



Design of a deep learning-based driver posture monitoring system

by
Kemin Shi

Supervisor: Muhammad
Second Marker:

Project submitted as part of the requirements for the
award of MSC-DAI Data and Artificial Intelligence

September 2022

DECLARATION

I, Kemin Shi declare that I am the sole author of this Project; that all references cited have been consulted; that I have conducted all work of which this is a record, and that the finished work lies within the prescribed word limits.

This work has not previously been accepted as part of any other degree submission.

Signed : Kemin Shi

Date : 2022/9/7

FORM OF CONSENT

I Kemin Shi hereby consent that my Project, submitted in candidature for the MSC-DAI Data and Artificial Intelligence degree, if successful, may be made available for inter-library loan or photocopying (subject to the law of copyright), and that the title and abstract may be made available to outside organisations.

Signed : Kemin Shi

Date : 2022/9/7

Abstract – With the changes in autonomous driving technology, it will largely liberate the driver, but the driver still plays a key role in the field of traffic safety. For quite a long time in the future, the safe driving of vehicles still depends on the close cooperation between human and system, so a monitoring system is needed that can accurately estimate the current posture of the human body in real time and alert the driver when he is in an incorrect posture, correct the driving posture and return the driver's attention to the road, thus ensuring a safe and reasonable human-computer interaction process.

The implementation of the entire monitoring system is essentially a classification of driving behavior. For the classification task alone, there are many kinds of powerful classification networks for deep learning, but the test results of all these networks after fine-tuning on distracted datasets are unsatisfactory, and the classification performance of each type is severely uneven. This is due to the fact that the network only learns local features in the image and does not learn the driver's human pose, which when extended to actual usage conditions can produce misclassification due to the complex driving environment, and the driver pose monitoring system relies on the pose classification information output by the network, which can seriously affect the robustness of the driver supervision system. In this paper, we propose how to design accurate, reliable and applicable classification networks using the estimated information of human posture.

This paper first investigates the development process of deep learning techniques in the field of human pose estimation, detailing and comparing the structures of various networks and summarizing key developments in the technology. Then, we propose to build the classification network on the basis of human pose estimation network to improve the accuracy and robustness of classification, and design a network that can fully learn, extract and utilize human pose information. The network uses RMPE as the basic framework, YOLOv3 for human detection, PyraNet for human pose information extraction, and modifies the intermediate supervised meaning according to the compositional model theory, and finally uses Resnet-18 for classification. In the experimental part, this paper builds and trains the model based on open source projects and datasets; finally, we

summarize the advantages and disadvantages of the network based on the experimental results and evaluate the feasibility of practical applications.

Table of Contents

Chapter 1	4
1.1 Introduction	4
1.2 Objectives	4
Chapter 2	6
2.1 Literature Review	6
2.1.1 Literature review of human pose estimation	7
2.2.2 Literature review of skeleton-based human action recognition	15
Chapter 3 Network	20
3.1 YOLOv3 section	20
3.1.1 The basic structure of YOLO	20
3.1.2 YOLOv3 improvements	23
3.2 SPPE section	26
3.2.1 Bounding box positioning false positive problem	27
3.2.2 Data enhancement of SSTN	30
3.2.3 Stacked Hourglass Network	31
3.2.4 Design of Hourglass Network	32
3.2.5 Design of the residual module	33
3.2.6 Design of Pyramid Hourglass Network	35
3.2.7 Design of intermediate supervision position	39
3.2.8 Elimination of redundant poses	42
3.3 Driving behavior monitoring section	45
3.3.1 SPPE Network Design with Fusion Composition Model	45

3.3.2 General expression of the composition model	47
3.3.3 Convolutional representation of the constituent model	49
3.3.4 Skeleton-based representation of human body parts	51
3.3.5 Stacked Hourglass Network Design with Fusion Composition Model	52
3.3.6 Defects of direct classification network.....	53
3.3.7 Classification network design based on human posture information	54
Chapter 4 Experiments	55
4.1 Data set and evaluation criteria	55
4.1.1 COCO human key point dataset.....	55
4.1.2 State-Fram distracted driving dataset.....	59
4.2 Training environment and hyperparameter setting.....	59
4.3 Training results.....	60
4.3.1 SPPE training results	60
4.3.2 Distracted Driving Classification Network Training Results	62
Chapter 5 Conclusion	67
References	69

Chapter 1

1.1 Introduction

With the development of autonomous driving technology, the driver will be freed up to engage in more non-driving related activities while the vehicle is in motion, resulting in new postures that are important entry points for optimizing traditional passive safety systems. And according to the American Automobile Association (AAA), the driving of self-driving vehicles will continue to rely on close cooperation between humans and systems for a considerable period of time in the future, so a system that can accurately estimate the current posture of the human body in real time and alert the driver when he or she is in an incorrect posture is needed to ensure a safe and reasonable human-computer interaction process [1]. The objectives of this paper are: i) to study the development process of deep learning techniques in the field of human posture estimation, detailing and comparing the structure of various networks and summarizing key developments in the technology. Second, to build and train models based on open source projects and datasets to achieve effective camera-based human posture estimation in real time and to develop corresponding monitoring systems to remind drivers to maintain proper contact with the driving system.

1.2 Objectives

(1) With the change of autonomous driving technology, it will largely liberate the driver, but the driver still plays a key role in the field of traffic safety. Autonomous driving can be divided into L1-L5 according to the level, L1 and L2 are currently realized and also used for general commercialization, and L3 is the current direction actively promoted by car companies and Internet companies. It is believed that in the near future, driving most situations will be taken over by intelligent systems, allowing drivers to devote themselves to activities not related to driving, such as working, eating, watching movies, reading, etc. But the L3 level is not perfect, and in many cases the system is unable to make timely and reliable judgments, so the driver needs to realize the need to take over as early as possible, make effective judgments, and intervene in a timely manner to maneuver the vehicle to ensure

driving safety. The driver needs a process from distracted state to intervene driving, and the driver posture monitoring system can ensure the process is comfortable and safe. The system uses human posture estimation technology to locate key parts of the driver's body and use them to infer their activity status and determine their ability to take over, and combined with real-time road condition feedback, it can promptly remind the driver to adjust his posture.

(2) The core of the driver posture monitoring system is the human posture estimation technology. In the past, contact sensors were often needed to locate key points on the human body in real time and accurately, but with the rapid development of computers and the gradual maturation of deep learning technology, it is possible to use monocular cameras to monitor the driver's posture in real time and steadily. As a major subfield of computer vision, human pose estimation requires consideration of background occlusion, illumination, clothing shape and texture, skin color and image defects, and the need to make the computer learn the correct human pose structure, but at the same time the amount of human pose database is small compared to other similar fields, making it one of the most complex tasks. In order to improve the performance of human pose estimation, besides expanding the database and deepening the network to improve the number of training, how to fully utilize the human structure characteristics and network learning characteristics to innovate the algorithm process and network structure has become a topic of concern for engineers in related fields. From 2013 to now, various algorithm structures have emerged, and the performance of human pose estimation has been significantly improved. In this paper, we will study the development process of deep learning technology in the field of human posture estimation, introduce and compare various network structures in detail, and summarize the key developments in the technology.

(3) Most of the current networks for driving posture classification tasks use direct classification but suffer from the drawback that the classification performance is unstable and cannot be generalized to more general scenarios. This is due to the fact that the network only learns the local features of each image in the training set and does not learn the human pose of the driver. Good classification can

only be performed when the training and validation sets have close images and distinct features. However, when extended to actual usage conditions, misclassification can occur due to the complex driving environment, and the driver pose monitoring system relies on the pose classification information output from the network, which can seriously affect the robustness of the driver supervision system. This paper proposes how to design an accurate, reliable and applicable classification network using the estimated information of human posture.

Chapter 2

2.1 Literature Review

Human action recognition can be divided into two main schemes: one recognizes human action directly by video, and the other first estimates the joint position of the human body in the video and judges the action class from the joint position.

Direct recognition generally uses convolutional neural networks (CNN), and since the convolution of 2D cannot model temporal information, Simonyan et al [1] proposed a dual-stream network. The dual-stream network consists of a spatial network and a temporal network. The spatial network takes a single frame RGB image as input for modeling appearance features, and the temporal network takes a stacked optical flow image as input for modeling motion features. Besides, the actions in video sequences are three-dimensional spatio-temporal signals, and modeling the spatio-temporal signals using 3D convolution provides a natural and effective way for action recognition. Ji et al [2] first proposed to extract spatio-temporal features of videos using 3D convolution. This architecture obtains grayscale, gradient and optical flow channel information from between adjacent frames in the video, then performs convolution and down-sampling operations for each channel separately, and finally combines the information from all channels to obtain the final feature representation.

Direct recognition includes the interaction between the human body and the external environment during the training process, so the description of the

recognition action is more accurate and includes specific interaction objects [3], such as "picking up the phone" and "folding the clothes". However, due to the dependence on the external environment, the model trained from one environment needs to re-understand this part of the environment when migrating to another environment that has changed significantly [4], and therefore requires a large number of datasets for migration training.

As a comparison, the scheme to estimate the joint positions of the human body in the video and determine the action class from the joint positions has several features: the human pose estimation part has a large number of publicly available datasets due to being a popular research area [5]. The trained model has a strong generalization performance [6] and can be deployed directly without making adjustments in most cases; the action category is determined by joint position although there is no environmental information, and the actions identified by joint information alone are not precise and specific enough and do not contain specific interaction objects, such as "hold", "fold", etc. "folding", etc., but the absence of environmental noise allows the action recognition algorithm to focus on the powerful features of the action [7] and the model is generalizable.

2.1.1 Literature review of human pose estimation

The traditional approach to human pose estimation is to use image features to composite the body structure. It uses a collection of "parts" linked by "springs" to represent the human body. The "parts" are the key points of the human body obtained in the image using feature matching (e.g., Haar features), the "springs" represent the deformable links of the key points of the human body, and the "springs" between different "parts" are The "springs" between different "parts" have different "stiffness". This approach transforms the human pose estimation problem into a structured prediction task, and the accuracy depends on the establishment of the pose model rather than on the richness of the data set. Therefore, the classical approach focuses on the representational capability of the enriched model.

Among the classical approaches, the Deformable part models [8] created by Yang and Ramanan are highly representative. This model uses a mixture of "part"

models to represent complex keypoint relationships, i.e., each "part" model has both global and local features. These features are matched in the image to detect the object. This hybrid approach is better at identifying local keypoints, but less effective at the global scale.

The Kinect device [9], brought by Microsoft, is the most successful case of human pose estimation in commercialization so far. It uses depth information to estimate the position of human joint points, specifically, Kinect projects light with certain structural characteristics onto the subject through a near-infrared laser, which is then captured by a dedicated infrared camera. This light with a certain structure will collect different image phase information depending on the different depth areas of the subject, and then convert this structural change into depth information through the computing unit, so as to obtain the 3D structure, and use the random forest machine learning method to achieve the positioning of joints. However, the method relies on dedicated equipment and the structured light perception has strict requirements for distance. As with other traditional methods, the accuracy of recognition can be drastically reduced or even have severely biased estimation results once the joints overlap with other joints or are obscured by obstacles, or due to environmental factors such as differences in clothing, skin color, light, respective differences in body parts, camera angle, etc.

Non-contact, low-cost, high-accuracy, real-time human pose estimation through computer vision has gradually become a reality in the past few years with the development of deep learning techniques. In 2013, Jain, A. et al. first published a paper on human pose estimation using convolutional neural networks [10]. In the paper, the researchers used convolutional neural networks to classify whether a certain joint point was present in each calibration frame and regressed the position coordinates of the joint point relative to the root joint. In practical tests, although it is higher in accuracy than previous classical machine vision methods such as template matching [8], it runs slowly and cannot use global information [11], so it cannot estimate the occluded human joint points, and in the paper the authors clearly show that the method cannot be applied to practical application scenarios. In 2015, Alexander Toshev et al. introduced DeepPose [12] can be considered as the

first major paper that applies deep learning to achieve real-time human pose estimation, the main difference of this paper is that it no longer classifies the presence or absence of human nodes in each area, but returns the position coordinates of all human nodes, so it runs fast and it can estimate the position of nodes even if they exist hidden or occluded. DeepPose is run in multiple stages, the first stage input is the original whole image and the output is the x, y coordinate values of each joint point. The local cropping of the predicted nodes is used as the input to the next stage as a way to optimize the coordinates of the nodes.

Tompson, J et al [13] proposed that regression of the joint point coordinates by images is not reasonable, and in actual labeling one cannot label the positions of the joint points precisely, so learning the coordinate positions introduces errors and increases the complexity of network learning. As an alternative, the paper outputs a heat map to represent the joint point locations, with darker colors representing a higher probability of the joint point being here. This change optimizes the training process of the model and effectively improves the learning performance. As a result, most of the subsequent network outputs from this paper onwards are heat maps.

The authors concluded that the ability of DeepPose to estimate the occluded human joint points lies in the ability of deep learning to understand and apply the global pose of the human body, i.e., the relationship between the various parts of the body [12]. However, at this point the network's understanding of the global pose is shallow and the estimation of the occluded body nodes often shows serious bias phenomenon. So far, how to enhance the network's understanding of the global pose is still a topic of discussion in this research area. The existing approaches can be broadly classified into two categories: "introducing external algorithms to assist the model in understanding the global pose of the human body" and "strengthening the model's own performance in understanding the global pose".

The "introduction of external algorithms to assist the model in understanding the global pose of the human body" involves incorporating algorithms dedicated to global pose derivation into the inference or training process of the deep learning network. Tompson [14] et al. proposed in 2014 to use the position of one joint to

refine the position of another joint. For example, for shoulder position, we know that under normal circumstances the face should be above the shoulder and within a certain range. Based on this idea, the researcher builds Markov Random Field (MRF) model in addition to the regular deep learning network to restrict the position region of human joint point estimation. This method can eliminate some of the unreasonable prediction results.

Wei,S et al [15] proposed CPM (Convolutional Pose Machines), a convolutional neural network structure modified by Pose Machine [16]. The Pose Machine is the first algorithm proposed to use machines to learn tight connections between body parts and to infer unknown parts from known parts. Convolutional neural networks can easily learn certain features with high consistency, such as face, shoulder, etc.; while it has much lower recognition accuracy for motion, occlusion, and similar joints. CPM is a combination of both: the joints that are easier to locate are identified by a deep network and then inferred by the pose machines. CPM is structured in the network in multiple phases, with the input of each phase being heat map output of the previous stage and image features, i.e., the image feature information is used to optimize the heat map using the pose machine algorithm. Another reason for using multiple stages is to expand the receptive domain of the network so that the network can accept more information for learning. The network starts with a small receptive domain and requires constant convolution to expand the receptive domain, but too deep networks can cause problems such as gradient disappearance. Multi-stage proceeding can avoid this problem by using intermediate supervision after each stage. The paper experimentally shows that the network is indeed learning step by step in the phased proceeding and the performance can be improved even further by increasing the number of phases of CPM.

Ning et al [17] proposed a method to guide model training with external knowledge. The authors first encoded the labeled human pose and visual features in the dataset as external knowledge to characterize the constraints and relationships of human pose. During the training process of the network, the output is further encoded and compared with the encoding of the external knowledge in addition to comparing with the true values, which is used to evaluate the reasonableness of the

output human posture. This approach can better guide the network to learn higher information about the interconnection constraints and dependencies of the human body joints rather than the slow purely data-driven approach. XuechengNie et al [18] proposed to use a human semantic segmentation model to guide the training of a human pose estimation model. The human body semantic segmentation model is able to output the attributes of each human part in the picture (e.g., head, body, extremities, etc.). The authors use the output of this model as an input to help the human pose estimation model to efficiently learn the occurrence locations and reasonable constraints of the human joint points.

"Enhancing the performance of the model itself to understand global pose" is to improve the learning performance of the model for both global and local pose, and to build a learning system that infers the global from the local and optimizes the local globally. One of the limitations of convolutional neural networks is that the network extracts features from small to large, and the features learned by shallow networks can accurately locate the position of joints but cannot understand the global connection; the features learned by deep networks can better understand the global relationship between human postures but cannot precisely locate the position, so it is necessary to use these features together to improve the accuracy. Tompson, J et al [19] proposed a network structure divided into a coarse heat map model and a refined heat map model; the coarse heat map model inputs images of multiple resolutions, and the network can capture features of various scales simultaneously during the generation of the heat map to improve the network's ability to capture global information; the refined heat map model is used to compensate for the negative effect of pooling and to recover the loss of spatial accuracy due to multiple resolutions in the coarse heat map model. merging in the coarse heat map model to recover the spatial accuracy lost due to the merging of multiple resolutions. Unlike DeepPose where each part needs to be trained separately, this network is more applicable since it is trained end-to-end in general.

Ning et al [17], in addition to designing the mechanism for external knowledge injection. In terms of network structure, it combines the advantages of initial network (Inception network) and residual module (Resnet model) to design the

initial-residual module (Inception-Resnet) as the basic block and branch line network. This structure is adopted because: as the network proceeds, the acceptance domain becomes larger and larger, and will inevitably include noise, for example, when the human left knee is close to the right knee, the network will produce misjudgment. Therefore, at the beginning of the network, when the acceptance domain is still relatively small, it is necessary to focus not only on the local but also on the dependence between the local and the nearby, and this dependence needs to be captured using the initial-residual module; at the same time, a line is branched outside the main network to use the high-level network acceptance domain for the low-level local situation, so that the low-level network can also have an overall understanding of the acceptance domain.

In 2016, Newell, A. et al [20] proposed a stacked hourglass network model (Stacked Hourglass Networks). This model operates in multiple stages similar to CPM, where the input of each stage is the output of the previous stage, and the spatial connections of the human pose model are gradually learned as the multiple stages proceed. Each phase is a down-sampling followed by an up-sampling process: down-sampling is used to expand the receptive domain of the network and target global information; up-sampling focuses on local information and improves the accuracy of node localization. Down-sampling and up-sampling are equally weighted in the network to capture as much information as possible at each scale and to preserve and deliver this information through the jump connection layer, so the structure is hourglass-like with up-and-down circulation and has excellent performance in human pose estimation tasks. There are many improved structures of the stacked hourglass network model: for example, Chu, X. et al [21] introduced the Multi-Context Attention (MCA) mechanism, where the original network no longer outputs the heat map directly, but outputs three attention intervals, which are used to gradually distinguish the human body region from the environment region and optimize the output of the heat map on this basis; Yang, W. et al. [22] proposed the structure of Learning Feature Pyramids to enhance the inference of the original network at multiple scales and designed a new scheme for the assignment of the initial weights of the network; LipengKe et al [23] proposed the

structure of Multi-Scale Structure-Aware (MSA) network that makes four main improvements to the original network to compensate for the two major problems of over-fitting the network at one scale, making the regression effect of the final model unstable and not having sufficient prior knowledge when calculating the loss function.

KeSun et al [24] proposed the Deep High-Resolution Representation Learning (DHRL) structure. The model maintains the resolution of the original image throughout the process, allowing the model to learn global features to the maximum extent. It also builds multiple parallel lines, each maintaining half the resolution of the previous one, which can better extract local features and synthesize the information extracted through the upper and lower layer linkage. The method is the most computationally intensive, but the network can obtain the most and complete information during the training process. It achieves 92.3% accuracy on the COCO dataset, which can represent the current optimal performance of convolutional neural networks on human pose estimation networks.

The introduction of human pose estimation into video streams brings new challenges: (1) the motion will make some frames blurred; (2) the introduction into the field of real-time computing requires increasing the processing speed per frame, and the computational complexity and computational volume of the network cannot be large. However, it also brings new opportunities, where the video can contain more contextual information, which is beneficial to improve the accuracy of the model.

Arjun Jain et al [25] proposed MoDeep, a system that incorporates motion features into a deep learning framework. motion features are extracted from changes between upper and lower frames, a high-dimensional feature information. Two main types of motion features are discussed in the paper: video frame based motion features and optical flow based motion features. Among them, the video frame based motion features are simple but high dimensional and easy to cause data redundancy. The paper adopts the relatively complex optical flow based motion features and processes the FLIC dataset to obtain the FLIC-motion, which is a data

set containing the original image and image optical flow motion features. By introducing this motion feature, the richness of the data is enhanced on the one hand; on the other hand, the motion feature can assist the network in estimating human nodes even if the original image becomes blurred, which improves the robustness. After this paper, the optical flow-based motion feature data becomes an indispensable part of the video-based human pose estimation task.

The introduction of motion features enables the network to have continuity for the estimation between the upper and lower frames, which can solve the motion blur problem to a certain extent. However, to solve such problems as occlusion, deformation, and prolonged motion blur, it is necessary to further improve the learning capability of the network for frame-to-frame connections. For example, if some joint point is occluded in the current frame, the current position of that joint point can also be inferred from the past and future estimation results, instead of relying only on the connection and constraint relationship between the body parts. YueLuo et al [26] proposed to introduce recurrent neural networks into the video human pose estimation task. Recurrent neural networks are able to input the output of the previous frame as implicit information to the human pose estimation process in the next frame. The authors use a recurrent neural network with LSTM (Long Short Term Memory Model), which has a strong learning ability for long time connections and dependencies, and some difficult human pose estimation tasks that cannot be solved in a single frame can get good estimation results with multiple frames.

The above estimation for human pose is based on a single target, i.e., only one human body in the picture or video. However, the actual tasks often contain plural human bodies, and even difficult recognition of dense crowds. In this case, in addition to the need for the model to accurately identify each human body joint, it also needs to distinguish the attribution of each human body joint. This places a higher demand on the model to understand the connections and constraints among the human body parts.

Hao-Shu Fang et al [27] proposed a framework for multi-target human pose estimation: first, target detection is performed to obtain the location of each human body, and then the region where each human body is located is cropped and input to a single-target human pose estimation model to obtain the joint point location of each human body. This method is straightforward and effective, but when the human bodies overlap, it is easy to have a plural number of human bodies within a single target detection frame, and serious miscalculations occur when performing human pose estimation at this time. Jiefeng Li et al [28] improved this framework by adding a global linkage algorithm to categorize the human body joints within the obtained human detection frame, and re-inferring them after categorization to compensate for the joint point overlap and occlusion to compensate for the recognition loss.

Zhe Cao et al [29] proposed another framework for multi-objective human pose estimation: instead of target detection, the positions of all human joints in the whole picture are estimated directly, and then Part Affinity Fields [30] are used to categorize and connect the individual human joints. The Part Affinity Fields algorithm learns the intrinsic connections and constraints of each part, and it connects all the joints first, and then determines whether each connection belongs to the part, and if so, it establishes the connection, otherwise it rejects the connection. When all the connections of a certain joint point are rejected, this joint point is also rejected. By this method, some cases of misclassification can also be reduced.

2.2.2 Literature review of skeleton-based human action recognition

Early skeleton-based human action recognition used manual features to model the spatial structure and temporal dynamics of the human skeleton and recognized human actions by elaborate classifiers. Wang et al [31] proposed an action integration-based approach to extract spatial locations between skeletons. Hussein et al [32] used covariance matrices to represent the relationships between joint trajectories. Vemulapalli et al [33] used relative rotations and translations between body parts to represent each skeleton. Shott et al [34] proposed a new method for

fast and accurate prediction of the 3D position of human joints from a single depth image without temporal information, mapping the difficult pose estimation problem to a simpler pixel-by-pixel classification problem. Based on this method Xia et al [35] used 3D joint point histograms to represent human pose and modeled the actions by a discrete Hidden Markov Model.

When the action categories are enriched, the traditional method will have blurred classification boundaries, for example, the two actions of tilting backward and raising the head, which are difficult to go through a clear definition by manual features only. Moreover, when the main action is performed, the human body often has accompanying actions, which can bring great interference to the recognition.

In addition to the application of deep learning to image data, the processing of sequence data has also been developing at a high speed in the past few years. Recurrent neural networks (RNNs) are one of the main types of networks dedicated to modeling sequential data, which can establish the dependencies of individual data over long and short time periods. Du et al [36] first applied recurrent neural networks (RNNs) to model time series contextual information by action-based dynamic modeling, but there are also often strong dependencies between joints in the spatial domain, and the spatial dependency structure is also discriminative for action classification. To make fuller use of spatial information, Liu et al [37] proposed a spatio-temporal LSTM-based model that mines both temporal and spatial features and uses a skeleton tree traversal method based on the human joint neighborhood graph to mine the structure of the spatial input sequence of skeleton points. Although LSTM networks are designed to model long-term temporal dependencies, it is difficult to learn high-level features directly from the skeleton because temporal modeling is performed on the original input space.

Convolutional neural network (CNN) models with good high-level information extraction have been used to learn spatio-temporal features from the skeleton. Li et al [38] proposed an end-to-end convolutional co-occurrence feature learning framework that can learn joint co-occurrence features simply and efficiently using the global aggregation capability of convolutional neural networks. Ke et al [39]

proposed an approach using body parts and joints instead of using joint node coordinates by selecting a specific joint as the reference joint, converting the coordinates of other joints into different vector representations depending on their relative reference joints, and calculating distance measures between these vectors for each video frame. Chao Li et al [40] used 3D convolutional neural networks to aggregate spatio-temporal information of joint points, and compared to 2D convolutional neural networks for temporal and spatial information separately, 3D convolution is able to extract both temporal and spatial information, further enhancing the model's ability to learn spatio-temporal features of bones. Although convolutional neural networks can learn high-level spatio-temporal features, they ignore the connection of the human joints themselves in the learning process, i.e., they cannot learn the connection and constraint relationship between the human posture itself and the action. Most of the models using convolutional neural networks have only about 60% recognition accuracy on the NTU RGB+D dataset.

In 2018, Yan et al [41] first proposed the application of graph convolutional neural networks to skeletal action-based recognition and extended graph convolution from the spatial domain to the spatio-temporal domain. Graph neural networks were created to better study non-Euclidean criteria, i.e., graph data. Scarselli et al [42] first proposed to model neural networks on graphs, i.e., graph neural networks (GNN). In a graph neural network, each node updates its own expression based on the mechanism of information propagation, using the surrounding nodes and connected edges as source information, and the model can handle most graph types such as acyclic, cyclic, directed, and undirected graphs. Qi et al [43] used GNN to solve the task of human-object interaction in images and videos. To handle graph data more flexibly, Brnna et al [44] constructed the first graph convolutional neural networks (GCNs), which define the convolution of graphs in spectral space by generalizing CNNs to more general intervals without the role of translation groups, starting from the convolution theorem. Original spectral methods suffer from high time and space complexity, Defferrard et al [45] and Kipf et al [46] parametrized convolution kernels in spectral methods, which greatly reduced the time and space complexity.

The application of graph convolutional neural networks to joint data not only exploits the good high level information extraction capability of convolutional neural networks, but also further learns the connection and constraint relationship between the human pose itself and the action. The model using graph convolutional neural network has at least 80% recognition accuracy on the NTU RGB+D dataset.

Based on the spatio-temporal graph convolution, Kong et al [47] added attention units to further improve the recognition of skeletal actions by spatio-temporal graph convolution. Tang et al [48] proposed a deep progressive reinforcement learning skeletal action recognition method that uses reinforcement learning to select frames in a skeletal sequence, retaining the most informative video frames and removing those with unclear information. Gao et al [49] used a graph regression-based approach to optimize the skeletal map, capturing not only the strong and weak connection relationships between each joint and adjacent joints in the same frame, but also the connection relationships of related joints between the preceding and following frames.

In addition to the joint position information itself, richer human features and motion information can be obtained through preprocessing techniques. Shi et al [50] added skeletal features, including bone length and bone angle of each part, and constructed a dual-stream network, i.e., joint features and skeletal features were input to separate networks for processing, and information was fused at the end, and the fused information was used for motion recognition. Lei et al [51] added motion features containing the velocity and acceleration of each joint, etc. These motion features can respond to the dynamic change information in time on the spatial map, which compensates the information cut in time and space from the original spatio-temporal map convolution.

For action recognition from human keypoint maps, the ideal algorithm should go beyond local keypoint connections and extract multi-scale structural features and long-distance dependencies, because structurally separated keypoints may also have strong correlations. For example, heterotopic hands and feet swing simultaneously when walking, when the model is required to establish and learn

connections between hand and leg articulation points, which are difficult to establish through the human keypoint graph alone. Liu et al [52] expanded the number of connections in the human articulation graph by using absolute distances instead of relative distances, i.e., each articulation point can aggregate information from no more than one absolute distance of articulation points, even if these Li et al [53] used a higher-order operator to enable a node to aggregate the information of more distant connected nodes (the distance depends on the order). Also, an attention mechanism is added so that an articulation point can aggregate information of articulations with similar characteristics to it at an unrestricted distance.

Chapter 3 Network

Driver behavior classification relies on the estimated information of human key points, which in turn relies on the correct human detection. In this chapter, a whole set of networks for driver human pose estimation and driving behavior classification is designed based on the above concepts. The overall framework of the network is based on RMPE [54] and consists of three major components: YOLOv3 [55] for human detection, PyraNet [12] based SPPE for human keypoint estimation, and a residual module-based classification network for classifying driving behavior.

3.1 YOLOv3 section

3.1.1 The basic structure of YOLO

YOLO (YOU ONLY LOOK ONCE) is one of the representative algorithms in deep learning object detection, and compared to its powerful competitors Faster-RCNN [56], SSD [57], and RetainNet [58] and their many improved algorithms, YOLO is the fastest among them, but it has higher localization error and lower recall (which measures the localization effect of all objects) is lower and is suitable for real-time object detection tasks in environments where the complexity is not very high.

Since the initial release of YOLO (YOLOv1 [59]), YOLO has gone through several major version upgrades, YOLOv2 & YOLO9000 [60], YOLOv3 [55], and the network structure has been continuously improved to keep up with the current mainstream network structure, making it more and more powerful. Meanwhile, thanks to the active community, the latest versions of YOLO support various deep learning architectures and hardware configurations. In practical development, researchers can easily deploy YOLO for object detection or as a part of a large deep learning task.

Although there are many versions of YOLO, the core idea of its implementation is to build an end-to-end regression network. YOLO first divides the input image into a grid of size $S \times S$, and each grid cell detects only one object. Specifically, each grid cell generates B bounding boxes and C conditional class probabilities. Each box corresponds to a confidence level that reflects the likelihood of the rectangle containing the object and the accuracy of the bounding rectangle, and the box with a high confidence level represents the region where the object is detected. The

conditional class probability represents the probability that the detected object belongs to a specific class.

Take YOLOv1's paper [59] as an example, as shown below, which divides the input image into a 7×7 size grid. Each grid cell generates 2 bounding boxes and 20 conditional class probabilities. Each bounding box has 5 elements: (x, y, w, h) and a confidence level. Where w and h denote the width and height of the bounding box, and x and y denote the offsets that correct the width and height of the bounding box. Therefore, the dimension of the YOLO output tensor in this paper is $(S, S, B \times 5 + C) = (7, 7, 30)$, where the high confidence predictions will be retained as the final output.

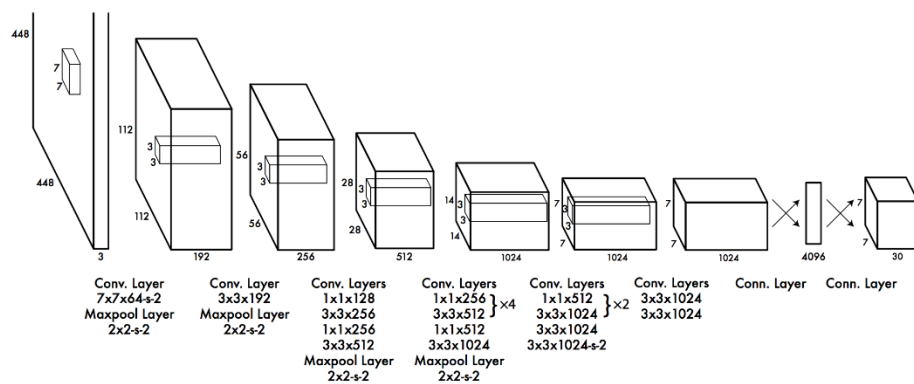


Figure 3.1 YOLOv1 network structure

YOLO's end-to-end is the most distinguishing feature that distinguishes RCNN from requiring multiple processes. This feature makes it fast, real-time, and easy to improve accuracy. Also, YOLO accepts the whole image when predicting the boundaries and gets a richer spatial context, so it gets fewer false positives when detecting background regions. Moreover, YOLO is more general and outperforms other methods when generalized from experimental settings to general usage scenarios.

In subsequent versions (YOLOv2 & YOLO9000 [60]), YOLO absorbed the new network structure and continuously made improvements to make it faster while improving accuracy, with the following main improvements.

One is the addition of batch normalization to the convolutional layer, replacing the dropouts structure. dropouts is designed to prevent overfitting with as little impact on accuracy as possible, but still with a 2%-5% impact, and the replacement of this structure allows YOLO to improve accuracy.

Second, the classification network is trained using high-resolution images. YOLO is actually trained in two stages: the first stage uses a pre-trained classification network, on which the final fully connected layer is replaced with a convolutional layer, and the second stage performs end-to-end training. So the performance of YOLO depends partly on the training process of the classification network. The higher the resolution of the images used for the training of the classification network, the faster the training of the object detection network will be and the accuracy will be improved somewhat (about 4% in the paper).

Third, the concept of anchor box is proposed. the bounding box used in YOLOv1 is arbitrarily specified, resulting in some bounding boxes may be far from the area of the object, which leads to oscillation of the loss during training, which is detrimental to the final performance. In contrast, in real life, bounding boxes are not arbitrary and objects have a certain shape in most cases. So one or several specific bounding boxes, i.e., anchor boxes, can be set based on statistical and computational methods to constrain the shape of bounding boxes for some specific application scenarios, which makes the initial training more stable.

Fourth, the dimensional clustering approach is proposed to find the appropriate number and shape of anchor boxes for a particular environment. For a certain training data, the top K bounding boxes with the best coverage exist. Determining these bounding boxes requires K-average clustering of the training data to obtain the center of mass of the top K clusters. Since the bounding boxes are dealt with, a measure of spatial distance using IoU is required. According to the curve of the paper, the more clusters the classification has, the larger its average IoU is, but its curve rises rapidly to the plateau period, where increasing the number of clusters covering the category will not increase significantly, but will lead to an increase in training complexity. Therefore, the number of clusters at the intersection of the plateau phase and the rise phase is the best, and the number and shape of anchor boxes can be determined based on the number of clusters.

Fifth, the fully connected layer is removed, which allows the network to be trained on more scales. Since the YOLO network downsamples the input 32 times, after removing the fully connected layer, it is only necessary to ensure that the images have the same aspect ratio and are multiples of 32. According to this

requirement, images of different scales can be prepared. During training, after every certain number of iterations, YOLO randomly selects another batch of images of different sizes as input. On the one hand, it acts as a data enhancement; on the other hand, the network is able to make good predictions for different input image sizes. This makes YOLO more usable scenarios: on GPUs with low performance or power limitations, real-time object detection can be performed with lower resolution images without any significant decrease in accuracy. In the paper, YOLO obtains an average accuracy similar to Fast R-CNN at 90 frames/second of real-time video using an image resolution of 288×288 as input.

Sixth is a CNN structure that further simplifies the body of the network, which is called DarkNet. darknet has only 3×3 and 1×1 convolutional kernels. 3×3 convolutional kernels are used to extract image features, and 1×1 convolutional kernels are used to reduce the number of channels. With the help of DarkNet, YOLO further increases the computational speed and improves the accuracy at the same time. The detailed network structure is shown in Fig.

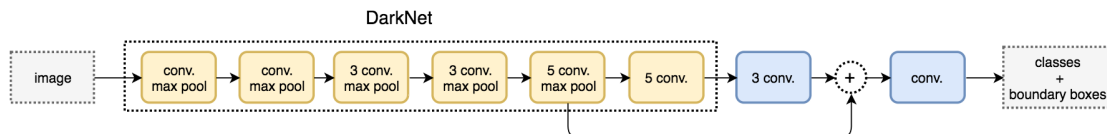


Figure 3.2 DarkNet network structure

Seventh, constant mapping is added to solve the problem of detecting small objects in images. In Darknet, there are 30 layers in the network, and it is often difficult to detect small objects because of the constant downsampling that causes the loss of fine features in the image. To solve this problem, a constant mapping is added to the network, and image features from the upper layers are concatenated to capture low-level image features.

3.1.2 YOLOv3 improvements

Among object detection algorithms, many have emerged in recent years with excellent real-time performance and higher localization accuracy and recall than YOLO, making it imperative for YOLO to achieve greater improvements in localization accuracy without sacrificing too much detection speed. YOLOv3 [55] accomplishes this task brilliantly by keeping up with the current mainstream network architecture and absorbing the best parts of it. In YOLO v3, it incorporates,

for example, residual modules, jump connection layers, and up-sampling, which are now essential elements in most recent algorithms.

YOLOv3 uses Darknet-53 as the base network structure. It has 106 layers, which is twice as many layers as its original structure, Darknet. This is one of the main reasons for the slow operation of YOLO v3 compared to previous versions. Its network structure is shown in the figure.

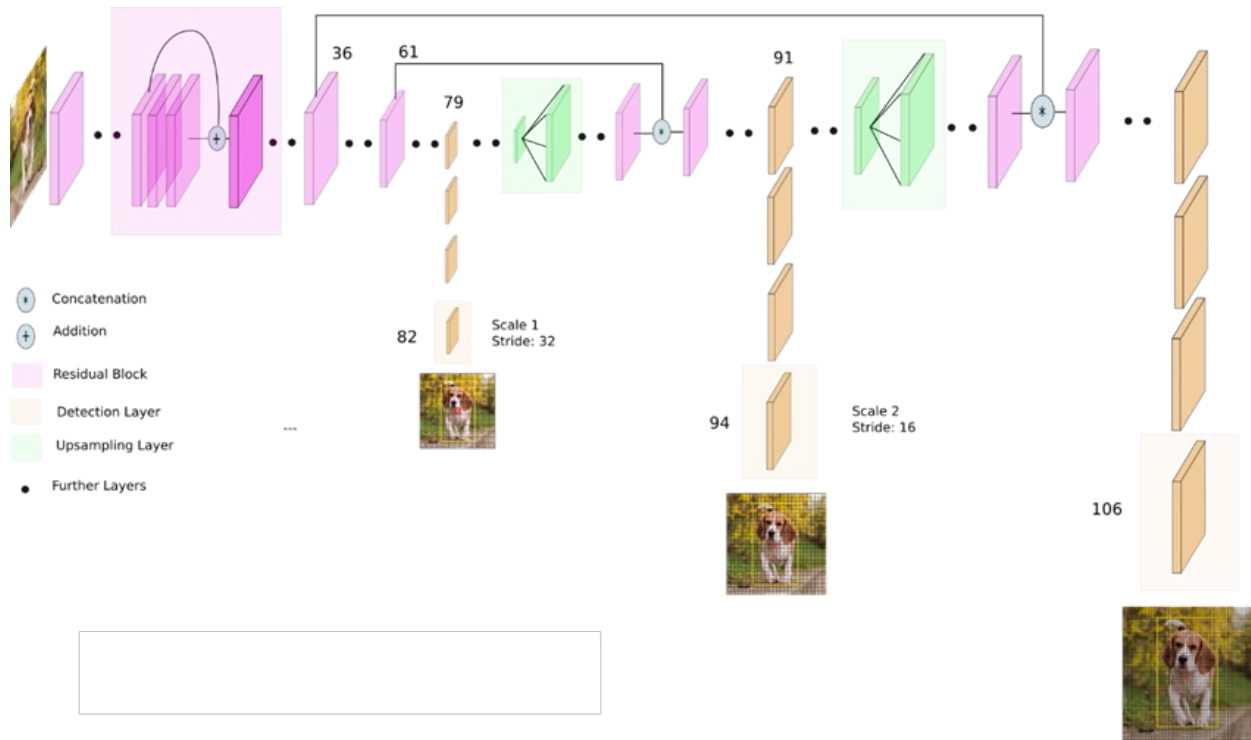


Figure 3.3 Network structure of YOLOv3

The most remarkable feature of YOLOv3 is that it obtains detection results at three different scales by using 1×1 convolution on three image features of different sizes at three different locations. The output of the first detection is at layer 82. In the paper, the size of the input image is 416×416 , and after down-sampling at $32\times$, the size of the image feature at layer 81 will be 13×13 , and after a 1×1 convolution kernel at layer 82, a detection result of $13 \times 13 \times 255$ will be obtained.

The output of the second detection is at layer 94, but the image feature formation used for the 1×1 convolution is more complex and comes from the combination of the image features at layer 79 and layer 61 of the network. Specifically, the network down-samples the original image at a rate of 32 before layer 79, so the image feature at layer 79 is 13×13 . The network down-samples the

original image at a rate of 16 before layer 61, so the image feature size at layer 79 is 26×26 . Before the image features of the two layers are fused, the image feature size at layer 79 needs to be adjusted by performing a 2-fold. The image feature size of layer 79 needs to be adjusted by up-sampling by a factor of 2 to obtain the image feature size of 26×26 so that the image feature can be fused with the image feature of layer 61. After the image features are combined, some 1×1 convolution is performed to make the two layers fuse better. Subsequent 1×1 convolution at layer 94 results in a detection of $26 \times 26 \times 255$.

The output of the third detection is at layer 106 (the last layer) and is similar to the process of the second one. The image features from layer 91 are up-sampled twice by a factor of 2 to obtain a 52×52 image feature, which is subsequently fused with the image features from layer 36. The fusion is followed by a final layer of 1×1 convolution to obtain a $52 \times 52 \times 255$ detection result.

In the original YOLO version, constant mapping is used to preserve image features, which solves the problem of detecting small objects to some extent, but causes interference with the detection of objects of other sizes. In YOLOv3, the up-sampling layer connected to the upper layer helps to preserve the features in the fine details of the image, and the detection output of three different scales makes YOLO get good results for objects of different sizes, with a 13×13 grid scale responsible for detecting larger objects, a 26×26 grid scale for detecting medium objects, and a 52×52 grid scale for detecting smaller objects.

In terms of the number of bounding boxes detected, YOLOv3 is far more than the previous version, about 10 times more than YOLOv2. This is one of the main reasons why the detection speed of YOLOv3 is slower than the previous version.

In addition, YOLOv3 also makes modifications to the loss function. the squared error used in YOLOv2 to calculate the loss is replaced by the cross-entropy error in v3. This allows YOLOv3 to make predictions by more powerful logistic regression during forward propagation, and also allows multi-label classification of objects detected in images, making the results more reasonable.

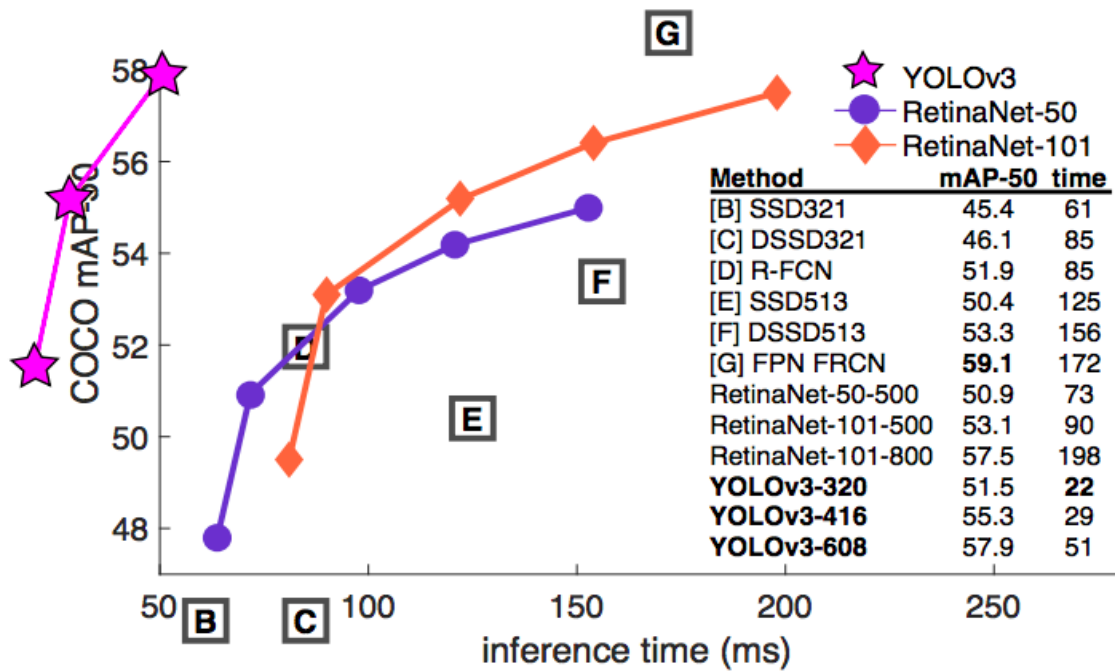


Figure 3.4 Performance comparison of YOLOv3 with each other algorithm

According to the experimental results of each algorithm in the paper [58] for the COCO mAP 50 benchmark test, as shown above, YOLOv3 is comparable to other advanced object detection algorithms such as RetinaNet in terms of localization accuracy, and it still retains extremely fast computing speed, which is much smaller than other algorithms in terms of running time.

However, YOLOv3 does not perform COCO benchmarking because the bounding box it generates is not well aligned with the object's calibration area, making it impossible to pass the COCO benchmarking test for high IoU values (which would be marked as false positives). This is also a shortcoming of YOLOv3 at present.

3.2 SPPE section

After the area of the person in the picture is given by YOLOv3, the accurate estimation of the pixel positions of important key points of the human body will be performed by SPPE (Single Person Pose Elimination). These pixel positions can derive the state of human pose and limb joints, which is an important basis for advanced tasks such as human action recognition and behavior monitoring, and plays an irreplaceable role in fields such as human-computer interaction and driving simulation.

3.2.1 Bounding box positioning false positive problem

Most of the current SPPE networks are built on the COCO dataset for training. In this dataset, the images are normalized so that the center of mass in the foreground is very close to the center of the image and the amount of content is much more than the background, i.e., the person is in the middle of the image and occupies most of the image content. Therefore, when migrating the training results from the COCO dataset to the practical application, it is necessary to use object detection algorithms such as YOLO to frame the area of the person so that it is as close as possible to the state at the time of training, and then hand over to the SPPE network for estimating the location of the key points of the human body.

Since the process involves multiple processes, in order to achieve real-time human pose estimation, this part of human region detection must be fast enough. yolov3 is the best in terms of speed, but it has a more serious problem of false positives for bounding box localization. According to the experimental results in this paper by RMPE [28], the accuracy of human pose estimation is very dependent on the accuracy of human bounding box localization. As shown in the figure below, the red box is the correct human bounding box and the yellow box is the detection bounding box for $\text{IoU} > 0.5$. Comparing the thermal maps obtained from these two inputs to SPPE, it is clear that the red box is much more effective in pose recognition than the yellow box. It is worth noting that in the criteria for object detection in the COCO dataset, if $\text{IoU} > 0.5$, the yellow box is considered as the "correct" detection, but the result of this bounding box localization is not sufficient for subsequent tasks, so it is called a false positive.

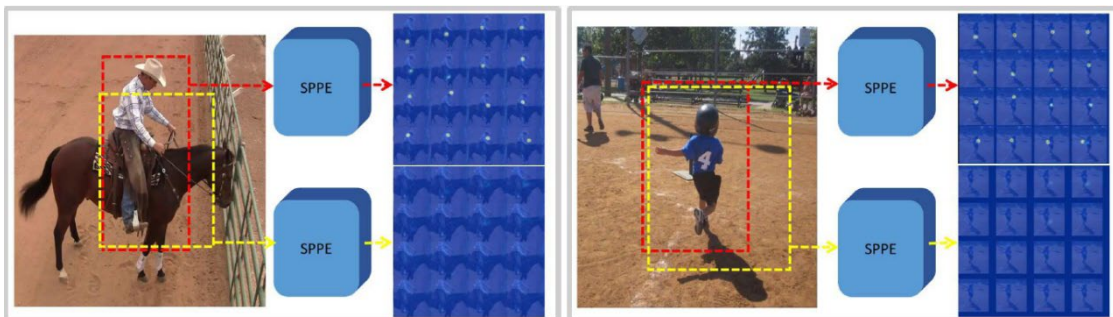


Figure 3.5 Effect of false positive boundary on human posture estimation

The individual bounding box output of YOLOv3 is (w, h, x, y) and a bounding box confidence, where x and y are the corrected values for the bounding box. Since YOLO is a complex multi-task regression network, modifying or retraining YOLOv3 for the accuracy of two parameters is more than worth the cost and will affect the excellent balance of speed and accuracy of YOLOv3.

Following the idea provided in the RMPE paper [28], a spatial transformation network (STN) is attached after YOLOv3, which is connected to SPPE and allows the extraction of high-quality single human regions from inexact bounding boxes. Specifically, this network takes the human region border correction value $\{x_i^s, y_i^s\}$ mapped to $\{x_i^t, y_i^t\}$, and the new correction value can well correct the bounding box to match the correct bounding box and improve the performance of SPPE.

The mathematical expression of this 2D mapping is

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = [\theta_1 \quad \theta_2 \quad \theta_3] \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} g \quad (3.1)$$

of which $\theta_1 \theta_2 \theta_3$ is the $\mathbb{R}^2$ vectors, the $\{x_i^s, y_i^s\}$ and $\{x_i^t, y_i^t\}$ are the bounding box correction values before and after mapping. After SPPE, the output heat map is transformed into the pixel coordinates of the key points of the human body, which are shifted to some extent due to the STN mapping, so it is necessary to correct this relationship by reflecting the projection (SDTN), and the mathematical expression of this reflective projection is

$$\begin{pmatrix} x_i^t \\ y_i^t \end{pmatrix} = [\gamma_1 \quad \gamma_2 \quad \gamma_3] \begin{pmatrix} x_i^s \\ y_i^s \\ 1 \end{pmatrix} \quad (3.2)$$

Since it is a mapping and reflection projection, they and θ are consistent with the following relationship.

$$[\gamma_1 \quad \gamma_2] = [\theta_1 \quad \theta_2]^{-1} \quad (3.3)$$

$$\gamma_3 = -1 \times [\gamma_1 \quad \gamma_2] \theta_3 \quad (3.4)$$

The parameters in the spatial transformation networks STN and SDTN need to be trained and optimized in order to correct the bounding box well. In the actual completion process, SPPE uses a pre-trained model that will be fine-tuned (fine-

tuning) according to the actual output. If STN, SDTN and SPPE are fine-tuned together, both of them are trained for different purposes and on different bases, which will lead to a decrease in the effectiveness of using the original SPPE. The paper RMPE proposes to use parallel SPPE to accomplish this training. After the STN output of the original network, the new human region will not only pass through the normal SPPE network, but will also pass through another parallel SPPE network to get the human pose. The pose will not be reflected by SDTN and restored to the original image, but will be compared with the human pose under the correct human bounding box. From the experiments performed previously, it is known that a large error is obtained when the body region bounding box is a false positive. To improve the speed of training, the parameters of the parallel SPPE network are frozen and this error is only used for the gradient update of the STN. The parallel SPPE is designed so that the training process of the STN does not interfere with the original human pose estimation network, and this parallel line is removed after the training is completed.

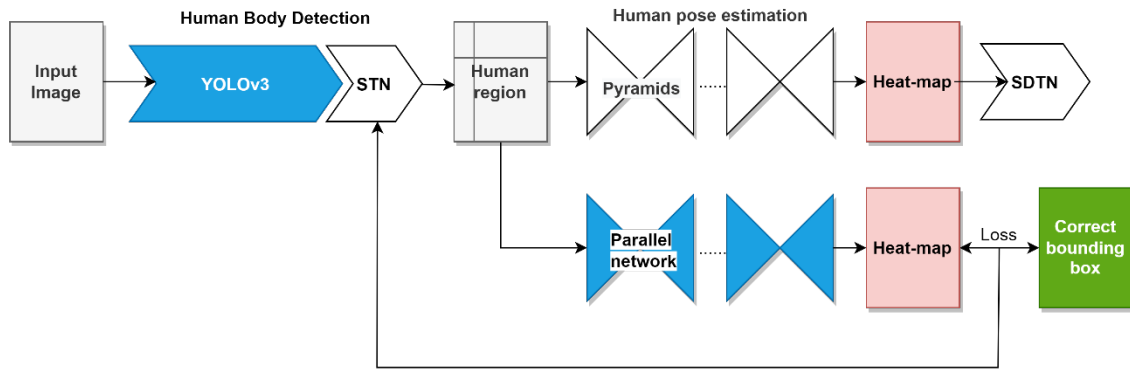


Figure 3.6 Parallel network structure

STN back propagation in θ the gradient equation of $\frac{\partial J(W,b)}{\partial \theta}$: The

$$\frac{\partial J(W,b)}{\partial [\theta_1 \ \theta_2]} = \frac{\partial J(W,b)}{\partial [\gamma_1 \ \gamma_2]} \times \frac{\partial [\gamma_1 \ \gamma_2]}{\partial [\theta_1 \ \theta_2]} + \frac{\partial J(W,b)}{\partial \gamma_3} \times \frac{\partial \gamma_3}{\partial [\gamma_1 \ \gamma_2]} \times \frac{\partial [\gamma_1 \ \gamma_2]}{\partial [\theta_1 \ \theta_2]} \quad (3.5)$$

$$\frac{\partial J(W,b)}{\partial \theta_3} = \frac{\partial J(W,b)}{\partial \gamma_3} \times \frac{\partial \gamma_3}{\partial \theta_3} \quad (3.6)$$

Among them $\frac{\partial[\gamma_1 \ \gamma_2]}{\partial[\theta_1 \ \theta_2]}$, and $\frac{\partial\gamma_3}{\partial\theta_3}$ can be obtained by a simple derivation of (3) and (4). after the STN training is completed, the parameters of the SDTN are automatically updated by reflecting the projection equation.

3.2.2 Data enhancement of SSTN

SSTN is designed to correct the non-standard human detection frames obtained from YOLOv3 in the previous stage, so its training requires a series of standard frames and a large number of non-standard frames corresponding to each standard frame. The simplest way to get non-standard frames is to generate them with YOLOv3, but then there is only one non-standard frame for each human body, which is too small a sample size for deep learning training. Therefore, a suitable non-standard frame generation method is needed to enhance the dataset. Since we already have a series of standard frames and detection frames corresponding to each standard frame, we can calculate the error distribution of the detection frames relative to the standard frames, and use this distribution to generate a large number of non-standard frames for SSTN training.

According to the experimental findings of the paper [28], the error distribution of the human detection frame is related to the specific pose, i.e., the distribution can be expressed as $P(\delta B|P)$, where P denotes the pose and δB is the magnitude of the error of the human detection model with respect to the standard frame. Due to the existence of a large number of human postures, it is not practical to generate distributions for each one. According to the atomic pose s method proposed by Andriluka et al. in their paper [61], the distribution expression can be modified as $P(\delta B|atom(P))$, where $atom(P)$ is denoted as the atomic pose s of the human body, which is a highly generalized expression of the human pose. To obtain the atomic pose s from the annotation, it is first necessary to align all the poses so that the lengths of their torsos are consistent; then the K-proximity algorithm is used to cluster these poses, and finally the centers of each category are calculated. Each center represents an atomic pose. For the poses belonging to a certain atomic pose, the errors in x and y between the human detection frame and the standard frame are first calculated, then normalized, and finally the error distribution curves are generated. Taking several of these poses as examples, the generated distribution curves are shown in the following figures.

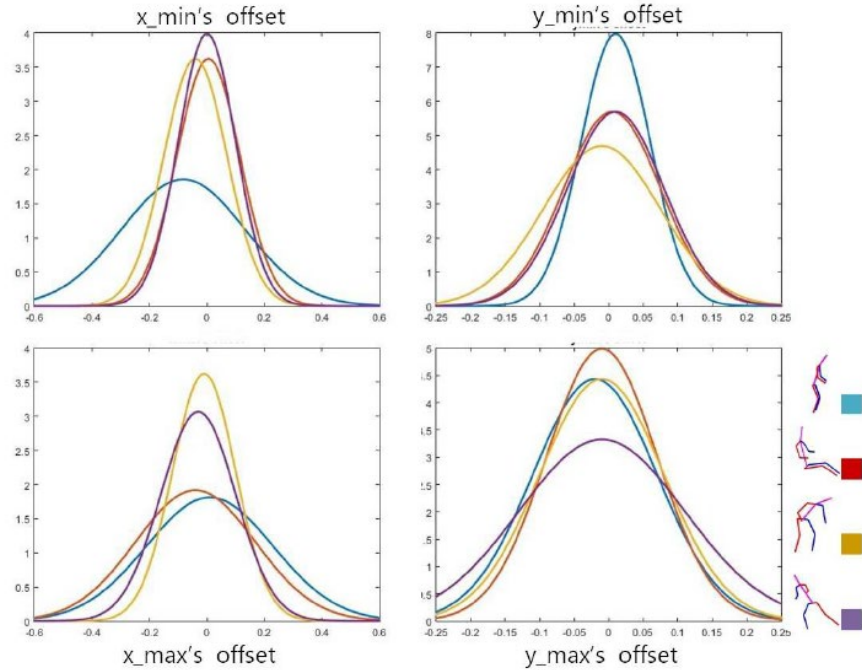


Figure 3.7 Distribution curve of human posture

3.2.3 Stacked Hourglass Network

Human pose estimation is a recognized challenge in computer vision. A good and practical human pose estimation system can achieve the following functions: it can infer from known key points when they are occluded and severely deformed, clearly distinguish when multiple people cross and similar key points overlap, discriminate and optimize by human features when illumination changes and the background of a person blends, and also have good prediction ability on certain rare poses. The implementation of these functions requires a pose estimation system that can locally discriminate features such as faces and hands, and overall has deep learning and coherent understanding of the human model, inference and correction capabilities, and good computing speed and strong robustness.

The advent of convolutional networks allows computer systems to actively extract features from images and learn them, and different network structures can be designed to accomplish different focused learning tasks. In this aspect of human pose estimation, the structure of Stacked Hourglass Networks [10] is designed to achieve the learning goal well. Its input at each stage is the output of the previous stage, and the spatial connections of the human pose model are gradually learned in

the course of multiple stages. As shown in the figure below, each of its stages is a process of down-sampling followed by up-sampling: down-sampling is used to expand the receptive domain of the network to target global information; up-sampling focuses on local information and improves the accuracy of regression. Down-sampling and up-sampling are equally weighted in the network to capture as much information as possible at each scale, and save and deliver this information through the jump connection layer, so the structure is hourglass-like with up-and-down circulation, and has excellent performance in human pose estimation tasks.

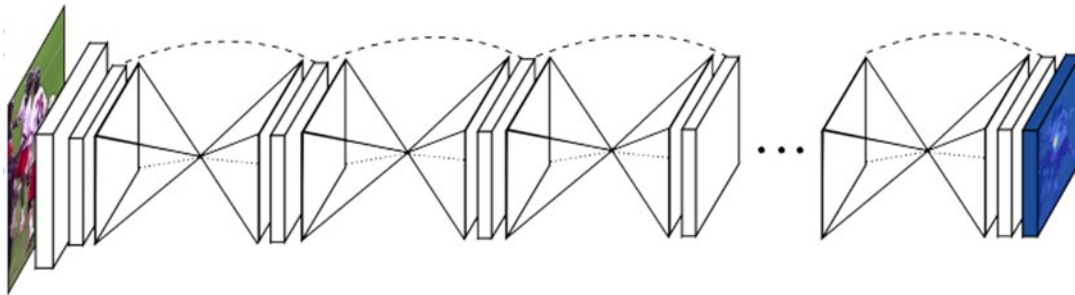


Figure 3.8 Stacked hourglass network structure

3.2.4 Design of Hourglass Network

The hourglass structure is designed to capture information at each scale. This is because local features are needed to capture, for example, the face, hands, etc., while overall information is needed about the orientation of the person, the arrangement of the limbs and the relationship of adjacent joints, and the final pose estimation requires a coherent and in-depth understanding of the whole human model. The hourglass structure is a basic structural unit designed to accomplish this task. Through its stacking, it gives the network the ability to capture features at each scale and combine them together to achieve high precision key point estimation.

The most important implementation of the hourglass structure is the capture and integration of features at multiple scales. The usual approach is to use multiple processes, each with a separate network to process information at each scale, and then combine these features. This approach is inherently independent of the information at each scale, and multiple processes are not conducive to training and analysis. The Hourglass module is designed with a jump connection layer in each

module to store and deliver spatial information at each scale, enabling end-to-end training.

Unlike the previous network structure where the number of down-sampling is much more than up-sampling, the hourglass network adopts a completely symmetric structure, i.e., the number of down-sampling is the same as up-sampling, and the whole network is structured like an hourglass. Its specific structure is related to the number of orders, and the following figure shows the first-order hourglass. Its input first passes through the maximum pooling layer, and then through several residual networks. After reaching the lowest resolution, the network starts to upsample and fuse the image features between different layers by jumping layers. The upsampling and cross-scale feature fusion process was first proposed by Tompson et al. in the paper Joint Training of a Convolutional Network and a Graphical Model [5], and according to the paper, in order to fuse two image features of different resolutions, the lower resolution needs to be up-sampling, where the method used for up-sampling is the nearest neighbor interpolation. By this method spatial features of different scales can be preserved. Based on the description in the paper [10], I draw the first-order hourglass structure as shown below. n-order structure is the replacement of the middlemost residual network with an n-1 order hourglass.

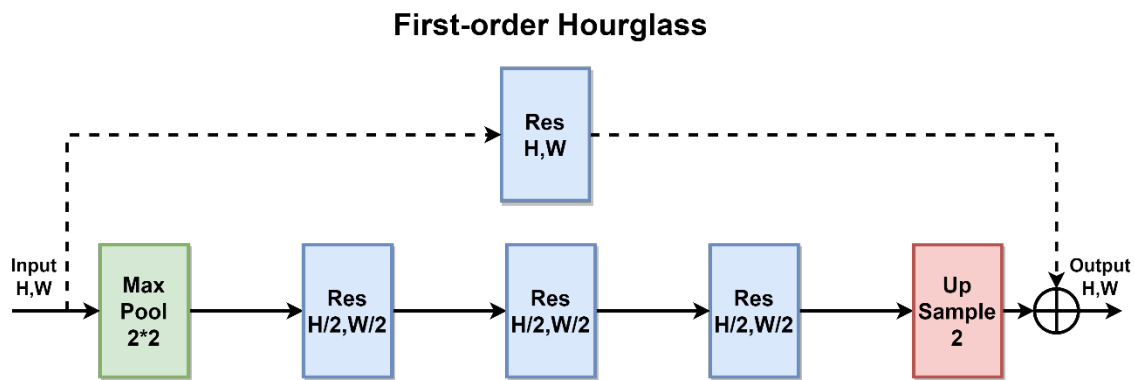


Figure 3.9 First-order hourglass network structure

3.2.5 Design of the residual module

The method of implementing each layer of the Hourglass network is very important. In the paper Stacked Hourglass Networks [10], after extensive

experimental exploration, the authors found that more spatial context can be captured by using consecutive 1x1 convolutional kernels and replacing 5x5 convolutional kernels with two separate 3x3 convolutional kernels. The pattern can be summarized as increasing the depth of the network instead of increasing the size of the convolutional kernels will help the network to learn the human pose more deeply. Based on the above rule, the authors finally chose to use the Residual Modules proposed by He et al [62] as the basic blocks of the hourglass network, which allows the network to be more fully extended in depth and the performance of human pose estimation to be significantly improved.

The residual module is designed to deepen the network as much as possible. In any deep learning task, network depth is critical, but deeper networks are difficult to train. This is because as the network depth increases, the difference in the loss value at the final convergence of the training becomes smaller and smaller. At this point, if we continue to increase the depth of the network, the loss value will increase instead. This phenomenon is called degradation of the network. This phenomenon is not caused by overfitting, so the commonly used regularization or Dropout cannot work.

In the paper, the authors explain from an information-theoretic perspective that in the process of forward propagation, the image information contained in image features decreases layer by layer as the network layers deepen, a phenomenon known as DPI (data processing inequality). That is, in the deeper layers of the network, the network performance degradation occurs because the amount of input information is too small and the image features learned by convolution do not yet have as much information processing loss.

This problem is solved by the advent of the residual module, whose basic expression is

$$y_l = h(x_l) + \mathcal{F}(x_l, W_l) \quad (3.7)$$

$$x_{l+1} = f(y_l) \quad (3.8)$$

in the formula $\mathcal{F}(\cdot)$ is the convolution function, the $f(\cdot)$ is the activation function, and $h(\cdot)$ (is the constant reflection). As shown in the figure below, the core of the residual network is the direct mapping, which is the direct mapping of the l The

image features of the layer are passed to the $l + 1$ layer, which ensures that the $l + 1$ layer contains more image information than the l layer. This structure can avoid the network degradation in deep learning. Since the direct mapping will not be worse than the direct learning, the module can be applied to any layer. In fact, since the publication of this study, the residual module has replaced the direct convolution module as the basic block in large deep learning networks.

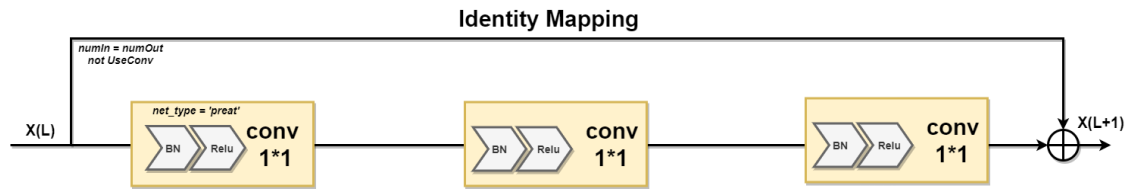


Figure 3.10 Structure diagram of residual module

3.2.6 Design of Pyramid Hourglass Network

Stacked Hourglass Networks [10] This paper has been widely used and improved after introducing the hourglass network structure. It has two main shortcomings: first, although the network captures each scale for learning as much as possible, the human body model is still difficult to be learned well, and the problems of human key point occlusion, blurring, and overlapping cannot be solved well, and the main solution at present is to use external knowledge [17] or external network [18] injection, but considering the actual operation speed and the cockpit environment human upper body occlusion is limited. original network structure has been able to meet the demand better.

The second drawback is that when the angle of the human body or camera changes, when the relative position or angle of the human body to the camera changes, it is easy to have a change in the size of each key point, when the network cannot capture the local features well, i.e., it cannot make a good estimation of the human pose. As shown in the figure below, the paper Learning Feature Pyramids for Human Pose Estimation [12] provides several pose situations that are difficult to be learned by the stacked hourglass network. The proportions of human parts in these images change significantly, such as the hand proportions in the left image and the leg proportions in the middle and right images. The part detectors trained under the

same proportion of parts in the normal pictures are likely to fail in this case. These abnormal proportions will not only affect the detection of human parts in the early stage of the hourglass network, but also affect the learning and recognition of the overall human model in the later stage. Since the human body changes frequently relative to the camera in the actual driving environment, the problem has a strong solution value.



Figure 3.11 Local feature scaling problem

In other vision tasks, such as object detection and semantic recognition, the feature pyramid structure [25, 26] has been widely used for its strong ability to handle the size variation of local features. In human pose recognition, the paper [12] proposes a multi-scale feature pyramid structure, namely the pyramid residual module (PRM), which is used to replace the residual module in part of the higher-order hourglass. Given the input features, the multi-branch network in the PRM will obtain features at different scales by down-sampling at different ratios of sampling rates, allowing the convolution to learn input features at different scales as a way to enhance the robustness of the hourglass network to changes in the scale of human body parts.

Specifically, making $\mathbf{x}^{(l)}$ and $\mathbf{W}^{(l)}$ be the input features and convolution parameters of the l th layer, respectively. The PRM can be expressed as follows

$$\mathbf{x}^{(l+1)} = \mathbf{x}^{(l)} + \mathcal{P}(\mathbf{x}^{(l)}; \mathbf{W}^{(l)}) \quad (3.9)$$

where $\mathbf{x}^{(l)}$ denotes the direct mapping. $\mathcal{P}(\mathbf{x}^{(l)}; \mathbf{W}^{(l)})$ denotes the feature pyramid, which has a specific expansion as

$$\mathcal{P}(\mathbf{x}^{(l)}; \mathbf{W}^{(l)}) = g\left(\sum_{c=1}^C f_c(\mathbf{x}^{(l)}; \mathbf{w}_{f_c}^{(l)}); \mathbf{w}_g^{(l)}\right) + f_0(\mathbf{x}^{(l)}; \mathbf{w}_{f_0}^{(l)}) \quad (3.10)$$

As shown in the figure below, the equation $f_0(\cdot)$ denotes the middle convolution line, and $w_{f_0}^{(l)}$ is the corresponding convolution parameter. $g(\cdot)$ denotes the multi-scale feature pyramid line below, and C denotes the number of pyramids, and $f_c(\cdot)$ is the convolution formula in the c th pyramid. $\{\mathbf{w}_{f_c}^{(l)}, \mathbf{w}_g^{(l)}\}_{c=0}^C$ is a set of convolution parameters. To reduce the computational complexity, the input features are first convolved by 1×1 to reduce the feature dimension, and then input to the maximum pooling layer with different ratios to obtain different scales of features. These features with different ratios are learned by 3×3 convolution. The convolved features will be upsampled to the same resolution and fused together by $g(\cdot)$ further convolution.

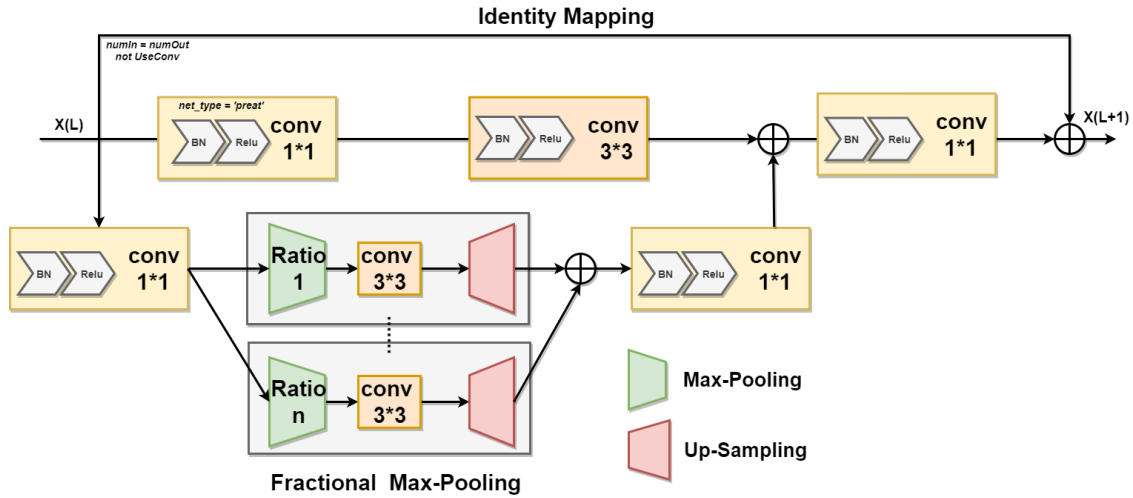


Figure 3.12 PRM network structure

The ratio of the maximum pooling needs to be selected according to a certain formula, and the ratio of the first C level pyramid's ratio is calculated as follows.

$$s_c = 2^{-M \frac{c}{C}}, \quad c = 0, \dots, C, M \geq 1 \quad (3.11)$$

where $s_c \in [2^{-M}, 1]$, represents the ratio of image features relative to the input image. For example, when $c = 0$ when the output has the same resolution as the input. When $M = 1$ when $c = C$, the output is half the resolution of the input. In the specific experiment of the paper [12], M is set to 1 and C is set to 4, i.e., there are 4 scales of the feature pyramid structure, and the minimum scale output is half of the input resolution.

The residual module in the hourglass network is not completely replaced by the PRM, the first-order network will not use the PRM structure, the second-order network only uses the PRM in the jump connection layer, and the higher-order hourglass network (third-order that is, above), only then the PRM completely replaces the residual module, the following figure shows the specific structure of the second-order and third-order hourglass network.

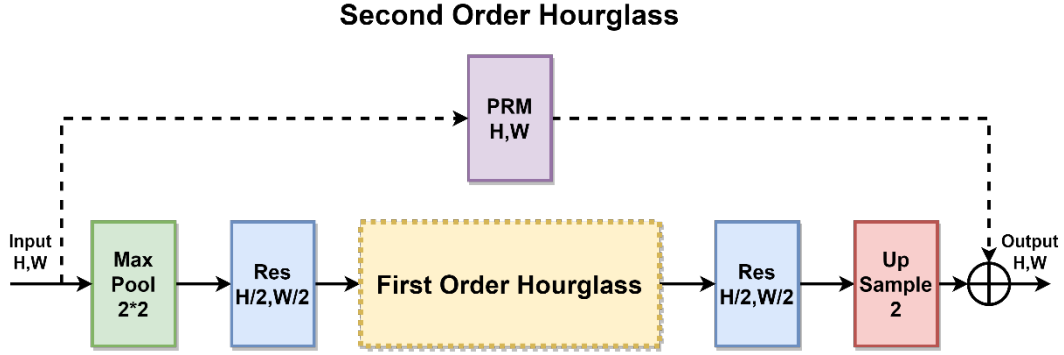


Figure 3.13 Second-order hourglass network structure

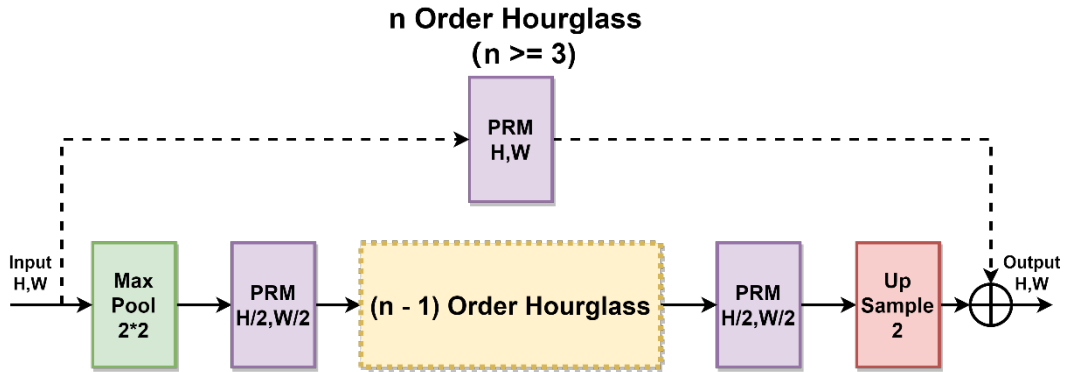


Figure 3.14 Third-order and higher network structure

The direct mapping of the residual module sums the outputs directly, according to the generic expression for the residual module, which yields

$$\text{Var}[\mathbf{x}^{(l+1)}] = \text{Var}[\mathbf{x}^{(l)}] + \text{Var}[\mathcal{F}(\mathbf{x}^{(l+1)}; \mathbf{W}^{(l)})] > \text{Var}[\mathbf{x}^{(l)}] \quad (3.12)$$

The formula shows that after one layer of residual modules, the output has almost twice the variance of the input. In very deep networks, the variance accumulates. Even if the initial variance is small, it can increase rapidly, leading to optimization difficulties and network performance degradation. So it is not desirable to use direct mapping exclusively in residual modules or PRM structures.

The paper [12] provides a simple but effective method to suppress the variance generation by replacing the direct mapping in it with BN-ReLU-Conv module every certain module interval. As shown in the figure below, the variance of the replaced hourglass network (red columns) is much smaller than that of the original version (yellow columns).

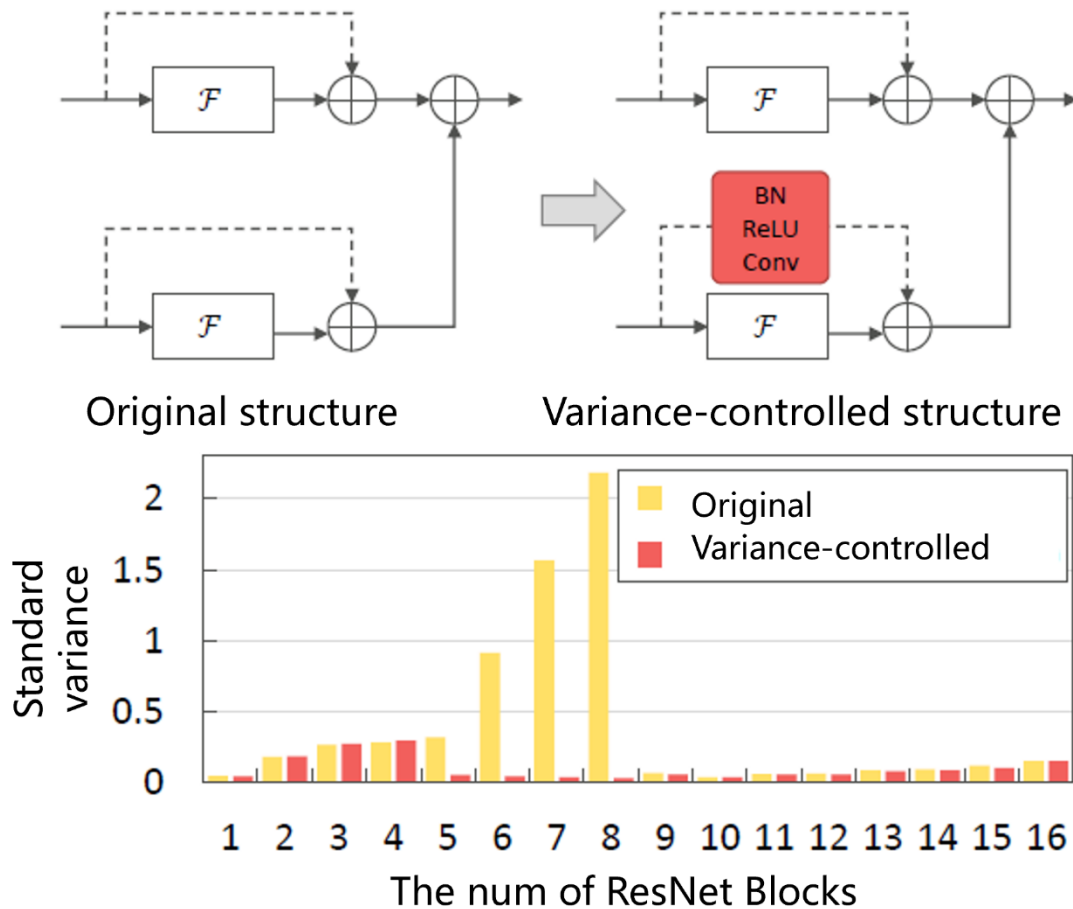


Figure 3.15 Structural design of hourglass with controlled variance

3.2.7 Design of intermediate supervision position

The stacked hourglass network [10] uses the hourglass network as the basic block. Its input of each hourglass comes from the output of the previous hourglass, and its output is used as the input of the next hourglass. During the multi-stage non-stop downsampling and upsampling, the hourglass gradually learns the spatial connections of the human posture model.

The network controls the number of layers of each hourglass and adopts the structure of stacked hourglasses to deepen the network instead of deepening one hourglass for two main reasons: First, according to the changing characteristics of image features in the hourglass, higher-order image features only appear at low resolution (after downsampling). To preserve the higher-order image features, the output needs to be set after downsampling, but this will lose important local image features. That is, although the network learns local and overall information during downsampling and upsampling, it cannot retain the output at the same time. Using the structure of multiple hourglasses, the information learned by the previous hourglass will be learned by the next hourglass, which can retain feature information better.

Second, the problem of gradient disappearance or gradient explosion occurs during network training. Learning the human pose model needs to cover the receptive domain of the full image element, which will put higher demands on the depth of the network due to the limitation of the convolutional kernel size. The overly deep network will have the concatenation effect in gradient backpropagation, which leads to unstable network weight update, i.e., the problem of gradient clearing or gradient explosion at a certain point in backpropagation, making the network unable to continue learning. Using the structure of multiple hourglasses, the output of each hourglass has the same dimension as the final output heat map, which allows for intermediate supervision, i.e., using the heat map generated by each hourglass with the calibration value to calculate the loss through the loss function to avoid gradient disappearance or gradient explosion during training.

In the intermediate supervision, the location of the loss calculation needs to consider the retention of high and low order features of the image mentioned in the first point above. To enable the network to make an accurate estimate of the human pose, then the intermediate supervision position of the hourglass should be performed in a global range, but setting the intermediate supervision position after downsampling will lose important local information, so this scheme is not desirable. Due to the adoption of the jump connection layer in the hourglass module, the output heat map has actually integrated local and global information. In the paper [10], the intermediate supervision position of the hourglass network is set at the

last layer of the output heat map, but this output first passes through the 1×1 convolution used to reduce the global information. This intermediate supervisory position is set up to value the role of global information in human pose estimation without losing local information, making full use of the learning performance of the hourglass network structure.

Based on the descriptions in paper [10] and paper [12], I drew the structure of the network as shown in the following figure. The network input 256×256 images, after image feature extraction the network resolution is reduced to 64×64 , followed by 8 stacked hourglass networks to obtain the final key point heat map output. The loss of each hourglass module is computed separately, which enables better re-estimation for later hourglass modules.

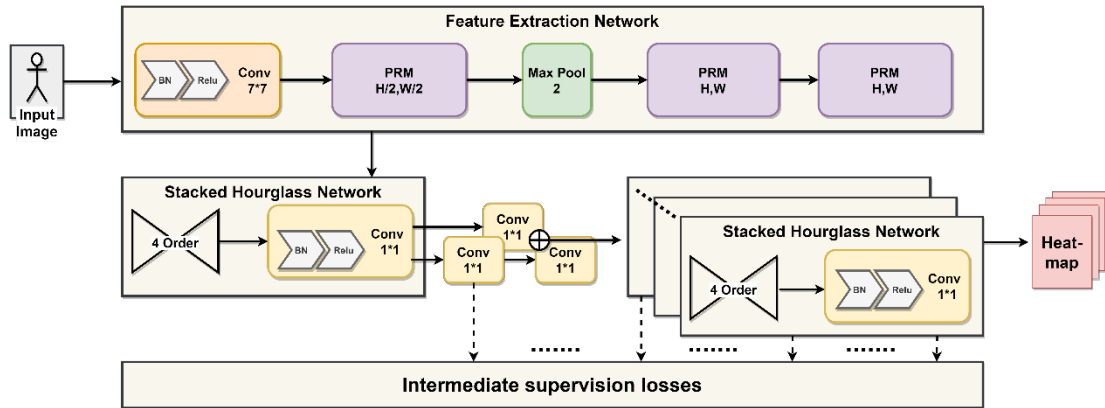


Figure 3.16 Overall network structure and intermediate supervision position

3.2.8 Elimination of redundant poses

SPPE generates a set of heat maps of human pose for each image, and since it is a regression problem, it inevitably generates redundant pose estimates, which need to be removed by a technique called pose non-maximum suppression (NMS) to obtain the correct pixel coordinates of key points. This technique is used in every SPPE algorithm, and the process can be summarized as follows: first, the pose with the highest confidence level among them is selected as the benchmark, and the part of the pose close to the benchmark is eliminated by a certain elimination criterion. Subsequently, the benchmark is reselected in the remaining set of poses and the process is repeated until the redundant poses are eliminated and only the unique pose remains.

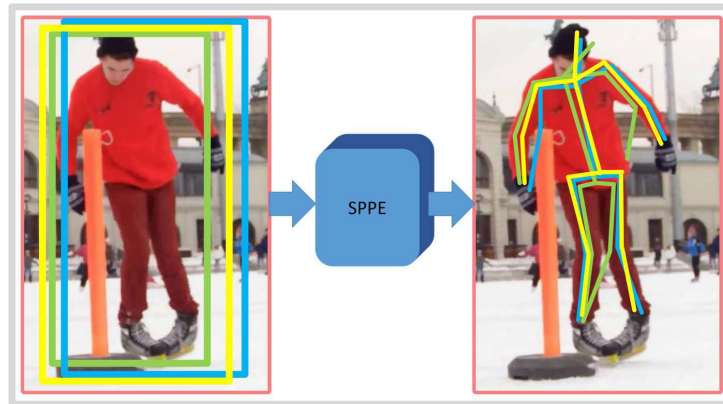


Figure 3.17 Generation of redundant poses

In the framework adopted in this thesis, SPPE does not directly estimate a particular image, but generates a set of human pose estimates for each bounding box given by YOLO. As shown in the figure below, since YOLO recognizes the human body in a large number of bounding boxes, the bounding boxes recognized near the center of the human body will be retained with a high probability, and although a part of the bounding boxes are eliminated by setting a threshold before output, a part of the redundant bounding boxes will inevitably be retained, and the redundant pose estimates generated on these redundant bounding boxes will have a large impact on the final recognition results. The off-the-shelf NMS algorithm in SPPE is not efficient and accurate, and cannot make timely and effective judgments in the face of a large number of redundant pose estimates.

This problem is further amplified in the usage scenario of the paper Crowd-Pose [37], so it proposes a new parametric pose non-maximal suppression method. The method uses a new pose distance representation to compare the similarity between poses, and poses with similarity up to a certain threshold are eliminated from each other. The parameters in this pose distance representation method are continuously optimized with continuous training to achieve the optimal elimination effect.

Specifically, assuming the posture P_i, P_j both have m critical points, labeled as $\{\langle k_i^1, c_i^1 \rangle, \dots, \langle k_i^m, c_i^m \rangle\}$ and $\{\langle k_j^1, c_j^1 \rangle, \dots, \langle k_j^m, c_j^m \rangle\}$ where k and c are the positions and confidence scores of the key points, respectively. $d(P_i, P_j | \Lambda)$ denotes the distance between these two poses, which is used to measure the similarity of the two poses, where Λ is the function $d(\cdot)$ of the set of parameters, and defines a threshold value η , and the elimination criterion can be written as

$$f(P_i, P_j | \Lambda, \eta) = \mathbb{1}[d(P_i, P_j | \Lambda, \lambda) \leq \eta] \quad (3.13)$$

If $d(\cdot)$ is less than or equal to η , then $f(\cdot)$ the output of should be 1, which indicates that with the pose P_j as a reference, the P_i is redundant and should be eliminated.

Distance function $d_{pose}(P_i, P_j)$ of consists of two parts: one is the number of key points matched between two poses, and the other is the spatial distance between two key points, as follows.

$$K_{Sim}(P_i, P_j | \sigma_1) = \begin{cases} \sum_n \tanh \frac{c_i^n}{\sigma_1} \cdot \tanh \frac{c_j^n}{\sigma_1}, & \text{if } k_j^n \text{ is within } \mathcal{B}(k_i^n) \\ 0 & \text{otherwise} \end{cases} \quad (3.14)$$

which $\mathcal{B}(k_i^n)$ is k_i is the center of the box, and $\mathcal{B}(k_i^n)$ Each dimension of the original box is B_i is 1/10 of the original box. $\tanh$ The operation can filter the low confidence poses. When the corresponding keypoints in both poses have high confidence, the output will be close to 1, i.e. the set of keypoints match. This formula calculates the number of keypoints that match the two poses.

$$H_{Sim}(P_i, P_j | \sigma_2) = \sum_n \exp \left[-\frac{(k_i^n - k_j^n)^2}{\sigma_2} \right] \quad (3.15)$$

The coordinates of the two attitude key points are subtracted and calculated with $\exp$. The spatial distance between the two attitude key points can be obtained after all the corresponding key points are calculated as above.

The final attitude distance function is obtained by combining the above two partial formulas.

$$d(P_i, P_j | \Lambda) = K_{\text{Sim}}(P_i, P_j | \sigma_1) + \lambda H_{\text{Sim}}(P_i, P_j | \sigma_2) \quad (3.16)$$

where the weights λ balance the relationship between the two posture distances, the $\{\sigma_1, \sigma_2\}$ is a set of distance parameters. Their specific values are not set manually, but are determined in a data-driven way to be continuously updated during the training iterations to make them optimal.

3.3 Driving behavior monitoring section

Driving a car is a complex task. The driver needs to make different behaviors in the face of different driving conditions to maintain or change the driving status of the car and make the car drive according to the traffic norms. However, in reality, distracted driving is inevitable, but the hazards caused are enormous. According to a report given by the National Highway Traffic Safety Administration (NHTSA), about 36,120 people died in motor vehicle crashes in 2019, 14% of which were due to distracted driving [1]. Distracted driving is any activity that takes the driver's attention away from the road and has two main characteristics: firstly, taking the eyes off the road; and secondly, wrong posture, i.e. the hands also fail to take over the steering wheel in time. Many countries now have laws in place that prohibit drivers from messaging, talking on cell phones and other distracted driving behaviors while driving, and are also accelerating the development of driving behavior monitoring systems to eliminate traffic accidents from the ground up.

With the current rapid development of computer vision and deep learning technologies, human posture estimation technology has become an important and fundamental part of a huge driving behavior supervision system. The system consists of camera incoming images, which are processed by neural network to get the driver's posture, and different levels of distracted driving classification based on the posture information, and human-computer interaction according to the reminder designed by the classification results to avoid traffic accidents.

3.3.1 SPPE Network Design with Fusion Composition Model

The classification of distracted driving needs to be built on the pose information of the overall human model. According to the characteristics of the stacked hourglass network, the acceptance domain of the network expands continuously during the training process, from learning local features of human parts to learning the overall human model gradually. Since the final output is a heat map of key points, the network needs to go through several hourglasses to optimize the judgment of local features after learning the overall human model. This property of stacked hourglass networks is well demonstrated by the network attention model generated in the paper [21]. To summarize, the stacked hourglass network has a

local inference of the whole and then the optimization of the local from the whole in the process of estimating the key points of the human body. As shown in the figure below, for example, the key points of human hip, knee, and ankle are obtained in the initial hourglass, the middle hourglass will gradually generate the upper and lower extremities of the leg, and then the leg, and the later hourglass will optimize the key points based on the whole of the leg.

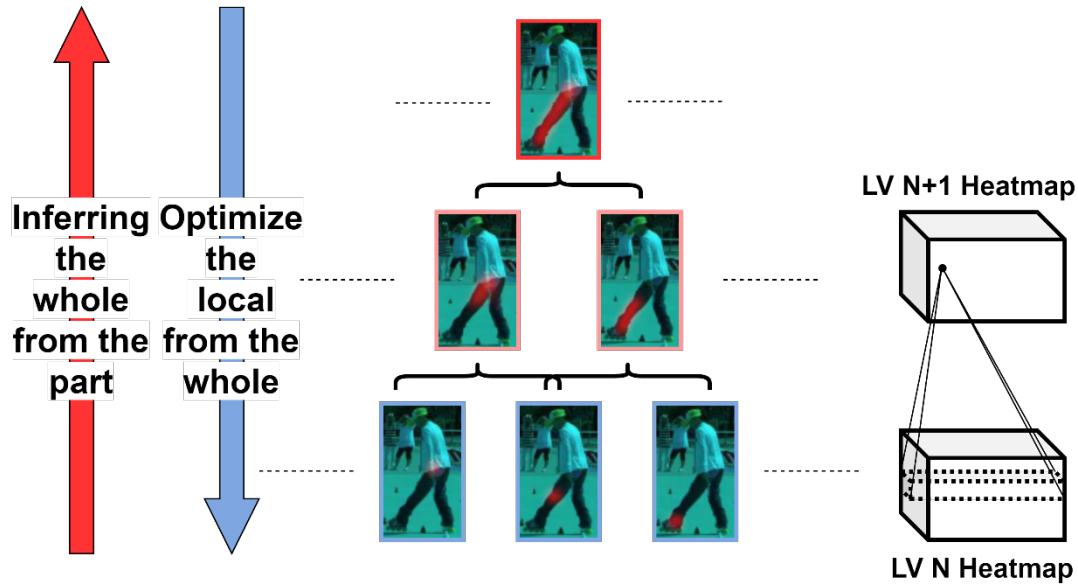


Figure 3.18 The process of learning human posture in the network

In the hourglass network as well as in several improved networks, the actual training process has an intermediate supervision session, but the session does not take advantage of the properties of the hourglass network mentioned above. As mentioned in the previous section, the intermediate supervision is to compare the output of the heat map of each hourglass with the Gaussian distribution generated by calibrating the key points to obtain the loss value and propagate backwards from that. However, in practice, the output of each hourglass in the stacked hourglass network has different meanings, and treating it uniformly does not positively affect the training of the network.

In the paper Deeply Learned Compositional Models [22], the authors modify the Compositional models algorithm in machine learning to the structure of a convolutional neural network and successfully apply the idea of compositional

models to the task of human posture estimation with good results. The main contribution of the paper can be summarized as setting specific tasks for the intermediate supervision of the network, such as comparing the hourglass output to the body key points in the first stage, the hourglass output to the body parts in the middle stage, and the body key points in the later stage. In this way, it is no longer only the last layer in the network's output that has a practical meaning; each layer's output contains specific information about the human pose, providing a basis and new ideas for designing distracted driving classification networks.

3.3.2 General expression of the composition model

The core of the composition model is the bottom-up feature connection and top-down feature update between hierarchical graphs, which do not share nodes in their general representation and only consider the relationships between levels. It can be expressed in terms of $(\mathcal{V}, \mathcal{E}, \phi^{and}, \phi^{leaf})$. These four parameters are expressed, where $(\mathcal{V}, \mathcal{E})$ denotes the structure of the hierarchical graph, and $(\phi^{and}, \phi^{leaf})$ denotes the potential function of the operation between each node. As shown in the figure below, the $\mathcal{V}$ denotes the nodes in the hierarchical graph, and there are two types of $\mathcal{V} = \mathcal{V}^{and} \cup \mathcal{V}^{leaf}$. $\mathcal{V}^{leaf}$ is the base of the hierarchical graph, i.e., the lowest level part. $\mathcal{V}^{and}$ is the nodes of the next level that are connected and modeled to form the nodes of the previous level. $\mathcal{V}^{and}$ The highest level of the hierarchical diagram, that is, the highest level of the hierarchy, and the final modeling target is called the root node.

Each node has a state variable w_u , and $u \in \mathcal{V}$. For human posture estimation, the $w_u = \{p_u, t_u\}$, where p_u denotes the position of the body part (or key point) and t_u denotes the state of the body part. In the paper [], the use of t_u denotes the orientation, scale and semantic class (straight or bent arm) of the body part.

Ω denotes the set of all state variables in the hierarchical graph, which has the following probability distribution form.

$$p(\Omega|\mathbf{I}) = \frac{1}{Z} \exp\{-E(\Omega, \mathbf{I})\} \quad (3.17)$$

where $\mathbf{I}$ is the input image, and $E(\Omega, \mathbf{I})$ is used to measure the energy magnitude of the set of state variables in the hierarchical graph with respect to the input image,

and Z is the collocation function. The probability formula in $-E(\Omega, \mathbf{I})$ is denoted as $S(\Omega)$, its expansion is given by

$$S(\Omega) \equiv -E(\Omega, \mathbf{I}) = \sum_{u \in \mathcal{V}^{leaf}} \phi_u^{leaf}(w_u, \mathbf{I}) + \sum_{u \in \mathcal{V}^{and}} \sum_{v \in ch(u)} \phi_{u,v}^{and}(w_u, w_v) \quad (3.18)$$

where $ch(u)$ denotes a node u child nodes v of the set. The computational equation consists of two parts that are added together to calculate V^{leaf} and V^{and} the potential functions of the nodes. The first part acts similar to a detector and is used to determine the location of the nodes modeled by V^{leaf} . The modeled hierarchical map appears at the location p_u and state t_u of the nodes. The second part is used to detect child nodes v and their parent nodes u compatibility between them. For the input image I the best set of state variables exists Ω^* such that $S(\Omega)$ maximum.

$S_u(w_u)$ is called the node u of the score map (score map). In the hierarchical map, the root node maximum can be obtained by recursively computing bottom-up $S_u(w_u)$ and the top-down recursive calculation of the optimal non-root node w_u to obtain the optimal set of state variables Ω^* . This recursive computation process is called compositional inference.

In the bottom-up phase, the recursive computation starts from V^{leaf} starts and ends at the root node, and the recursive goal is to obtain $\max_{\Omega} S(\Omega)$: the

$$(Leaf) S_u^\uparrow(w_u) = \phi_u^{leaf}(w_u, \mathbf{I}) \quad (3.19)$$

$$(And) S_u^\uparrow(w_u) = \sum_{v \in ch(u)} \max_{w_v} [\phi_{u,v}^{and}(w_u, w_v) + S_v^\uparrow(w_v)] \quad (3.20)$$

Computational equation (19) sets the initial value of the recursive calculation, i.e. V^{leaf} node u of the score map. Equation (3.20) sets the recursive definition of V^{and} . The node u 's $S_u^\uparrow(w_u)$ is the maximum score graph that can be obtained by combining this node with all its children. Since the V^{and} node's $S_u^\uparrow(w_u)$ are all maximal, so $S(\Omega)$ reaches the maximum.

In human pose modeling, there is a spatial distance limitation between the child nodes and the parent nodes, so the process of recursive parent maximum score graph can be simplified, and it is not necessary to traverse all the child nodes. When $p_v - p_u \notin \mathbb{D}_{uv}$ When, let $\phi_{u,v}^{and}(w_u, w_v) = 0$. $\mathbb{D}_{uv}$ be the distance range between the parent and child nodes, which is set in the paper [] as $[-50, 50] \times [-50, 50]$.

In the top-down phase, the same recursive computation is performed, starting from the root node and V^{leaf} ends, and the recursive goal is to obtain Ω^* : the

$$(\text{Root})w_u^* = \operatorname{argmax}_{w_u} S_u^\downarrow(w_u) \equiv \operatorname{argmax}_{w_u} S_u^\uparrow(w_u) \quad (3.21)$$

$$(\text{Non-root})w_v^* = \operatorname{argmax}_{w_v} S_v^\downarrow(w_v) \equiv \operatorname{argmax}_{w_v} [\phi_{u,v}^{and}(w_u^*, w_v) + S_v^\uparrow(w_v)] \quad (3.22)$$

Equation (21) sets the initial value of the recursive calculation: the root node optimal state variable w_u^* determined by the maximum score graph of the root node obtained recursively by equation (3.20). Equation (3.22) sets the recursive definition: the non-root node optimal state variable w_v^* is determined by the unique parent node u and the maximum score graph obtained by the recursion of equation (3.20). Since the state variables of each node reach the optimal state, the Ω is the set of optimal state variables.

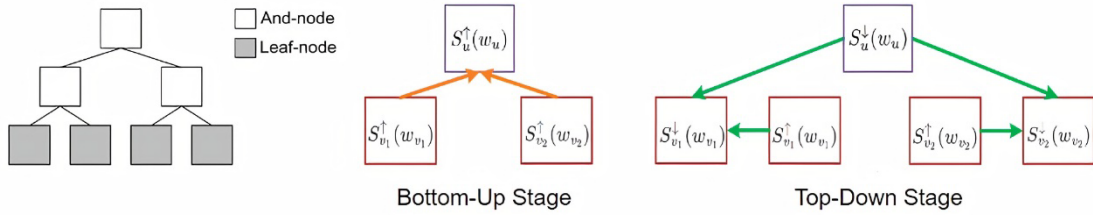


Figure 3.19 The general process of composing a model

3.3.3 Convolutional representation of the constituent model

Intuitively, there are very many similarities between the inference and optimization process of the compositional model and the learning process of the convolutional neural network. The paper [22] provides a further interpretation of the computational equation of the compositional model to find common points between the compositional model and the convolutional layer. The process is to first analogize the computational equation of the compositional model to the pooling layer and then to generalize from the pooling layer to the more general and powerful convolutional layer.

For non-root nodes and non V^{leaf} nodes, its computational equations for the bottom-up and top-down phases can be written as

$$S_u^\uparrow(w_u) = \sum_{v \in \text{ch}(u)} \max_{w_v} [\phi_{u,v}^{and}(w_u, w_v) + S_v^\uparrow(w_v)] \quad (3.23)$$

$$S_v^\downarrow(w_v) = \sum_{w_u} \phi_{u,v}^{and}(w_u, w_v) \bar{S}_u^\downarrow(w_u) + S_v^\uparrow(w_v) \quad (3.24)$$

Equation (23) is the same as equation (20). Another expression of equation (3.24) is used to calculate equation (3.22), where $\bar{S}_u^\downarrow(w_u)$ is $S_v^\downarrow(w_v)$ the threshold expression when $w_u = w_u^*$ when $\bar{S}_u^\downarrow(w_u)$. In the statistical definition, a pool is a way of aggregating information in order to retain task-relevant information while removing irrelevant detailed features, resulting in a more compact information representation and more robustness to noise and interference. In the bottom-up and top-down phases of the recursion that comprise the model, what is actually performed is to combine the score graphs of certain nodes to obtain the state information of other relevant nodes. So the two computational equations in $\sum_{v \in ch(u)}$ $\max_{w_v}$ and $\sum_{w_u}$ can be analogous to the average pooling layer (average-pool) and the maximum pooling layer (maximum-pool).

Pooling layer is a special kind of layer in convolutional neural network, which works in a similar way as convolutional layer, but it works in a fixed mode and does not update parameters with back propagation, that is, it does not have learning function. For the complex constraints and compositional relationships within the human pose, the pooling layer completed in a compositional model or analogous way cannot make effective inference and modeling. The convolutional layer is a more general extension of the pooling layer, which not only supports the aggregation of information, but also has the ability to reason spatially approximatively and learn a more effective way of modeling the human body through a data-driven approach. Specifically, the computational equation of the convolutional layer after replacing the compositional model is

$$S_u^\uparrow(w_u) = \mathbf{c}_u^\uparrow \left(\{S_v^\uparrow(w_v)\}_{v \in ch(u)}; \theta_u^\uparrow \right) \quad (3.25)$$

$$S_v^\downarrow(w_v) = \mathbf{c}_v^\downarrow (S_u^\downarrow(w_u), S_v^\uparrow(w_v); \theta_v^\downarrow) \quad (3.26)$$

Among them, the $\mathbf{c}_u^\uparrow$ and $\mathbf{c}_v^\downarrow$ are the convolution equations, and $\theta_u^\uparrow$ and $\theta_v^\downarrow$ are the parameters of the convolution kernel. It can be seen that the computational equation is actually a general convolutional expression. It follows that all

convolutional neural networks can double as compositional models, but in a data-driven way, will be far more powerful than the general compositional models.

3.3.4 Skeleton-based representation of human body parts

In the process of composing models to model human postures, human parts are usually represented by the location and state of the part. However, this representation has many drawbacks: in the representation of location, a series of assumptions are usually made to simplify, such as the assumption that the generation of key points conforms to the Gaussian distribution and the location of key points is simplified to a local average. However, this scheme is not suitable for the training of human pose modeling networks, because the assumed part does not match the actual situation, and the simplified part cannot describe the complex compositional relationships among the body parts.

In the state representation method, there will be too much space for high-level body part types. Assuming that both the left calf and left thigh have n types in a batch of data sets, the entire left leg can have $O(N^2)$ types, while the composition of the left leg and right leg can have $O(N^4)$ types. This will lead to $S_u^\uparrow(w_u)$ and $S_u^\downarrow(w_u)$ to store a very high dimension of types, which places higher demands on both storage and computation.

The paper [22] proposes a novel skeleton-based representation of body parts, which encodes the type information of the parts such as orientation, scale and shape and embeds them into the score map, and the encoded information is called body skeleton information. Representing body parts with bones can make the score map into a 2D tensor with no type dimension, significantly reducing storage requirements and computational complexity. In training, the output heat map with labeled bones along the part direction generates a Gaussian distribution, as shown below, and the loss is calculated by the loss function and back propagated for training.



Figure 3.20 Skeleton-based representation of human posture

3.3.5 Stacked Hourglass Network Design with Fusion Composition Model

Through the above analysis, the previous stacked hourglass network fusion composition model actually does not need to make changes to the network structure, but only needs to apply the new human body part representation to modify the intermediate supervised objects and give them actual meanings. The modified SPPE network is shown in the figure below. The heat map of key points is set to Lv0, the heat map of two key points is set to Lv1, and the heat map of three key points is set to Lv3. The final output of the network is the heat map of Lv0. At the intermediate supervision, with 8 stacked hourglasses, for example, the intermediate supervision output for the 2nd and 6th hourglass is Lv1, and the intermediate supervision output for the 4th hourglass is Lv2. The output heat map of each level is compared with the labeled value to obtain the mean square error (MSE) loss, which is used to back propagate the parameters of the updated convolutional kernel. In this way, the output of each level of the convolutional neural network is made to have a practical meaning, providing sufficiently complete information about the human pose for the subsequent distracted driving network.

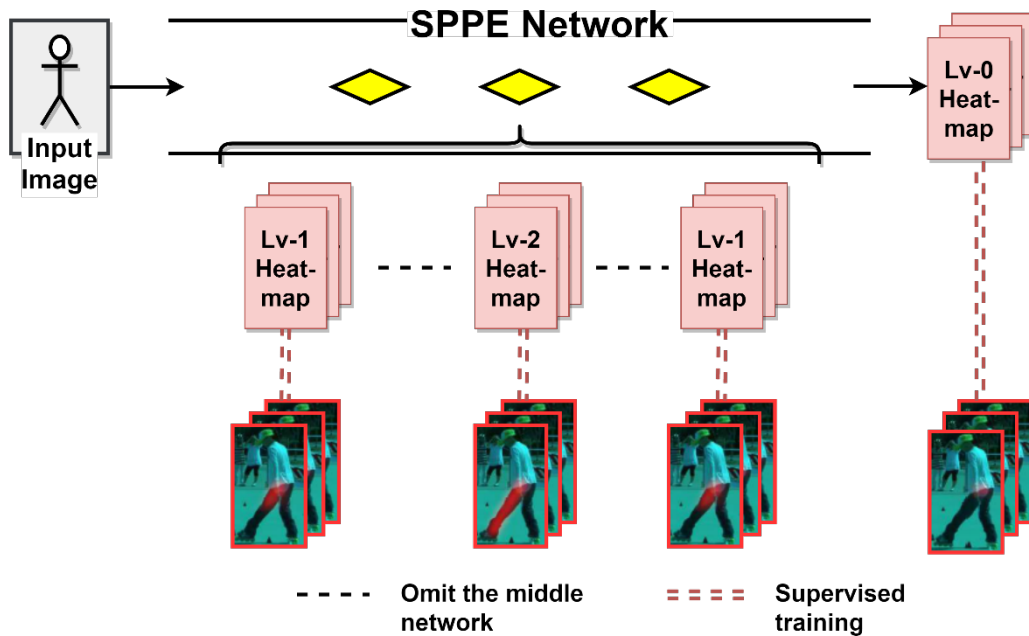


Figure 3.21 Stacked Hourglass Network Design with Fusion Composition Model

3.3.6 Defects of direct classification network

Images from distracted datasets that are directly trained by migration through common classification networks suffer from the drawback of unstable classification performance and cannot be generalized to more general scenarios. As shown in the figure below, Xception [64], Resnet-50 [65], VGG-16 [66], and Mobilenet [67] are four classification networks with superior performance and maturity, and the migration training on the distracted dataset is completed by freezing the parameters before the last layer and training only the last layer of the network. After the training on this dataset, the overall accuracy in the validation set can all reach more than 90%. By deepening the network and the number of training sessions, even 98% accuracy can be achieved. However, when drilling down into the classification results for each category of distracted driving on the validation set, it can be seen that the classification performance for each category is severely uneven. For example, the network is able to perform good recognition of driver typing behavior, but is unable to achieve 50% classification accuracy in terms of grooming and talking to passengers. This is due to the fact that the network only learns the local features of each image in the training set, and does not learn the driver's

human posture. Good classification can only be performed if the training and validation sets have similar images and distinct features. However, extending to real-world usage conditions can produce serious misclassifications due to the complex driving environment and affect the robustness of the driver supervision system.

	Xception	Resnet 50	VGG 16	Mobilenet
Safe Driving	78.9%	92.5%	54.4%	78.2%
Texting -Right	93.2%	92.3%	96.2%	89.1%
Talking - Right	99.4%	94.3%	95.1%	97.4%
Texting - Left	98.2%	98.2%	98.5%	98.5%
Talking - Left	76.5%	75.8%	87.9%	77.3%
Operating the Radio	94.8%	99.0%	96.8%	97.5%
Drinking	80.4%	86.2%	90.3%	79.3%
Reaching Behind	100.0%	99.4%	81.1%	99.7%
Hair and Makeup	40.1%	53.3%	63.6%	59.6%
Talking to passenger	58.8%	68.5%	83.1%	79.5%

Figure 3.22 Direct classification results of four common classification networks

3.3.7 Classification network design based on human posture information

Classification based on human pose information rather than direct images will undoubtedly improve the accuracy and robustness of the system substantially. However, if the key point heat map output from the SPPE network is used directly, the presence of very large permutations of body parts will make the state space too sparse and large, which will make the network learning cost surge and inference become very complicated. This places high demands on both computation and data volume. Therefore, the human pose information cannot contain only key points, but also needs the output of each part and state information, i.e., the heat map of multiple levels of the above SPPE network. The specific network design is shown in the figure below. Since the modeling of human pose has been done by the previous complex network, the subsequent classification can be done by simple machine learning methods, such as kernel SVM or decision tree. In this thesis, the classification task is done using a shallow network consisting of residual modules [62] in order to maintain a consistent network structure. The input to the network is

a collection of heat maps from multiple layers output by the SPPE network, which is learned by the residual module, the maximum pooling layer compresses the data space, and the classification is done with a fully connected layer.

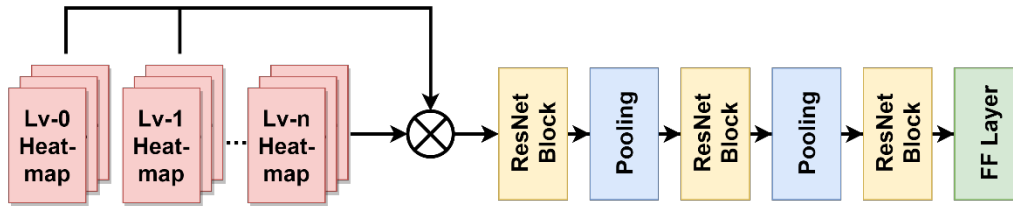


Figure 3.23 Design of classification network based on human posture information

Chapter 4 Experiments

4.1 Data set and evaluation criteria

4.1.1 COCO human key point dataset

MS COCO [68], whose full name is Microsoft Common Objects in Context, originated from the Microsoft COCO dataset funded by Microsoft in 2014 for labeling, and is regarded as one of the most popular and authoritative competitions in computer vision, along with the ImageNet competition.

COCO provides datasets required for a range of challenging tasks commonly found in computer vision, such as object detection, object segmentation, and object keypoint detection. These datasets are collected from everyday scenes, are very rich in content, and give pixel-level positional calibration of the task targets in each image.

For human keypoint detection, COCO provides a dataset of 105,698 training images and over 1 million labeled keypoints; and about 80,000 test instances split into four roughly equal-sized parts: test-challenge, test-dev, test-Standard, and test-Reserve. The dataset covers a variety of challenging scenarios such as single-player, multi-player, occlusion, and overlap, and a set of evaluation metrics and corresponding evaluation codes are designed. Some of the labeled images are shown in the following figures.



Figure 4.1 Partial COCO human key point labeling image

The COCO critical point task requires the network to detect object regions and locate their critical point locations. To fairly and reasonably measure the performance of each detection algorithm, COCO proposes a novel evaluation metric inspired by object detection evaluation metrics, namely average precision (AP) and average recall (AR).

The core of this evaluation metric, as in the case of object detection, is the similarity metric between the annotated values and the predicted objects. In the evaluation of object detection, IoU (intersection-over-union) is used as this similarity metric, which defines the degree of match between annotated values and predicted objects, and enables the calculation and plotting of accurate AP/AR curves. COCO defines a similar similarity metric OKS, object keypoint similarity, which indicates the degree of similarity between the estimated location of keypoints and the degree of similarity between the actual annotation and enables the calculation and plotting of accurate AP/AR curves.

Specifically, for each detection object, the key points are labeled in the format $[x_1, y_1, v_1, \dots, x_k, y_k, v_k]$, where (x, y) denotes the annotation position, and v indicates whether the annotation is visible. $v = 0, 1, 2$ represent unmarked, marked but not visible, and marked and visible, respectively. The format of the estimated keypoints is $[x_1, y_1, \dots, x_k, y_k]$ and confidence level, without v that the visibility of the annotation is not predicted.

The mathematical expression of OKS is.

$$\text{OKS} = \frac{\sum_i \left[\exp\left(-\frac{d_i^2}{2s^2\kappa_i^2}\right) \delta(v_i > 0) \right]}{\sum_i [\delta(v_i > 0)]} \quad (4.1)$$

where d_i is the Euclidean distance between each corresponding labeled keypoint and the predicted keypoint, v_i is the visibility marker of the labeled keypoint, s is the scale of the object, and κ_i is a constant to control the decay, and $s^2\kappa_i^2$ is the square of the standard deviation. In the OKS calculation, the distance between two keypoints is transformed into a similarity. This similarity is between 0 and 1. A perfect prediction will get an OKS close to or equal to 1, while all predicted keypoints deviating from the labeled keypoints by multiple standard deviations will get an OKS infinitely close to 0. κ_i OKS is calculated in a similar way to IoU, so IoU calculation and plotting of AP/AR curves can be used in a similar way.

The constants that control the attenuation κ_i needs to be adjusted. COCO gives specific adjustment steps [68]: first, the standard deviation σ_i is measured for each key point type i in the 5000 redundantly labeled images in val at different object scales s , i.e. $\sigma_i^2 = E[d_i^2 / s^2]$. The standard deviation σ_i is a parameter that characterizes the keypoint type. The keypoints in different parts of the human body vary greatly, and in general, keypoints located in body parts are much larger than those located in head parts.

After the measurement is complete, set $\kappa_i = 2\sigma_i$. According to the rule of thumb in statistics: with a normal distribution, 68%, 95% and 99.7% of the data will fall within the 1, 2 and 3 standard deviations respectively. In the $\kappa_i = 2\sigma_i$ setting, the key point similarity is calculated when 1, 2, and 3 standard deviations occur, respectively $\exp(-d_i^2/2s^2\kappa_i^2)$: The $e^{-1/8} = 0.88$, $e^{-4/8} = 0.61$ and $e^{-9/8} = 0.32$, i.e., the coordinates of 68%, 95% and 99.7% of the detected key points should have more than 0.88, 0.61 and 0.32 key point similarity with the corresponding labeled key point coordinates, respectively.

The following two curves represent the OKS distribution with $\kappa_i=2\sigma_i$ (blue curve) and the actual human OKS distribution curve (green curve) under the assumption of 10 independent key points per object.

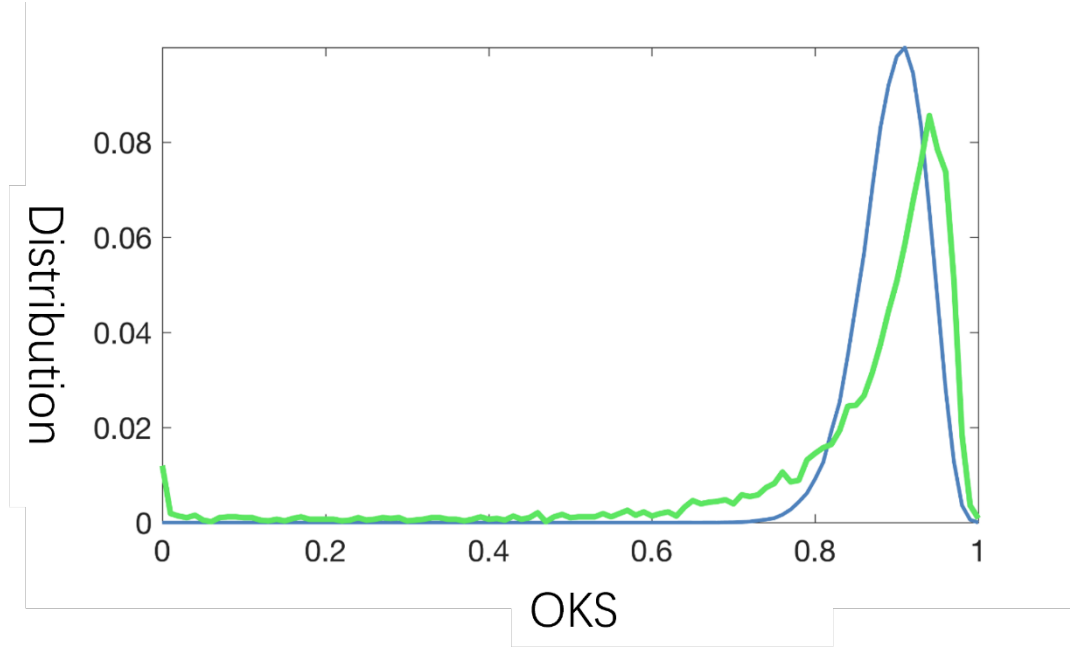


Figure 4.2 Comparison of theoretical and actual OKS distribution curves

The theoretical and actual OKS distribution curves are approximately the same. There are certain offsets and errors, mainly from the assumption that the independent key points do not match the reality. In addition, the actual labeling contains a small number of errors, most of which are caused by having multiple bodies overlapping and interfering with the labelers' judgment.

Based on the above OKS definition and evaluation curve, COCO provides 10 performance evaluation tools for characterizing key point detectors on COCO: 5 for average accuracy (AP) and 5 for average recall (AR). In this thesis, only AP is used as an evaluation metric, and the following are specific descriptions of the five metrics.

Table 4.1 Description of evaluation indicators for COCO

AP	Between 0.50 and 0.95, taking OKS at 0.05 intervals, the combined average accuracy (the most significant indicator)
AP^{50}	OKS takes an average accuracy of 0.50 (loose indicator)
AP^{75}	OKS takes an average accuracy of 0.75 (strict indicator)
AP^M	Average accuracy for images of medium human bodies

AP^L	Average accuracy for images of large human bodies
--------	---

4.1.2 State-Fram distracted driving dataset

There are many possible distracted driving situations, and considering the difficulty of obtaining training data under the influence of epidemics, the State-Farm dataset provided by Kaggle is used in this thesis. This dataset contains snapshots captured by cameras installed in cars [73] and classified at a later stage as shown in the figure below. There are 22.4 K labeled samples in the training set, which are equally distributed among the classes, and 79.7 K unlabeled test samples. There are 10 classes of images whose corresponding distracted driving profiles are described in the following figure.



Figure 4.3 State-Farm distraction dataset display by type

4.2 Training environment and hyperparameter setting

SPPE training and distracted driving classification network training were done on a single 4-block RTX2080Ti. Due to the epidemic, SSH connection to a remote server was used to debug the code and train. The remote server system was Ubuntu 18.04 and the training was done in docker. docker was configured with Python 3.68 + Pytorch 1.10 + CUDA 10.2 and some other packages.

SPPE is conducted based on the open source project AlphaPose in Github, modifying human detection to YOLOv3 and using PyraNet as the human keypoint detector. during SPPE training, a 512x512 input image is cropped from the resized image according to the labeled body position and scale, and the image is enhanced by scale transformation (0.7 to 1.35), rotation (-45 degrees to +45 degrees), flip and add color noise to enhance the image size training data. Adam was used as the

optimization strategy for training, setting the batch to 32 (8 per GPU), and the learning rate was initialized to 5e-4 and decreased to 1e-5 after the 150th iteration. 170 training sessions were performed in total.

During the training of the distracted driving classification network, the dataset images were adjusted to 512x512, and the number of trained parameters was small due to migration training, and no data enhancement was performed. Adam was used as the optimization strategy during training, setting the batch to 32 (8 per GPU), and the learning rate was initialized to 1e-3, decreasing to 1e-4 after the 40th iteration and 1e-5 after the 80th iteration. 100 training sessions were performed in total.

4.3 Training results

4.3.1 SPPE training results

The following table summarizes some representative methods of human critical point detection, where the data are from the results provided by each paper, and some parameters not provided are not listed in the table.

Table 4.2 Test results of each method on the COCO dataset

Methods	Basic Network	Parameters	GFLOPs	AP	AP ⁵⁰	AP ⁷⁵	AP ^M	AP ^L
Mask-RCNN [69]	ResNet-50-FPN	–	–	63.1	87.3	68.7	57.8	71.4
G-RMI [70]	ResNet-101	42.6M	57.0	64.9	85.5	71.3	62.3	70.0
Integral Pose Regression [53]	ResNet-101	45.0M	11.0	67.8	88.2	74.8	63.9	74.0
CPN [9]	ResNet-Inception	–	–	72.1	91.4	80.0	68.7	77.2
CFN [71]	–	–	–	72.6	86.1	69.7	78.3	64.1
Simple-Baseline [20]	ResNet-152	68.6M	35.6	73.7	91.9	81.1	70.3	80.0

HRNet-W48 [23]	HRNet-W48	63.6M	32.9	75.5	92.5	83.3	71.9	81.5
RMPE [28]	Hourglass [10]	–	–	61.8	83.7	69.8	58.6	67.6
RMPE (PyraNet)	PyraNet [12]	28.1M	26.7	72.3	89.2	79.1	68.0	78.6

The original version of RMPE, a 2016 paper, was limited by both human detector and keypoint detector performance, and the AP values achieved on the COCO dataset were somewhat different compared to the latest results. By changing the human detector and key point detector, the AP values obtained are leaps and bounds ahead, and the difference is not significant even compared to the latest HRNet. However, with the RMPE framework, the number of parameters required for training is dramatically reduced compared to other algorithms, and the number of floating point operations per second (GFLOPs) performed at runtime is much smaller than other algorithms. For in-vehicle conditions, the RMPE framework is definitely more suitable due to the hardware and heat dissipation level.

Due to the epidemic, it was not possible to actually capture images or videos for the study. The following figure shows some of the results obtained after processing the distraction dataset with the trained RMPE (PyraNet). As can be seen, the camera angle for capturing the distracted dataset is at the side, and there are many captured images where half or more of the human body is obscured. But the network learns the overall human pose very well and estimates the key points of the human body very effectively.

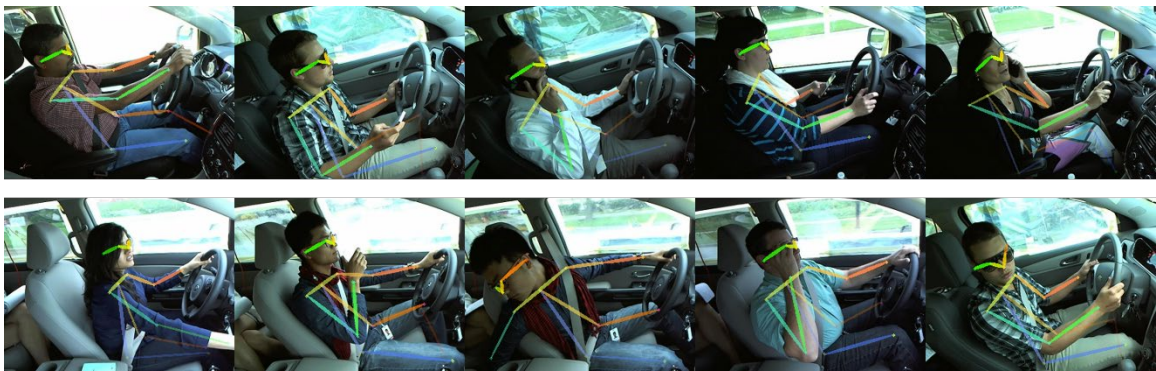


Fig. 4.4 Presentation of the correct human posture estimation results

There are also some images with obvious detection errors in the post-test images. As shown in the figure below, most of the detection errors occurred in the left foot area. Due to the shooting angle and the sitting posture of the human body, in many cases the images collected in the left foot area of the human body are completely obscured or dark, in which case the network cannot make a judgment on the left foot by image features or other postures, so many of the detection results of the left foot and knee joint or knee joint and hip connection directly become extended lines.

In addition, when there are other human bodies in the picture, the network cannot divide that part of the human body into a separate piece because the captured picture is not complete. In this case, when the driver's human body area overlaps with that part of the human body area, it will happen that a key point will drift. At this time, the system's judgment of the driver's behavior will produce a large error.

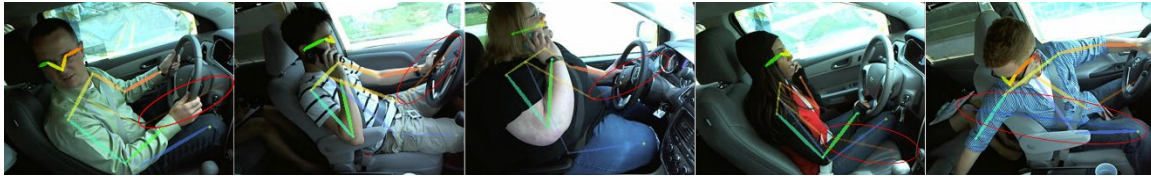


Fig. 4.5 Presentation of wrong human pose estimation results

For real-time video processing, it is impossible to give experimental results because of the lack of equipment and conditions. However, the results of 23 frames of operation are given on the project homepage of AlphaPose. It is worth noting that AlphaPose has not only RMPE to locate human body and key points, but also PoseFlow which is integrated for multi-person matching. This part of the network will reduce the operation efficiency of the network to some extent, which means that for the driving environment without multi-person matching, the speed of the project operation will be improved and basically meet the need of real-time driver posture supervision.

4.3.2 Distracted Driving Classification Network Training Results

To verify the effectiveness of the distracted driving classification network designed in this thesis, a set of controlled experiments is designed. The same distracted driving dataset was used, and 100 training sessions were conducted with

Resnet-50, SPPE+ classification network, and SPPE+ composition model + classification network, where Resnet-50 used a pre-training model, freezing the parameters except the last layer and modifying the last layer to 10 classification items; SPPE+ classification network in which SPPE used a pre-training model and only The model of classification network is shown above, and the training hyperparameters used are the same as above. After the training was completed, the same validation set was used for testing to obtain the total accuracy with the accuracy of each category.

The loss values of each iteration during training were recorded in csv files and curves were generated in excel as shown below.

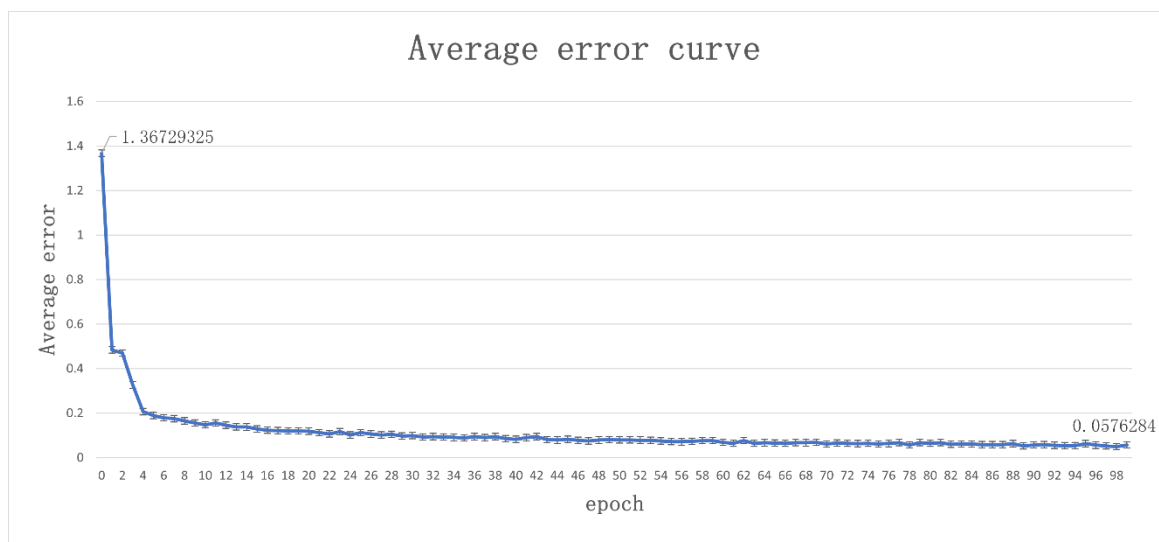


Figure 4.6 Resnet-50 error convergence curve

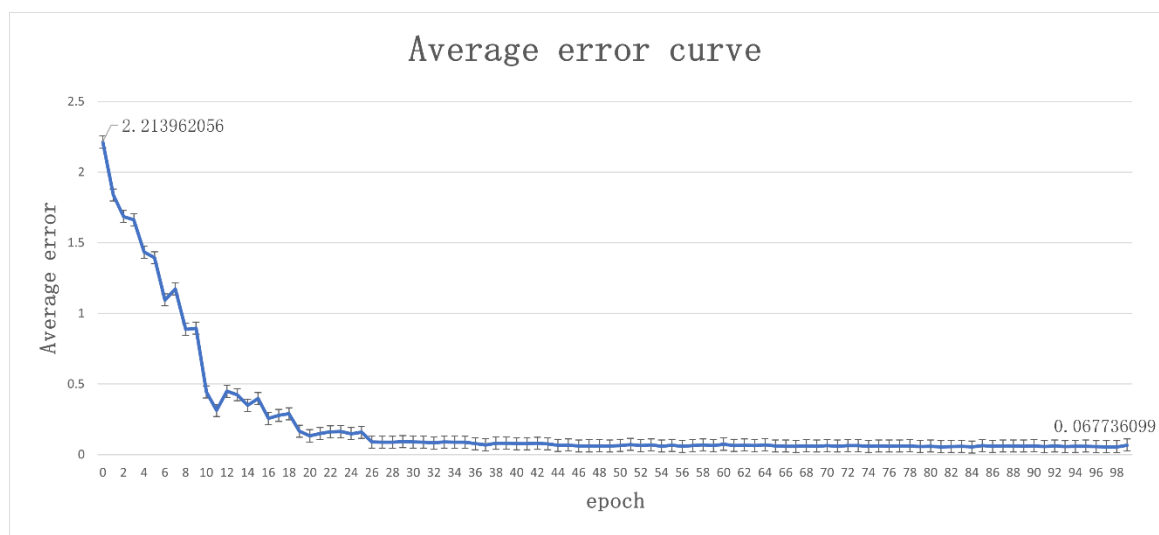


Figure 4.7 SPPE+ classification network error convergence curve

The curve of recorded accuracy with the number of iterations is shown in the following figure.

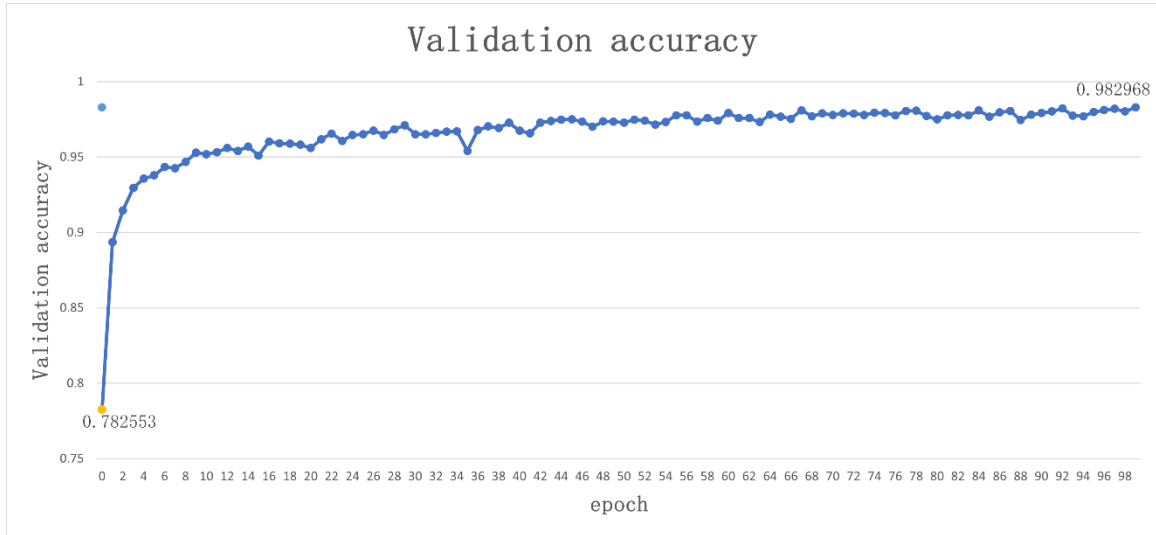


Figure 1 .8 Resnet-50 accuracy curve

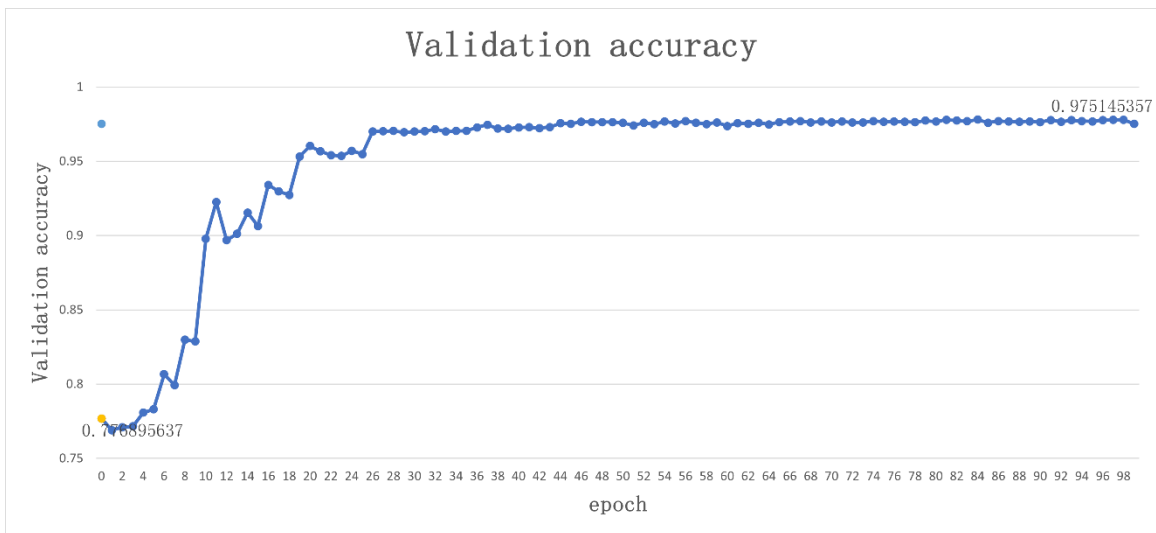


Figure 4.9 SPPE+ classification network accuracy curve

Record the classification accuracy of the classification network for each category of distracted driving after the training is completed.

<pre> No. 99 Epoch: Accuracy 0.982968 good:0.789403 R_type:0.932041 R_phone:0.994636 L_type:0.946731 L_phone:0.979334 radio:0.979650 back:1.000000 make:0.736412 talk:0.795700 </pre>	<pre> No. 99 Epoch: Accuracy 0.977767 good:0.925601 R_type:0.923065 R_phone:0.943112 L_type:0.943621 L_phone:0.98200 radio:0.862689 back:0.994312 make:0.553760 talk:0.685671 </pre>
---	--

Figure 4.10 Left: Resnet-50 accuracy rates for each category Right: SPPE+ classification network accuracy rates for each category

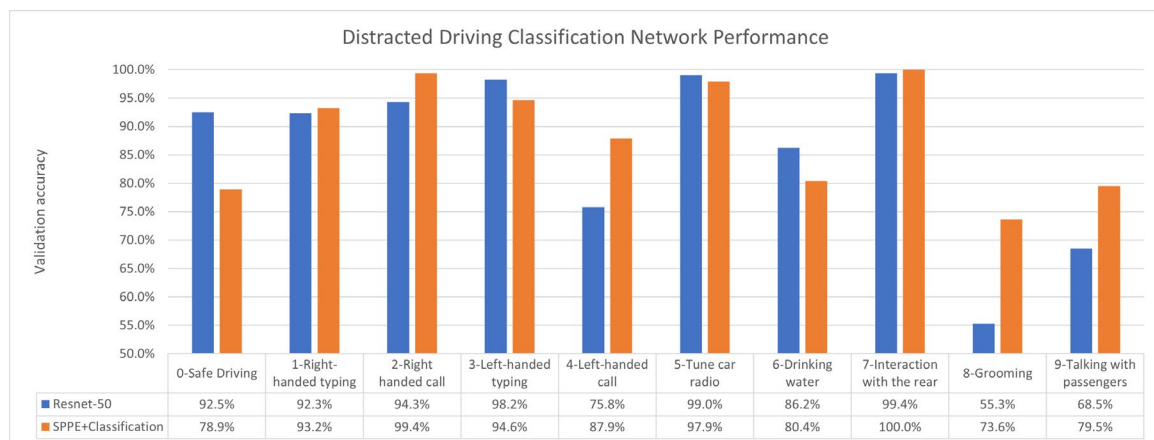


Figure 4.11 Performance comparison of distracted driving classification networks

The training results of Resnet-50 were analyzed: the distracted driving classification task is actually very simple to achieve excellent results if only the total verification accuracy is considered. However, it can be seen that the classification performance for each category is severely uneven, for example, the network is able to achieve very good recognition for the behaviors of typing with the driver's left hand, tuning the car radio and interacting with the rear, but it is unable to achieve 60% classification accuracy in organizing grooming. This is due to the fact that the network only learns the local features of each image in the training set and does not learn the driver's human posture. Good classification is only possible when the training and validation sets have similar images and distinct features. However, extending to real-world usage conditions can produce serious misclassifications due

to the complex driving environment and affect the robustness of the driver supervision system.

The training results of the SPPE+ classification network are analyzed: the key point heat map output of the SPPE network is classified directly, and the loss value will produce a sharp oscillation at the beginning and then stabilize. The learning rate setting is very important during training, too large or too small will cause the loss not to converge to 0. After a certain training, the learning rate needs to be decreased so that the loss can continue to decrease, otherwise it will produce oscillation and even increase the loss. After the training was completed, it was found through validation that the classification accuracy was similar to Resnet-50 with some improvement, but it was clear that the network did not fully utilize the information of human posture. This is due to the fact that inputting key information about the human body, the network first needs to organize and construct a model of the human body and subsequently classify it based on the model features. But at this point there are very large permutations of body parts, which can make the state space too sparse and large, thus making the network surging in learning costs and making inference very complex. This puts high demands on both computation and data volume. But unfortunately the amount of state-farm distraction dataset is not huge, the constructed network is shallow, and the number of training iterations is limited. To satisfy the need for loss convergence, the network needs to drop this part of inference learning, thus making the actual performance of the network similar to that of a direct classification network.

Chapter 5 Conclusion

As one of the important branches of computer vision, human pose estimation has started to be applied to the analysis of human behavior patterns, motion capture in animation and film production, and other commercially valuable application scenarios with high research enthusiasm in the past few years. Every year, new state-of-art methods are born, for example, there are 12 papers on human pose estimation among the papers accepted by CVPR2021 this year. Most of these methods continue to improve their performance on open-source datasets by fusing algorithms, improving network structures, and benefiting from enhanced computational power and data. At the same time, several major open source datasets are continuously expanding their data pools to add as many scenarios as possible, allowing for significant improvements in the generality of the trained networks.

Current research on human pose estimation is focused on solving problems such as pose recognition accuracy in 3D and human pose tracking in dense environments. At the same time, various methods continue to enhance the overall understanding of the network for human models, reduce the amount of floating-point operations per unit of practice, and increase the rate of real-time operation as much as possible.

Driver behavior monitoring is an application scenario based on human posture estimation technology, and with the advent of autonomous driving, the need for the development of this system is becoming more and more urgent. From the test, the current human posture estimation algorithm has been able to achieve a high accuracy rate in the driving environment, but it needs to be enhanced in terms of real-time and generality.

In terms of real-time, the current test environment is a server, and the processing speed of data can basically meet the needs of real-time. But the development of mobile scenarios will limit the power and cost, and self-driving cars are bound to give priority to sensing the external environment, and the computational resources allocated to the driver monitoring system will be further limited.

In terms of generality, the current open-source dataset is too homogeneous, without various driving environments and shooting angles, and the number of classifications is also very limited. To accomplish a generalized algorithm, the acquisition of datasets has become a problem that must be solved.

Currently this thesis has limited time to complete, and much time is spent on learning, setting up the environment and Debug, which relatively reduces the time for perfecting the network structure and adjusting the hyperparameters for repeated training. The loss curve given in Chapter 4 actually has a lot of room for optimization, the composition model is limited by time and not retrained, and there is a lot of explorable space in the way of feature fusion for the composition human posture. In conclusion, there are still many regrets about the experiments of this thesis. After finishing the graduation design, I will continue to study it in depth for a period of time until the practice conforms to the theory and verifies my ideas.

References

- [1] Simonyan, Karen, and Andrew Zisserman. "Two-stream convolutional networks for action recognition in videos." *Advances in neural information processing systems* 27 (2014).
- [2] Ji, Shuiwang, et al. "3D convolutional neural networks for human action recognition." *IEEE transactions on pattern analysis and machine intelligence* 35.1 (2012): 221-231.
- [3] Poppe, Ronald. "A survey on vision-based human action recognition." *Image and vision computing* 28.6 (2010): 976-990.
- [4] Hara, Kensho, Hirokatsu Kataoka, and Yutaka Satoh. "Learning spatio-temporal features with 3d residual networks for action recognition." *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017.
- [5] Dang, Qi, et al. "Deep learning based 2d human pose estimation: A survey." *Tsinghua Science and Technology* 24.6 (2019): 663-676.
- [6] Munea, Tewodros Legesse, et al. "The progress of human pose estimation: a survey and taxonomy of models applied in 2D human pose estimation." *IEEE Access* 8 (2020): 133330-133348.
- [7] Presti, Liliana Lo, and Marco La Cascia. "3D skeleton-based human action classification: A survey." *Pattern Recognition* 53 (2016): 130-147.
- [8] Yi Yang, Deva Ramanan, Articulated Human Detection with Flexible Mixtures-of-Parts, *IEEE Trans Pattern Anal Mach Intell*
- [9] Zhang, Zhengyou. "Microsoft kinect sensor and its effect." *IEEE multimedia* 19.2 (2012): 4-10.
- [10] Jain, A., Tompson, J., Andriluka, M., Taylor, G.W., & Bregler, C., Learning Human Pose Estimation Features with Convolutional Networks, *ICLR* 2013
- [11] Feng, Liqi, et al. "A comparative review of graph convolutional networks for human skeleton-based action recognition." *Artificial Intelligence Review* (2021): 1-31.
- [12] Toshev, A., & Szegedy, C., DeepPose: Human Pose Estimation via Deep Neural Networks, *CVPR* 2014

- [13] Tompson, J., Goroshin, R., Jain, A., LeCun, Y., & Bregler, C, Efficient Object Localization Using Convolutional Networks – T, CVPR 2015
- [14] Tompson, J., Jain, A., LeCun, Y., & Bregler, C., Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation, NIPS 2014
- [15] Wei, S., Ramakrishna, V., Kanade, T., & Sheikh, Y., Convolutional Pose Machines, CVPR 2016
- [16] Ramakrishna, Varun, et al. "Pose machines: Articulated pose estimation via inference machines." European Conference on Computer Vision. Springer, Cham, 2014.
- [17] Ning, G., Zhang, Z., & He, Z., Knowledge-Guided Deep Fractal Neural Networks for Human Pose Estimation, IEEE Transactions on Multimedia 2018
- [18] Xuecheng Nie, Jiashi Feng, Yiming Zuo, Shuicheng Yan, Human Pose Estimation with Parsing Induced Learner, CVPR 2018
- [19] Wei Tang, Pei Yu, Ying Wu, Deeply Learned Compositional Models for Human Pose Estimation, ECCV 2018
- [20] Newell, A., Yang, K., & Deng, J., Stacked Hourglass Networks for Human Pose Estimation, ECCV 2016
- [21] Chu, X., Yang, W., Ouyang, W., Ma, C., Yuille, A.L., & Wang, X., Multi-context Attention for Human Pose Estimation, CVPR 2017
- [22] Yang, W., Li, S., Ouyang, W., Li, H., & Wang, X., Learning Feature Pyramids for Human Pose Estimation, ICCV 2017
- [23] Lipeng Ke, Ming-Ching Chang, Honggang Qi, Siwei Lyu, Multi-Scale Structure-Aware Network for Human Pose Estimation, ECCV 2018
- [24] Ke Sun, Bin Xiao, Dong Liu, Jingdong Wang, Deep High-Resolution Representation Learning for Human Pose Estimation, CVPR 2019
- [25] Jain, A., Tompson, J., LeCun, Y., & Bregler, C., MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation, ACCV 2014
- [26] Yue Luo, Jimmy Ren, Zhouxia Wang, Wenxiu Sun, Jinshan Pan, Jianbo Liu, Jiahao Pang, Liang Lin, LSTM Pose Machines, CVPR 2018

- [27] Fang, Hao-Shu, et al. "Rmpe: Regional multi-person pose estimation." Proceedings of the IEEE international conference on computer vision. 2017.
- [28] Li, Jiefeng, et al. "Crowdpose: Efficient crowded scenes pose estimation and a new benchmark." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.
- [29] Osokin, Daniil. "Real-time 2d multi-person pose estimation on cpu: Lightweight openpose." arXiv preprint arXiv:1811.12004 (2018).
- [30] Cao, Zhe, et al. "Realtime multi-person 2d pose estimation using part affinity fields." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
- [31] Wang, Jiang, et al. "Mining actionlet ensemble for action recognition with depth cameras." 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012.
- [32] Hussein, Mohamed E., et al. "Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations." Twenty-third international joint conference on artificial intelligence. 2013.
- [33] Vemulapalli, Raviteja, and Rama Chellapa. "Rolling rotations for recognizing human actions from 3d skeletal data." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- [34] Shotton, Jamie, et al. "Real-time human pose recognition in parts from single depth images." CVPR 2011. Ieee, 2011.
- [35] Xia, Lu, Chia-Chih Chen, and Jake K. Aggarwal. "View invariant human action recognition using histograms of 3d joints." 2012 IEEE computer society conference on computer vision and pattern recognition workshops. IEEE, 2012.
- [36] Du, Yong, Wei Wang, and Liang Wang. "Hierarchical recurrent neural network for skeleton based action recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
- [37] Li, Chao, et al. "Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation." arXiv preprint arXiv:1804.06055 (2018).

- [38] Ke, Qihong, et al. "Skeletonnet: Mining deep part features for 3-d action recognition." *IEEE signal processing letters* 24.6 (2017): 731-735.
- [39] Li, Chao, et al. "Skeleton-based action recognition with convolutional neural networks." *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE, 2017.
- [40] Yan, Sijie, Yuanjun Xiong, and Dahua Lin. "Spatial temporal graph convolutional networks for skeleton-based action recognition." *Thirty-second AAAI conference on artificial intelligence*. 2018.
- [41] Scarselli, Franco, et al. "The graph neural network model." *IEEE transactions on neural networks* 20.1 (2008): 61-80.
- [42] Qi, Siyuan, et al. "Learning human-object interactions by graph parsing neural networks." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.
- [43] Bruna, Joan, et al. "Spectral networks and locally connected networks on graphs." *arXiv preprint arXiv:1312.6203* (2013).
- [44] Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. "Convolutional neural networks on graphs with fast localized spectral filtering." *Advances in neural information processing systems* 29 (2016).
- [45] Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." *arXiv preprint arXiv:1609.02907* (2016).
- [46] Kong, Yinghui, et al. "Attention module-based spatial-temporal graph convolutional networks for skeleton-based action recognition." *Journal of Electronic Imaging* 28.4 (2019): 043032.
- [47] Tang, Yansong, et al. "Deep progressive reinforcement learning for skeleton-based action recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
- [48] Gao, Xiang, et al. "Optimized skeleton-based action recognition via sparsified graph regression." *Proceedings of the 27th ACM International Conference on Multimedia*. 2019.

- [49] Zhang, Xikun, et al. "Graph edge convolutional neural networks for skeleton-based action recognition." *IEEE transactions on neural networks and learning systems* 31.8 (2019): 3047-3060.
- [50] Shi, Lei, et al. "Two-stream adaptive graph convolutional networks for skeleton-based action recognition." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.
- [51] Shi, Lei, et al. "Skeleton-based action recognition with multi-stream adaptive graph convolutional networks." *IEEE Transactions on Image Processing* 29 (2020): 9532-9545.
- [52] Liu, Ziyu, et al. "Disentangling and unifying graph convolutions for skeleton-based action recognition." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020.
- [53] Li, Maosen, et al. "Actional-structural graph convolutional networks for skeleton-based action recognition." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.
- [54] Fang, H. S., Xie, S., Tai, Y. W., & Lu, C. (2017). RMPE: Regional Multi-person Pose Estimation. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-October, 2353- 2362.
- [55] Redmon, J., & Farhadi, A. (2017). YOLOv3: An Incremental Improvement. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.
- [56] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [57] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. in *Lecture Notes in Computer Science*
- [58] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal Loss for Dense Object Detection. in *Proceedings of the IEEE International Conference on Computer Vision*.

- [59] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December, 779-788.
- [60] Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017.
- [61] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: new benchmark and state of the art analysis. cvpr 2014.
- [62] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. cvpr 2016.
- [63] Li, J., Wang, C., Zhu, H., Mao, Y., Fang, H. S., & Lu, C. (2019). Crowdpose: Efficient crowded scenes pose estimation and a new benchmark. in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.
- [64] Chollet, F. (2017). Xception: deep learning with depthwise separable convolutions. in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017.
- [65] He, K., Zhang, X., Ren, S., & Sun, J. (2016). ResNet. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.
- [66] Simonyan, K., & Zisserman, A. (2014). VGG-16. arXiv Preprint.
- [67] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... Adam, H. (2017). MobileNet V1. ArXiv Preprint ArXiv:1704.04861.
- [68] MSCOCO keypoint challenge 2017. <http://mscoco.org/dataset/keypoints-challenge2016>. 4326
- [69] K. He, G. Gkioxari, P. Doll'ar, and R. B. Girshick. mask R-CNN. in ICCV, pages 2980-2988, 2017.
- [70] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy. Towards accurate multiperson pose estimation in the wild. in CVPR, pages 3711-3719, 2017.

[71] S. Huang, M. Gong, and D. Tao. A coarse-fine network for keypoint localization. in ICCV, pages 3047-3056. IEEE Computer Society, 2017.