{sp} of a superpixel, we define the distance threshold T_{sp} as equation 3.8. Then, the candidate superpixels are eroded based on the threshold T_{sp}. The pixels in the shrunk region are put into unknown group while the others are selected as positive training samples.

$$T_{sp} = A_{sp} * W_{sp} \tag{3.8}$$

Following the previous pixel-level segmentation approach [10], we extract color features from RGB, HSV and LAB color spaces and texture feature using HOG [91]. By using a pool of combination of features and random forest classifiers [92], we classify the unknown pixels and obtain fine-grained hand segmentations. After that, we also get a more precision description of the confidence of a superpixel belonging to the hand region. The confidence

values of superpixels are updated with their portion of positive labeled pixels. Then, we re-train the superpixel-level classifier by using the superpixel having high confidence values. By doing this, we update the hand and background models on-the-fly which makes the method more robust to varying environment.

Note that the two-level classifiers select out the pixels that are most likely to be in the hands. The motion cue becomes salient and discriminative again when the interactive hand gradually moves out of the view field. Therefore, we still have to monitor the hand absence by aid of the egocentric saliency metric which is added a confidence term, as described in equation 3.9.

$$E_f^{OUT} = \sum_{i=1}^{W}\sum_{j=1}^{H}(\frac{1}{1+e^{\lambda}(h-j)} - 0.5) * M_f(i,j) + \sum_{t=f-n}^{f} sgn(N_{t-1} - N_t) - \frac{1}{m}\sum_{k=1}^{m} W_{sp_k} \tag{3.9}$$

Where, the first term denotes the motion saliency, the second term observes the consequent motion decrement, the third term is the average superpixel confidence of the frame f and m is the number of superpixels having confidence greater than 0.5.

3.3.3 Evaluation to Update Classifier

In evaluation stage, we use a bottom-up strategy. We evaluate bottom classifiers and feed-back loss to the upper levels. The superpixel-level classifier is directly affected by precision of pixel-level classification since the confidence of superpixel is calculated based on pixel classification results. In the initialization step, we consider frames of a sequence equally to contribute to pixel-level classifier. Since background changes constantly, the appearance of hand varies a lot and becomes different from previous situation, such as hand enters into a shadow place. Therefore, the history frames contribute differently and we calculate weights W_t for n history frames of the training set F_t to make their contributions more rational based on error of pixel-level classifier. The weight W_t consists of a local metric

W_L^t and a global metric W_G^t.

$$W_t = \frac{1}{(W_L^t + W_G^t)} \tag{3.10}$$

Given a labelled training set F_t, we train a collection of classifiers C_t. By using the classifier C_t, we get the confidence value $W_{sp_k}^t$ of a superpixel SPk belonging to the hand regions in current test frame f. The local metric SP_k restricts that the result of classifier C_t has low variance with other classifiers of the set. Therefore, we calculate the loss of using training data from frame t based on the difference between classification results of test frame f produced by C_t and the average classifier $\overline{C_{\{F_t\}}}$.

$$W_L^t = \frac{1}{m}\sum_{k=1}^{m}(W_{sp_k}^t - \overline{C_{\{F_t\}}}) \tag{3.11}$$

$$\overline{C_{\{F_t\}}} = \frac{1}{n}\sum_{l \in F_t} W_{sp_k}^l \tag{3.12}$$

Where, m is the number of superpixels in current test frame f, n is the number of frames in the training set F_t. From a global point of view, we estimate the loss of using training data from frame t based on the difference between the classification result of frame f produced by C_t and the classification result of frame $f-1$ produced by the previous classifier $C_{F_p}^{f-1}$ which is trained using data from F_p under the constraint of weight W_p. Generally speaking, precise classification can segment hand region from background with clear boundary while smooth and flat inside the region. We calculate gradient map of the classification probability map and define three gradient-based constraints to evaluate the global loss. Firstly, the magnitude of the biggest contour in the gradient map should be large. Then, the gradient in the conjunction of two superpixels should be small. That is, the number of contours in the gradient map should be small. And last, the shapes of the biggest contours in current and previous gradient maps should be similar. Based on these three constraints, we calculate a global loss function having terms based on the average magnitude G_f of the

biggest contour, the number N_f of contours, and the shape S_f of the biggest contour in the classification result of test frame f.

$$W_G^t = (G_f^t - G_{f-1}^{f-1}) + (N_f^t - N_{f-1}^{f-1}) + D(S_f^t, S_{f-1}^{f-1}) \tag{3.13}$$

Where, the right hand superscript denotes the classifier has been used, C_t or $C_{F_p}^{f-1}$. $D(\cdot, \cdot)$ is a function estimating the difference between two shapes. By combining W_L^t and W_G^t, we evaluate the effectiveness of training samples from frame t not only in local superpixels but also in the global hand region. Based on the weight W_t, we optimize the pixel-level classification result which is used to update the superpixel-level classifiers. Note that the terms of the weight function will be normalized before combination.

3.4 Results and Discussion

We evaluate our cascaded hand segmentation method on two types of egocentric data which correspond to different levels of human-computer interaction. The first type contains the both hands are exposed with little varying gesture and interacting with objects, such as holding a cup. The second type contains the hands performing gestures, such as virtual keyboard typing, without directly interacting with any object. We firstly compare our cascaded hand segmentation with the state-of-the-art methods and analyze the validity of our framework. Then, we illustrate that the egocentric human-computer interaction can benefit from our hand segmentation approach.

3.4.1 Evaluation on Benchmark Dataset

To compare with baseline methods, we first test our approach on the benchmark dataset CMU EDSH [10] which consists of egocentric videos containing diverse indoor and outdoor illumination and hand poses. The videos were collected by a subject wearing the head-mounted standard color camera and passing through scenes with varying illumination

Table 3.1: Comparison with state-of-the-art on F-score

Data	Li's[10]	Zhu's[11]	Baraldi's[13]	Ours
EDSH1	Training	Training	Training	0.8667
EDSH2	0.835	0.835	0.852	0.899
EDSHK	0.840	0.935	0.901	0.913

Table 3.2: Comparison with state-of-the-art on time

-	Li's[10]	Zhu's[11]	Baraldi's[13]	Ours
Time(ms)	2138	3277	1092	3497

including the extreme cases of underexposed and overexposed at a resolution of 720p and a speed of 30 FPS. Besides the change of skin color, the hand pose also changes during the subject doing daily activities. The dataset contains 19,788 frames and 743 ground truth labels from 3 video clips, including EDSH1, EDSH2 and EDSHK. EDSH1 and EDSH2 involve data of bare hands with a few intentional gestures while EDSHK records hand interacting with objects in a kitchen. In order to match the scale of the ground truth, we downsample the resolution of the frame from 1280x720 to 640x480 pixels. We conduct quantitative and qualitative evaluation on the benchmark dataset to compare our detection performance with the prior arts.

In Table 3.1, we compare our method with the three state-of-the-arts on F-score. Li et al. [10] predict hand pixel using color and gradient features based on Random Forest classifiers. Zhu et al. [11] extend the pixel-level method by introducing shape information of pixels based on structured forests. Baraldi et al. [13] utilize temporal and spatial coherence strategy to improve the hand segmentation of the pixel-level method. The state-of-arts use video clip EDSH1 as the training data and test their approaches on the rest clips of EDSH2 and EDSHK. The corresponding F-scores are provided by their papers. Since our approach using online training strategy, we give out our F-scores on all the clips. As the F-scores shown in Table 3.1, our approach improves the detection precision in most experiments. We have implemented our algorithm and tested the non-optimized code on an Intel based

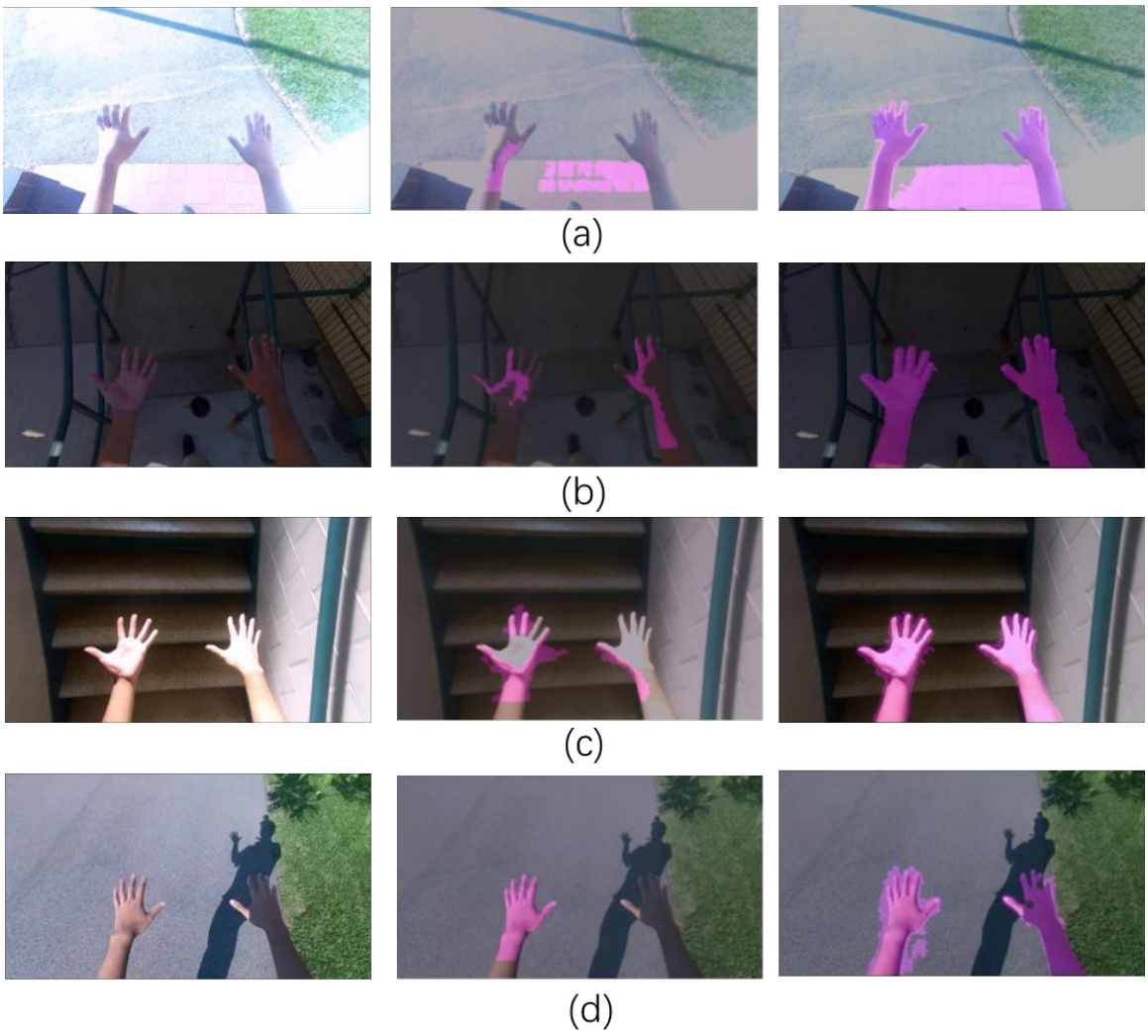


Figure 3.6: Challenge illumination cases comparison.
From top to bottom are cases of overexposed, underexposed, overexposed, and high contrast shadows. The first column shows the original images. The middle column shows results of Li et al. [10] and the last column shows our results.

PC, with a i7-4500U CPU that runs at 1.80 GHz. Most of time is spent on superpixel sample selection and online training. The time cost can be reduced by decreasing the number of samples used in all stages. In Table 3.2, we compare our method with of the three state-of-the-arts on time.

Figure 3.6 and Figure 3.7 show the visually comparison of test images overlapped by detection results provided by their papers and our method in the challenge cases of extreme lighting conditions and background color. In Figure 3.6, the test frames of EDSH2 were

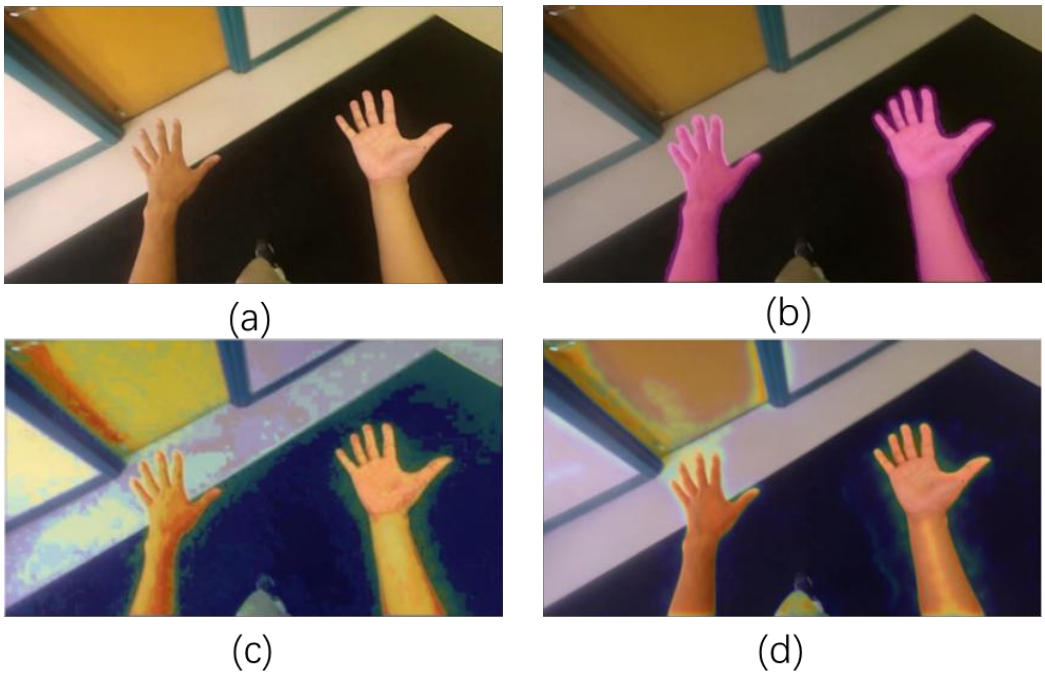
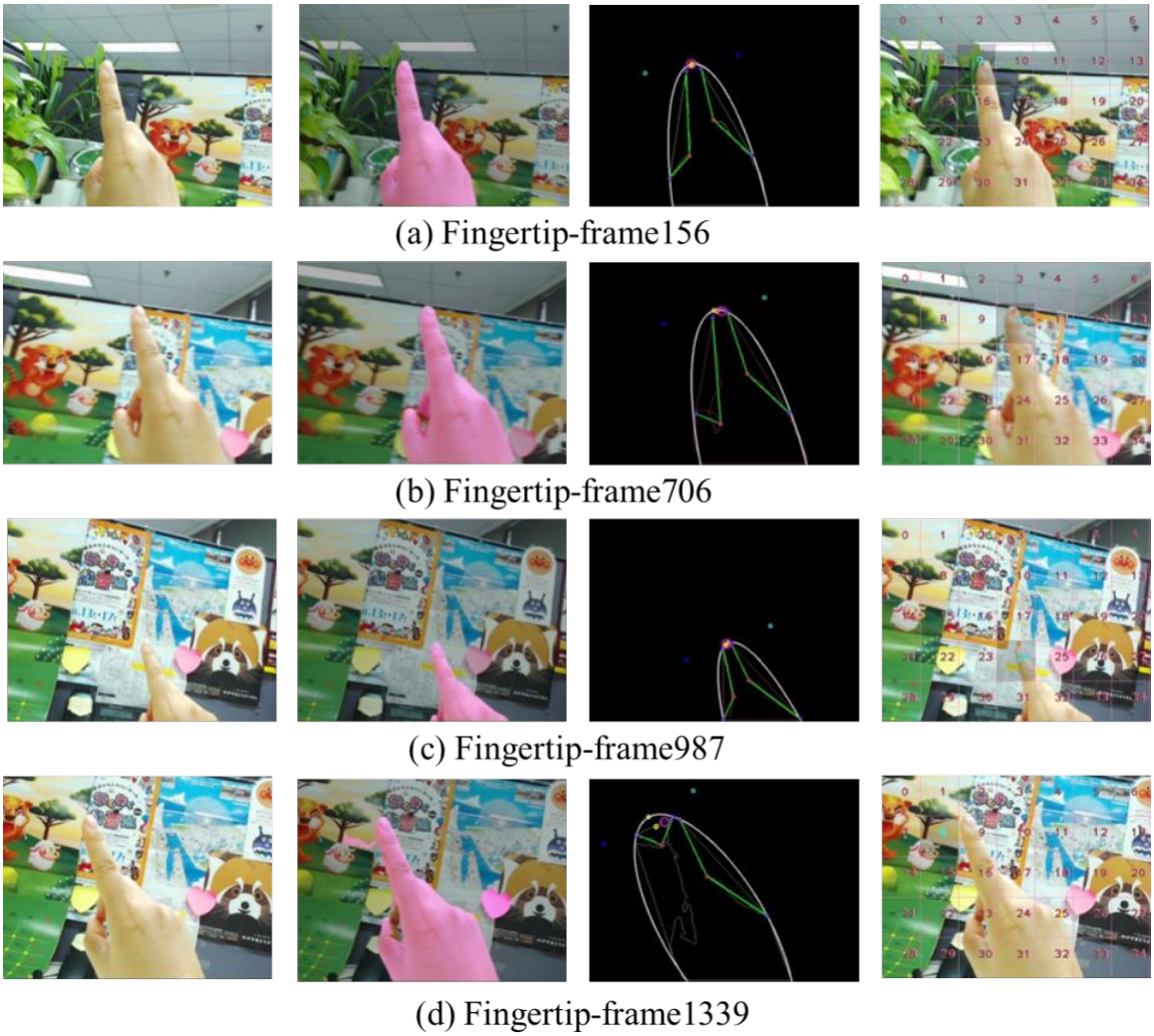


Figure 3.7: The original image has similar color in background and hand regions. From (b) to (c) are corresponding results of ours, Li et al. [10] and Zhu et al. [11].

taken under extreme lighting conditions of overexposed, underexposed and high contrast shadows. Parts of the hand are blended into background by the strong or insufficient light while the color and texture of the other parts are faded inordinately. Li et al. [10] fail to give good prediction in these cases. In contrast, our approach has much higher detection precision. Our continuously online training strategy makes the classifiers robust to varying illumination even in the extreme conditions. Figure 3.7 shows the case of background sharing similar color and texture with hand. Both methods of Li et al. [10] and Zhu et al. [11] fail to distinguish the hand from the textureless and skin-colored background. In contrast, our approach gives more correctly prediction in this case. By using online learning, our method gradually updates the hand and background models so that the classifiers are more robust to varying scene.



(a) Fingertip-frame156

(b) Fingertip-frame706

(c) Fingertip-frame987

(d) Fingertip-frame1339

Figure 3.8: Virtual keyboard interaction.

(a)-(d) illustrate examples of fingertip detection for virtual keyboard interaction. Columns from left to right are original images, hand segmentation results overlapping on original images, fingertip detection by analyzing convex of hand segmentation result, and virtual keyboard interaction.

3.4.2 Evaluation on Egocentric Application

Fingertip position is one the most practical information for egocentric vision based human-computer interaction, such as the user inputs command via a virtual keyboard. Fingertip detection can directly benefit or suffer from the precision of the hand segmentation. Therefore, we use a simple fingertip detection method to further evaluate our hand segmentation method from the practical point of view.

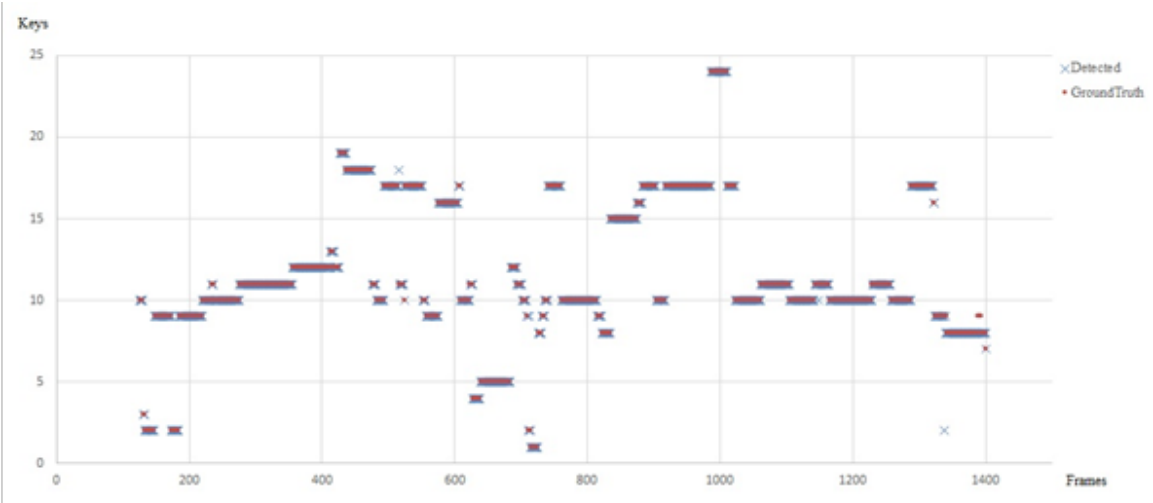
As shown in Figure 3.8, we evaluate the applicability of hand segmentation by an ap-

plication of virtual keyboard interaction. The ready gesture of index finger up triggers the virtual keyboard to show up. Then the egocentric view field is divided into girds each of which corresponds to a key. In the experiment, we divide the view field into 5x7 grids which provide relative comfortable interaction scale for the user. The duration of fingertip activates the key input and the corresponding position will light up. We extract tip position of the index finger from the hand segmentation result by convex hull analysis. The video was recorded by a subject wearing the head-mounted Logitech camera in the indoor scene at a resolution of 640x480 and a speed of 30 FPS. The test video totally contains 1439 frames consist of the whole interaction procedure including hand moving into the view field, ready gesture showing up, fingertip hovering and moving through keys, and hand moving out of the view field. Figure 3.8(a)-(c) illustrate the robustness of our hand segmentation-based fingertip detection. Figure 3.8(d) shows the failure case caused by the noise of the segmentation which could be removed by extra post-process.

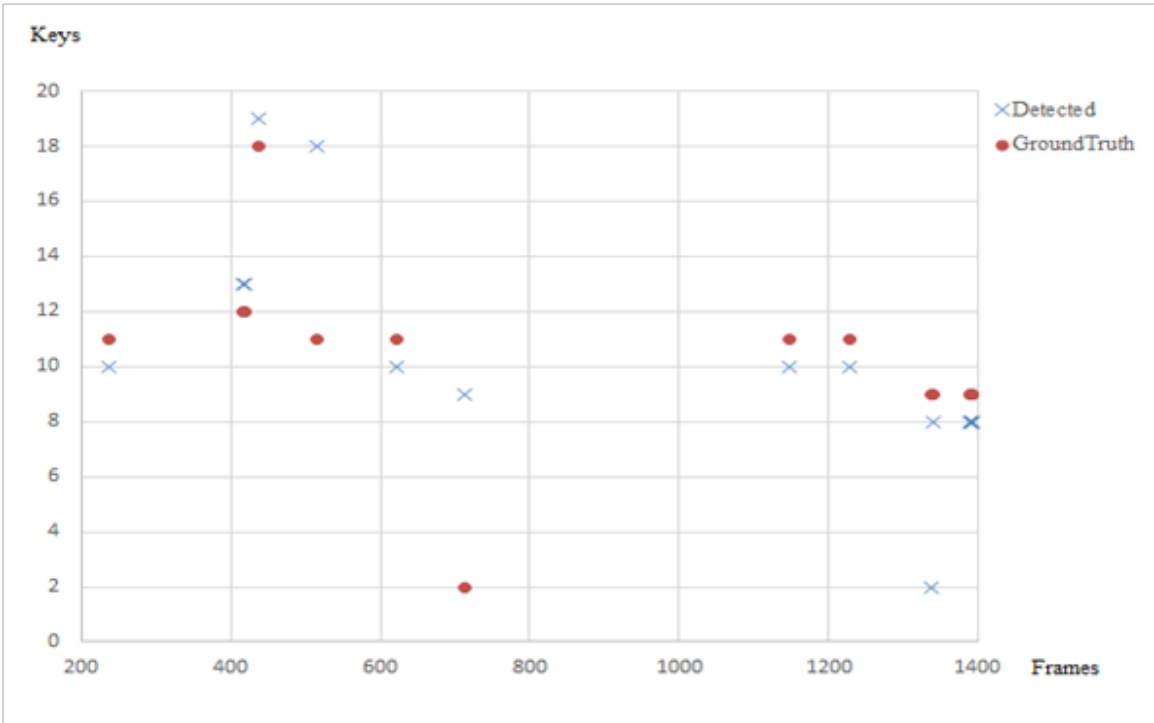
Figure 3.9 shows the performance of our hand segmentation method in the virtual keyboard interaction application. The red and blue dots are the detected fingertip interaction frames. We can see that the detected fingertip position and the ground truth respectively in the keyboard position is stable and with little jitter. And the fingertip detection accuracy rate is 0.9867 over the test video. Figure 3.9 (b) shows the total 17 failure cases over the 1277 interactive frames. It proves that our hand segmentation approach is reliable and prone to be used in egocentric vision based human-computer interaction.

3.5 Conclusion

In this chapter, we presented an unsupervised on-the-fly hand segmentation method which consists of top-down classification and bottom-up optimization. From the point of view of egocentric interaction loop, an unsupervised frame-level hand detector is proposed for the purpose of reducing the false positive caused by hand absence. We implement the frame-level detection by setting a non-interactive border based on an assumption that the hand is



(a) Fingertip detection results compared with ground truth.



(b) Close-up of errors in (a).

Figure 3.9: Performance of hand segmentation in virtual keyboard interaction.

hardly to enter into the view field from the top side for egocentric interaction. Based on the frame-level detection result, the superpixel-level and pixel-level classifiers are trained on-the-fly sequentially aimed at improving reliability of hand segmentation. To get stable samples for superpixel-level training, we select the candidates based on steps of confidence score calculation and energy optimization. In order to be robust to vary environmental conditions, the classifiers are updated from the bottom up based on the proposed performance evaluation method. Experiments carried on public datasets validate the generality of the proposed approach. This chapter shows the potential of unsupervised method for pixel-level hand segmentation in egocentric interaction. We believe that it can be transferred to the pixel-level object segmentation by combining with gaze analysis and contribute to activity recognition.

CHAPTER 4

CLUSTER-WISE LEARNING FOR MULTI-PERSON POSE ESTIMATION

4.1 Introduction

In this chapter, we propose a cluster-wise feature aggregation network that exploits multi-level contextual association for multi-person pose estimation. The recent popular approach for pose estimation is extracting the local maximum response from each detection heatmap that trained for a specific keypoint type. To exploit more contextual information, our network simultaneously learns complementary semantic information to encourage the detected keypoints subject to a certain contextual constraint. Specifically, our network uses dense and sparse branches to generate paired multi-peak detection heatmaps for clusters of keypoints. To enhance the feature passing through the network, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches. We demonstrate competitive performance of our network on the benchmark dataset for multi-person pose estimation.

To capture and consolidate information across all scales of the image, the pipeline of first encoding and then decoding is broadly used in both top-down and bottom-up approahces, such as the Stacked Hourglass Network [45] and Associative Embedding (PoseAE) [52]. The backbone of these networks are based on multiple Hourglass modules [45] which firstly pool down the feature maps to a very low scale, and then upsamples and combines features across multiple resolutions. This architecture provides the benefit of capturing both global and local features for the pixel-wise prediction tasks. One commonality of these approaches is that they separately regress keypoint locations from each other according to

their corresponding detection heatmaps. However, since body parts connect to each other, such as hip-knee-ankle, positions of some parts provide important contextual information and constraints on locating other related parts and the cues are hard to find if the keypoint representations are separated. This brings us an intuitive thinking of utilizing structural knowledge to enforce the model to exploit clues of related body parts to overcome ambiguous problem caused by occlusions and complex multi-person situations. Therefore, we propose to learn keypoints by clusters to exploit more contextual information. Our backbone structure for multi-scale feature extraction is same as the Stacked Hourglass Network's. However, the prediction heads are different. Our method utilizes multi-peak ground truth heatmaps to supervise the network exploiting contextual information between related body parts while the Stacked Hourglass Network uses a single-peak heatmap to present each keypoint separately.

Without prior assumption of person locations, the bottom-up methods have to estimate tags that guide the detected keypoints to form individual poses for person instances. For grouping the detected keypoints into instances, PoseAE [52] integrates associative embedding with the Stacked Hourglass Network [45], which produces a tagging heatmap for each detection heatmap. The idea of associative embedding is to predict an embedding in addition to the detection score for each keypoint. The embeddings serve as tags that group keypoints from same instances while separate keypoints from different instances. Following PoseAE [52], our network predicts tagging heatmaps in addition to detection heatmaps. Different from PoseAE [52], our tag embedding is applied cluster-wise. Since we divide keypoints into clusters, our tagging heatmaps contain multi-peak responses according to the keypoints in same cluster. Then the tags are normalized within each keypoint cluster. Specifically, we represent each tag by the cluser-wise mean value with an offset factor to distinguish the keypoints by their relative orders. We supervise the cluster-wise embedded tags with a pull loss that encourages similar tags for clusters from same group and a push loss that enforces different tags for clusters across different groups.

In this chapter, we propose a bottom-up method that learns keypoints by clusters with different configurations to exploit more contextual information. The keypoint cluster contains a group of keypoints to be learnt simultaneously according to the connection between body parts. Based on the number of related body parts, we define the dense and sparse cluster representations, such as divide the keypoints of head part into one cluster and the keypoints of limb part into another cluster. Comparing to the sparse cluster representation, the dense keypoint clusters reveal more global contextual information between more related parts. Therefore, our network is a multi-branch architecture that learns dense and sparse cluster of keypoints under the supervise of multi-peak heatmaps. This architecture imposes multi-level contextual relationships among multi-type abstractions of the keypoints.

To group the detected keypoints into instances, our network predicts tagging heatmaps in addition to the detection heatmaps. Therefore, each cluster branch consists of two sub-branches for keypoint detection and tag embedding. The pixels in the tagging heatmap serve to group keypoint clusters belong to the same instance. Besides that, the regions of person instances are simultaneously exposed in tagging heatmaps. Hence, our architecture can be taken as a multi-task learning, which can jointly handle the keypoint detection and coarse person segmentation problems. Despite the coarse boundaries of person regions generated by the tag embedding branch, it unveils holistic context information that benefits to keypoint detection. Based on this observation, we use an in-branch block to aggregate the tag features to the detection features in each branch to generate a region attention for keypoint detection. In addition, the detection features from dense and sparse branches complement each other. We design a cross-branch block to aggregate the sparse detection features to the dense detection features to improve the flow of information between different branches. Thus, our network optimizes the features at a specific keypoint by the received information from diverse context.

Figure 4.1 illustrates overview of the proposed cluster-wise learning network. The backbone of our network consists of multi-stack Hourglass modules. Each stack module

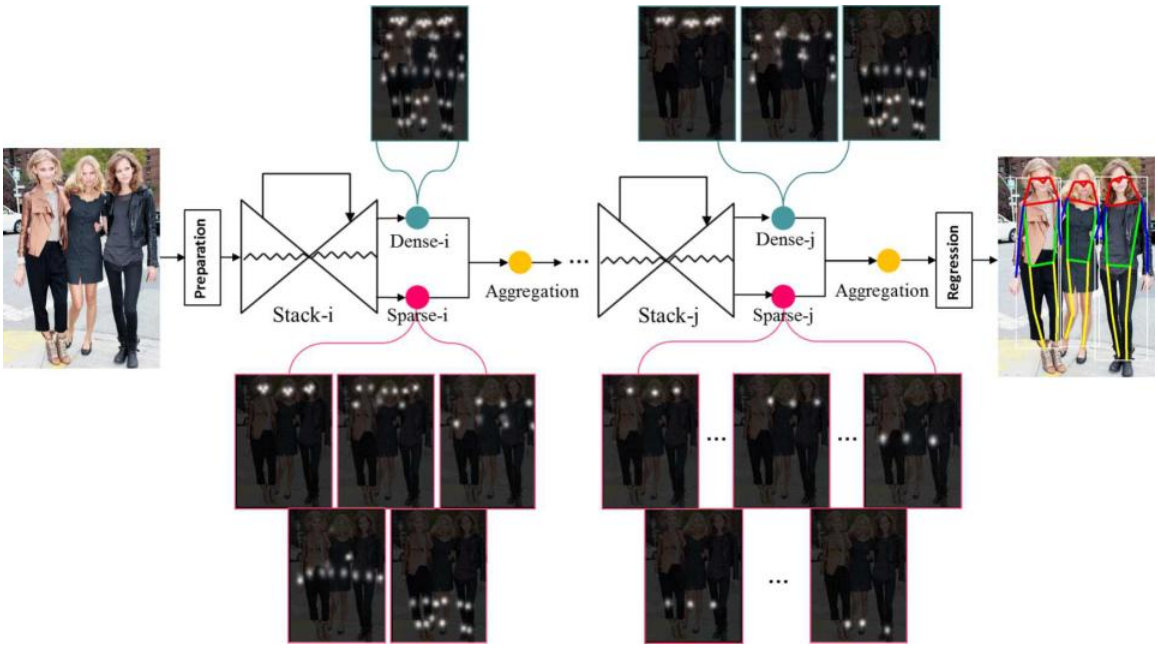


Figure 4.1: Overview of cluster-wise learning based keypoint detection.
Our network learns keypoints by cluster. The two branches of the network simultaneously learn different clusters containing dense and sparse keypoint divisions. To take advantage of multi-level contextual information, each stack is set with different cluster configuration in both dense and sparse flows.

consists of branches of dense and sparse keypoint clusters. Specifically, the dense keypoint cluster contains more related keypoints than the sparse keypoint cluster,such as entire body parts versus only one body part in each heatmap. By doing this, the network learns global and local contextual information for each keypoint. For each branch, the network predicts multi-peak detection heatmaps that indicate keypoint locations. To retrieve instance identification, our network exploits tag embedding to predict instance tagging heatmaps that naturally serves as a rough instance segmentation. Therefore, before passing features to next stack, we aggregate the tagging heatmaps and detection heatmaps in and cross branches to enforce the network focus on global and local consistency of entire body and different parts. In summary, this chapter makes three main contributions:

- Cluster-wise keypoint detection. Instead of detect each keypoint separately, our network predicts multi-peak detection heatmaps for clusters of dense and sparse keypoints, which exploits global and local contextual information to improve the detection robustness.

- Feature aggregation. To enhance feature passing from shallow stack to deep stack, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches.

- Cluster-wise tag embedding. To better grouping the detected keypoints into instances, our network embedds relationships among the intra-cluster and inter-cluster keypoints with offset learning, which not only benefits the instance grouping but also individual keypoint identification.

4.2 Related Work

Keypoint detection, human joint [42, 48, 49, 50, 51, 52, 53, 93] and rigid object keypoints [33, 94, 95, 96, 97] detection in particular, is an important issue extensively discussed and researched in computer vision in recent years. This problem is firstly converted to a supervised regression of the (x, y) coordinates of the keypoints. Note that human pose estimation requires type information of each keypoint but object keypoint detection may not. DeepPose [66] proposed by Toshev et al. is the pioneer work in using the deep neural network to estimate 2D human pose. Since DeepPose [66], the pose estimation solutions thrive in the presence of deep neural network and branch out into top-down and bottom-up approaches.

By integrating person detection with single person pose estimation, the top-down approaches generates keypoints for all persons in the given image. Since the pose estimator only have to focus on single instance, these two-stage methods usually produce fine localization. G-RMI [98] by Papandreou et al. predicts dense detection heatmaps and offsets for each keypoint type by using ResNet [64] and combined these outputs by an aggregation procedure to obtain highly localized keypoint predictions. In contrast to Gaussian detection heatmaps, the authors estimated a disk-shaped keypoint masks and 2-D offset vector fields

to accurately localize keypoints. G-RMI [98] also uses a novel form of keypoint-based NonMaximum-Suppression (NMS), instead of the cruder box-level NMS, and a novel form of keypoint-based confidence score estimation, instead of box-level scoring. Xia et al.[27] ultilize a PoseFCN and a PartFCN to provide initial estimation of pose joint potential and semantic part potential. They declare that the estimated pose provides object-level shape prior to regularize part segments while the part-level segments constrain the variation of pose locations. RMPE[43] by Fang et al. integrates the Symmetric Spatial Transformer Network (SSTN), Parametric Pose NonMaximum-Suppression (NMS), and Pose-Guided Proposals Generator (PGPG) with stacked Hourglass modules [45] to handle inaccurate bounding boxes. MaskRCNN [44] by He et al. solves multi-task of object detection, instance segmentation and keypoint prediction together based on the RoI aligned feature maps extracted from ResNet [64] integrated with FPN [99]. CMUpose [49] by Cao et al. uses a nonparametric representation named as Part Affinity Fields to learn to associate body parts with individuals in the image. The architecture is designed to jointly learn part locations and their association via two branches of the same sequential prediction process.

Bottom-up multi-person pose estimation approaches firstly detect body parts instead of full persons and subsequently associate these parts to human instances. Methods of this fashion are similar in part detectors that generate detection heatmaps and differ in how to associate parts with each other and group into person instances. Compare to top-down approaches, they are faster in test time and smaller in model size. However, their accuracy always lower than the top-down methods since they are lack of prior knowledge of the number of people, their locations and detailed high resolution features. Deepcut [93] by Pishchulin et al. proposes a partitioning and labeling formulation of a set of body-part hypotheses generated with CNN-based part detectors. Due to proposed clustering algorithm, Deepcut [93] is very slow and its processing time is in the order of hours even though it doesn't use person detections. Following Deepcut [93], Deepercut [50] by Insafutdinov et al. benefits from deeper ResNet[64] architectures as part detectors and improves the parsing

efficiency of Deepcut [93] with an incremental optimization strategy. Iqbal et al. [51] also formulate the problem as part grouping and labeling via a linear programming. Different from Deepcut [93] and Deepercut [50], Iqbal et al. [51] propose to solve the densely connected graphical model locally to improve time efficiency significantly. Following CPM [47], CMUPose by Cao et al. [49] uses a part affinity field learn to associate body parts and group them to person instances with greedy bottom-up parsing steps. PoseAE [52] proposes associative embedding to simultaneously generate and group detections. Chu et al. [42] use geometrical transform kernels and a bi-direction tree structured model to capture and pass relationships between joints. PersonLab [53] proposes a part-induced geometric embedding descriptor to associate semantic person pixels with their corresponding person instance. MultiPoseNet [100] receives keypoint and person detections, and produces accurate poses by assigning keypoints to person instances.

Among the most dominant approaches to pose estimation from videos is a two-stage approach, which first deploys a frame-level keypoint estimator, and then connects these keypoints in space and time using optimization techniques. For frame-level keypoint estimation, Detect-and-track by Girdhar et al. [101] experiment with Mask R-CNN [44], as well as their proposed 3D extension of this model, which leverages temporal information over small clips to generate more robust frame predictions. SimpleBaseline by Xiao et al. [102] builds a frame-level keypoint detector by simply adding a few deconvlutional layers over the last convolution stage in the residual network (ResNet) by He et al.[64]. The one-stage approach Poseflow by Xiu et al. [103] proposes a functionally structured spatial-temporal deep network, PoseFlow Net (PFN), to jointly solve the skeleton localization and matching problems of PoseFlow. Its spatial derivative reasoning branch is an encoder-decoder(convolution, pooling-deconvolution, upsampling) network architecture with input as the fth video frame.

Inspired by recent works, we propose a network simultaneously detecting and grouping keypoints with the help of multi-level contextual exploration. Overall, two branches of our

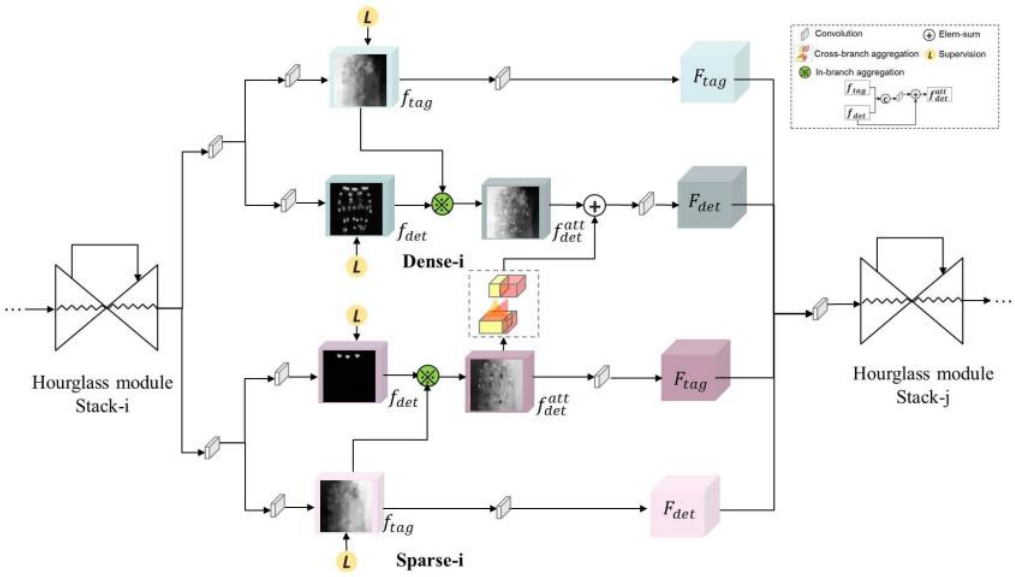


Figure 4.2: Illustration of our cluster-wise learning network.

The backbone of our network consists of multi-stack Hourglass modules. Each Hourglass module is followed by two branches for predicting keypoints based on dense and sparse clusters. For each branch, the network predicts detection heatmaps to indicate keypoint positions and tagging heatmaps to group the detected keypoints into instances. Additionally, the detection heatmaps and tagging heatmaps are fused to enhanced feature maps by in-branch and cross-branch aggregation. All feature maps are fused together by 1x1 convolutions and passed to next Hourglass module. The details of feature aggregation are shown in Figure 4.4.

network learns keypoints by clusters. The cluster structures are set in different keypoint densities. The denser the cluster, the higher the global information capacity. To group instance while retrieve individual keypoint identification, our network learns offset embedding. To enhance feature passing from stack to stack, the network aggregates features in and cross branches. The in-branch aggregation enhances keypoint detection features by fusing them with holistic instance region attention maps in each branch. The cross-branch aggregation further strengthens the keypoint detection feature by fusing global and local context information between dense and sparse branches.

4.3 Method

4.3.1 Network Overview

In this section, we introduce a multi-person pose estimation network consists of feature extraction backbone, keypoint detection and tag embedding. As illustrate in Figure 4.2, the network stacks multiple Hourglass modules [45] to extracts multi-scale features by repeating top-down and bottom-up information passing. The Hourglass module [45] has the symmetric topology that consists of a series of convolution, pooling and upsampling layers. This architecture provides the benefit of capturing both global and local features for the pixel-wise prediction tasks. We stack multiple Hourglass modules under supervision of multi-dense keypoint clusters to make the network gradually strengthen the capacity of accurate prediction. Figure 4.3 illustrates predictions of each stack module. To consider all keypoints simultaneously and learn configurations of keypoints, each stack outputs two branches of keypoint clusters. Different cluster conveys different semantic information and are complementary to each other. This enforces the module to have different localization abilities. To enhance features passing from stack to stack, the network aggregate information from different branches. As shown in Figure 4.5 (b), the tag value obviously changes across instances that makes the tapmaps to be a rough instance segmentation. Therefore, the combination of detection heatmaps and tagging heatmaps serves as a bootstrap for passing semantic enhanced features from stack to stack. The module embeds multiple relation of the keypoints and provides instance information for grouping the keypoints and individual identification. The design of these modules is motivated by the need to capture contextual correlation of keypoints and use that to produce accurate detection. Under supervisions of variant ground truth, these consecutive modules gradually transform the features into detection and tag heatmaps that respectively indicate keypoint locations and groups. More specifically, the network contains multiple stacks which consists of sparse and dense branches. The predictions of all branches in all stacks are used for training. The

sparse branch of the last stack is used for inference.

To detect keypoints of an object, a popular way is detecting each of them as an individual class. This kind of independent detection can produce accurate localizations for the keypoints having distinctive appearances. However, they often fail to output desirable results for the keypoints with ambiguous appearances. Since spatial distributions and semantic meaning of adjacent keypoints are highly correlated, context information should be learnt together with the appearance features. Thus, we take a cluster of keypoints as an entity of the network rather than each individual keypoint. Our network consists of branches of dense and sparse keypoint clusters. The denser the cluster, the higher the global information capacity. More specifically, the dense keypoint cluster contains keypoints of entire body while the sparse keypoint clusters are consists of part of them. By doing this, the network learns global and local contextual information for each keypoint. For each branch, the network predicts multi-peak detection heatmaps that indicate keypoint locations.

By supervising multiple keypoint detectors built on hierarchical features with multi-level semantic supervisors, our network fully explores the diversities of keypoint contextual structures. Then, the detection heatmaps produced by each branch will be unified to produce the final keypoint locations. So far, we only know where are the keypoints but have no idea about how do they relate to each other and which instances do they belong to. In addition to producing the detection heatmaps, the network correspondingly outputs tagging heatmaps for keypoint-wise relation embedding and instance retrieving. In the tagging heatmaps, pixels having similar values indicate the locations from the same instance. Finally, we use a loss function that enforces the keypoints detection close to the ground truth and encourages pairs of tags to have similar values if the corresponding detections belong to the same instance or different values otherwise.

4.3.2 Cluster-wise Keypoint Detection

In most recent keypoint detection network, ordered multi-channel detection heatmaps are widely used for representing keypoints. The number of channels is fixed by the object category, e.g. 17-channel detection heatmaps for human data from MSCOCO [74] dataset. Each channel detection heatmap associates one category specific keypoint and is learnt independently with the others. The relation and restriction among keypoints are not being fully explored and exploited. In addition, although these representations are compatible with invisible keypoints, it is difficult to learn categories with varying number of keypoints together. One brute force approach is to stack all detection heatmaps from all categories and certainly results in high computational costs.

In this chapter, we propose a novel keypoint representation which encodes multi-level relation of keypoints while provides flexibility to represent varying numbers of keypoints across different categories. Our representation consists of two level entities. One is the dense keypoint cluster that contains global information by using more keypoints in a cluster, such as all keypoints of the entire body. The other is the sparse keypoint cluster that are consists of partial keypoints to be learnt simultaneously. To extract multiple levels of relations, we stack multiple single stage modules. The dense and sparse cluster configurations can be varied from stack to stack and the detection heatmaps from dense and sparse flow are fused and passed to next stack. As shown in Figure 4.3, we predict C_d^i and C_s^i detection heatmaps for the dense and sparse keypoint cluster, where C^i is the number of keypoint clusters in *i-th* stack, such as $C_d^0 = 1, C_s^0 = 5$ in the *0-th* stack. Local peak values in each detection heatmap indicate the most probably locations for the entities from multiple instances of the same specific cluster. The StarMap [33] proposes a category-agnostic keypoint representation, which combines all keypoints of one object together and forms a multi-peak detection heatmap (called StarMap). This is in contrast to our method which uses multi-peak detection heatmaps to detect multiple instances of the same specific cluster of keypoints. We divide the keypoints into clusters based on the structure of body joints

connection, such as divide the keypoints of head part into one cluster and the keypoints of limb part into another cluster.

Given a set of ground truth keypoint locations, we divide them into C_d^i and C_s^i groups and generate a set of ground truth detection heatmaps for training the dense and sparse keypoint clusters. More specifically, the input labels for supervision are the keypoint coordinates which are converted to ground truth heatmaps by using Gaussian kernel. Thus, each pixel in the ground truth heatmaps represents the probability of location being the keypoint. The prediction penalty is increased along the radius with the amount given by a 2D Gaussian kernel. To detect the keypoint locations, we enforce the prediction loss that encourages the prediction detection heatmaps having consistent local peaks with the ground truth. The topK local peaks will be selected as the candidate keypoints for each channel. The K indicates the number of visible keypoints based on the maximum number of people can be detected and the cluster setting. In our experiments, we set K as $30c_i$ where the maximum number of people is 30 and the c_i is the number of keypoints in the ith cluster. Specifically, this loss is the mean Euclidean distance over all pairs of prediction and ground truth detection heatmap. For each keypoint cluster, the loss of location prediction is defined as

$$L_{det}^B = \frac{1}{M} \sum_i (p_i - g_i)^2 \tag{4.1}$$

Where, B indicates branch of dense or sparse cluster, M is the number of pixels in the detection heatmap, p_i and g_i are pixel values at position i in the detection heatmap and ground truth heatmap respectively.

4.3.3 Cluster-wise Tag Embedding

Following Newell et al. [52], we group detected keypoints into instances by tag embedding. For each detection heatmap, the network predicts a tagging heatmap which is responsible for guiding the corresponding keypoint to group with other keypoints. The difference is that our tag embedding is applied in a cluster-wise way. Similar as detection heatmaps,

our tagging heatmaps contain multi-peak responses since we divide keypoints into clusters. The tag values are normalized within each tagging heatmap. Specifically, we represent each tag by the cluser-wise mean value with an offset factor to distinguish the keypoints by their relative orders. By doing this, we can refine the detected position by checking whether the difference between two tags satisfies the offset setting. The generation of tagging heatmaps does not need specific ground truth labels for supervision since the clusters can be grouped into instances based on the differences between the tag values. Therefore, we supervise the cluster-wise embedded tags with a pull loss that encourages similar tags for clusters from same instance and a push loss that enforces different tags for clusters across different instances. Figure 4.3 visualizes the tagging heatmaps.

In general, the tagging heatmaps for Bth branch are trained with embedding loss.

$$L_{emb}^{B} = L_{pull} + L_{push} \tag{4.2}$$

The pull loss L_{pull} is defined as the squared distance between the normalized embedding and the reference embedding of each keypoint.

$$L_{pull} = \frac{1}{NJ} \sum_{n} \sum_{k} (\hat{h_k} - \overline{h_n})^2 \tag{4.3}$$

Where, N is the number of instances, J is the total number of visible keypoints and assuming all instances have J visible keypoints, $\hat{h_k}$ is the normalized embedding for the kth keypoint, $\overline{h_n}$ is the reference embeddings for the nth instance.

The normalized embedding is calculated as the cluster-wise mean value $\tilde{h_k}$ with the offset that encodes the keypoint index d_k in the cluster, such as $d_k = 0$ for the first keypoint.

$$\hat{h_k} = \tilde{h_k} + d_k * \delta \tag{4.4}$$

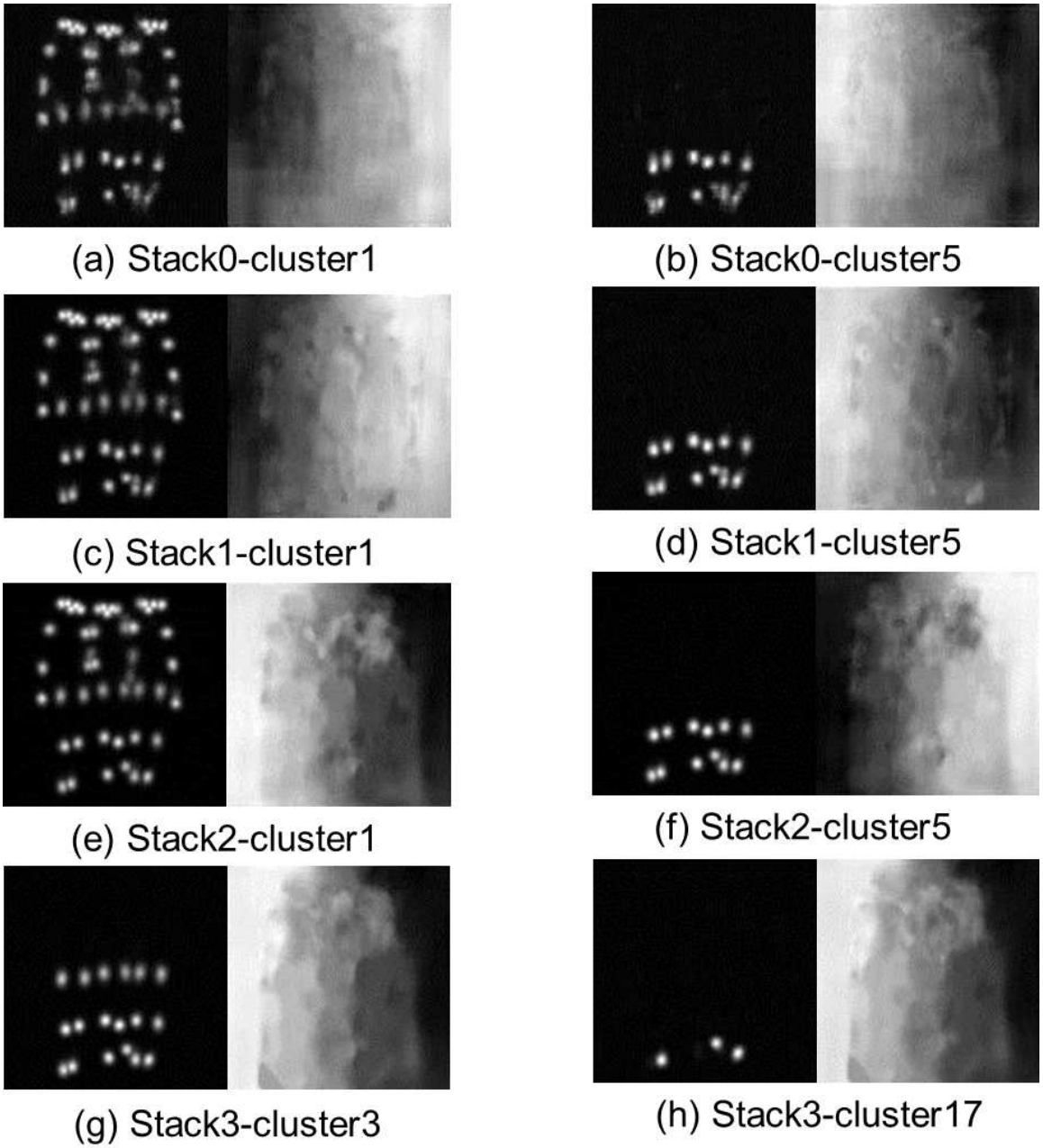


Figure 4.3: Illustration of heatmaps of each stack module.

The heatmaps indicate keypoints(left) and tags(right). The network consists of four stack modules with dense(a,c,e,g) and sparse(b,d,f,h) cluster settings (1,1,1,3) and (5,5,5,17). For simplicity, we only display a part of each cluster.

$$\tilde{h_k} = \frac{1}{c_i} \sum_k h_k \tag{4.5}$$

Where, $d_k \in [0, max(c_i)]$ and c_i is the number of visible keypoints in ith cluster, δ is the base factor for the offset, h_k is the predicted tagging heatmap for the kth keypoint.

The push loss L_{push} is computed between reference embeddings of different instances with a penalty that drops exponentially to zero as the increase of embedding difference.

$$L_{push} = \frac{1}{N(N-1)} \sum_n \sum_{m \neq n} exp(-\frac{1}{2\Delta^2}(\overline{h_n} - \overline{h_m})^2) \tag{4.6}$$

$$\overline{h_n} = \frac{1}{J} \sum_k \tilde{h_k} \tag{4.7}$$

Where, $\overline{h_n}$ and $\overline{h_m}$ are the reference embeddings for different instances, Δ leads a margin for distance between each pair of reference embeddings. Considering inter-instance difference should be much larger than intra-instance difference, we assign the margin Δ based on the offset $d_k\delta$ to far apart reference embeddings from each other. Then, the embedding values are separated both in clusters and instances and benefitted to individual keypoint retrieving. We set (Δ,δ) as (4, 0.03) in our experiments.

Given the predicted keypoints and tagging heatmaps, we group the keypoints across channels by retrieving the tag values at keypoint locations from corresponding channels. To decode the output of the network, the keypoints having similar tag values are matched up to form instances. The full training loss is

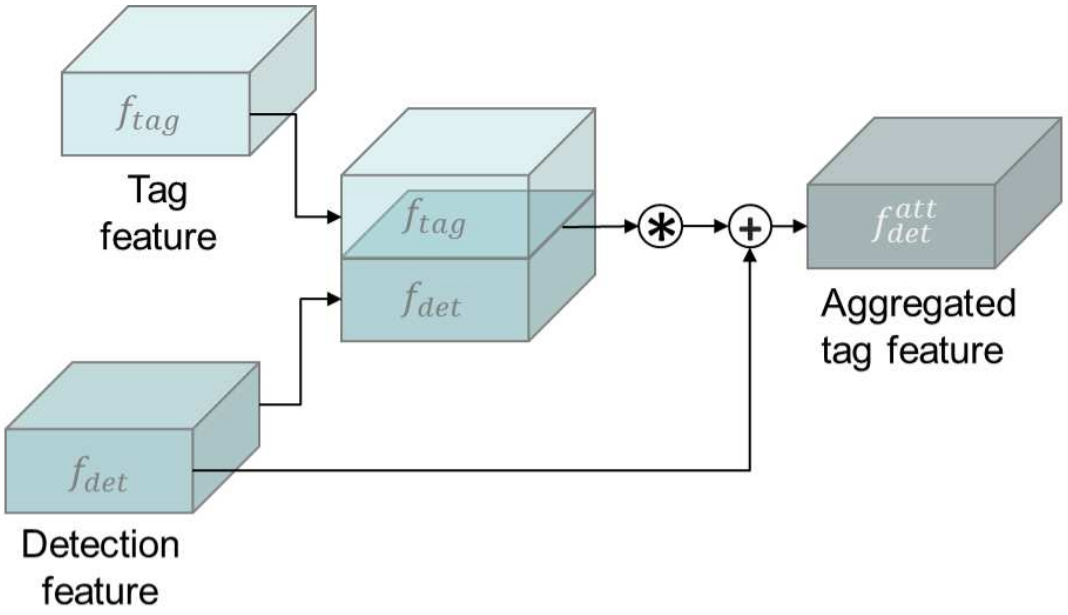
$$L = \sum_B (\lambda_d L_{det}^B + \lambda_e L_{emb}^B) \tag{4.8}$$

Where λ_d and λ_e are the weights for the detection and embedding loss respectively. We set (λ_d, λ_e) as (1, 1e-3) in our experiments.

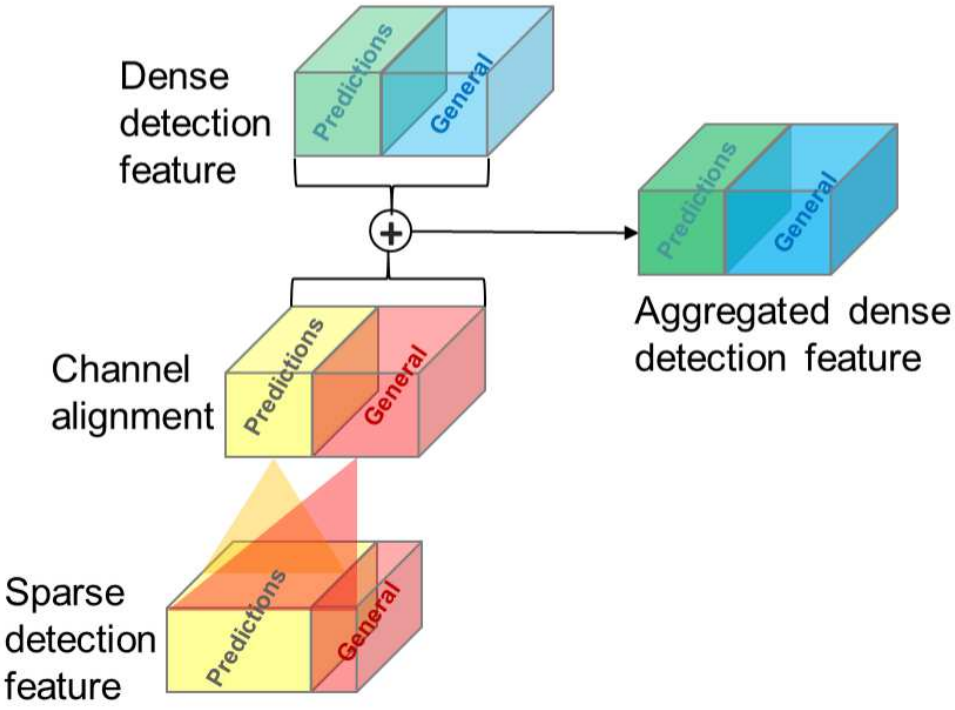
4.3.4 Feature Aggregation

For each detected keypoint, we have to estimate a tag to indicate which person instance it belongs to. Thus, we separate each cluster branch into two sub-branches for keypoint detection and tag embedding. Then, the intra and inter relation of keypoint clusters are embedded to tagging heatmaps. The pixels in the tagging heatmap serve to group keypoint clusters belong to the same instance. Besides that, the regions of person instances are simultaneously exposed, as shown in Figure 4.5. Hence, our architecture can be taken as a multi-task learning, which can jointly handle the keypoint detection and coarse person segmentation problems. Despite the coarse boundaries of person regions generated by the tag embedding branch, it unveils holistic context information that benefits to keypoint detection. Moreover, the stacking strategy of Hourglass modules can be taken as refinement process. Multi-scale features in shallow stack are aggregated into high-level features to be further exploited by deep stack. To enhance the features passing from shallow stack to deep stack, we aggregate information from different branches. More specifically, we enrich the features to be passed to next stack by aggregating tag to detection in-branch and sparse to dense detections cross-branch. We use an in-branch block to aggregate the tag features to the detection features in each branch to generate a region attention for keypoint detection. In addition, the detection features from dense and sparse branches complement each other. We design a cross-branch block to aggregate the sparse detection features to the dense detection features to improve the flow of information between different branches. Thus, our network optimizes the features at a specific keypoint by the received information from diverse context.

The in-branch aggregation targets to enhance keypoint detection features by fusing holistic human region attention maps that are byproducts of tag embedding. The tag embedding generates tagging heatmaps by pulling together the keypoints from same instance while pushing away those from different instances. The supervision is realized through comparison of relative tag values between keypoints but not all pixels in the map. Then,



(a). In-branch aggregation of tag features.



(b). Cross-branch aggregation of detection features.

Figure 4.4: Demonstration of feature aggregation.
In-branch aggregation combines tag and detection features in each branch and cross-branch aggregation fuses detection features from dense and sparse branches.

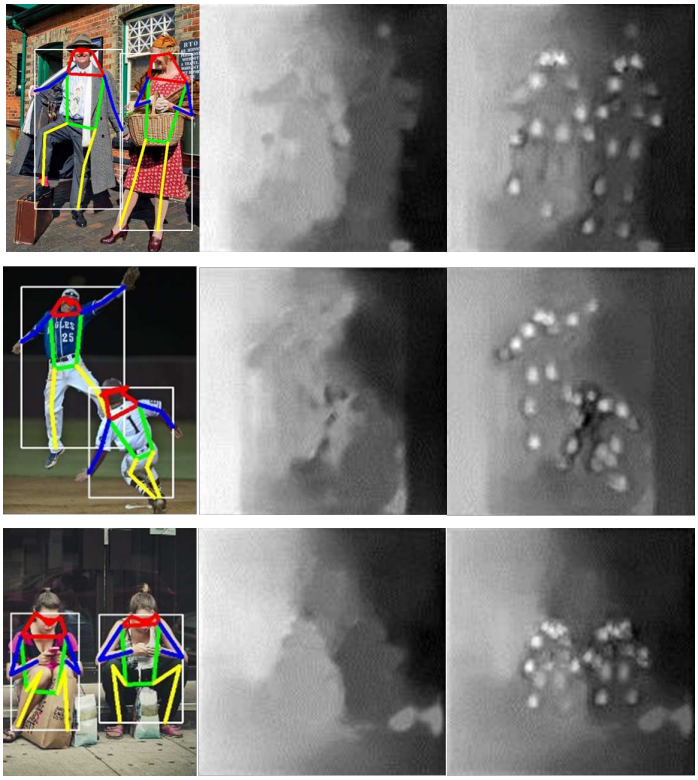


Figure 4.5: Visualization of tag embeddings of dense keypoint cluster.
Columns from left to right are pose predictions overlapping on inputs(cropped for simplicity illustration), tagging heatmaps and combination of detection heatmaps and tagging heatmaps. Note that, the tag value obviously changes across instances.

the keypoints can be grouped into instances based on the differences between the tag values. Moreover, the non-keypoint pixels obtain tag values through convolution propagation, and aggregate into body regions around keypoints and eventually form human regions. Therefore, tagging heatmaps also provide crucial clues of semantic regions for keypoint detection. Based on this observation, we aggregate tag features to detection features, as demonstrated in Figure 4.4 (a). The tag and detection features are firstly concatenated and then fused by a 1x1 convolution layer to generate merged features. Then, the residual connection adds the identity mapping of detection feature to the merged features. Figure 4.5 c visualizes some examples of in-branch aggregation.

The cross-branch aggregation focuses on fusing global and local context information between dense and sparse branches for keypoint detection. To balance features in different branches and make the network easier to extend, the number of channels are same in dense

and sparse branches. The feature maps of each branch are composed of the supervised predictions and the unsupervised general features. In addition, the proportions of these two components in dense and sparse branches are different. The dims of predictions correspond to the number of supervisions while the extra features are the rest channels of total amount. Therefore, we apply different aggregation strategy to these two kinds of feature maps, as illustrated in Figure 4.4 (b). The numbers of channels are different in two branches due to the dense and sparse cluster settings. Therefore, we firstly unify the numbers of channels of two branches before feature aggregation. We use group convolution to merge the feature maps of sparse branch and the group setting is according to the dense branch, such as the feature maps of keypoints in head part are merged together. Thus, we align the channels of sparse branch to dense branch based on their semantic meaning. Considering parameters and computation burden, we use 1x1 kernels to do the channel-wise fusion. Through 1x1 kernels, the prediction features are mapped from sparse to dense and the general features are enriched by aggregating the supervised information. We aggregate the aligned features from sparse branch to dense branch by element-wise sum.

4.4 Results and Discussion

4.4.1 Implementation

We train and evaluate our network on the MSCOCO [74] dataset which is introduced in Chaper 2. Our network is implemented under the PyTorch framework and trained and tested on two GTX 1080 GPUs. The test experiments are also developed on the same environment. We tried on combinations of different randomly initialization and found out that the training works well under the default setting of PyTorch. During training, we set the input resolution of the network to 512x512, which leads to an output resolution of 128x128. To reduce overfitting, we adopt standard data augmentation techniques including random horizontal flipping, scaling, cropping and adjusting the brightness, saturation and contrast of an image. We train the network using a batch size of 8 for 200 epochs with an initial

learning rate of 2e-4 (dropped to 1e-5 after 150 epochs). We find that larger values of learning rate lead to diverging gradient and kill some neural units. The forward passing is terminated by a 1x1 convolution and output detection heatmaps corresponding to each cluster. We use Adam to optimize the full training loss L defined in Eq(8). The loss minimizes the differences between the predicted and the ground truth locations and groupings. The relation embedding term is weighted by a factor of 1e-3 relative to the keypoint detection term. During each round of iterations, 1000 training steps and 10 evaluation steps are alternatively carried on. The network performance is evaluated and samples of each batch are randomly shuffled. Our pipeline needs totally around 0.229 seconds for generating the final detection heatmaps and keypoints converting. Following state-of-the-art, our methods are trained and evaluated on these MSCOCO and PoseTrack datasets.

4.4.2 Performance

To get an intuitive sense of the network's predictions, we firstly visualize the final keypoint locations of our system extracted from validation dataset. Figure 4.6 illustrates the typical cases of occlusion, crowding, deformation and low resolution. We see that our system has superior and robust performance on self-occluded joints with variant deformation of human poses. Our system also makes better balance between false detection and miss detection for the intractable cases caused by dense mutual occlusions and low resolutions. This is quite encouraging since our approach is designed to be a general purpose keypoint predictor. The quantitative evaluation for keypoint detection on MSCOCO is presented in Table 4.1. The primary challenge metric AP and AR are averaged over multiple OKS values with 0.05 interval. The AP^{50} means AP at $OKS = 0.50$. Small objects (segment area $< 32^2$) do not contain keypoint annotations. The AP^M is for the medium objects having areas between of 32^2 and 96^2 and the AP^L is for large objects having area larger than 96^2.

We test the effectiveness of cluster-wise learning by testing the network under different clustering configurations. The cluster configuration (1,1,1,1) means that the 17 keypoints

Table 4.1: Comparison results on MSCOCO testdev dataset with OKS.

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
CMUPose[49]	0.618	0.849	0.675	0.571	0.682	0.665	0.872	0.718	0.606	0.746
RMPE[43]	0.618	0.837	**0.698**	0.586	0.673	0.676	0.875	0.746	0.630	0.740
MaskRCNN[44]	0.631	**0.873**	0.687	0.578	**0.714**	-	-	-	-	-
G-RMI[98]	**0.649**	0.855	0.713	**0.623**	0.700	0.679	**0.887**	**0.755**	**0.644**	0.771
PoseAE[52](baseline)	0.633	0.857	0.689	0.580	0.704	**0.688**	0.884	0.742	0.620	**0.781**
Cluster1	0.617	0.852	0.675	0.564	0.691	0.682	0.894	0.735	0.617	0.769
Cluster2	0.620	0.852	0.677	0.564	0.698	0.682	0.891	0.734	0.615	0.773
In-branch	0.625	0.855	0.682	0.570	**0.703**	0.687	**0.897**	0.739	0.621	**0.777**
Cross-branch	0.625	0.854	0.681	0.571	0.701	0.687	0.895	0.739	0.622	0.776
In-Cross-branch	**0.627**	**0.857**	**0.687**	**0.572**	0.702	**0.688**	0.896	**0.742**	**0.623**	**0.777**

are divided into 1 cluster and will be learnt simultaneously in each stack. Similarly, the 17 keypoints are evenly divided into 3 clusters and 5 clusters with (5,6,6) and (3,3,3,4,4) points in each cluster. Thus, for experiment Cluster-1, the dense cluster configuration is (1,1,1,3) and the sparse cluster configuration is (5,5,5,17). And for experiment Cluster-2,the dense cluster configuration is (1,1,1,1) and the sparse cluster configuration is (17,17,17,17). Following prior arts, we use four stacks of corresponding networks for each experiment. Based on Cluster-2, we further study the benefits of feature aggregation by training and testing the networks with/without In-branch and Cross-branch aggregation blocks. Overall, the In-Cross branch achieves our best result by enforcing the network learns more complex contextual information and semantic consistency for keypoint detection. This result indicates that the more global information encoded, the finer the detection result.

In Table 4.1, we compare our network with other state-of-the-art detectors based on results from their public implementation. The result of PoseAE [52] is our baseline. Comparing with the baseline, our method has better performance at AR^{50} and AR^{M} and comparable performance at most other metrics, like AP^{50}, AP^{75}, AP^{L}, AR and AR^{75}. The overall performance of our network is higher than the state-of-the-art CMUpose [49] and RMPE [43] and reaches a competitive result over the other recent methods. Note that our

Table 4.2: Comparison results on PoseTrack validation dataset with mAP.

Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
DetectTrack[101]	67.5	70.2	62.0	51.7	60.7	58.7	49.8	60.6
PoseFlow[103]	66.7	73.3	68.3	61.1	67.5	67.0	61.3	66.5
SimpleBaseline[102]	81.7	**83.4**	**80.0**	**72.4**	**75.3**	**74.8**	**67.1**	**76.7**
In-Cross-branch	**83.4**	80.9	66.4	56.0	72.5	64.8	52.0	70.3

recalls (AR^{50}, AR^{50} and AR^{L}) are higher than most of the compared methods. The dense branch encodes global consistency information by using more keypoints in a cluster, such as all keypoints of the entire body. By using dense branch, our network explores more keypoints which are difficult to be found.

We retrain our network for multi-person pose estimation on the PoseTrack2018 training set which is introduced in Chapter 2 and we use the model pre-trained on MSCOCO dataset to initialize the network. Since it is a fine-tuning, the training iteration converges within 20 epochs and the base learning rate 1e-5 is changed to 1e-6 after 15 epochs. The other training configures are the same as that for MSCOCO. We use the PoseTrack evaluation toolkit for results on validation set and report results on test set from the evaluation server. Since our network is designed for single-frame pose estimation, we estimate our frame-wise prediction results on validation set. Figure 4.7 and Table 4.2 illustrate results of our approach on PoseTrack validation dataset. Overall, our network performs better in exploring keypoints of head region and competitive in detecting keypoints of shoulders and hips. Figure 4.8 shows some failure cases of our method on MSCOCO and PoseTrack datasets.

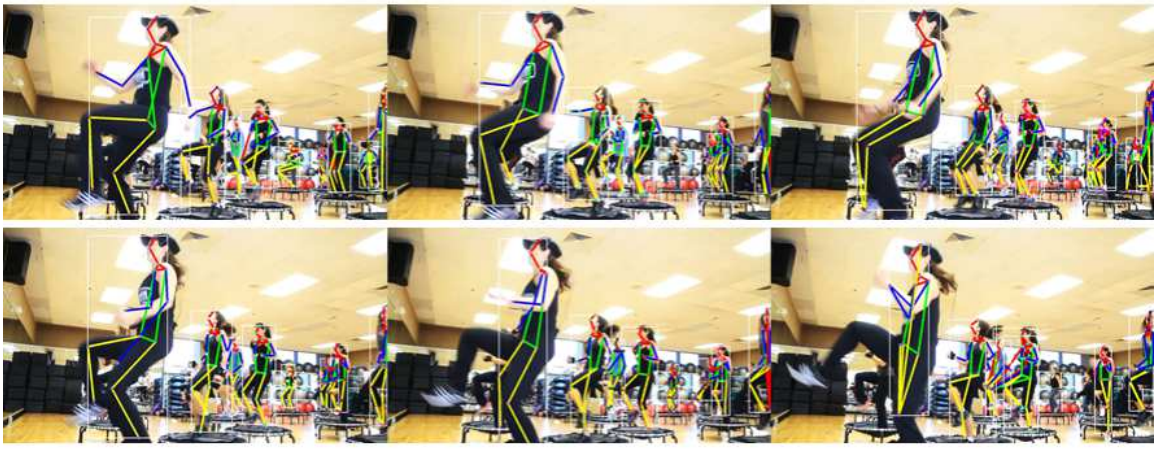
4.5 Conclusion

In this chapter, we propose the idea of exploring global and local contextual relations from dense and sparse cluster-wise multi-peak detection heatmaps for keypoint detection. To enhance repeated bottom-up and top-down feature passing, we aggregate information

Figure 4.6: Visualization of our results on MSCOCO validation dataset.

Figure 4.7: Visualization of our results on PoseTrack test dataset.

(a) Results on COCO validation dataset.



(b) Results on PoseTrack test dataset.

Figure 4.8: Visualization of failure cases.

Figure 4.9: More examples of feature aggregation.

from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches. Meanwhile, the intra-cluster and inter-cluster relationships are embedded with tag learning to guide the instance grouping and individual keypoint identification. However, since the network stacked hourglass modules , its parameters become huge and difficult to be trained under limited computational resource. In next chapter, we introduce a lite version of the network to solve the problems.

CHAPTER 5

LITE HOURGLASS NETWORK FOR MULTI-PERSON POSE ESTIMATION

5.1 Introduction

Recent multi-person pose estimation networks rely on sequential downsampling and upsampling procedures to capture multi-scale features and stacking basic modules to reassess local and global contexts. However, the network parameters become huge and difficult to be trained under limited computational resource. Motived by this observation, we design a lite version of Hourglass module that uses hybrid convolution blocks to reduce the number of parameters while maintaining performance. The hybrid convolution block builds multi-context paths with dilated convolutions with different rates which not only reduces the number of parameters but also enlarges the receptive field. Moreover, due to the limitation of heatmap representation, the networks need extra and non-differentiable post-processing to convert heatmaps to keypoint coordinates. Therefore,we propose a simple and efficient operation based on integral loss to fill this gap specifically for bottom-up pose estimation methods. We demonstrate that the proposed approach achieves better performance than the baseline methods on the challenge benchmark MSCOCO dataset for multi-person pose estimation.

To capture and consolidate information across all scales of the image, the pipeline of repeated downsampling and upsampling are broadly used, such as the classical architecture of the Hourglass network [45, 52]. The Hourglass network pools down the feature maps to a very low resolution, and then upsamples and combines features across multiple resolutions. The most significant attribute of the network is the symmetric topology that consists of a series of convolution, pooling and upsampling layers. This architecture provides the benefit of capturing both global and local features for the pixel-wise prediction

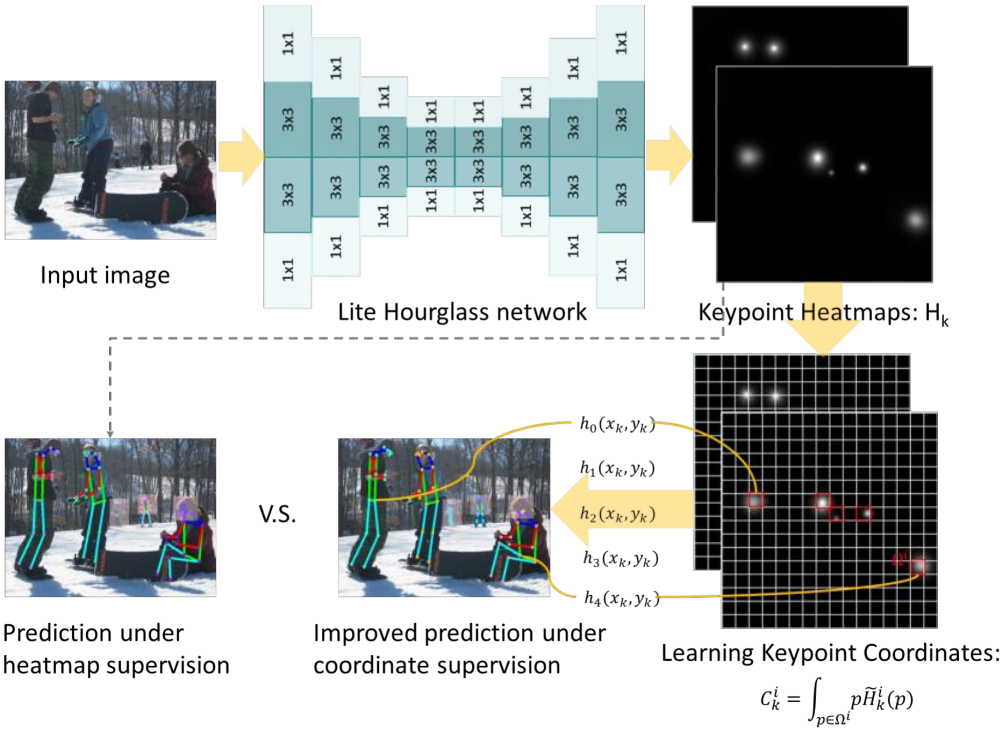


Figure 5.1: Overview of proposed approach.
We propose a lite Hourglass network with bottom-up integral regression.

tasks. However, with the using of stacking multiple Hourglass modules, the network parameters become huge and difficult to be trained under limited computational resource. The top-down pipeline method Stacked Hourglass Network (SHN) [45] uses the bottleneck [64] in each layer of the Hourglass module and achieves good performance. However, due to the needs of simultaneously detecting and grouping keypoints, the bottom-up pipeline method PoseAE [52] uses standard 3x3 convolutional kernels in each layer of the Hourglass module. Motivated by this observations, we propose a lite version of Hourglass module which is able to use much less parameters while maintaining the performance. As shown in Figure 5.1, our module uses hybrid convolution kernels in each scale rather than uniformly using 3x3 kernels. It reduces the number of parameters by using combination of 1x1 and 3x3 convolutional kernels in each scale. The network outputs heatmaps which are usually converted to keypoint coordinates by non-differentiable post-processing. Integral loss [104] proposed by Sun et. al reduces the quantization error. However, it can't be directly used for bottom-up pipeline method. Therefore, we propose a bottom-up integral regression

approach to improve the network performance.

The main contributions of this chapter can be summarized as follows:

- **Lite multi-context block.** We propose a novel hybrid convolution block which builds multi-context paths with dilated convolutions with different rates. The proposed block not only reduces the number of parameters but also enlarges the receptive field. We perform extensive experiments to analyse the properties and the performance of the proposed block.

- **Bottom-up integral regression.** We propose a bottom-up integral regression to transform the heatmap representation to keypoint coordinates by a differentiable way specifically for bottom-up multi-person pose estimation methods. Our approach results in performance improvement over the baseline methods.

5.2 Related Work

Since DeepPose [66], the pose estimation solutions thrive in the presence of deep neural network. The pose estimation network branched out into bottom-up and top-down structures. In the bottom-up branch, the approaches firstly detect body keypoints (or joints) and then group them into person instances. Deepcut [93] proposes a partitioning and labeling formulation of a set of body-part hypotheses generated with CNN-based part detectors. Deepercut [50] and Iqbal et al. [51] also formulate the problem as part grouping and labeling via a linear programming. Following PoseMachine [47], OpenPose [49] uses a part affinity field learn to associate body parts and group them to person instances with greedy bottom-up parsing steps. PoseAE [52] proposes associative embedding to simultaneously generate and group detections. Chu et al. [42] use geometrical transform kernels and a bi-direction tree structured model to capture and pass relationships between joints. PersonLab [53] proposes a part-induced geometric embedding descriptor to associate semantic person pixels with their corresponding person instance. MultiPoseNet [100] receives keypoint and

person detections, and produces accurate poses by assigning keypoints to person instances. On the contrary, top-down methods [42, 43, 44, 45, 46, 47, 48] detect people first and then apply a single person pose estimation to each person detection result.

From the architecture point of view, recent pose estimation networks have universally adopted the architecture consisting of successive bottom-up (from high resolutions to low resolutions) and top-down (from low resolutions to high resolutions) processing. This high-low-high architecture captures and consolidates features across all scales of image with a sequence of downsampling and upsampling. The typical case is the Hourglass Network [45] proposed by Newell et al. which produces keypoint heatmaps by pooling down features to a very low resolution, then unsampling and combining features across multi-resolution. Methods of [52, 54, 95] are also based on the Hourglass module proposed by [45]. In addition, Xia et al. [27] detect keypoints of the human pose by using a fully convolution network (FCN) [105] which also processes spatial information at multiple scales for dense prediction. Compare to FCN [105], the Hourglass has more symmetric distribution of capacity between bottom-up and top-down processing. Unlike the classical high-low-high structures, the recent pose estimation network names HRNet [106] maintains high-resolution representations through the whole network and gradually connects to lower resolutions in parallel.

5.3 Method

5.3.1 Network Overview

As shown in Figure 5.1, the main pipeline of the Hourglass module is the repeated process of bottom-up encoding and top-down decoding. It firstly downsamples the feature maps several times to reach the lowest resolution and then upsamples the feature maps to restore the original scale. Meanwhile, it builds a residual connection between the feature maps from shallow and deep layers having same scale and merges them together. By doing this, it captures and consolidates the information across all scales of the input image.

For each scale, the computation consists of a main path for the fully multi-scale feature extraction and an identity path for current scale feature strengthen. As points verified by methods [45, 52], stacking multiple Hourglass modules with intermediate supervision gradually produces a more accurate final prediction due to the repeated bottom-up and top-down inference. However, the network parameters become huge and difficult to be trained under limited computational resource. Motived by this observation, we design a lite version of Hourglass module that uses hybrid convolution blocks to reduce the number of parameters while maintaining performance.

To detect keypoints from multiple instances, our network is constructed based on the bottom-up pipeline which stacks multiple lite Hourglass modules with two head networks: one head network predicts confidence and precise locations of keypoints, and the other head network predicts associations between keypoints. The network first decreases the input resolution with convolutions and poolings to reduce computational burden. Moreover, the feature maps from previous stack are merged and passed to next stack repeatedly. This processing makes the network process features at both local and global contexts and gradually strengthen the capacity of accurate prediction. With intermediate supervision, each Hourglass module outputs a set of heatmaps indicating keypoints positions and tagmaps for grouping the keypoints into instances. Since the body keypoints only occupy a small area in images, the heatmap matrix is very sparse. Therefore, we propose a bottom-up integral regression approach to reduce the noise of background areas and improve the network performance.

5.3.2 Lite Multi-context Block

The standard convolution layer with 3x3 kernels has more flexible ability of expression while higher computational cost than the one with 1x1 kernels. With the increasing number of the 3x3 convolution layers, the amount of parameters increases to a large extent and at the risk of being redundant, especially for the case of stacking four or more Hourglass

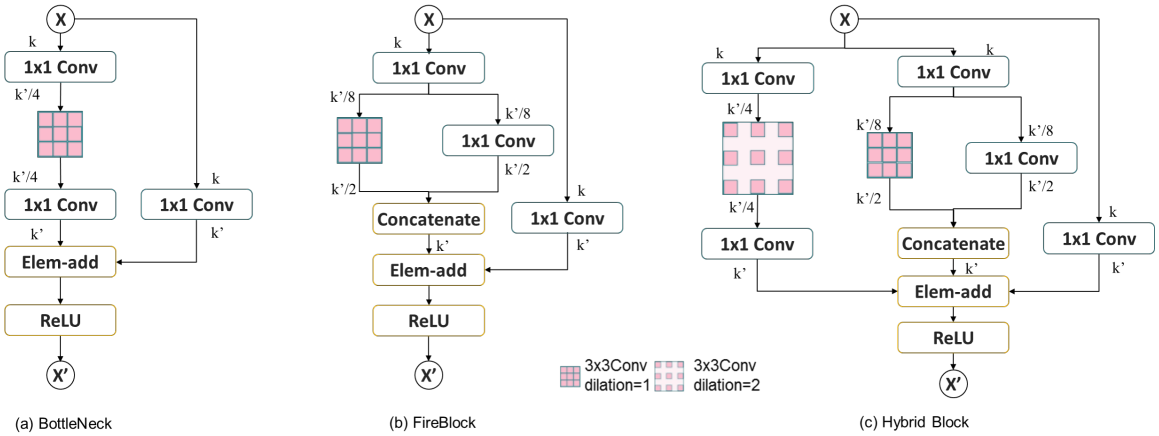


Figure 5.2: Illustration of lite blocks for compacting Hourglass module.
(a) is the BottleNeck of ResNet [64]. (b) is the FireBlock of SqueezeNet [65]. (c) is our proposed hybrid block.

modules. To reduce the amount of parameters, the 1x1 convolution kernels are integrated with 3x3 kernels in a block to harmonize the feature map dimensions. Generally, each block consists of head, core and tail parts. The core part is a 3x3 convolution layer that responses for heavy feature extraction. The block uses a head 1x1 convolution layer to reduce the input dimension and a tail 1x1 convolution layer to harmonize the number of output channels. The typical examples of this kind of block are the BottleNeck from ResNet [64] and the FireBlock from SqueezeNet [65], as shown in Figure 5.2 (a) and 5.2 (b).

The BottleNeck from ResNet [64] is a quite effective structure and broadly used in many computer vision tasks. It consists of a sequence of 1x1, 3x3, and 1x1 convolution layers, where the 1x1 layers are responsible for reducing and then restoring the amount of feature channels, leaving the 3x3 layer a bottleneck with smaller input/output dimensions. The shortcut connection with 1x1 convolutions is used to harmonize dimensions. To compact network complexity, SqueezeNet [65] proposes a FireBlock and uses it to replace 3x3 convolution layers. The FireBlock utilizes 1x1 convolution kernels to squeeze the amount of input channels that fed to the following 3x3 convolution kernels and expands the number of output channels by the other 1x1 convolution kernels. The output of the fire module is a concatenation of 1x1 and 3x3 convolution results. Figure 5.2 (a) and 5.2 (b) show the detail structures of the BottleNeck and FireBlock and input/output dimensions of each layer.

Inspired by the insight of ResNet [64] and SqueezeNet [65], we propose a HybridBlock that integrates ResBlock and FireBlock by using 3x3 convolution kernels with different dilation rates and sharing a projection path based on 1x1 convolution kernels. The basic intention of our hybrid convolution block is to generate more expressive feature maps with less parameters. As shown in Figure 5.2 (c), the proposed HybridBlock is a multi-branch convolution block which integrates the BottleNeck and FireBlock with different squeezing and expansion factors. The numbers beside each layer indicated the input and output dimensions. To be specific, its inner structure can be components: the squeezing convolution layer with 1x1 kernels to decrease the number of channels in the feature map, the expanding convolution layers with 1x1 and 3x3 kernels, and the shortcut connection with 1x1 convolution kernels to match the output dimensions. Take FireBlock branch as an example, the input channels are firstly squeezed into 1/8 of the amount of output k by 1x1 convolution kernels. Then, it expands the feature maps by concatenating two sets of k'/2 channels from 1x1 and 3x3 kernels. Similar with the BottleNeck, to aggregate feature and avoid gradient vanishing, the shortcut connection is set through 1x1 convolution kernels in the HybridBlock. Eventually, the feature maps of all the branches are concatenated and merged into the output feature maps.

By using the HybridBlock, we reduce the computational cost and make it possible to use higher resolution feature maps in the limited computation budget. Retaining high resolution provides more spatial information and is important for spatially detailed image understanding, especially for keypoint detection from smaller objects. However, it also reduces the receptive field that corresponds to context cues for prediction. According to [107], for a unit in a certain layer in the network, its receptive field is defined as the region in the input that the unit affected by. As for this point, we use the dilated convolution [108] to enlarge the receptive field of the higher resolution layers.

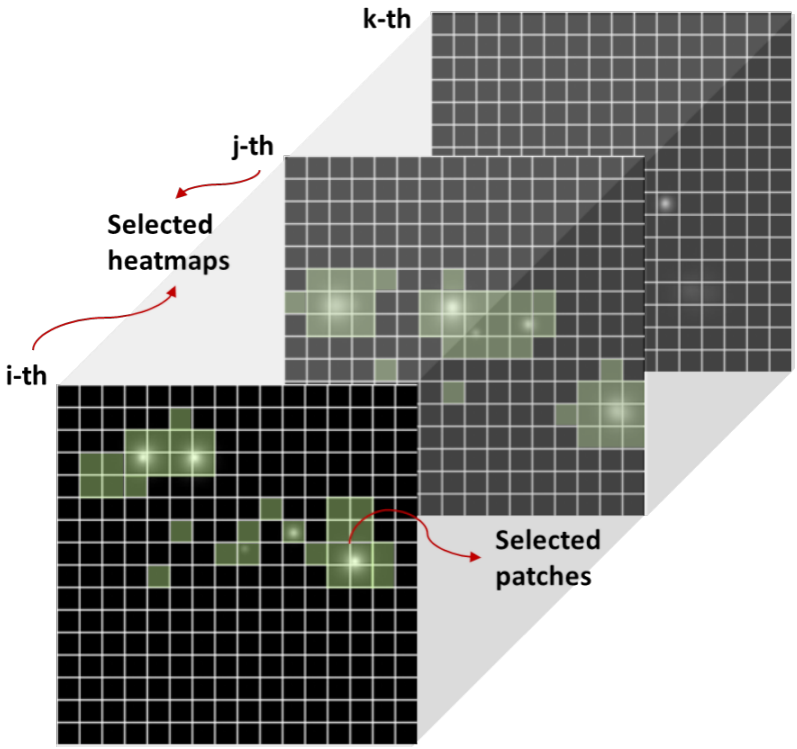


Figure 5.3: Illustration of bottom-up integral regression.

5.3.3 Bottom-up Integral Regression

For each type of keypoint, there is only a few ground-truth positive locations and all other locations are negative. Instead of equally penalizing negative locations, the common approaches [44, 52, 98] reduce the penalty given to negative locations within a radius of the positive location. Therefore, by using a 2D Gaussian, the ground-truth locations are converted to a ground-truth heatmap in which each pixel represents the probability of the location being the keypoint. Thus, the amount of penalty is reduced with the reduction of distance from the keypoint location. The groundtruth heatmap representation can be formed as:

$$H_k(i,j) = exp(-\frac{(i-i_k)^2 + (j-j_k)^2}{2\sigma^2}) \tag{5.1}$$

where (i_k, j_k) is the groundtruth coordinate of the *k-th* keypoint and H_k is the generated heatmap by using the Gaussian kernel with the standard deviation σ.

Since the body keypoints only occupy a small area in images, the heatmap matrix is

very sparse. Moreover, the integral loss [104] is designed for top-down pipeline method which assumes only one person exists in the cropped region of input image. Therefore, we propose a bottom-up integral regression approach to reduce the noise of background areas and quantization error of converting heatmaps to coordinates specifically for bottom-up pipeline method. Our network is trained by minimizing the bottom-up integral loss which is defined as follows:

$$L_k^{det} = \frac{1}{G} \sum_k \sum_g (L_I^{g,k} + L_H^{g,k}) \tag{5.2}$$

where L_k^{det} is the bottom-up integral loss integrates local grid loss to global heatmap loss. For each grid, the loss consists of a coordinate loss L_I^g and a heatmap loss L_H^g. The L_I^g enables the network learns keypoint locations in a differentiable way without quantization errors and L_H^g forces the network to focus on relevant areas. As shown in Figure 5.3, we select top-m grids and top-n heatmaps for final loss calculation.

The coordinate loss L_I^g is calculated in a local region between the predicted heatmap and groundtruth heatmap which are equally divided into grids. The softmax function is applied to each heatmap grid along the spatial axis to generate the coordinate C_k^g for the *k-th* keypoint of multi-person in a differentiable manner. The definition of the L_I^g is as follows:

$$L_I^{g,k} = ||C_{g,k} - C_{g,k}^*|| \tag{5.3}$$

$$C_{g,k} = (\sum_i^w \sum_j^h i\hat{H}_{g,k}(i,j), \sum_i^w \sum_j^h j\hat{H}_{g,k}(i,j)) \tag{5.4}$$

where K is the number of keypoints, $C_{g,k}$ is the groundtruth coordinates for the *g-th* grid of *k-th* heatmap, (w, h) indicates the size of a grid patch, $\hat{H}_{g,k}$ is the predicted heatmap with softmax applied, the groundtruth heatmaps are processed in the same way. For the predicted detection heatmap of each grid, the Mean Square Error (MSE) loss is computed

by

$$L_H^{g,k} = \frac{1}{M}\sum_i (p_i - g_i)^2 \tag{5.5}$$

Where, M is the number of pixels in the heatmap grid, p_i and q_i are pixel values at position i in the heatmap grid and ground-truth map grid respectively.

Given a set of keypoints extracted from local peaks of the heatmaps, we have to group them into person instances. Following PoseAE [52], we calculate the loss for grouping keypoints based on "pulling" together those from the same instance and "pushing" away the others from different instances. Therefore, in addition to producing the detection heatmaps, the network correspondingly outputs tagmaps for keypoint-wise relation embedding and instance retrieving. In each tagmap, pixels having similar values indicate the locations belong to the same instance. The loss for grouping keypoints into instances is calculated by

$$\begin{aligned} L_{emb} = \frac{1}{NK}\sum_n \sum_i (h_i - \overline{h}_n)])^2 + \\ \frac{1}{N(N-1)}\sum_n \sum_{m \neq n} exp(-\frac{1}{2\delta^2}(\overline{h}_n - \overline{h}_m)^2) \end{aligned} \tag{5.6}$$

Where, N is the number of instances, K is the number of visible keypoints, h_i is pixel values at position i in the tagmap, $\overline{h}_n$ and $\overline{h}_m$ are the reference tag embeddings for different instances.

Finally, the fully training loss L enforces the keypoints detection close to the ground-truth and encourages pairs of indexes to have similar values if the corresponding detections belong to the same instance or different values otherwise.

$$L = \lambda_d L_{det} + \lambda_e L_{emb} \tag{5.7}$$

Where λ_d and λ_e are the weights for the detection and embedding loss respectively.

Table 5.1: Dilation setting on COCO held-out500 with training resolution of 512x512.

Blocks	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
OurBlock-f1r1	0.590	0.829	0.645	**0.522**	**0.699**	**0.635**	**0.850**	0.677	0.555	0.749
OurBlock-f2r1	0.590	**0.835**	**0.656**	0.521	0.695	0.633	**0.850**	**0.684**	**0.556**	0.744
OurBlock-f2r2	0.584	0.825	0.645	0.513	0.696	0.629	0.841	0.675	0.548	0.745
OurBlock-f1r2	**0.591**	0.834	0.650	**0.522**	0.695	**0.635**	**0.850**	0.681	**0.556**	**0.747**

Table 5.2: Performances of lite blocks on COCO test-dev with training resolution of 512x512.

Blocks	# Parameters	GFLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
FireBlock [65]	**25,799,568**	**112.16**	0.562	0.821	0.609	0.51	0.636	0.62	0.852	0.666	0.555	0.708
Bottleneck [64]	30,447,248	116.54	0.579	0.831	0.63	0.527	0.651	0.635	0.863	0.683	0.572	0.721
OurBlock	41,789,328	127.52	**0.623**	**0.848**	**0.682**	**0.578**	**0.689**	**0.672**	**0.877**	**0.782**	**0.615**	**0.751**
3x3Conv	*138,933,904*	222.57	0.633	0.857	0.689	0.580	0.704	0.688	0.884	0.742	0.620	0.781

5.4 Results and Discussion

5.4.1 Performance

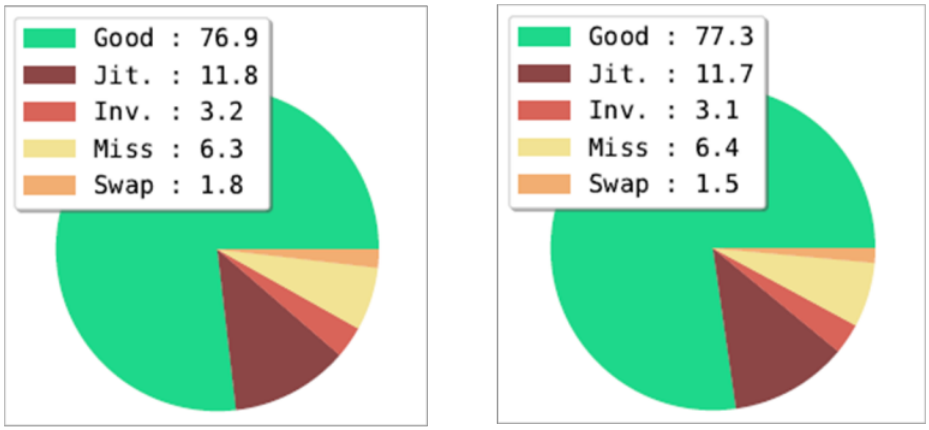
Our network is trained MSCOCO [74] and its details can be found in Chapter 2. And the training implementation is similar with Chapter 4. We evaluate our networks from two aspects: computational cost under the number of parameters and GFLOPs and performance under OKS metrics. We study each major change in our network to understand its contribution to achieve the performance. Table 5.1 shows the results with different dilation settings. By using dilation rates of 1 and 2 for paths of fireblock and bottleneck respectively, our

Table 5.3: Comparison on COCO test-dev dataset.

Methods	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
CMU-Pose [49]	0.618	0.849	0.675	0.571	0.682	0.665	0.872	0.718	0.606	0.746
RMPE[43]	0.618	0.837	0.689	**0.586**	0.673	0.676	0.875	0.746	0.630	0.740
Mask-RCNN(ResNet50) [44]	0.629	**0.871**	0.689	0.576	0.713	**0.697**	**0.913**	**0.751**	**0.639**	0.776
PoseAE [52]	0.633	0.857	0.689	0.580	0.704	0.688	0.884	0.742	0.620	0.781
OurBlock+OurLoss	0.621	0.851	0.683	0.58	0.685	0.677	0.886	0.732	0.622	0.751
PoseAE+OurLoss	**0.642**	0.863	**0.706**	**0.586**	**0.719**	0.696	0.892	**0.751**	0.630	**0.785**

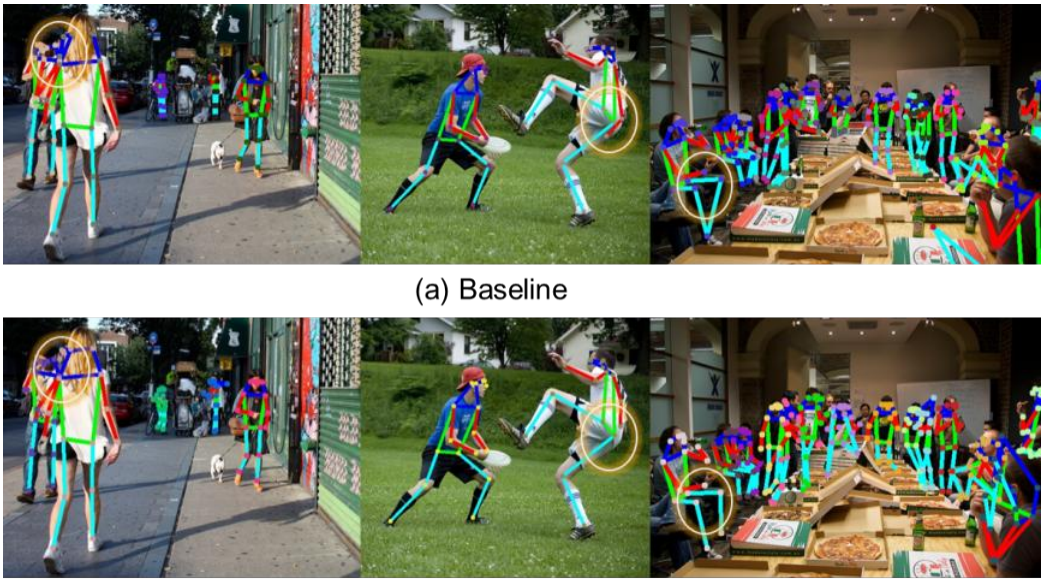
hybrid block achieves best performance on a held-out set containing 500 images. The primary challenge metric AP and AR are averaged over multiple OKS values with 0.05 interval. Small objects (segment area $< 32^2$) do not contain keypoint annotations. The AP^M is for the medium objects having areas between of 32^2 and 96^2 and the AP^L is for large objects having area larger than 96^2. Table 5.2 shows our results compared with the baselines under the input size 512x512. The original structure of PoseAE [52] uses the standard 3x3 convolutional filters in most layers and results in huge number of parameters and GFLOPs. By replacing the 3x3 convolution with the conventional lite blocks of Bottle-Neck or FireBlock, the number of parameters is greatly reduced but the performance is also drastically degraded. On the contrast, our network not only reduces the computational cost but also maintains the performance compare to using standard 3x3 convolutional layers.

We also study the error distributions in our network to understand its contribution to achieve the performance. As introduced in Chapter 2, Ronchi et al. [78] categorize the error distributions include the frequency of pose errors of jitter, inversion, swap, and miss according to the keypoint type, number of visible keypoints, and overlap in the input image. There may also be keypoints that do not have any error and are called good status. More specifically, good status is defined as a very small displacement from the groundtruth



(a) Without our loss. (b) With our loss.

Figure 5.4: Analysis of error distributions.
Error distributions change after applying the proposed bottom-up integral loss to PoseAE [52] network.

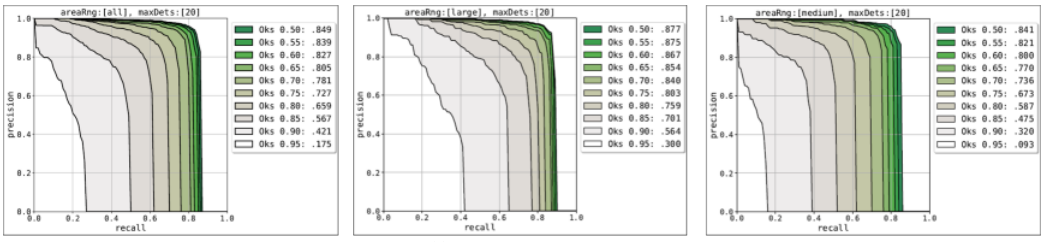


(a) Baseline

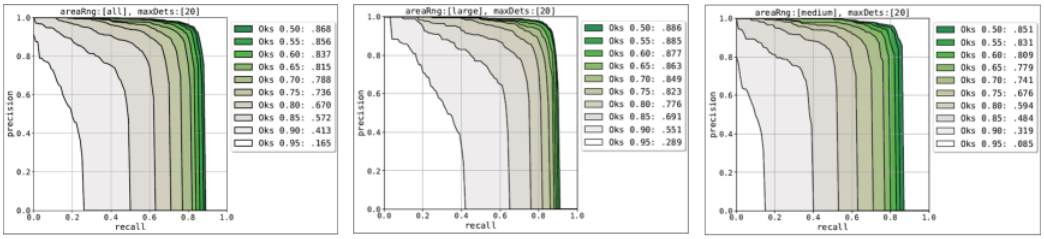
(b) Ours

Figure 5.5: Visualization of pose errors correction comparison.

keypoint. Jitter error is a small displacement error around the groundtruth keypoint locatin. Inversion error occurs when a pose estimation model is confused between semantically similar parts that belong to the same instance. Swap error represents a confusion between the same or similar parts which belong to different instances. Miss error represents a large displacement from the groundtruth keypoint position. The part experiments are carried on the held-out 500 images set. Figure 5.4 shows the error distributions change after applying the proposed bottom-up integral regression to the PoseAE [52] network. Our method not

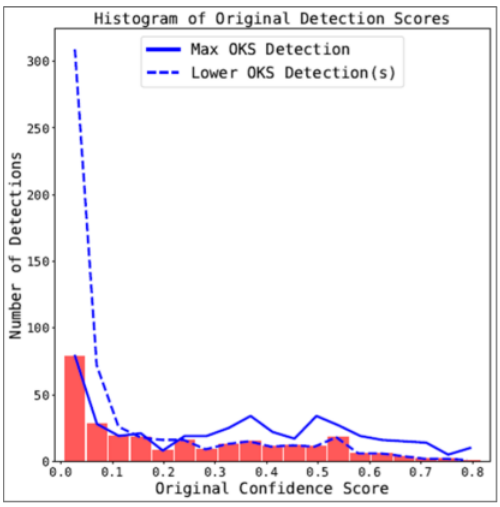


(a) Without our loss.

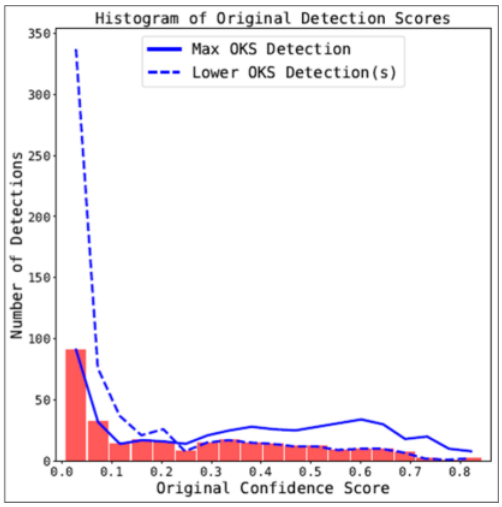


(b) With our loss.

Figure 5.6: Illustration the performance improvement according to the size of targets.



(a) Without our loss.



(b) With our loss.

Figure 5.7: Confidence score and OKS comparison.

only corrects the small displacement error of jitter but also the large displacement errors of inversion and swap. To give an intuitive sense of the improvement, Figure 5.5 visualizes the comparison results of pose errors correction. Figure 5.6 illustrates the performance improvement according to the size of targets. The precision-recall curves showing the performance of PoseAE [52] without and with our loss by progressively increasing the OKS evaluation threshold with 0.05 increment. For both large and medium size targets, the proposed loss improves the network performance. Figure 5.7 represents the changes of heatmap confidence scores which should be OKS monotonic increasing and reflect as much

as possible the probability of being a True Positive. It shows the enlarged amount of overlap between the histogram of detection scores with the highest with a given groundtruth (solid line) and all other detections with a lower OKS (dash line). It indicates that the improved heatmap prediction makes less error and its confidence score is a better OKS predictor.

Finally, we compare our method with state-of-the-art of methods on COCO test-dev set and the results are shown in Table 5.3. Except Mask-RCNN [44], all methods follow the bottom-up detection pipeline which detects keypoints firstly and groups them into instances. As shown in Figure 5.4, the PoseAE [52] achieves better performance with the help of our proposed bottom-up integral loss. It obtains 1.3 point higher AP than the top-down method Mask-RCNN [44] which utilize the extra knowledge of person region. The lite Hourglass network degrades the performance comparing to the PoseAE [52] since it uses much less parameters but it still outperforms CMU-Pose [49] and RMPE [43].

5.5 Conclusion

In this chapter, we present a hybrid block for compact Hourglass module while maintaining the network performance. Unlike standard Hourglass module, we use the hybrid convolution block instead of uniformly using 3x3 convolutional kernels in all layers. The hybrid block builds multi-context paths with dilated convolutions with different rates which not only reduces the number of parameters but also enlarges the receptive field. We also introduce a novel bottom-up integral regression operation to reduce the quantization error of converting heatmaps to keypoint coordinates specifically for bottom-up pipeline multi-person pose estimation methods. However, the network performance is reduced by the lite version. In next chapter, we present an improved method which is easy to train and achieves better performance.

CHAPTER 6
JOINT BOTTOM-UP AND TOP-DOWN LEARNING FOR MULTI-PERSON POSE ESTIMATION

6.1 Introduction

Multi-person pose estimation is mostly challenged by occlusion and variant postures. Existing bottom-up and top-down methods have their own advantages and limitations. The bottom-up approaches detect body keypoints without advance knowledge of person locations which are firstly explored in the top-down pipelines. However, the benefit of unifying the two pipeline is lack of exploration. Motived by this observation, we propose a joint bottom-up and top-down learning method to better predict multi-person join locations. Our network not only learns the body keypoints but also the centers which indicate different person instances. Meanwhile, the offsets from body keypoints to the centers are encoded for both retrieving and grouping the keypoints. Based on shared features of keypoints-to-instances and instances-to-keypoints branches, our method can efficiently perform multi-person pose estimation. Moreover, due to the limitation of heatmap representation, the networks need extra and non-differentiable post-processing to convert heatmaps to keypoint coordinates. Therefore,we propose a simple and efficient operation based on integral loss to fill this gap specifically for bottom-up pose estimation methods. We demonstrate that the proposed approach achieves better performance than the baseline methods on the challenge benchmark MSCOCO dataset for multi-person pose estimation.

The two major approaches in multi-person pose estimation are structured in top-down and bottom-up. On one hand, the top-down methods [42, 43, 44, 45, 46, 47, 48] firstly detect persons and then repeatedly estimate pose for each of them. On the other hand, the bottom-up methods [49, 50, 51, 52, 53, 109] detect body keypoints without advance

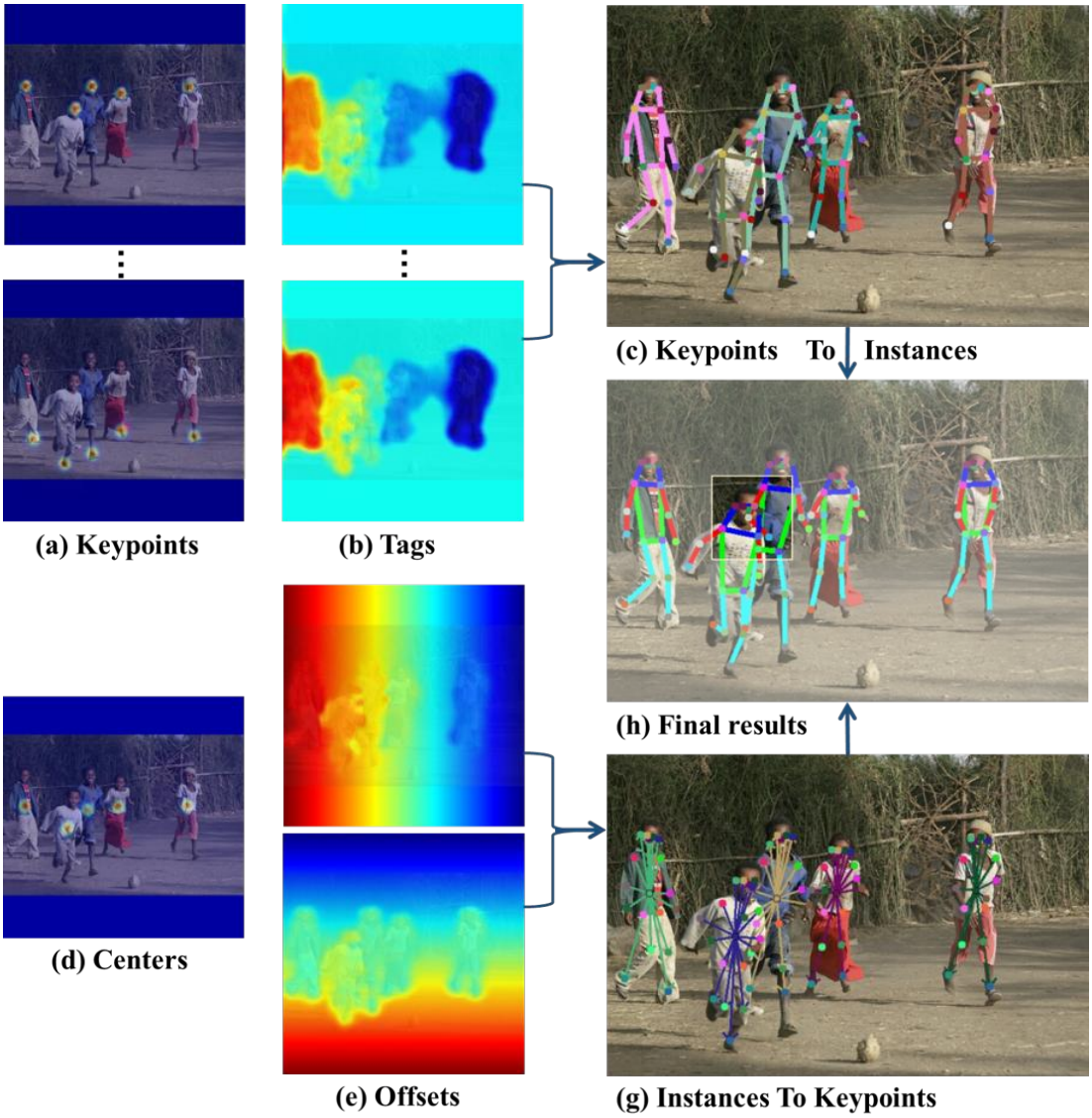


Figure 6.1: Overview of proposed joint bottom-up and top-down learning.

knowledge of person locations and then group them into person instances. Most top-down methods use an isolated human detector to firstly extract instance reigons which are further feeded into the single-person pose estimation network. Without simultaneously training, the context feature from instance regions is not fully contributed to keypoint detection. Recently, instance center with other object properties is creatively utilized for object detection, such as CenterNet [110] and SPM [111]. From this point of view, the human detector of top-down method can be converted to point prediction with heatmap learning which is consistent with the bottom-up methods. Then, the keypoints can be retrieved based on the center points and their offsets with respect to the keypoints. Thus, the top-down pipeline encodes more geometric information which complements the appearance features learnt from bottom-up pipeline. Based on these consideration, we integrate the bottom-up and top-down learning in a single framework to improve the performance, as shown in Figure 6.1.

Existing methods prove that the heatmap learning is more suitable for keypoint detection than the directly coordinates regression. The network outputs heatmaps which are usually converted to keypoint coordinates by non-differentiable post-processing. Integral loss [104] proposed by Sun et. al reduces the quantization error. However, it can't be directly used for bottom-up pipeline method. Therefore, we propose a bottom-up integral regression approach to improve the network performance.

The center offset is the displacement from the keypoint to its instance center. It encodes the retrieving cue to calculate keypoint locations with the detected centers. Its variation acts as an extra grouping tag to assign individual keypoint detections to their closest person instance. Due to the situation of occlusion, different type of keypoints from different instances may be at same location of the image. The occluded keypoints will unable to be retrieved from the missing information. To conquer this problem, our network simultaneously learns holistic and separate offset maps for each keypoints.

The main contributions of this chapter can be summarized as follows:

- **Joint bottom-up and top-down learning.** We propose a unified pipeline to predict body keypoints from both bottom-up and top-down perspectives. Our meth-od explores the benefit of jointly learning muli-person body keypoints from branches of keypoints-to-instan-ces and instances-to-keypoints.

- **Bottom-up integral regression.** We propose to integrate local losses of predicting heatmaps and coordiates together and use a bottom-up integral regression to transform the heatmap representation to keypoint coordinates by a differentiable way specifically for bottom-up multi-person pose estimation methods. The coordinate loss complements the over-segmentat-ion of heatmaps caused by local calculation. Our approach results in performance improvement over the baseline methods.

- **Offset encoding.** We explore the affection of holistically and separately encoding the offsets from body keypoints to centers. We propose a refine method by utilizing both holistically and separately encoded offsets.

6.2 Related Works

Since DeepPose [66], the pose estimation solutions thrive in the presence of deep neural network. The pose estimation network branched out into bottom-up and top-down structures.

In the bottom-up branch, the approaches firstly detect body keypoints (or joints) and then group them into person instances. Deepcut [93] proposes a partitioning and labeling formulation of a set of body-part hypotheses generated with CNN-based part detectors. Deepercut [50] and Iqbal et al. [51] also formulate the problem as part grouping and labeling via a linear programming. Following PoseMachine [47], CMU-Pose [49] uses a part affinity field learn to associate body parts and group them to person instances with greedy bottom-up parsing steps. PoseAE [52] proposes associative embedding to simultaneously generate and group detections. PersonLab [53] introduces a part-induced geometric em-

bedding descriptor to associate semantic person pixels with their corresponding person instance. Clusterwise [109] proposes to learn connected keypoints simultaneously. It uses dense and sparse branches to generate paired multi-peak detection heatmaps for clusters of keypoints. To enhance the feature passing through the network, it aggregates information from different branches. STIE [112] introduces a fully differentiable pose-guided grouping module for both pose estimation and tracking.

On the contrary, top-down methods detect people first and then apply a single person pose estimation to each person detection result. Since the pose estimator only have to focus on single instance, these two-stage methods usually produce fine localization. RMPE [43] by Fang et al. integrates the symmetric spatial transformer network, parametric pose nonmaximum-suppression, and pose-guided proposals generator with stacked Hourglass modules [45] to handle inaccurate bounding boxes. G-RMI [98] by Papandreou et al. predicts dense detection heatmaps and offsets for each keypoint type by using ResNet [64] and combined these outputs by an aggregation procedure to obtain highly localized keypoint predictions. Mask-RCNN [44] by He et al. solves multi-task of object detection, instance segmentation and keypoint prediction together based on the RoI aligned feature maps extracted from ResNet [64] integrated with FPN [99]. MultiPoseNet [100] receives keypoint and person detections, and produces accurate poses by assigning keypoints to person instances. Xia et al. [27] ultilize a PoseFCN and a PartFCN to provide initial estimation of pose joint potential and semantic part potential.

6.3 Method

6.3.1 Joint Bottom-up and Top-down Learning

As shown in Figure 6.1, we unify the bottom-up and top-down pose estimation in a single framework which is referred as BUTDI in the following chapters. The bottom-up pipeline learns keypoint heatmaps and their association tagmaps and captures appearance features, while the top-down pipeline explores instance center point heatmaps and their offset maps

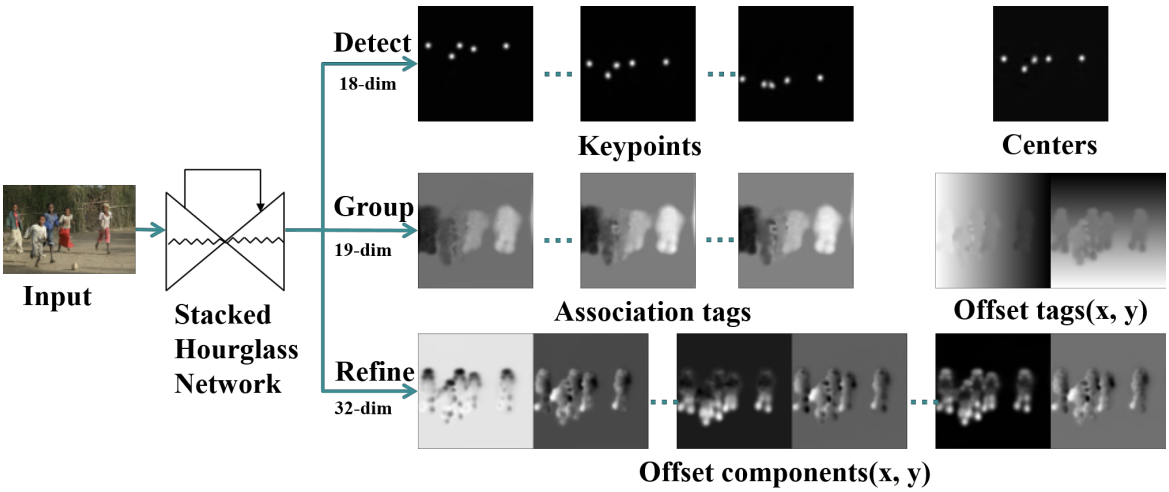


Figure 6.2: The overview of our model architecture.
The network outputs three type of outputs for different tasks. The heatmaps in the top row are detected keypoints and centers. The middle row illustrates the tagmaps for grouping the detected keypoints into instances. The bottom row shows the separately learnt offsets between keypoints and their centers.

with respect to the keypoints and extracts geometric features. Therefore, the network combines them together can simultaneously learn the complementary information. Moreover, since the detection part of both pipelines are points prediction with heatmaps, it is more reasonable and suitable to integrate the bottom-up and top-down learning in a single framework.

Figure 6.2 shows the detail of our framework which consists of three types of outputs: heatmaps for 17-dim keypoints and 1-dim center point detection, 17-dim association tags and 2-dim offset tags for grouping the detected keypoints into instances, and 32-dim offset components for retrieving keypoints and refining results. To use the pre-trained model for initialization, we keep the total output dim as 68. Note that, the offset tags and components are not used at same time. For using offset tag, the supervised dim is 37. For using offset components, the first three keypoints (nose, left-eye, right-eye) are packed as one component and the dim of offset components is 32. In this case, the supervised dim is 67. In summary, 68-dim outputs are produced from a common backbone with different supervisions. We take the Stacked Hourglass Network [52] as the backbone. The network first decreases the input resolution with convolutions and poolings to reduce computational

burden. Moreover, the feature maps from previous stack are merged and passed to next stack repeatedly. This processing makes the network process features at both local and global contexts and gradually strengthen the capacity of accurate prediction. With intermediate supervision, each Hourglass module outputs a set of heatmaps indicating keypoints positions and tagmaps for grouping the keypoints into instances. Since the body keypoints only occupy a small area in images, the heatmap matrix is very sparse. Therefore, we propose a bottom-up integral regression approach to reduce the noise of background areas and improve the network performance.

6.3.2 Bottom-up Integral Regression

We introduce a bottom-up integral regression method in the Chapter 5. Since the heatmap is evenly divided into patches, the keypoint regions may be over-segmented and reduce the regression precision. Therefore, we improve the bottom-up integral regression approach by adding a coordinate loss. Our network is trained by minimizing the bottom-up integral loss which is defined as follows:

$$L_k^{det} = \frac{1}{KG} \sum_k (\sum_g (L_H^{g,k} + L_I^{g,k}) + L_C^k) \tag{6.1}$$

where L_k^{det} integrates each local grid loss into a global detection loss for the *k-th* keypoint, *K* is the number of keypoints, *G* is the number of grids. For each grid, the loss consists of a heatmap loss $L_H^{g,k}$, an integral loss $L_I^{g,k}$ and a coordinate loss $L_C^{g,k}$. The $L_H^{g,k}$ forces the network to focus on relevant areas. The L_I^g enables the network learns keypoint locations in a differentiable way without quantization errors. The L_C^g complements the over-segmentation caused by gridding. As shown in Figure 6.3, we select top-m grids and top-n heatmaps for final loss calculation.

The predicted heatmap and groundtruth heatmap are equally divided into grids. For the predicted detection heatmap of each grid, the heatmap loss L_H^g is computed based on the

Mean Square Error.

$$L_H^{g,k} = \frac{1}{M}\sum_i (p_i - g_i)^2 \tag{6.2}$$

Where, M is the number of pixels in each grid, p_i and q_i are pixel values at the *i-th* grid of the predicted heatmap and ground-truth map respectively.

The integral loss L_I^g is calculated with the coordinates generated from both the predicted heatmap and ground-truth map. The softmax function is applied to each grid of the predicted heatmap and groundtruth map along the spatial axis to generate the coordinate $C_{g,k}$ and $C^*_{g,k}$ in a differentiable manner. The definition of the L_I^g is as follows:

$$L_I^{g,k} = ||C_{g,k} - C^*_{g,k}|| \tag{6.3}$$

$$C_{g,k} = (\sum_i^w \sum_j^h i\hat{H}_{g,k}(i,j), \sum_i^w \sum_j^h j\hat{H}_{g,k}(i,j)) \tag{6.4}$$

where (w, h) indicates the size of the *g-th* grid patch, (i, j) is the coordinate refers to the grid, $\hat{H}_{g,k}$ is the predicted heatmap with softmax applied, and $C^*_{g,k}$ is the ground-truth coordinates for the *g-th* grid of *k-th* ground-truth map in the same way of Equation 6.4.

The coordinate loss L_C^k measures the accuracy of coordinate generation given a neighbourhood of the *k-th* keypoint, as the blue regions shown in Figure 6.3. L_C^k is calculated by distance between the positions of predicted point $C_{b,k}$ and ground-truth keypoint T_k.

$$L_C^k = ||C_{b,k} - T_k|| \tag{6.5}$$

where *b* indicates the neighbourhood of the *k-th* keypoint and its size is 5 in our experiments. The $C_{b,k}$ is generated by Equation 6.4 and converted to the position in image coordinate.

Given a set of keypoints extracted from local peaks of the heatmaps, we have to group them into person instances. Following PoseAE [52], we calculate the loss for grouping key-

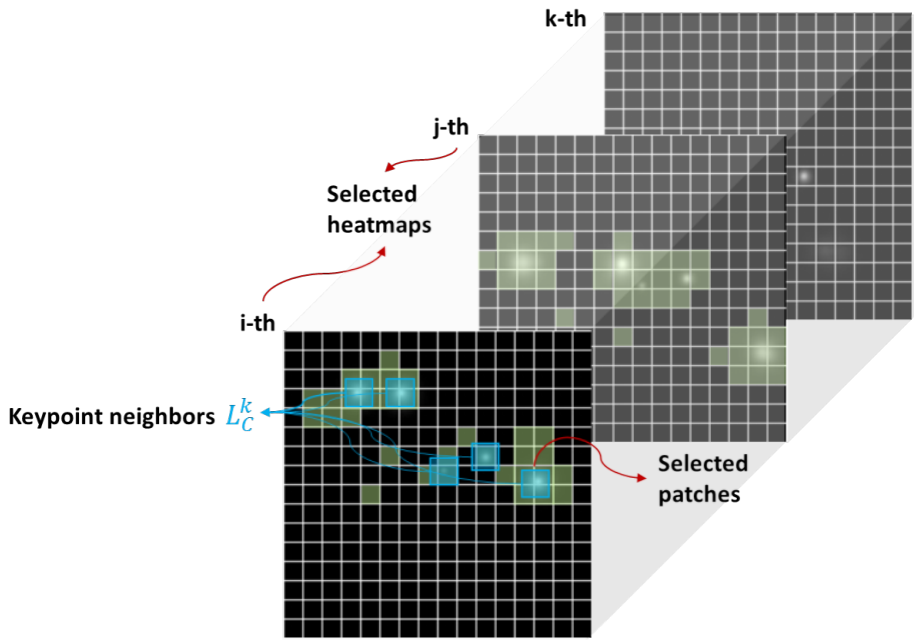


Figure 6.3: Illustration of bottom-up integral regression.

points based on "pulling" together those from the same instance and "pushing" away the others from different instances. Therefore, in addition to producing the detection heatmaps, the network correspondingly outputs associative tagmaps for keypoint-wise relation embedding and instance retrieving, as shown in Figure 6.4 (d). Compare to PoseAE [52], our associative tagmap removes the noise of background. In each tagmap, pixels having similar values indicate the locations belong to the same instance. The loss for grouping keypoints into instances is calculated by

$$\begin{aligned} L_{emb} = \frac{1}{NK}\sum_{n}\sum_{i}(h_i - \overline{h}_n)])^2 + \\ \frac{1}{N(N-1)}\sum_{n}\sum_{m\neq n} exp(-\frac{1}{2\delta^2}(\overline{h}_n - \overline{h}_m)^2) \end{aligned} \tag{6.6}$$

Where, N is the number of instances, K is the number of visible keypoints, h_i is pixel values at position i in the tagmap, $\overline{h}_n$ and $\overline{h}_m$ are the reference tag embeddings for different instances.

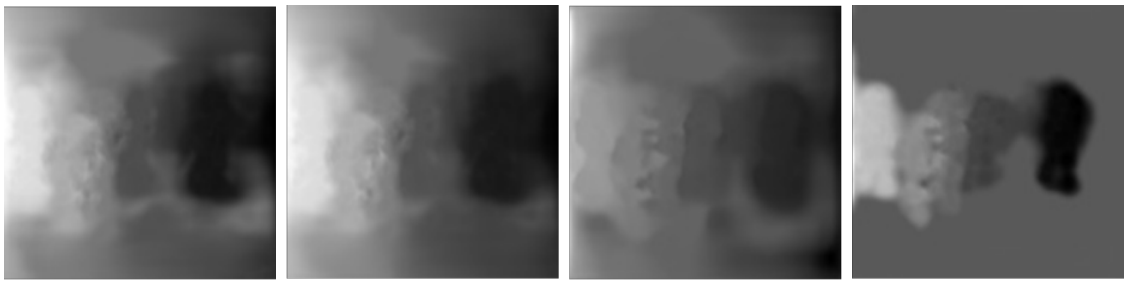


Figure 6.4: Illustration of associative tagmaps.
The tagmaps are respectively generated by (a) PoseAE [52], (b) BUI [113], (c) BUTD and (d) BUTDI. The input is shown in Figure 6.2.

6.3.3 Offset Encoding

The center offset is the displacement from the keypoint to its instance center, as shown in Figure 6.5 (a). It encodes the retrieving cue to calculate keypoint locations with the detected centers. Its variation acts as an extra grouping tag to assign individual keypoint detections to their closest person instance. We construct two dense offset maps S_x and S_y which encode *x* and *y* component of the relative displacement from the keypoints to their instance center locations. Since the offsets of all keypoints are encoded together, we call these two offset maps as holistic offset maps. We denote the center location by (x_c, y_c), the neighbours of each keypoint by (x_k^i, y_k^j) and the ground truth of offset learning by $D(x_k^i, y_k^j)$.

$$D(x_k^i, y_k^j) = (x_k^i, y_k^j) - (x_c, y_c), where(i, j) \in R_k \tag{6.7}$$

where R_k denotes the neighbouring positions of the *k-th* keypoint. Then, we directly regress to the keypoint offsets with an *L1* loss.

$$L_{offset} = \frac{1}{NK} \sum_n \sum_k ||S_n(x_k^i, y_k^j) - D_n(x_k^i, y_k^j)|| \tag{6.8}$$

where *N* is the number of instances, *K* is the number of visible keypoints, the neighbourhood size is set as 3 in our experiments. Figure 6.5 (c) and 6.5 (d) show examples for the constructed holistic offset maps S_x and S_y respectively. By subtracting the offset value

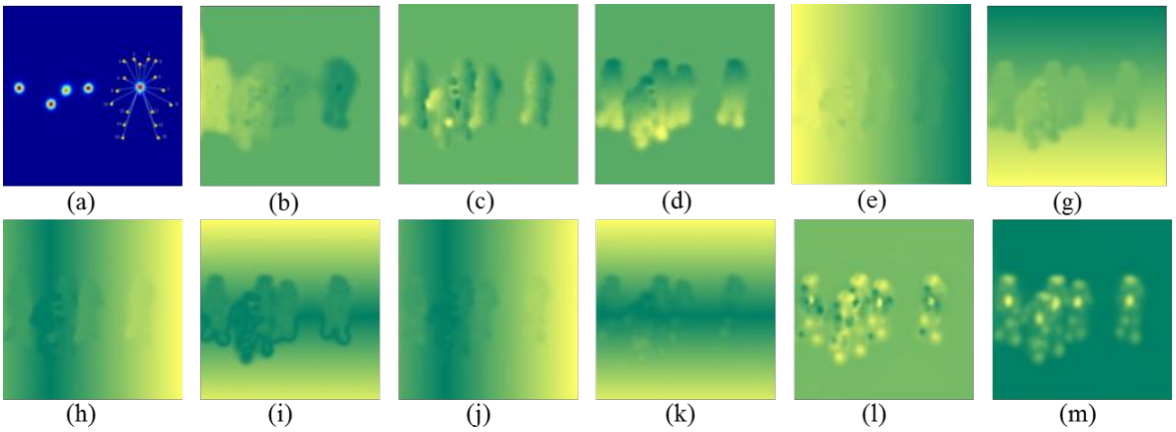


Figure 6.5: Illustration of holistically and separately learnt offsets.
(a) Center heatmap overlap with an offset structure example. (b) Tagmap in center-based offset pipeline. (c) and (d) are holistically learnt offset maps in x and y directions. (e) and (g) are converted tagmaps from (c) and (d). (h) and (i) are instance-based tagmaps from (e) and (g). (j) and (k) are separately learnt keypoint-based offset maps in x and y directions. (l) and (m) are separately learnt center-based offset maps in x and y directions.The input is shown in Figure 6.2.

with its coordinate, the holistic offset maps can be converted to offset-based tagmaps in which the keypoint tags are the coordinates of their instance centers. Figure 6.5(e) and 6.5(g) visualize the offset-based tagmaps corresponding to the x and y coordinates of the instance centers.

By combining the detected centers and offset-based tag-maps together, we can roughly get each instance region, as the dark green zones shown in Figure 6.5 (h) and 6.5 (i). Due to the situation of occlusion, different type of keypoints from different instances may be at same location of the image. The occluded keypoints will unable to be retrieved from the missing information. To conquer this problem, our network simultaneously learns a set of separate offset maps S_n^k for each keypoints. Different from the holistic offset maps, each separate offset map encodes the offset for a specific keypoint. Figure 6.5 (j) and 6.5 (k) show example of the left-shoulder keypoint. Compare with Figure 6.5 (h), Figure 6.5 (j) has more complete instance region in the highlight zone. Since the offsets are encoded at the locations of keypoints, we call this kind of separate offset map as keypoint-based offset map.

Optionally, we can encode the offset value at the location of the center. Once the center

point is detected, the keypoints can be retrieved by subtracting the offset from the coordinates of the centers. This process implicitly uses center locations as the tag to group keypoints. Therefore, we call them as center-based offset maps. Figure 6.5 (l) and 6.5 (m) illustrate the examples of the center-based offset maps for the left-ankle keypoint. We note that the centers are also encoded into the tagmaps if we use the center-based offset learning, as shown in Figure 6.5 (b).

Based on the offset maps, we refine the bottom-up detected keypoints with the top-down reference. As shown in Figure 6.5 (i) and 6.5 (j), the keypoint-based offset maps are firstly converted to tagmaps according to the locations of detected centers. Given a detected keypoint, we search a reference position that has the minimum value in its sliding neighbourhood in the tagmaps. Then, we update the keypoint location with the weighted reference and detected positions. Similarly, we construct reference maps for each keypoint by adding the center coordinates to its center-based offset maps. In a neighbourhood of the center point, the nearest point to the detected keypoint is selected out and its offset is used to update the keypoint location.

Finally, the fully training loss *L* enforces the keypoints detection close to the ground-truth and encourages pairs of indexes to have similar values if the corresponding detections belong to the same instance or different values otherwise.

$$L = \lambda_d L_{det} + \lambda_e L_{emb} + \lambda_o L_{offset} \tag{6.9}$$

Where λ_d, λ_e and λ_o are the weights for the detection, embedding and offset learning loss respectively.

Table 6.1: Results on MSCOCO validation500 dataset.

Experiemnts	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
PoseAE [52]	0.593	0.820	0.649	0.505	0.727	0.639	0.839	0.685	0.540	0.780
BUI [113]	0.599	**0.829**	0.649	0.508	0.734	0.642	0.840	0.684	0.544	**0.782**
BUTD	0.600	**0.829**	0.666	0.515	0.732	0.642	0.843	0.693	0.548	0.775
BUTDI	**0.604**	**0.829**	0.662	0.515	**0.738**	0.648	**0.846**	0.694	0.553	**0.783**
BUTDI-k	0.601	0.827	**0.670**	0.516	0.732	0.646	0.843	**0.701**	0.554	0.776
BUTDI-kr	0.600	0.827	0.664	0.515	0.732	0.645	0.843	0.700	0.553	0.775
BUTDI-c	**0.604**	0.826	0.659	0.519	0.733	0.648	0.843	0.692	0.557	0.778
BUTDI-cr	**0.604**	0.826	0.662	**0.520**	0.734	**0.649**	0.843	0.695	**0.558**	0.778

Table 6.2: Multi-scale results on MSCOCO validation500 dataset.

Experiments	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
PoseAE [52]	0.665	0.849	0.726	0.612	0.744	0.701	0.867	0.755	0.640	0.789
STIE [112]	**0.680**	**0.878**	0.747	0.626	**0.761**	–	–	–	–	–
BUI [113]	0.671	0.852	0.732	0.617	0.749	0.706	**0.890**	0.760	0.647	0.789
BUTDI	0.678	0.862	**0.751**	0.630	0.751	0.713	0.878	0.772	0.657	0.792
BUTDI-k	0.678	0.862	**0.751**	0.628	0.752	0.713	0.879	**0.776**	0.656	**0.794**
BUTDI-cr	0.679	0.860	**0.751**	**0.631**	0.747	**0.714**	0.877	**0.776**	**0.661**	0.790

6.4 Results and Discussion

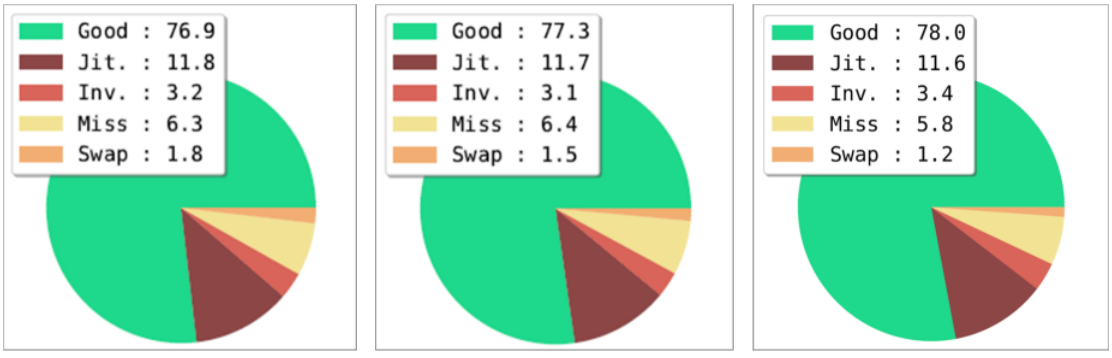
6.4.1 Performance

We train the proposed network on MSCOCO [74] under the similar setting of Chapter 4. We evaluate our networks performance under OKS metrics. The primary challenge metric AP and AR are averaged over multiple OKS values with 0.05 interval. Small objects (segment area $< 32^2$) do not contain keypoint annotations. The AP^M is for the medium objects having areas between of 32^2 and 96^2 and the AP^L is for large objects having area larger than 96^2. We study each major change in our network to understand its contribution to achieve the performance.

Table 6.1 shows the results with different supervisions for training. By using bottom-up integral (BUI) loss to supervise the heatmap learning, our network achieves better performance than the baseline PoseAE [52] on a held-out set containing 500 images, referred as MSCOCO validation500 dataset. The performance of our network is further improved by introducing the supervision of top-down pipeline (BUTD) which learns the instance centers and the holistic offsets. Compare to result of the second experiment, the introducing of keypoint-based (BUTDI-k) separate offset improves AP^{75} and AR^{75} but reduces the performance on other metrics. The refine process (BUTDI-kr) with keypoint-based offset map just keeps the performance. The center-based (BUTDI-c) offset improves the performance on AP^M and AR^M. And the refine process (BUTDI-cr) further improves its performance. Table 6.2 shows the results of multi-scale testing on the held-out set. Moreover, we compare our method with state-of-the-art of methods on MSCOCO test-dev set and the results are shown in Table 6.3. We can see that the proposed joint bottom-up and top-down learning achieves overall 0.648 AP which is better than the most competitors. We also note that the proposed method performance better even on the challenge case of medium size objects with 0.603 AP^M and 0.641 AR^M. Overall, our method achieves better performance than the baseline and competitive performance with state-of-the-arts.

Table 6.3: Comparison on MSCOCO test-dev dataset.

Experiments	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
CMU-Pose [49]	0.618	0.849	0.675	0.571	0.682	0.665	0.872	0.718	0.606	0.746
RMPE [43]	0.618	0.837	0.689	0.586	0.673	0.676	0.875	0.746	0.630	0.740
Mask-RCNN[44]	0.629	**0.871**	0.689	0.576	0.713	**0.697**	**0.913**	**0.751**	0.639	0.776
PoseAE[52]	0.633	0.857	0.689	0.580	0.704	0.688	0.884	0.742	0.620	0.781
Clusterwise[109]	0.627	0.857	0.687	0.572	0.702	0.688	0.896	0.742	0.623	0.777
G-RMI[98]	**0.649**	0.855	**0.713**	**0.623**	0.700	**0.697**	0.887	0.755	**0.644**	0.771
BUI[113]	0.642	0.863	0.706	0.586	**0.719**	0.696	0.892	**0.751**	0.630	**0.785**
BUTDI	0.646	**0.861**	0.707	0.593	**0.720**	0.694	0.887	0.749	0.630	**0.782**
BUTDI-k	0.645	0.860	0.707	0.593	0.719	0.693	0.887	0.749	0.631	0.779
BUTDI-cr	**0.648**	0.860	**0.712**	**0.603**	0.712	**0.698**	**0.890**	**0.753**	**0.641**	0.776



(a) (b) (c)

Figure 6.6: Comparison of error distributions.
Results are from (a) PoseAE [52], (b) BUI [113] and (c) BUTDI.

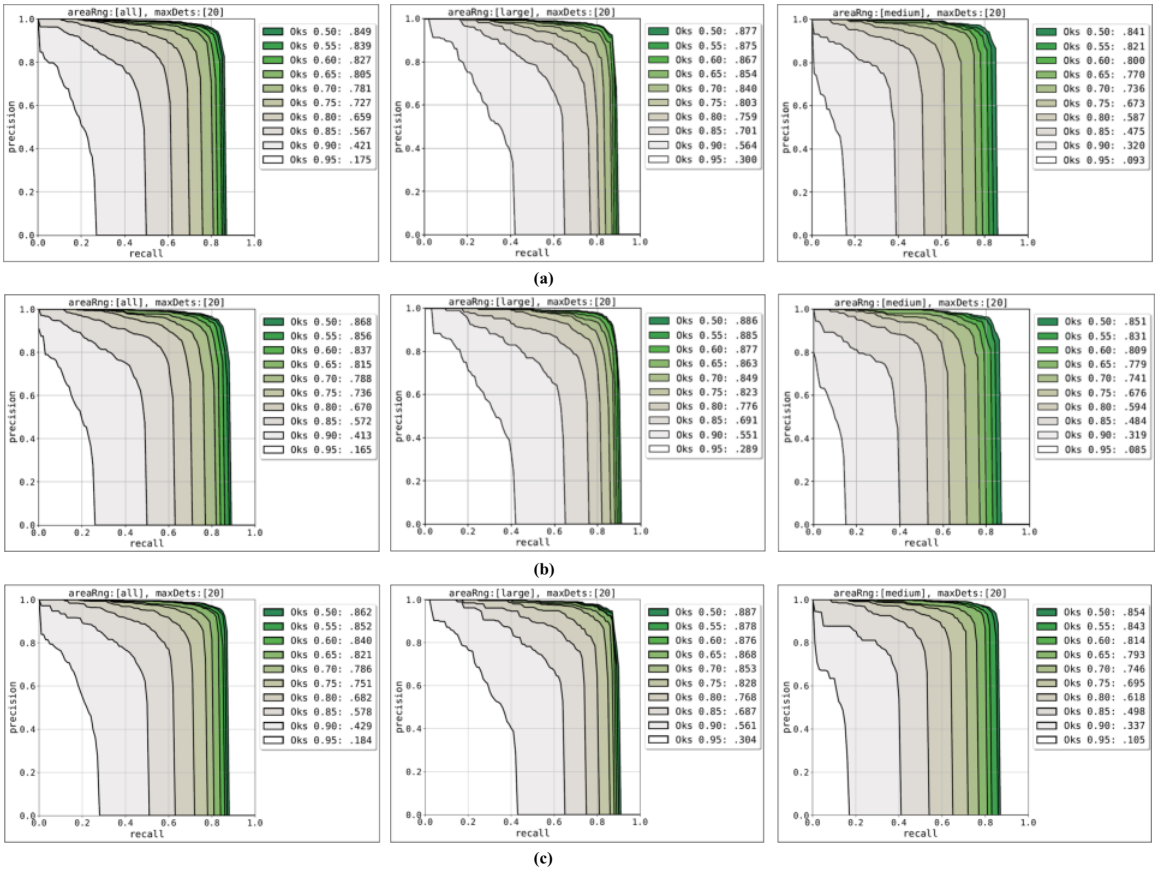


Figure 6.7: Comparison of the performance improvement according to the size of targets. Results are from (a) PoseAE [52], (b) BUI [113] and (c) BUTDI.

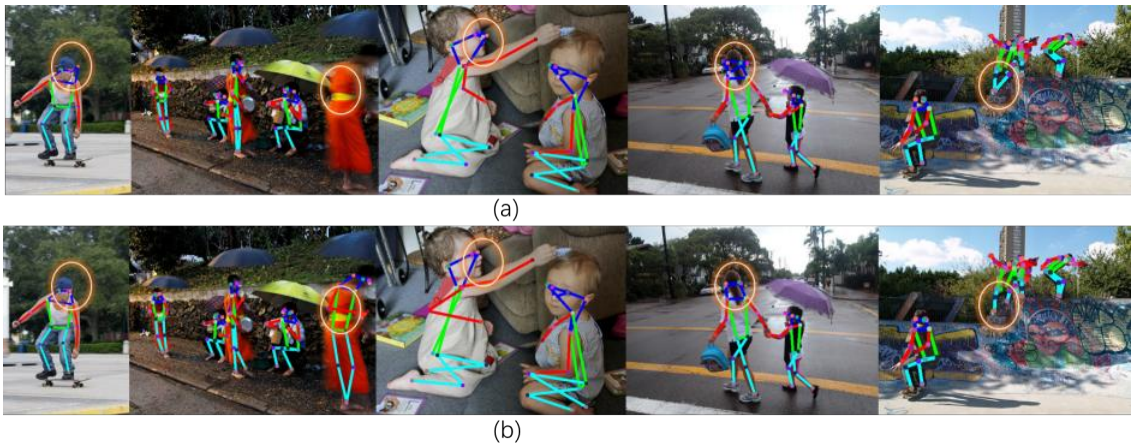


Figure 6.8: Qualitative results on MSCOCO validation500 dataset.
Results of (a) PoseAE [52] and (b) Ours.

We also study the error distributions in our network to understand its contribution to achieve the performance. We follow the analysis method proposed by Ronchi et al. [78] which categorize the error distributions include the frequency of pose errors of jitter, inversion, swap, and miss according to the keypoint type, number of visible keypoints, and overlap in the input image. Figure 6.6 shows the error distributions change after applying the proposed bottom-up integral regression to the PoseAE [52] network. Our method not only corrects the small displacement error of jitter but also the large displacement errors of inversion and swap. Figure 6.7 illustrates the performance improvement according to the size of targets. The precision-recall curves showing the performance of PoseAE [52] without and with our loss by progressively increasing the OKS evaluation threshold with 0.05 increment. For both large and medium size targets, the proposed loss improves the network performance. To give an intuitive sense of the improvement, Figure 6.8 visualizes the comparison results of pose errors correction.

6.5 Conclusion

In this chapter, we propose a joint bottom-up and top-down learning method to better predict multi-person join locations. Our network not only learns the body keypoints but also the centers which indicate different person instances. Meanwhile, the offsets from body

keypoints to the centers are encoded for both retrieving and grouping the keypoints. Based on shared features of bottom-up and top-down branches, our method can efficiently perform multi-person pose estimation. We also introduce an improved bottom-up integral regression operation to reduce the quantization error of converting heatmaps to keypoint coordinates specifically for bottom-up pipeline multi-person pose estimation methods.

CHAPTER 7

CONCLUSION

The thesis characterizes the benefits of image-based learning methods for natural Human-Computer Interaction (HCI) that unconsciously recognizes diversity human activities in the wild. Due to unpredictable environmental condition in the natural application scene, such as rapid changes in illuminations, background clutter, multiple persons with variant articulation of body limbs, diverse appearance, self-occlusion, different scales, significant overlap between neighbor persons and crowd, it is challenging to extract human body information from the images. The recent promising wearable cameras for egocentric vision complement the traditional way of vision data capturing. This thesis demonstrates a hand segmentation approach that exploits the traits of egocentric viewpoint. Meanwhile, the thesis presents three approaches of multi-human pose estimation using unconstrained vision which offers extensive and diverse clues for not only the human activity but also the scene understanding. Experiments carried on public datasets validate the generality of the proposed approaches.

To robust segment user's hand from varying illumination and appearance conditions, we propose a coarse-to-fine online learning approach. We introduce a frame-level hand presence detection approach that utilizes hand motion saliency in the egocentric HCI, which reduces the false positive rate for the final target of pixel-level hand segmentation. We present a top-down cascaded classification method which segments hand hierarchically in levels of frame, superpixel and pixel so as to reduce computational cost, in which the classifiers are trained on-the-fly so as to be robust to diverse users. We analyze and optimize the online trained classifiers by a bottom-up method which makes the hand segmentation robust to varying environmental conditions. Experiments carried on public datasets validate the generality of the proposed approach. This method shows the potential of unsupervised

method for pixel-level hand segmentation in egocentric interaction. In the future, it is worth to try to combine hand segmentation with hand keypoint estimation to provide more dimensions for HCI.

To robust estimate human pose from practical application scene, we discuss three methods from perspectives network head and backbone structure. The cluster-wise feature aggregation method illustrates the benefits of using multiple task heads to enforce the network learn richer contextual information. The lite hourglass network introduces the potential way of balancing performance and computational cost by using hybrid dilated blocks in the network backbone. The joint bottom-up and top-down learning method encodes more geometric information which complements the appearance features and improves the network performance.

Firstly, we introduce our cluster-wise learning approach for multi-person pose estimation. Instead of detect each keypoint separately, our network predicts multi-peak detection heatmaps for clusters of dense and sparse keypoints, which exploits global and local contextual information to improve the detection robustness. To enhance feature passing from shallow stack to deep stack, we aggregate information from different branches. The in-branch aggregation enriches the detection features in each branch by absorbing the holistic human region attention. The cross-branch aggregation further strengthens the detection features by fusing global and local context information between dense and sparse branches. To better grouping the detected keypoints into instances, our network embeds relationships among the intra-cluster and inter-cluster keypoints with offset learning, which not only benefits the instance grouping but also individual keypoint identification. We demonstrate competitive performance of our network on the benchmark dataset for multi-person pose estimation. In the future, we are keeping on exploring more effective restriction of keypoint relations.

Secondly, we present our lite version of hourglass network for reducing the computational cost while maintaining the performance. We propose a novel hybrid convolution

block which builds multi-context paths with dilated convolutions with different rates. The proposed block not only reduces the number of parameters but also enlarges the receptive field. We perform extensive experiments to analyse the properties and the performance of the proposed block. We propose a bottom-up integral regression to transform the heatmap representation to keypoint coordinates by a differentiable way specifically for bottom-up multi-person pose estimation methods. Our approach results in performance improvement over the baseline methods. We demonstrate competitive performance of our network on the benchmark dataset for multi-person pose estimation.

Last but not least, we introduce a joint bottom-up and top-down learning approach that integrates complementary appearance and geometric information. We propose a unified pipeline to predict body keypoints from both bottom-up and top-down perspectives. Our approach explores the benefit of jointly learning muli-person body keypoints from branches of keypoints-to-instances and instances-to-keypoints. We explore the affection of holistically and separately encoding the offsets from body keypoints to centers. We propose a refine method by utilizing both holistically and separately encoded offsets. Moreover, due to the limitation of heatmap representation, the networks need extra and non-differentiable post-processing to convert heatmaps to keypoint coordinates. Therefore,we propose a simple and efficient operation based on integral loss to fill this gap specifically for bottom-up pose estimation methods. We demonstrate that the proposed approach achieves better performance than the baseline methods on the challenge benchmark dataset for multi-person pose estimation.

In the future, we believe that these methods can be extended to or integrated with other computer vision tasks, such as object detection by exploring keypoints, sparse-to-dense tasks like depth recovering, image matting, instance segmentation, etc. Our hand segmentation method also can be transferred to the pixel-level object segmentation by combining with gaze analysis and contribute to activity recognition. Moreover, it also can be integrated with deep convolution neural networks to achieve better performance by using higher di-

mension features. For pose estimation, further improvement is expected by integrating with more advanced existing techniques of flownet [114] tracking, relation extraction [115] and information passing [116]. Moreover, to meet the end of real-time application, further improvement will be expected by integrating the architecture with more lightweight kernels and multi-scale supervision to achieve better performance. Considering integration of hand segmentation and human pose estimation, it brings us to a more exciting and challenging future topic of whole-body keypoint detection [9].

REFERENCES

[1] https://www.microsoft.com/en-us/hololens/.

[2] https://theta360.com/en/about/theta/.

[3] A. Betancourt, P. Morerio, C. S. Regazzoni, and M. Rauterberg, "The evolution of first person vision methods: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 5, pp. 744–760, 2015.

[4] G. Serra, M. Camurri, L. Baraldi, M. Benedetti, and R. Cucchiara, "Hand segmentation for gesture recognition in ego-vision," in *Proceedings of the 3rd ACM international workshop on Interactive Multimedia on Mobile & Portable Devices*, Association for Computing Machinery, 2013, pp. 31–36.

[5] Z. Lu and K. Grauman, "Story-driven summarization for egocentric video," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2013, pp. 2714–2721.

[6] Y. J. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2012, pp. 1346–1353.

[7] B. A. Singletary and T. E. Starner, "Learning visual models of social engagement," in *IEEE International Conference on Computer Vision, ICCV Workshop*, IEEE Computer Society, 2001, pp. 141–148.

[8] E. Ng, D. Xiang, H. Joo, and K. Grauman, "You2me: Inferring body pose in egocentric video via first and second person interactions," *CoRR*, vol. abs/1904.09882, 2019.

[9] G. Hidalgo, Y. Raaj, H. Idrees, D. Xiang, H. Joo, T. Simon, and Y. Sheikh, "Single-network whole-body pose estimation," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2019, pp. 6982–6991.

[10] C. Li and K. M. Kitani, "Pixel-level hand detection in ego-centric videos," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2013, pp. 3570–3577.

[11] X. Zhu, X. Jia, and K. K. Wong, "Pixel-level hand detection with shape-aware structured forests," in *Asian Conference on Computer Vision, ACCV*, Springer, 2014, pp. 64–78.

[12] C. Li and K. M. Kitani, "Model recommendation with virtual probes for egocentric hand detection," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2013, pp. 2624–2631.

[13] L. Baraldi, F. Paci, G. Serra, L. Benini, and R. Cucchiara, "Gesture recognition using wearable vision sensors to enhance visitors' museum experiences," in *IEEE Sensors Journal*, vol. 15, IEEE, 2015, pp. 2705–2714.

[14] X. Ren and C. Gu, "Figure-ground segmentation improves handled object recognition in egocentric video," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2010, pp. 3137–3144.

[15] P. Morerio, L. Marcenaro, and C. S. Regazzoni, "Hand detection in first person vision," in *Information Fusion*, IEEE, 2013, pp. 1502–1507.

[16] A. Betancourt, "A sequential classifier for hand detection in the framework of egocentric vision," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR Workshops*, IEEE Computer Society, 2014, pp. 600–605.

[17] A. Betancourt, P. Morerio, E. I. Barakova, L. Marcenaro, M. Rauterberg, and C. S. Regazzoni, "A dynamic approach and a new dataset for hand-detection in first person vision," in *International conference on Computer Analysis of Images and Patterns, ICAIP*, ser. Lecture Notes in Computer Science, vol. 9256, Springer, 2015, pp. 274–287.

[18] J. Kumar, Q. Li, S. Kyal, E. A. Bernal, and R. Bala, "On-the-fly hand detection training with application in egocentric action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR Workshops*, IEEE Computer Society, 2015, pp. 18–27.

[19] Y. Zhao, Z. Luo, and C. Quan, "Coarse-to-fine online learning for hand segmentation in egocentric video," *EURASIP Journal on Image and Video Processing*, vol. 2018, p. 20, 2018.

[20] J. C. Núñez, R. Cabido, J. J. Pantrigo, A. S. Montemayor, and J. F. Vélez, "Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition," in *Pattern Recognition*, vol. 76, Elsevier, 2018, pp. 80–94.

[21] L. Ge, Y. Cai, J. Weng, and J. Yuan, "Hand pointnet: 3d hand pose estimation using point sets," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2018, pp. 8417–8426.

[22] G. Garcia-Hernando, S. Yuan, S. Baek, and T. Kim, "First-person hand action benchmark with RGB-D videos and 3d hand pose annotations," in *IEEE Confer-*

ence on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, 2018, pp. 409–419.

[23] L. Huang, Y. Huang, W. Ouyang, and L. Wang, "Part-aligned pose-guided recurrent network for action recognition," in *Pattern Recognition*, vol. 92, Elsevier, 2019, pp. 165–176.

[24] L. L. Presti and M. L. Cascia, "3d skeleton-based human action classification: A survey," in *Pattern Recognition*, vol. 53, Elsevier, 2016, pp. 130–147.

[25] M. Liu, H. Liu, and C. Chen, "Enhanced skeleton visualization for view invariant human action recognition," in *Pattern Recognition*, vol. 68, Elsevier, 2017, pp. 346–362.

[26] S. Qi, W. Wang, B. Jia, J. Shen, and S. Zhu, "Learning human-object interactions by graph parsing neural networks," in *Proceedings of the European Conference on Computer Vision, ECCV*, 2018, pp. 407–423.

[27] F. Xia, P. Wang, X. Chen, and A. L. Yuille, "Joint multi-person pose estimation and semantic part segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2017, pp. 6080–6089.

[28] Y. Yang and D. Ramanan, "Articulated human detection with flexible mixtures of parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence,TPAMI*, vol. 35, no. 12, pp. 2878–2890, 2013.

[29] W. Wang, Z. Zhang, S. Qi, J. Shen, Y. Pang, and L. Shao, "Learning compositional neural information fusion for human parsing," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2019, pp. 5703–5713.

[30] C. Patruno, R. Marani, G. Cicirelli, E. Stella, and T. D'Orazio, "People re-identification using skeleton standard posture and color descriptors from RGB-D data," in *Pattern Recognition*, vol. 89, Elsevier, 2019, pp. 77–90.

[31] Y. Lin, L. Zheng, Z. Zheng, Y. Wu, Z. Hu, C. Yan, and Y. Yang, "Improving person re-identification by attribute and identity learning," in *Pattern Recognition*, vol. 95, Elsevier, 2019, pp. 151–161.

[32] S. Tulsiani and J. Malik, "Viewpoints and keypoints," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2015, pp. 1510–1519.

[33] X. Zhou, A. Karpur, L. Luo, and Q. Huang, "Starmap for category-agnostic keypoint and viewpoint estimation," in *Proceedings of the European Conference on*

Computer Vision, ECCV, ser. Lecture Notes in Computer Science, vol. 11205, Springer, 2018, pp. 328–345.

[34] W. Wang, J. Shen, R. Yang, and F. Porikli, "Saliency-aware video object segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI*, vol. 40, no. 1, pp. 20–33, 2018.

[35] W. Wang, J. Shen, H. Sun, and L. Shao, "Video co-saliency guided co-segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,TPAMI*, vol. 28, no. 8, pp. 1727–1736, 2018.

[36] W. Wang, Y. Xu, J. Shen, and S. Zhu, "Attentive fashion grammar network for fashion landmark detection and clothing category classification," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2018, pp. 4271–4280.

[37] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, "Category-specific object reconstruction from a single image," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2015, pp. 1966–1974.

[38] X. Zhang, Z. Tang, J. Hou, and Y. Hao, "3d human pose estimation via human structure-aware fully connected network," in *Pattern Recognition Letters*, vol. 125, Elsevier, 2019, pp. 404–410.

[39] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum, "Marrnet: 3d shape reconstruction via 2.5d sketches," in *Advances in Neural Information Processing Systems, NeurIPS*, MIT Press, 2017, pp. 540–550.

[40] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. L. Yuille, "Detect what you can: Detecting and representing objects using holistic models and body parts," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2014, pp. 1979–1986.

[41] Z. Liu, S. Yan, P. Luo, X. Wang, and X. Tang, "Fashion landmark detection in the wild," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 9906, Springer, 2016, pp. 229–245.

[42] X. Chu, W. Ouyang, H. Li, and X. Wang, "Structured feature learning for pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2016, pp. 4715–4723.

[43] H. Fang, S. Xie, Y. Tai, and C. Lu, "RMPE: regional multi-person pose estimation," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2017, pp. 2353–2362.

[44] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2017, pp. 2980–2988.

[45] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 9912, Springer, 2016, pp. 483–499.

[46] W. Yang, S. Li, W. Ouyang, H. Li, and X. Wang, "Learning feature pyramids for human pose estimation," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2017, pp. 1290–1299.

[47] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2016, pp. 4724–4732.

[48] S. Huang, M. Gong, and D. Tao, "A coarse-fine network for keypoint localization," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2017, pp. 3047–3056.

[49] Z. Cao, T. Simon, S. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2017, pp. 1302–1310.

[50] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele, "Deepercut: A deeper, stronger, and faster multi-person pose estimation model," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 9910, Springer, 2016, pp. 34–50.

[51] U. Iqbal and J. Gall, "Multi-person pose estimation with local joint-to-person associations," in *Proceedings of the European Conference on Computer Vision, ECCV Workshops*, ser. Lecture Notes in Computer Science, vol. 9914, Springer, 2016, pp. 627–642.

[52] A. Newell, Z. Huang, and J. Deng, "Associative embedding: End-to-end learning for joint detection and grouping," in *Advances in Neural Information Processing Systems, NeurIPS*, MIT Press, 2017, pp. 2274–2284.

[53] G. Papandreou, T. Zhu, L. Chen, S. Gidaris, J. Tompson, and K. Murphy, "Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model," in *Proceedings of the European Conference on Computer Vision (ECCV)*, ser. Lecture Notes in Computer Science, vol. 11218, Springer, 2018, pp. 282–299.

[54] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis, "Coarse-to-fine volumetric prediction for single-image 3d human pose," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2017, pp. 1263–1272.

[55] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2015, pp. 648–656.

[56] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[57] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

[58] M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S. Hu, "Global contrast based salient region detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI*, vol. 37, no. 3, pp. 569–582, 2015.

[59] A. Borji, M. Cheng, Q. Hou, H. Jiang, and J. Li, "Salient object detection: A survey," *Computational Visual Media*, vol. 5, no. 2, pp. 117–150, 2019.

[60] F. Liu and M. Gleicher, "Region enhanced scale-invariant saliency detection," in *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME*, IEEE, 2006, pp. 1477–1480.

[61] Z. Yu and H. Wong, "A rule based technique for extraction of visual attention regions based on real-time clustering," in *IEEE Transactions on Multimedia*, vol. 9, IEEE, 2007, pp. 766–784.

[62] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2015, pp. 448–456.

[63] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[64] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2016, pp. 770–778.

[65] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size," *CoRR*, vol. abs/1602.07360, 2016.

[66] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2014, pp. 1653–1660.

[67] S. Johnson and M. Everingham, "Learning effective human pose estimation from inaccurate annotation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2011, pp. 1465–1472.

[68] F. Wang and Y. Li, "Beyond physical connections: Tree models in human pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2013, pp. 596–603.

[69] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2015, pp. 648–656.

[70] Y. Yang and D. Ramanan, "Articulated pose estimation with flexible mixtures-of-parts," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2011, pp. 1385–1392.

[71] L. Pishchulin, M. Andriluka, P. V. Gehler, and B. Schiele, "Strong appearance and expressive spatial models for human pose estimation," in *IEEE International Conference on Computer Vision, ICCV*, 2013, pp. 3487–3494.

[72] P. Hu and D. Ramanan, "Bottom-up and top-down reasoning with hierarchical rectified gaussians," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2016, pp. 5600–5609.

[73] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2016, pp. 4724–4732.

[74] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 8693, Springer, 2014, pp. 740–755.

[75] M. Andriluka, U. Iqbal, E. Insafutdinov, L. Pishchulin, A. Milan, J. Gall, and B. Schiele, "Posetrack: A benchmark for human pose estimation and tracking," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2018, pp. 5167–5176.

[76] M. Andriluka, L. Pishchulin, P. V. Gehler, and B. Schiele, "2d human pose estimation: New benchmark and state of the art analysis," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2014, pp. 3686–3693.

[77] V. Ferrari, M. J. Marín-Jiménez, and A. Zisserman, "Progressive search space reduction for human pose estimation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2008, pp. 1–8.

[78] M. R. Ronchi and P. Perona, "Benchmarking and error diagnosis in multi-instance pose estimation," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2017, pp. 369–378.

[79] T. Ishihara, K. M. Kitani, W. Ma, H. Takagi, and C. Asakawa, "Recognizing hand-object interactions in wearable camera videos," in *IEEE International Conference on Image Processing, ICIP*, IEEE, 2015, pp. 1349–1353.

[80] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending A hand: Detecting hands and recognizing activities in complex egocentric interactions," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2015, pp. 1949–1957.

[81] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2011, pp. 3281–3288.

[82] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2012, pp. 2847–2854.

[83] A. Betancourt, P. Morerio, E. I. Barakova, L. Marcenaro, M. Rauterberg, and C. S. Regazzoni, "Left/right hand segmentation in egocentric videos," in *Computer Vision and Image Understanding*, vol. 154, Elsevier, 2017, pp. 73–81.

[84] X. Zhu, W. Liu, X. Jia, and K. K. Wong, "A two-stage detector for hand detection in ego-centric videos," in *IEEE Winter Conference on Applications of Computer Vision, WACV*, IEEE Computer Society, 2016, pp. 1–8.

[85] Y. Sheikh, O. Javed, and T. Kanade, "Background subtraction for freely moving cameras," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2009, pp. 1219–1225.

[86] M. Narayana, A. R. Hanson, and E. G. Learned-Miller, "Coherent motion segmentation in moving camera videos using optical flow orientations," in *IEEE Inter-*

national Conference on Computer Vision, ICCV, IEEE Computer Society, 2013, pp. 1577–1584.

[87] X. Liang, C. Zhang, and T. Matsuyama, "Inlier estimation for moving camera motion segmentation," in *Asian Conference on Computer Vision, ACCV*, ser. Lecture Notes in Computer Science, vol. 9006, Springer, 2014, pp. 352–367.

[88] C. Zhang, Z. Chen, M. Wang, M. Li, and S. Jiang, "Robust non-local tv-l^1 optical flow estimation with occlusion detection," in *IEEE Transactions on Image Processing*, vol. 26, IEEE, 2017, pp. 4055–4067.

[89] E. Ising, "Beitrag zur theorie des ferromagnetismus," in *Zeitschrift Für Physik*, vol. 31, Springer Berlin Heidelberg (Germany), 1925, pp. 253–258.

[90] S. G. Brush, "History of the lenz-ising model," in *Reviews of modern physics*, vol. 39, American Physical Society, 1967, p. 883.

[91] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2005, pp. 886–893.

[92] L. Breiman, "Random forests," in *Machine Learning*, vol. 45, Springer, 2001, pp. 5–32.

[93] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, and B. Schiele, "Deepcut: Joint subset partition and labeling for multi person pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2016, pp. 4929–4937.

[94] H. Altwaijry, A. Veit, and S. J. Belongie, "Learning to detect and match keypoints with deep architectures," in *Proceedings of the British Machine Vision Conference, BMVC*, BMVA Press, 2016.

[95] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis, "6-dof object pose from semantic keypoints," in *IEEE International Conference on Robotics and Automation, ICRA*, IEEE, 2017, pp. 2011–2018.

[96] J. Thewlis, H. Bilen, and A. Vedaldi, "Unsupervised learning of object landmarks by factorized spatial embeddings," in *IEEE International Conference on Computer Vision, ICCV*, IEEE Computer Society, 2017, pp. 322 9–3238.

[97] Y. Zhang, Y. Guo, Y. Jin, Y. Luo, Z. He, and H. Lee, "Unsupervised discovery of object landmarks as structural representations," in *IEEE Conference on Computer*

Vision and Pattern Recognition, CVPR, IEEE Computer Society, 2018, pp. 2694–2703.

[98] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy, "Towards accurate multi-person pose estimation in the wild," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2017, pp. 3711–3719.

[99] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2017, pp. 2117–2125.

[100] M. Kocabas, S. Karagoz, and E. Akbas, "Multiposenet: Fast multi-person pose estimation using pose residual network," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 11215, Springer, 2018, pp. 437–453.

[101] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, and D. Tran, "Detect-and-track: Efficient pose estimation in videos," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2018, pp. 350–359.

[102] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 11210, Springer, 2018, pp. 472–487.

[103] Y. Xiu, J. Li, H. Wang, Y. Fang, and C. Lu, "Pose flow: Efficient online pose tracking," in *Proceedings of the British Machine Vision Conference, BMVC*, BMVA Press, 2018, p. 53.

[104] X. Sun, B. Xiao, F. Wei, S. Liang, and Y. Wei, "Integral human pose regression," in *Proceedings of the European Conference on Computer Vision, ECCV*, ser. Lecture Notes in Computer Science, vol. 11210, Springer, 2018, pp. 536–553.

[105] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CoRR*, vol. abs/1411.4038, 2014.

[106] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2019, pp. 5693–5703.

[107] W. Luo, Y. Li, R. Urtasun, and R. S. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Advances in Neural Information Processing Systems, NeurIPS*, MIT Press, 2016, pp. 4898–4906.

[108] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations, ICLR*, 2016.

[109] Y. Zhao, Z. Luo, C. Quan, D. Liu, and G. Wang, "Cluster-wise learning network for multi-person pose estimation," in *Pattern Recognition*, Elsevier, 2019, p. 107 074.

[110] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *CoRR*, vol. abs/1904.07 850, 2019. arXiv: 1904.07850.

[111] X. Nie, J. Zhang, S. Yan, and J. Feng, "Single-stage multi-person pose machines," *CoRR*, vol. abs/1908.09220, 2019. arXiv: 1908.09220.

[112] S. Jin, W. Liu, W. Ouyang, and C. Qian, "Multi-person articulated tracking with spatial and temporal embeddings," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2019, pp. 566 4–5673.

[113] Y. Zhao, Z. Luo, C. Quan, D. Liu, and G. Wang, "Lite hourglas network for multi-person pose estimation," in *International Conference on MultiMedia Modeling*, Springer, 2020, pp. 226–238.

[114] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2017, pp. 1647–1655.

[115] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation networks for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, 2018, pp. 3588–3597.

[116] A. Kumar and R. Chellappa, "A convolution tree with deconvolution branches: Exploiting geometric relationships for single shot keypoint detection," *CoRR*, vol. abs / 1704.01880, 2017.

PUBLICATIONS

[1] Y. Zhao, Z. Luo, C. Quan, D. Liu, and G. Wang, Joint Bottom-up and Top-down Learning for Multi-person Pose Estimation[J], Neurocomputing, 2019. (Under Review)

[2] Y. Zhao, Z. Luo, C. Quan, D. Liu, and G. Wang, Lite Hourglass Network for Multi-person Pose Estimation[C], 26th International Conference on MultiMedia Modeling, MMM 2020, Daejeson, Korea, January 5-8, 2020, Springer, 2020, pp.226-237. (Published)

[3] Y. Zhao, Z. Luo, C. Quan, D. Liu, and G. Wang, Cluster-wise Learning Network for Multi-person Pose Estimation[J], Pattern Recognition, 2020, 98:107074. (Published)

[4] Y. Zhao, Z. Luo, C. Quan, Coarse-to-fine Online Learning for Hand Segmentation in Egocentric Video[J], EURASIP Journal on Image and Video Processing, 2018, 2018(1): 20. (Published)

[5] Y. Zhao, Z. Luo, C. Quan, Unsupervised Online Learning for Fine-grained Hand Segmentation in Egocentric Video[C], 14th Conference on Computer and Robot Vision, CRV 2017, Edmonton, AB, Canada, May 16-19, 2017, IEEE Computer Society, 2017, pp. 248–255. (Published)

Doctor Thesis, Kobe University
”Image-based Learning of Human Body Information
for Natural Human-Computer Interaction ”, 149 pages
Submitted on January, 23rd, 2020
The date of publication is printed in cover of repository
version published in Kobe University Repository Kernel.