

基于 BlazePose 算法的机器人人体姿势识别 与模仿

陶冶

院 (系): 计算学部 专 业: 计算机科学与技术

学 号: 1180300204 指导教师: 傅忠传

2022 年 6 月

哈爾濱工業大學

毕业设计（论文）

题 目 基于 BlazePose 算法的
机器人人体姿势识别与模仿

专 业 计算机科学与技术

学 号 1180300204

学 生 陶冶

指 导 教 师 傅忠传

答 辩 日 期 2022 年 6 月 10 日

摘要

人机交互的主要目的是使机器人能够学习和了解人，能领会和模仿人的语言和行为。为了使人机交互自然，必须引入类似于人与人之间的沟通方式，即依赖语音与视觉。在这种背景下，人体姿态估计在人机交互方面有着举足轻重的作用。本文就是对此提出了基于人体姿态估计的机器人姿态跟踪算法，系统地完成了足够轻量化的人体姿态估计算法，并落实到了移动端，机器人端。

本文首先系统全面地介绍了国内外对于单人姿态估计算法的发展历程以及各自的优势。并且介绍了人体姿态估计所涉及领域的主要内容，包括卷积神经网络的组成、反向传播算法、激活函数以及正则化。为后续的 BlazePose 算法研究打下基础。其次研究并复现了 BlazePose 算法，对比了复现的模型与官方模型以及一些主流模型的性能差异，在得到一个可以实用的模型后，将该算法成功部署到电脑端、手机端和机器人上。最后，得出结论：从机器人模仿的准确度和实时性而言，本项目以达到了预期的效果。

本文的主要贡献如下：

1. 摒弃了传统的 NMS 算法来进行目标检测的后处理步骤，而假设人脸必须出现在图像中并采用了一个全新的人脸检测器。
2. 将目前最主流的基于热图的姿态估计和基于回归的姿态估计相结合。使用编码器-解码器网络架构来预测所有关节的热图。同时后面跟着另一个编码器，直接回归到所有关节的坐标。使得足够轻量化，可以架构在移动端和机器人端。

关键词：人体姿态估计；机器人；轻量化；实时性；模仿人体姿态

Abstract

The main purpose of human-computer interaction is to enable robots to learn and understand people, to comprehend and imitate human language and behavior. In order for the human-computer interaction to be natural, a communication method similar to that between humans must be introduced, that is, relying on speech and vision. In this context, human pose estimation plays a pivotal role in human-computer interaction. This paper proposes a robot posture tracking algorithm based on human body posture estimation, and systematically completes a sufficiently lightweight human body posture estimation algorithm, and implements it on the mobile side and the robot side.

This paper firstly introduces the development history of single-person pose estimation algorithms at home and abroad and their respective advantages. Then we introduced the main contents of the field of human pose estimation, including the composition of convolutional neural network, back-propagation algorithm, activation function and regularization. It lays the foundation for the follow-up BlazePose algorithm research. Secondly, the BlazePose algorithm is researched and reproduced, and the performance differences between the reproduced model and the official model and some mainstream models are compared. After obtaining a practical model, the algorithm was successfully deployed on computers, mobile phones and robots. Finally, it is concluded that this project has achieved the expected effect in terms of the accuracy and real-time performance of robot imitation.

The main contributions of this paper are as follows:

1. The post-processing step of the traditional NMS algorithm for object detection is abandoned, and a new face detector is adopted assuming that the face must be present in the image.
2. Combining the most mainstream heatmap-based pose estimation with regression-based pose estimation. Heatmaps for all joints are predicted using an encoder-decoder network architecture. At the same time, it is followed by another encoder, which directly returns to the coordinates of all joints. It is lightweight enough to be built on mobile and robotics.

Keywords: Human Posture Estimation, robot, lightweight, Real-time, action imitation

缩略词注释表

简称	英文全称	中文全称
CNN	Convolutional Neural Network	卷积神经网络
PCK	Percentage of Correct Keypoints	关键点正确估计的比例
HOG	Histogram of Oriented Gradient	方向梯度直方图
ROI	region of interest	感兴趣区域
NMS	non maximum suppression	非极大抑制
IoU	Intersection over Union	交并比
CPM	Convolutional Pose Machine	卷积姿态网络
SHN	Stacked Hourglass Networks	级联的沙漏网络
HRUs	Hourglass Residual Units	沙漏残差单元

目 录

摘要	I
Abstract	II
缩略词注释表	III
第1章 绪论	1
1.1 课题背景与研究意义	1
1.2 国内外研究现状	2
1.2.1 基于图的人体姿态估计	2
1.2.2 基于深度学习的人体姿态估计	3
1.3 论文主要研究内容及创新点	6
1.4 论文章节安排	7
第2章 姿态估计相关理论基础	8
2.1 引言	8
2.2 卷积神经网络的组成	8
2.2.1 卷积层	8
2.2.2 池化层	10
2.3 卷积神经网络(CNN)反向传播算法	11
2.3.1 DNN 的反向传播算法	11
2.3.2 CNN 的反向传播算法思想	11
2.4 激活函数	12
2.4.1 Sigmoid 激活函数	12
2.4.2 tanh 激活函数	14
2.4.3 整流线性单元 (ReLU)	14
2.5 正则化	15
2.6 本章小结	16
第3章 基于BlazePose的人体姿态估计算法研究	17
3.1 推理通道	17
3.2 神经网络结构	17
3.3 人体检测器	18

3.4 人体拓扑结构	19
3.5 输入输出.....	19
3.6 本章小结.....	20
第4章 基于BlazePose的训练、姿态识别与模仿	23
4.1 训练	23
4.2 性能比较.....	23
4.3 API	25
4.4 基于BlazePose的姿态识别	26
4.4.1 电脑端的姿态识别.....	26
4.4.2 移动端的姿态识别.....	29
4.5 基于BlazePose的姿态模仿	30
4.5.1 虚拟机器人的姿态模仿	30
4.5.2 真实机器人的姿态模仿	30
4.6 本章小结.....	32
结 论	34
参考文献	36
哈尔滨工业大学本科毕业设计（论文）原创性声明	40
致 谢	41
附录1 BlazePose各关键点的名称	42

第1章 绪论

1.1 课题背景与研究意义

单人姿态估计是计算机视觉领域的热点基础问题之一，已经研究了 20 多年。它之所以重要，是因为有大量的应用程序可以从这种技术中受益。例如，人体姿势估计可以在人机交互和活动识别的背景下进行更高层次的推理；它也是无标记运动捕捉（MoCap）技术的基本构建块之一。MoCap 技术适用于角色动画、电影和游戏，以及病理步态的临床分析等应用。

人体姿态估计具体是指基于观察图像中的人体来恢复关节和躯干。^[1]

近年来，已有大量的成果使用到了神经网络训练姿态估计，人体姿态估计在实际生活中的应用也愈发广泛。具体如下：

1. 虚拟试衣

随着单人姿态估计的技术越发成熟，现在一些服饰店（如优衣库、男衣邦）推出虚拟试衣服务。以便用户省去脱衣试衣的繁琐操作，另外此项技术在疫情期间网上购物的应用场景下显然有着更光明的应用前景。图 1-1 便是来自 ICCV2021 的一篇虚拟试衣论文

2. 智能安防

随着 5G 网络的普及，摄像头遍布公共场所，可以采集丰富的人类行为数据。现有的人体姿态估计算法已经可以分析并自动识别出监控视频中人群的活动。如存在着异常的行为，则可以即使给出异常提示或警报。^[3]

3. 体育健身

疫情场景下的居家健身成为当下时代热门，如何规范、无错误动作的健身则显得尤为重要。因此，基于人体姿态估计的健身程序应运而生。该类程序运用单人姿态估计算法获取人体姿态，检测姿势是否标准，一定程度上减少了因动作不规范而引发的安全隐患。图 1-2 就是来自谷歌的 AI 健身。

4. VR 技术

在元宇宙概念爆火的同时，也带来了一个问题，那就是 VR 设备动辄几万美元的费用并不是大部分人所能承担得了。作为 VR 体感技术的代替品，姿态估计有着得天独厚的优势，如低廉的价格、更少的硬件设备等。就目前而言，微软已有相关应用：Kinect^[4]。



图 1-1 不限定品类和数量的多件单品试穿，来自 DiOr (Cui et al.[2])



图 1-2 基于姿态估计的 AI 健身，来自谷歌官网

1.2 国内外研究现状

1.2.1 基于图的人体姿态估计

图模型、优化算法和组件外观模型是基于图的人体姿态估计方法的三个部分。该方法定义的图结构如图 1-3 所示。图结构模型的工作原理主要是设计一些人体部件检测器，并将各部件之间联通起来，再根据人体运动学的一些要求

实现人体姿态估计。这种设计方法的主要缺点有：

1. 一方面，图模型结构的时间复杂度虽然较低，但是它主要提取的是 HOG 和 SHIEFT 特征，导致其无法重复使用图像的底层信息和语义信息，从而使得图像中的底层信息对于算法的制约很大。另一个，由于部件的模型较为简单，当人体运动幅度过大时，算法不能很好地识别出姿态，同一种姿态存在着多个可行解，使得图结构模型方法的应用场景比较狭隘。
2. 另一个方面，图结构模型方法基本上都是基于传统的数字图像提取特征算法，所以需要较为昂贵的专业传感设备，导致应用场景进一步受到限制。另外，这种算法通常需要多视角摄像来减少遮蔽问题，导致姿态数据的获取很是繁琐。因此，基于传统的数字图像提取特征算法来进行姿态估计无法推广，而且效率极为低效。

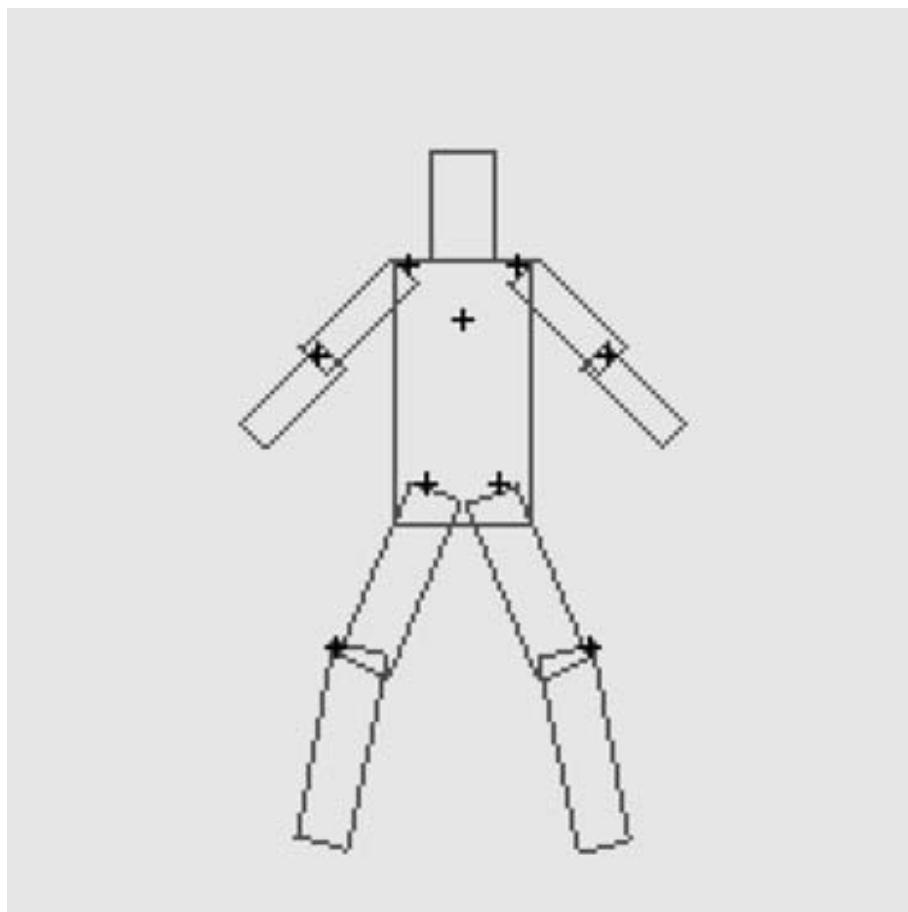


图 1-3 人体图结构, Felzenszwalb et al.[5]

1.2.2 基于深度学习的人体姿态估计

由于技术的掣肘，在 2015 年之前，大部分基于深度学习的人体姿态估计都

是回归精确的关键点坐标 (x, y) ，由于人体的刚性程度很低，导致这种单一方式的模型拓展性较差。因此我们主要分析 15 年之后的单人姿态估计算法。

MPII 单人数据集是目前单人姿态估计的主流数据集之一，这可以通过计算这个数据集的 PCK (Percentage of Correct Keypoints，如式 (1-1) 所示) 值来评估模型的优劣。目前 MPII 单人数据集的排名如表 1-1 所示。

$$\begin{aligned} PCK_i^k &= \frac{\sum_p \delta\left(\frac{d_{pi}}{d_{p,def}} \leq T_k\right)}{\sum_p 1} \\ PCK_{mean}^k &= \frac{\sum_p \sum_i \delta\left(\frac{d_{pi}}{d_{p,def}} \leq T_k\right)}{\sum_p \sum_i 1} \end{aligned} \quad (1-1)$$

表 1-1 Overall performance

	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
Pishchulin et al., ICCV'13 ^[6]	74.3	49.0	40.8	34.1	36.5	34.4	35.2	44.1
Tompson et al., NIPS'14 ^[7]	95.8	90.3	80.5	74.3	77.6	69.7	62.8	79.6
Carreira et al., CVPR'16 ^[8]	95.7	91.7	81.7	72.4	82.8	73.2	66.4	81.3
Tompson et al., CVPR'15 ^[9]	96.1	91.9	83.9	77.8	80.9	72.3	64.8	82.0
Hu&Ramanan, CVPR'16 ^[10]	95.0	91.6	83.0	76.6	81.9	74.5	69.5	82.4
Pishchulin et al., CVPR'16 ^[11]	94.1	90.2	83.4	77.3	82.6	75.7	68.6	82.4
Lifshitz et al., ECCV'16 ^[12]	97.8	93.3	85.7	80.4	85.3	76.6	70.2	85.0
Gkioxary et al., ECCV'16 ^[13]	96.2	93.1	86.7	82.1	85.2	81.4	74.1	86.1
Rafi et al., BMVC'16 ^[14]	97.2	93.9	86.4	81.3	86.8	80.6	73.4	86.3
Belagiannis & Zisserman, FG'17 ^[15]	97.7	95.0	88.2	83.0	87.9	82.6	78.4	88.1
Insafutdinov et al., ECCV'16 ^[16]	96.8	95.2	89.3	84.4	88.4	83.4	78.0	88.5
Wei et al., CVPR'16 ^[17]	97.8	95.0	88.7	84.0	88.4	82.8	79.4	88.5
Bulat & Tzimiropoulos, ECCV'16 ^[18]	97.9	95.1	89.9	85.3	89.4	85.7	81.7	89.7
Newell et al., ECCV'16 ^[19]	98.2	96.3	91.2	87.1	90.1	87.4	83.6	90.9
Tang et al., ECCV'18 ^[20]	97.4	96.4	92.1	87.7	90.2	87.7	84.3	91.2
Ning et al., TMM'17 ^[21]	98.1	96.3	92.2	87.8	90.6	87.6	82.7	91.2
Luvizon et al., arXiv'17 ^[22]	98.1	96.6	92.0	87.5	90.6	88.0	82.7	91.2
Chu et al., CVPR'17 ^[23]	98.5	96.3	91.9	88.1	90.6	88.0	85.0	91.5
Chou et al., arXiv'17 ^[24]	98.2	96.8	92.2	88.0	91.3	89.1	84.9	91.8
Chen et al., ICCV'17 ^[25]	98.1	96.5	92.5	88.5	90.2	89.6	86.0	91.9
Yang et al., ICCV'17 ^[26]	98.5	96.7	92.5	88.7	91.1	88.6	86.0	92.0
Ke et al., ECCV'18 ^[27]	98.5	96.8	92.7	88.4	90.6	89.4	86.3	92.1
Tang et al., ECCV'18 ^[28]	98.4	96.9	92.6	88.7	91.8	89.4	86.2	92.3
Zhang et al., arXiv'19 ^[29]	98.6	97.0	92.8	88.8	91.7	89.8	86.6	92.5
Su et al., arXiv'19 ^[30]	98.7	97.5	94.3	90.7	93.4	92.2	88.4	93.9
Bulat et al., FG'2020 ^[31]	98.8	97.5	94.4	91.2	93.2	92.2	89.3	94.1

大体上表 1-1 中的算法可以分成两类：

- 基于热图的方法

- 基于回归的方法

热力图方法能以较小的算力开销拓展到多人姿态识别，但会使得模型更复杂。

而基于回归的算法虽然计算简单，但也有一定的缺陷。最致命的是，基于回归的方法通过预测平均值并不能解决多义性问题。

Pfister et al.[32] 首次提出了回归 heatmap，同时使用了网络层次较深的卷积神经网络进行姿态估计，将单人姿态估计的鲁棒性进一步提高，并且可以可视化观察训练过程，以便即使调整网络结构，避免不必要的能耗。Pfister 最具创意的地方在于提出了空间融合模型，即将 CNN 的第三层和第七层分别提取出来再进行一次卷积操作；同时还使用了光流信息，预测出相邻帧的热力图，减少了模型的复杂度。最后经过一个池化层将对齐的热力图合成一个置信图。

Wei et al.[17] 提出了 CPM(Convolutional Pose Machine) 算法，使用卷积神经网络进行人体姿态估计，它的创新点在于卷积结构是高度顺序化的，表现在网络分成了多个阶段，每个阶段都会监督训练，可以更好地融合空间、纹理信息。另外还使用了多尺度处理输入，提升了模型的准确度。

Newell et al.[19] 提出了 SHN(Stacked Hourglass Networks) 算法，使用了沙漏结构，在参数量减少的同时，又能兼顾到基于回归的姿态估计方法的准确度。SHN 算法重复使用自底向上/自顶向下地网络结构，看起来像一个沙漏，故此得名。另外，同时引入了中间监督学习，从而显著提高了模型的准确率。

王晓刚组于 2017 年提出 structured pose^[33]，网络结构同样是基于 CNN，它在卷积层使用了几何变换核，并引入双向树概念，使得关键点的通道可以互相接受信息，从而达到信息传递的作用。与王晓刚组不同的是，Chen et al.[25] 网络结构基于 GAN，性能提升效果并不显著，更多的是对于沙漏网络之后的参数微调。

Ke et al.[27] 将多内容信息注意力机制迁移到卷积神经网络，得到了单人姿态估计的端到端框架，并且改进了 hourglass 网络架构，设计出新颖的 HRUs(Hourglass Residual Units)，以增加网络接受野。同时还是用了多尺度监督来训练模型，多尺度回归来优化人体结构。最后又使用了 keypoint masking 作为数据扩增的方式。

Tang et al.[28] 设计了基于 DNN 的网络结构，新颖的地方在于，这个网络结构是分层组成的，并且在推理阶段使用了自下而上/自上而下。

百度研究院和香港科技大学联合于 2019 年出品了一篇单人 pose 检测文章^[29]。这篇文章的贡献主要是提出了两个简单高效的模块，第一个是 Cascade

Prediction Fusion(CPF) 网络，可以用来预测人体姿态关键点；另外一个是 Pose Graph Neural Network(PGNN)，用于修正预测的关键点。

2019 年南京开发团队（平安科技所著）的一篇论文^[30]里的算法在 mpII 数据集中的准确率达到 93.9%，与之前其他的单人姿态估计算法相比，准确度有着明显的提升。这主要归功于作者提出了三点创新。第一，最后将多个热图的平均值作为最后输出，提高了模型的鲁棒性；第二，联合了 resnet101 模型和 resnet50 模型，使得效果达到最佳；第三，在数据集方面，引入 AI Challenger 数据集起到正则化的作用。

Bulat et al.[31] 设计小的 block 和 feature map 的融合方式，提人体姿态估计高计算效率和精度。该模型在 MPII 和 LSP 数据集上实现了 SOTA。此外，在模型的复杂度降低三倍的情况下，运行速度提升了两倍，并且性能没有下降。在 mpII 数据集结果达到 94.1%。

1.3 论文主要研究内容及创新点

本文以谷歌闭源工业级模型 BlazePose 为切入点，优化其网络结构，完成代码复现工作，得到一个较好的模型，并利用该模型完成了对于图片、视频和摄像头实时的检测工作，并且成功将其移植到手机端、unity 虚拟机器人和真实机器人上。

与之前的姿态估计算法相比，BlazePose 的主要创新性工作如下：

- 提出了 detector-tracker 设计的推理通道

流程包括一个轻量级的人体姿态估计检测器和一个姿态跟踪网络。跟踪网络预测关键点坐标。当画面中第一次出现人体时，我们启动人体检测器，定位出 ROI (region of interest)，随后跟踪器在 ROI 上预测 33 个关键点；当画面中不是第一次出现人体（即当前帧的前一帧出现了人体），我们不会启用检测器，而是直接从上一帧的关键点推导出 ROI。这样会大大提升模型的轻量化。

- 提出了基于人脸的人检测器，

目前主流的人体姿态估计算法在检测人体的后处理步骤中，都是使用的 NMS (non maximum suppression) 算法，不过 NMS 对于非刚性物体并不是很友好，尤其是类似与人体这类关节复杂度高的姿势场景。这是因为有大量的，模糊的候选框都满足非最大值抑制的交并比阈值。

注意到人脸比较刚性，特征的对比度相对较高，所以我们假设人脸是始终可见的。对于如何检测人脸，我们使用了一个由 Bazarevsky et al. 提出的 Blaze-

Face^[34] 模型。

- 将基于热图方法和基于回归方法相结合

结合两者的优势，使用编码器-解码器网络架构来预测所有关节的热图。同时后面跟着另一个编码器，直接回归到所有关节的坐标。值得注意的是，热图分支可以在推理过程中被丢弃，使得模型足够轻量级，可以在移动端上运行。

1.4 论文章节安排

本文共五章。绪论主要介绍了论文工作的研究背景和应用场景，对国内外的人体姿态估计算法做出了一个较为详细的概况，最后概述了本项目的工作内容和创新点。

第二章介绍了有关深度学习的基础知识和基本概念以及本文用到的损失函数，激活函数，优化方法等。

第三章主要介绍了 Blaze 算法的推理通道，神经网络结构，人体检测器，人体拓扑结构，复现过程等。

第四章是成果展示，主要包括了五个方面：

- 与官方模型及其他模型的性能对比；
- 适用于 pc 端的图片检测、视频检测和摄像头检测；
- 适用于手机移动端的摄像头实时检测；
- 在 unity 端虚拟机器人的检测；
- 使用 arduino 开发板驱动 sg90 舵机完成最终的机器人姿态模仿。

第2章 姿态估计相关理论基础

2.1 引言

人体姿态估计是人机交互的基础，学者们做了很多工作。从基于传感器的模型，到基于图模型的研究，再到如今基于深度学习方法的流行，其热度始终不减。但依旧无法满足应用场景。从绪论我们可以看出，当今人体姿态估计算法的方向是轻量化，快速化，准确化。

本章节主要介绍一些比较基础但十分重要的深度学习概念，为下面 BlazePose 算法的诞生做铺垫。首先我们会介绍卷积神经网络的各个组成部件，其次会深入了解反向传播算法，然后介绍常见的激活函数，最后为抑制过拟合现象介绍一些正则化方法。

2.2 卷积神经网络的组成

一个基本的卷积神经网络是由多个卷积层与池化层组成的卷积块以及全连接层相互连接而成。如图 2-1 所示。由 $M(2 \leq M \leq 5)$ 个连续的卷积层和 $b(b\text{取}0\text{或}1)$ 个池化层组成一个卷积模块。再由 N 个连续的卷积模块和 $K(0 \leq K \leq 2)$ 个全连接层堆叠而成。

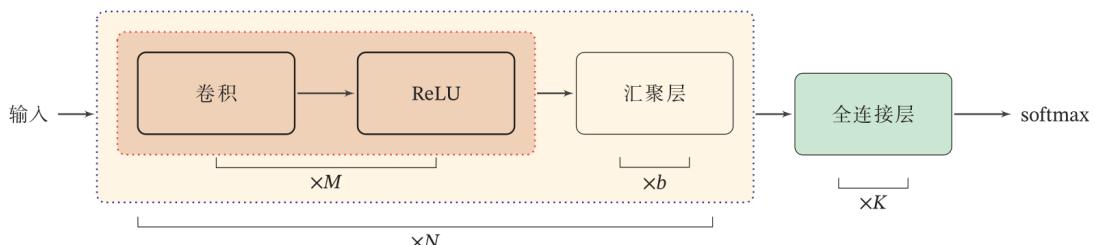


图 2-1 常用卷积网络整体结构

2.2.1 卷积层

卷积层的作用是提取特征，浅层网络可以提取到边缘，颜色等浅层信息；深层网络可以提取到形状，图案等深层的语义信息。其中，使用不同的滤波器即卷积核，提取到的信息也不尽相同。不失一般性，我们假设卷积神经网络处理的是二维图像，那么我们将通常使用三维的、大小是 $M \times N \times D$ 神经层，其中 M, N, D 分别为高度，宽度，深度。

由此可以假设卷积层的结构如下：

1. 由三维张量 $x \in \mathbb{R}^{M \times N \times D}$ 组成的输入特征映射组，其中 $1 \leq d \leq D$ ；
2. 由三维张量 $y \in \mathbb{R}^{M' \times N' \times P}$ 组成的输出特征映射组，其中 $1 \leq p \leq P$ ；
3. 由四维张量 $w \in \mathbb{R}^{U \times V \times P \times D}$ 组成的卷积核，其中 $1 \leq p \leq P, 1 \leq d \leq D$ 。

卷积层的三维结构表示如图 2-2 所示。

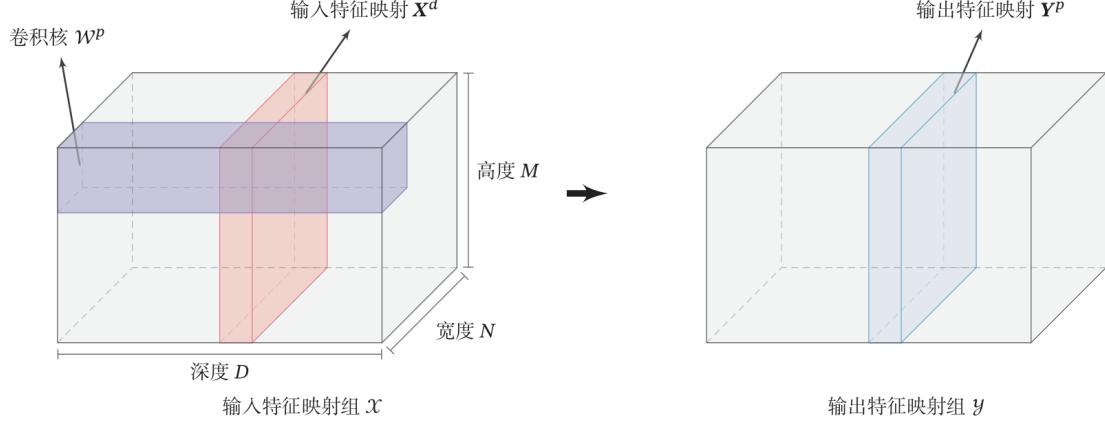


图 2-2 卷积层的三维结构表示

将卷积核 $W^{p,1}, W^{p,2}, \dots, W^{p,D}$ 和输入特征映射 X^1, X^2, \dots, X^D 一一对应分别进行卷积操作，并与偏置量 b 求和。可以得到卷积层输出 Z^p ，最后再经过激活函数的激活，即可得到输出特征映射 Y^p 。

$$Z^p = W^p \otimes X + b^p = \sum_{d=1}^D W^{p,d} \otimes X^d + b^p \quad (2-1)$$

$$Y^p = f(Z^p)$$

值得注意的是，对于激活函数 $f(\cdot)$ ，我们一般使用 ReLU 函数。

计算过程的流程图如图 2-3 所示。

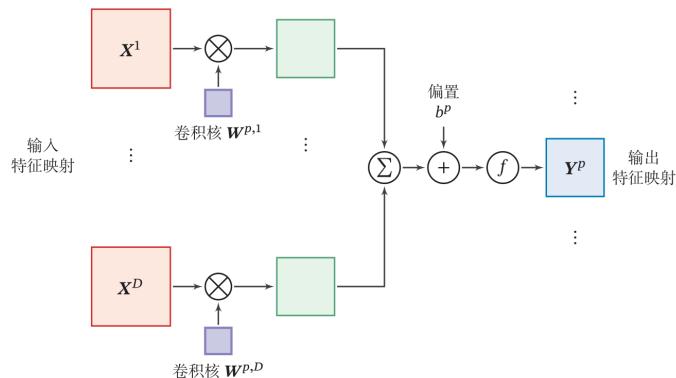


图 2-3 卷积层中的计算流程图

2.2.2 池化层

池化层通俗地说，池化就是将矩阵的各个子矩阵压缩。例如，如果我们想对一个 4×4 的矩阵进行 2×2 的池化，那么就会将 4×4 的矩阵拆分成4个 2×2 的子矩阵。并将每个子矩阵经过池化变成一个标量。一般而言，池化标准可以是最大池化，也可以是平均池化。对于前者，在每个子矩阵的元素中取出一个最大的作为结果；对于后者，则是将子矩阵的各个元素的平均值作为结果。从而达到降维的效果。

接下来，我们举一个采用 2×2 最大池化，步幅为2的例子。如图 2-3 所示。

首先，我们以一个 4×4 的输入矩阵为例，将其拆分成4个 2×2 的子矩阵。对左上角的红色子矩阵使用最大池化，得到池化结果是该区域的最大值6。以此类推，右上角的绿色矩阵结果为8，左下角的黄色区域池化结果为3，右下角的蓝色区域结果为4。最终我们得到了一个 2×2 的矩阵，即池化的最终结果。可以看出，池化可以在保留一定的原信息的同时有效地降低矩阵维数，简化后续计算。

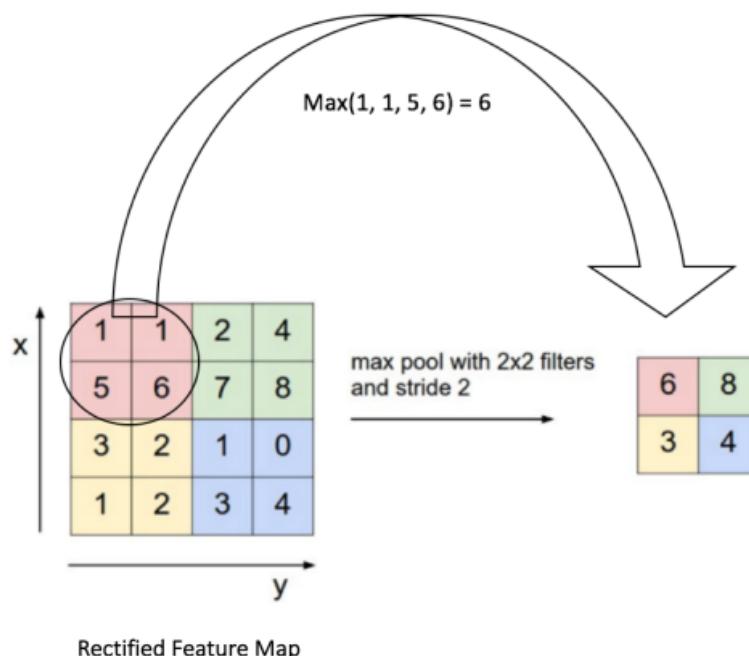


图 2-4 卷积层中从输入特征映射组 X 到输出特征映射 Y^p 的计算示例

2.3 卷积神经网络(CNN) 反向传播算法

2.3.1 DNN 的反向传播算法

要计算 DNN 的反向传播，我们首先得出输出层的 δ^L

$$\delta^L = \frac{\partial J(W, b)}{\partial z^L} = \frac{\partial J(W, b)}{\partial a^L} \odot \sigma'(z^L) \quad (2-2)$$

此外，使用第一数学归纳法，可以由 δ^{l+1} 的值算出 δ^l 如下：

$$\delta^l = \delta^{l+1} \frac{\partial z^{l+1}}{\partial z^l} = (W^{l+1})^T \delta^{l+1} \odot \sigma'(z^l) \quad (2-3)$$

最后，我们可以得到出 W, b 的梯度：

$$\begin{aligned} \frac{\partial J(W, b)}{\partial W^l} &= \frac{\partial J(W, b, x, y)}{\partial z^l} \frac{\partial z^l}{\partial W^l} = \delta^l (a^{l-1})^T \\ \frac{\partial J(W, b, x, y)}{\partial b^l} &= \frac{\partial J(W, b)}{\partial z^l} \frac{\partial z^l}{\partial b^l} = \delta^l \end{aligned} \quad (2-4)$$

简单介绍完 DNN 的反向传播算法的推到思路后，我们将公式套用到 CNN 中，但需要注意的是，CNN 与 DNN 还是有些许不同，这导致了我们需要解决几个问题，才能正确地套用。

2.3.2 CNN 的反向传播算法思想

鉴于 DNN 与 CNN 的区别，我们再研究 CNN 反向传播的时候遇到了如下几个关键问题：

1. 池化层没有激活函数，这导致了我们无法从池化结果反向传播到输出。对此我们可以令激活函数是 $\sigma(z) = z$ ；
2. 与 DNN 不同的是，图像经过池化层的前向传播时，受到了压缩，在数据被压缩的情况下，如何反向推导 δ^{l-1} ；
3. 与 DNN 还有一点不同的是，DNN 全连接层的输出是矩阵乘法得到的，而 CNN 的卷积层是通过卷积得到当前层的输出；
4. DNN 中没有卷积层的概念，所以在求 W 时，要考虑到于 W 使用的是卷积运算。

可以看出，第一点很容易就给出答案，所以问题 2, 3, 4 是解决 CNN 反向传播的难点。由于篇幅限制，下面直接给出结论：

对于问题 2，我们有

$$\delta_k^{l-1} = \frac{\partial J(W, b)}{\partial a_k^{l-1}} \frac{\partial a_k^{l-1}}{\partial z_k^{l-1}} = \text{upsample}(\delta_k^l) \odot \sigma'(z_k^{l-1}) \quad (2-5)$$

对于问题 3，我们有

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \delta_{11} & \delta_{12} & 0 \\ 0 & \delta_{21} & \delta_{22} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} * \begin{pmatrix} w_{22} & w_{21} \\ w_{12} & w_{11} \end{pmatrix} = \begin{pmatrix} \nabla a_{11} & \nabla a_{12} & \nabla a_{13} \\ \nabla a_{21} & \nabla a_{22} & \nabla a_{23} \\ \nabla a_{31} & \nabla a_{32} & \nabla a_{33} \end{pmatrix} \quad (2-6)$$

对于问题 4，我们有

$$\frac{\partial J(W, b)}{\partial W^l} = \frac{\partial J(W, b)}{\partial z^l} \frac{\partial z^l}{\partial W^l} = \delta^l * \text{rot180}(a^{l-1}) = \sum_{u,v} (\delta^l)_{u,v} \quad (2-7)$$

如此我们便可以得到算法 2-1。

2.4 激活函数

本质上，卷积神经网络就是由多个感知机组成的，所以，如果没有其他操作，最后的结果肯定相当于一个线性变换，这使得神经网络处理信息的能力很弱。故而我们引入激活函数，来模拟人脑神经元突触的激活效果，从而使得卷积网络脱离线性关系，以获得更强的学习能力。这里我们主要介绍三种激活函数：Sigmoid, tanh, ReLU。

2.4.1 Sigmoid 激活函数

Sigmoid 函数及其导数如图 2-5 所示。

从 sigmoid 函数的导数我们可以看出，它可以用 sigmoid 函数自身来表示，所以梯度计算较为方便。此外它的梯度也比较平滑，它将一个 $(-\infty, +\infty)$ 之内的实数值变换到区间 $[0, 1]$ 之间，还满足处处连续。但是它也有着很严重的缺点：一方面，求导计算量大，反向传播过程中计算 loss 值的梯度时还会涉及除法；另一方面，由于当输入值在 $[-4, 4]$ 之间时导数值比较大，其余部分导数值趋于 0，很容易出现梯度消失的情况，从而无法实现深层网络的训练。

算法 2-1 反向传播算法（批量梯度下降法）

Input: $m, L, K, F, P, S, k, \alpha, MAX, \epsilon$, 它们分别是图片样本数, 模型层数, 卷积核大小, 卷积核子矩阵维度, 填充大小, 步幅, 池化区域大小, 梯度迭代步长, 最大迭代次数, 停止迭代的阈值。

Output: CNN 各输出层的 W, b

- 1 将所有层的 W, b 初始化成随机值。
- 2 **for** $iter=1$ to MAX **do**
- 3 **for** $i = 1$ to m **do**
- 4 将 CNN 输入 a^1 设置为 x_i 对应的张量
- 5 **for** $l = 2$ to $L - 1$ **do**
- 6 分 3 种情况计算前向传播:
- 7 如果当前是全连接层: 则有 $a^{i,l} = \sigma(z^{i,l}) = \sigma(W^l a^{i,l-1} + b^{i,l})$
- 8 如果当前是卷积层: 则有 $a^{i,l} = \sigma(z^{i,l}) = \sigma(W^l * a^{i,l-1} + b^{i,l})$
- 9 如果当前是池化层: 则有 $a^{i,l} = pool(a^{i,l-1})$
- 10 **end**
- 11 对于第 L 层输出
- 12 层: $a^{i,L} = softmax(z^{i,L}) = softmax(W^{i,L} a^{i,L-1} + b^{i,L})$, 通过损失函数计算输出层的 $\delta^{i,L}$
- 13 **for** $l = L$ to 2 **do**
- 14 分 3 种情况计算反向传播:
- 15 如果当前是全连接层: $\delta^{i,l} = (W^{l+1})^T \delta^{i,l+1} \odot \sigma'(z^{i,l})$
- 16 如果当前是卷积层: $\delta^{i,l} = \delta^{i,l+1} * rot180(W^{l+1}) \odot \sigma'(z^{i,l})$
- 17 如果当前是池化层: $\delta^{i,l} = upsample(\delta^{i,l+1}) \odot \sigma'(z^{i,l})$
- 18 **end**
- 19 **end**
- 20 **for** $l = 2$ to L **do**
- 21 根据下面 2 种情况更新第 l 层的 W^l, b^l :
- 22 如果当前是全连接层: $W^l = W^l - \alpha \sum_{i=1}^m \delta^{i,l} (a^{i,l-1})^T b^l = b^l - \alpha \sum_{i=1}^m \delta^{i,l}$
- 23 如果当前是卷积层, 对于每一个卷积核有:
- 24 $W^l = W^l - \alpha \sum_{i=1}^m \delta^{i,l} * rot180(a^{i,l-1}), b^l = b^l - \alpha \sum_{i=1}^m \sum_{u,v} (\delta^{i,l})_{u,v}$
- 25 若 W, b 的变化值不大于阈值 ϵ , 跳出循环。
- 26 输出各层的 W 和 b , 其中 W 是线性关系系数矩阵, b 是偏倚向量。

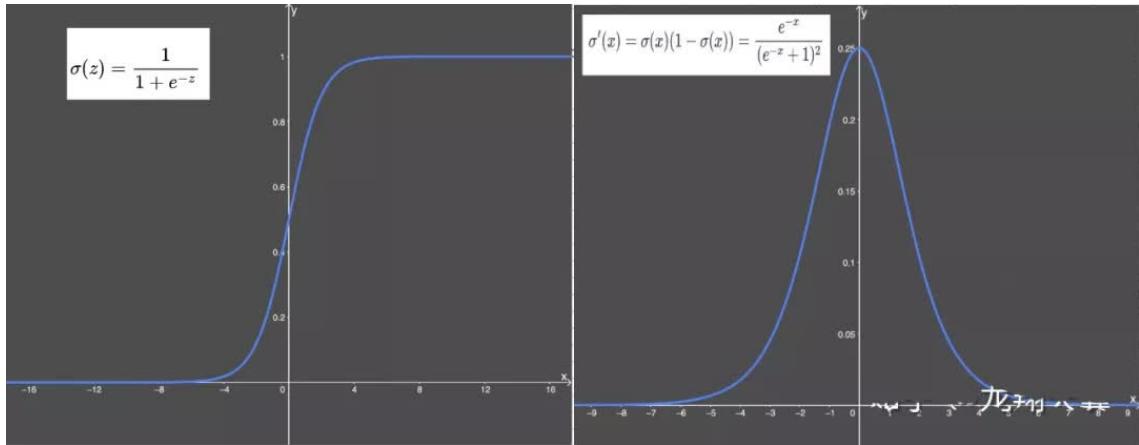


图 2-5 sigmoid 函数及其导数

2.4.2 tanh 激活函数

tanh 函数及其导数如图 2-6 所示。

与 sigmoid 函数不同的是，tanh 函数将输出域从 $(0, 1)$ 改到 $(-1, 1)$ ，让输出以 0 为中心，有便于神经网络归一化特征的学习。此外，tanh 函数的梯度消失问题比 sigmoid 要轻，收敛更快。但是，当输入太大或者太小的时候，tanh 函数的值无限接近 -1 或者 1 ，此时的梯度为 0，梯度消失问题依旧很严重。

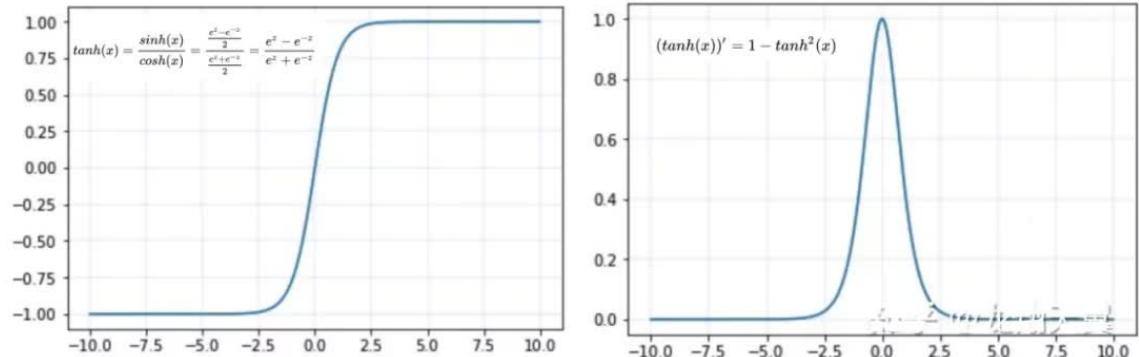


图 2-6 tanh 函数及其导数

2.4.3 整流线性单元 (ReLU)

ReLU 函数及其导数如图 2-7 所示。

为了缓解梯度消失问题，ReLU 函数被提出来。它简单高效，不涉及指数等运算；而且 ReLU 激活函数的导数在变动很大的情况下，会远大于 0；此外，ReLU 激活函数会使神经网络学习效率变高；最后，当 $x > 0$ 时，ReLU 函数的导数是常数，这有效地解决了梯度弥散现象。这些使得它成为当今最主流的激活函数之一。

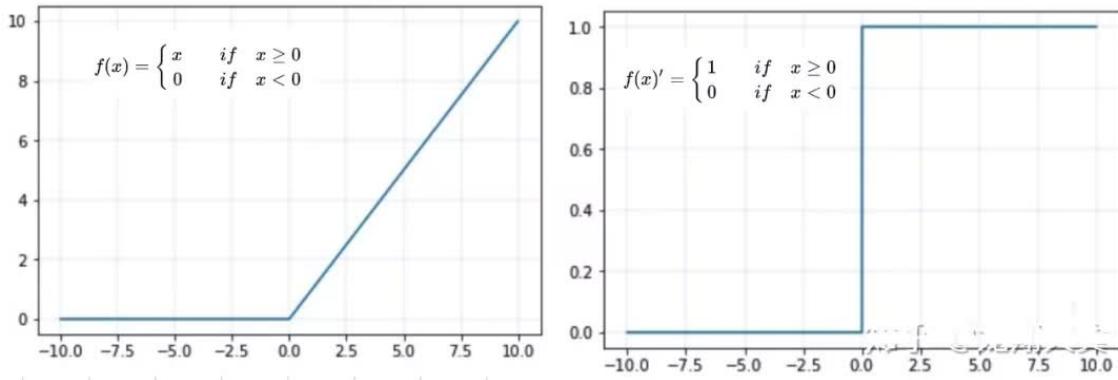


图 2-7 ReLU 函数及其导数

2.5 正则化

在机器学习中一切抑制过拟合现象的方法都可以被称作是正则化方法，例如集合学习，数据增强，dropout，修改损失函数等。这里我们主要介绍接下来工作中所用到的通过修改损失函数的正则化方法。

修改损失函数的正则化方法主要有 L1 正则化(见公式 2-8)，L2 正则化(见公式 2-9) 和 Smooth L1 正则化(见公式 2-10)。

$$L_1(x) = |x| \quad (2-8)$$

$$L_2(x) = x^2 \quad (2-9)$$

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \quad (2-10)$$

其中 \$x\$ 为预测值与真值之间的差异。

三个损失函数对 \$x\$ 的导数分别为：

$$\frac{dL_1(x)}{dx} = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{otherwise} \end{cases} \quad (2-11)$$

$$\frac{dL_2(x)}{dx} = 2x \quad (2-12)$$

$$\frac{ds_{smooth_{L_1}}(x)}{dx} = \begin{cases} x & \text{if } |x| < 1 \\ \pm 1 & \text{otherwise} \end{cases} \quad (2-13)$$

根据公式 2-11，\$L_1\$ 对 \$X\$ 的导数是常数。这将带来一个问题，那就是训练后期时，真值和预测值之间的差值很小，但 \$L_1\$ 对 \$X\$ 的导数仍然不变，导致神经网络难以继续收敛。更致命的是，\$L_1\$ 在零点导数不唯一，会影响训练的收敛。

观察公式 2-12， L_1 对 X 的导数与 X 呈正相关。所以刚开始训练时真值和预测值之间的差值很大，导致损失函数的梯度也很大，会使得训练不稳定。同时离群点会占据 loss 的主要部分，造成训练的失败。

而 Smooth L1 Loss(公式 2-13) 结合了 L1 Loss 以及 L2 Loss 的优点，它既可以较快地收敛，又能降低对离群点的敏感度，使得梯度变化较小，训练不易波动。

2.6 本章小结

本章主要介绍了卷积神经网络的一些信息，为后续研究 BlazePose 算法打好基础，重点阐述了 CNN 的组成部件，CNN 的反向传播算法，以及激活函数的选择和一些常见的正则化方法。

第3章 基于BlazePose的人体姿态估计算法研究

3.1 推理通道

在推断过程中，采用了 detector-tracker 设计。流程（见图 3-1）包括一个轻量级的人体姿态估计检测器，和一个紧随其后的姿态跟踪网络。当目前帧上有人体出现时，使用跟踪网络预测关键点坐标；当跟踪器不能检测出关键点坐标，即没有人出现时，检测器会在下一帧重新启用。

具体来说，模型会使用检测器定位图像的姿态 ROI，然后传给跟踪器，预测出 33 个关键点的坐标。对于视频流来说，检测器只会在第一次出现人脸之前运行。对于检测到人脸之后，会从前一帧的 33 个关键点中预测出 ROI。

3.2 神经网络结构

我们将两种主流的方法，即基于热图和基于回归相结合，如图 3-2 所示。热力图所在的网络层只在训练过程中出现，不包含回归。当训练完成时，热力图相对应的输出层将会被删除，从而降低模型的复杂度，提升模型的轻量化。同时，还使用了堆叠沙漏方法^[19]，但与之不同的是，本项目同时堆叠了一个 encoder-decoder 热图网络和一个回归 encoder 网络。

中间上面是输入图片，然后逐步向下，是个 bottom-up 的过程，每个 scale 都有向左和向右的横向链接。

左边从下到上，是个 top-down 的过程，和中间部分有横向链接 skip connections，这都和“Hourglass”一样，最上面是“Hourglass”部分输出的 heatmap，这个 heatmap 仅仅用来应用 loss 监督训练“Hourglass”部分生成中间的 embedding

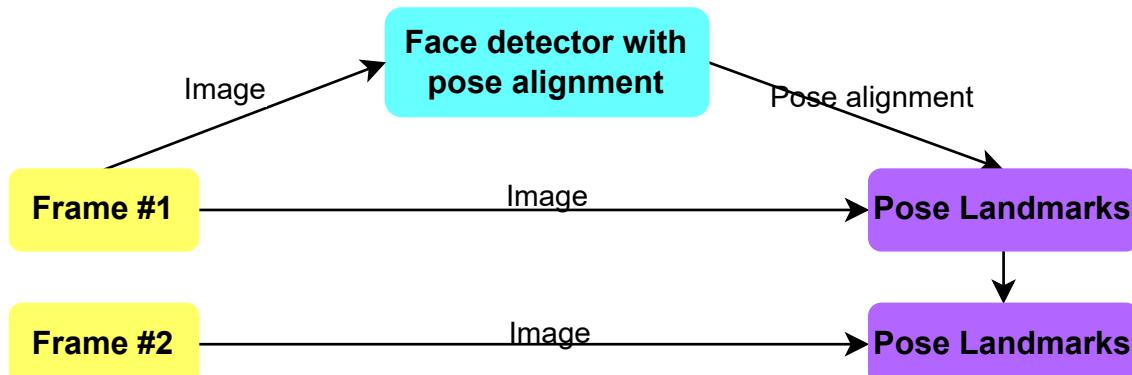


图 3-1 推理通道

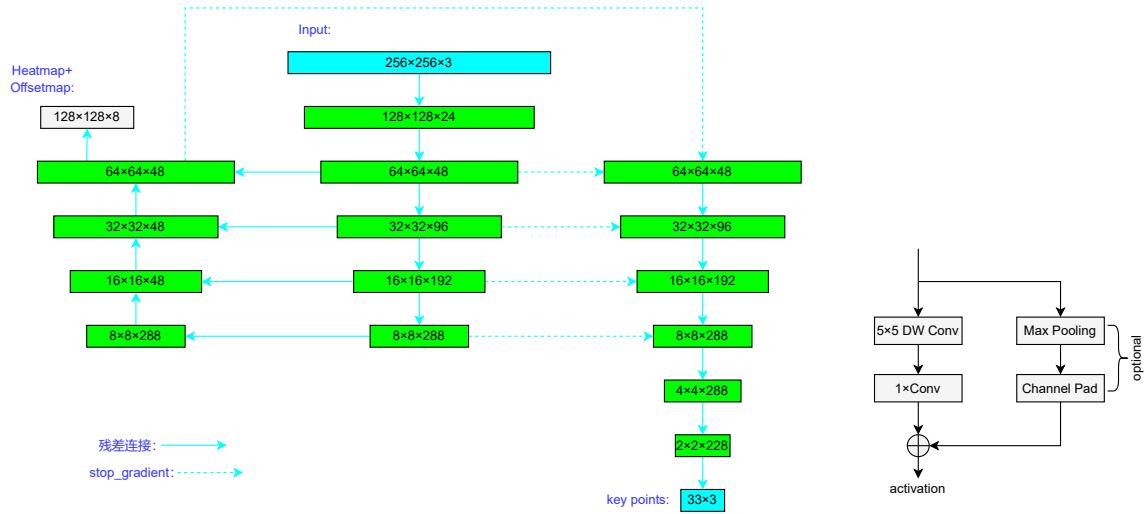


图 3-2 神经网络结构

特征，在预测以及 regression 部分都不参与。

右边从上到下整个是 regression encoder 网络，这部分不参与训练“Hourglass”部分，仅仅用来“后处理”。它每层有对应的输入，其中最上面的第一层输入来自两部分，分别是 bottom-up 以及 top-down 的同级分辨率特征（heatmap 没有参与过来，也就是砍掉了），最后输出 33 个关键点信息。

从图 3-2 中可以看出，本项目的模型，充分连接了网络的各个阶段，使其不再孤立，从而平衡了深浅层的特征。不同层次之间存在跨层连接，这样既可以发挥深层网络的特化语义信息，抽象信息；也可以充分发挥浅层网络提取出的细粒度的边缘、颜色、转角、斑块，这些底层的图像信息。图中的实线是残差连接，即可以回传梯度，虚线表示停止回传梯度。这样使得热图的预测准确率和坐标的精度都大大提升。

图 3-3 和图 3-4 分别展示了本项目模型每层的连接关系和每一层的具体实现细节。可以看出网络一共由二十三层，其中 Conv7a 至 Conv11 为热力图分支，Conv11 为热力图输出，Conv12a 至 Conv15 为回归分支，Conv15 为全连接层。中间层激活函数均为 Relu，最后一层的激活函数为 Sigmoid。

3.3 人体检测器

目前主流的人体姿态估计算法在检测人体的后处理步骤中，都是使用的 NMS 算法，不过 NMS 对于非刚性物体并不是很友好，尤其是类似与人体这类关节复杂度高的姿势场景。这是因为有大量的，模糊的候选框都满足非最大值抑制的交并比阈值。

因此，本项目摒弃了 NMS 算法，而采用人脸检测框。这是因为在大多数

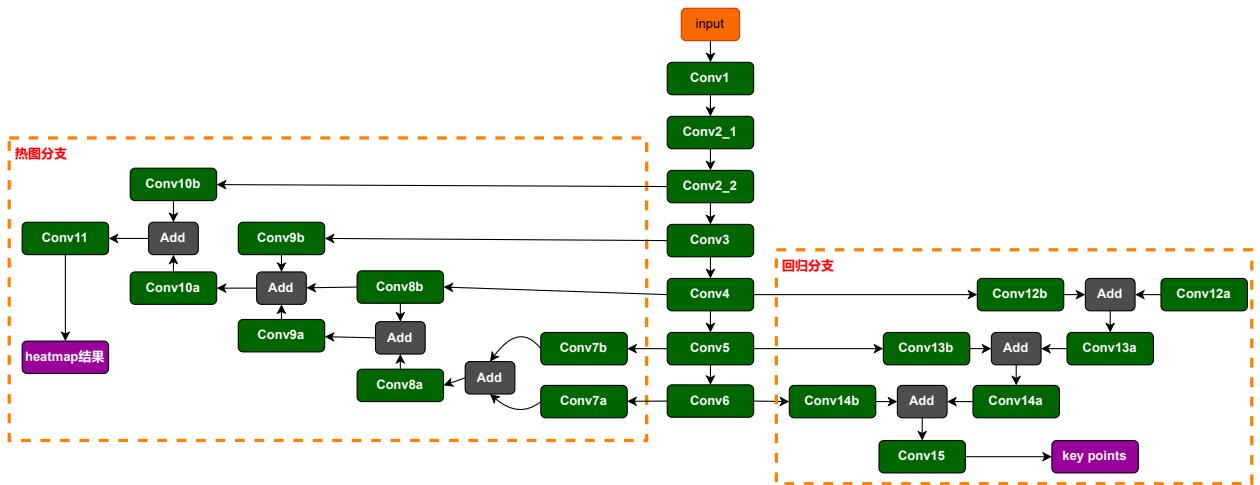


图 3-3 层与层之间的连接关系

情况下，人脸较为刚性，是关于躯干的最强信号。因此，我们做出了一个假设，即在单人姿态估计中，人脸是必须要出现的。

该检测器(见图 3-5)来自谷歌的轻量级 BlazeFace 模型，它预测了人体两个髋关节的中点作为整个人体的中心，并以此为圆心，画一个能够包含整个人体的最小圆，这就是本项目中的 ROI。对于人体倾斜的情况，还预测了人体中心与两个眼睛的中心之连线和铅垂线的夹角，将图像旋转这个夹角即可得到标准的人体图像。

3.4 人体拓扑结构

我们结合了 BlazeFace、BlazePalma 和 Coco 使用的数据集，并研究了他们的并集之超集，提出了人体 33 个关键点。

和 OpenPose^[35] 以及 Kinect^[4] 的关键点不同的是，本文在脸部，手部和脚部添加了更多的关键点，用来估计后续模型的 ROI。拓扑结构如图 3-6 所示。

此外，各个关键点的名称详见附录 1 [BlazePose 各关键点的名称](#)。

3.5 输入输出

由复现的网络结构图 3-4 可知，模型的输入是：

视频帧中检测到人的区域。大小是 $256 \times 256 \times 3$ ，在垂直身体姿势中以两个髋关节的中部为中心。通道顺序是 RGB。

模型的输出是 33 对 3 元组 (X, Y, Z) ，其中 33 对表示 33 个关键点，3 对分别是关键点在 X, Y, Z 轴上的坐标：

1. X, Y 坐标是感兴趣区域的局部坐标，范围为 $[0.0, 255.0]$ ；

2. Z 坐标与 X 和 Y 坐标一样以“图像像素”测量，表示相对于臀部平面的距离。正值表示在臀部的后面，负值表示在臀部和相机之间。这样就可以知晓 33 个关键点，尤其是手部和腿部的 Z 轴位置，可以充分发挥图像的信息。但 Z 坐标与 X 和 Y 坐标不一样的是， X 和 Y 坐标通过人工注释获得，而 Z 坐标是通过将合成数据（GHUM 模型^[36-37]）拟合到 2D 模型中，是按比例计算的。

3.6 本章小结

在本章中，我们对 BlazePose 算法做出了细致的研究，包括其推理通道，神经网络结构，人检测器，关键点模型等等，其中着重介绍了本项目复现出的神经网络结构。

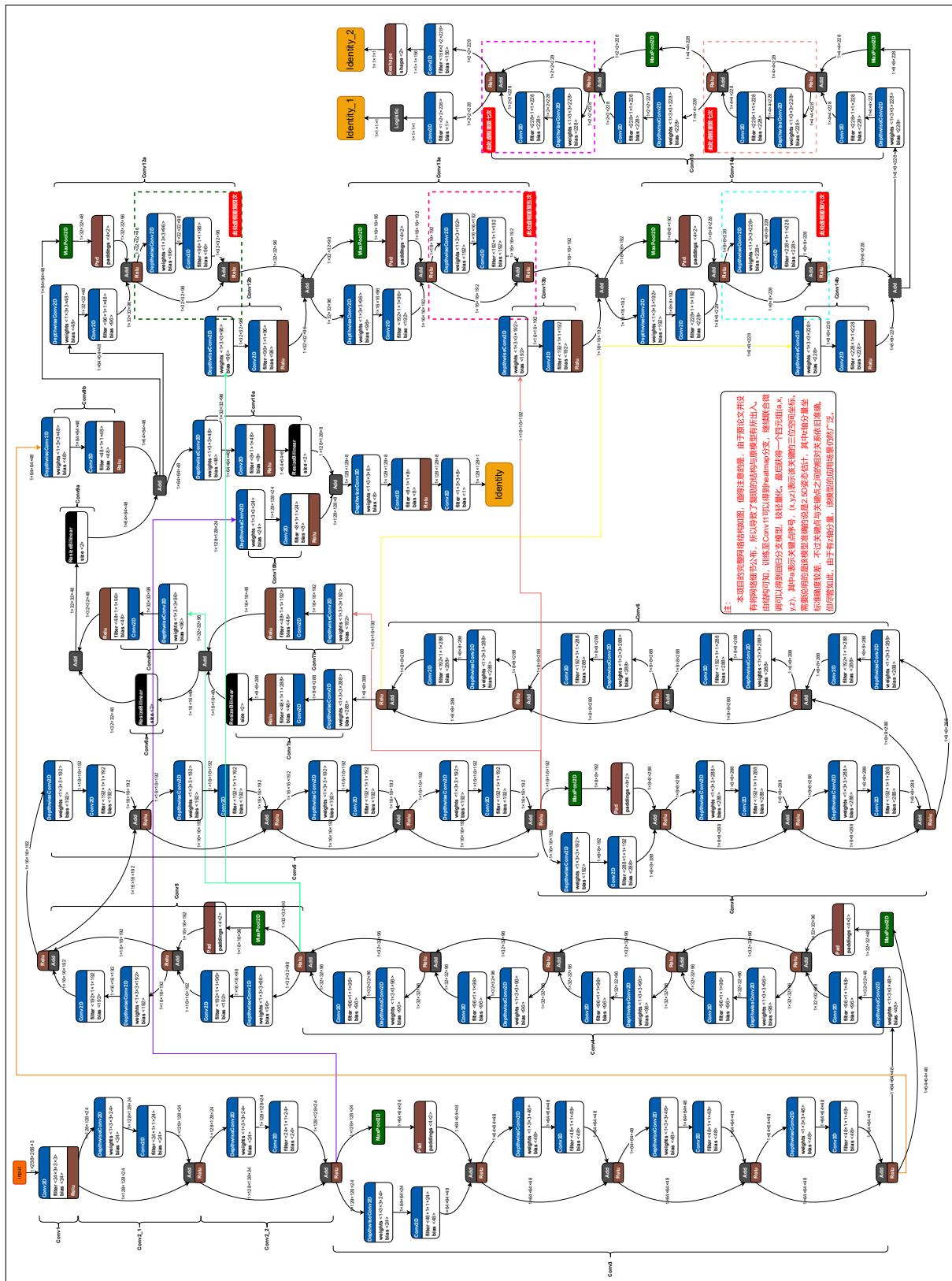


图 3-4 层与层之间的连接关系和每层的实现细节

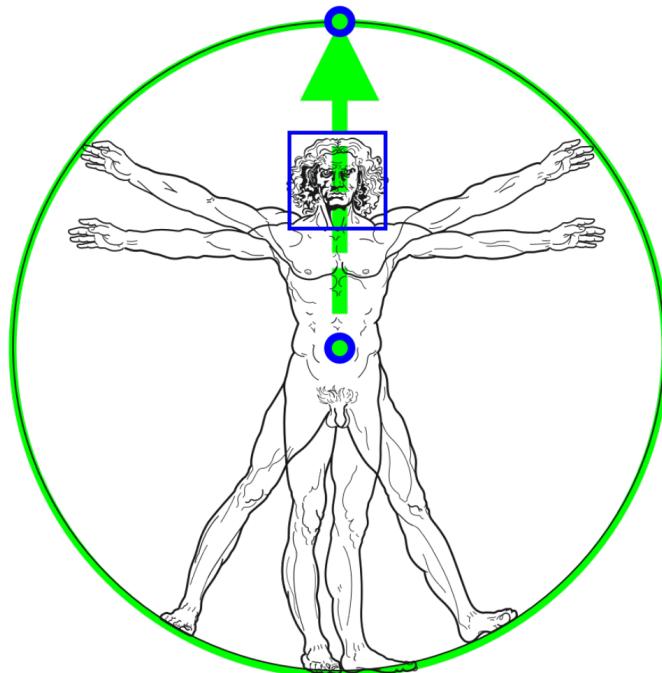


图 3-5 从维特鲁威人中受到启发的检测器

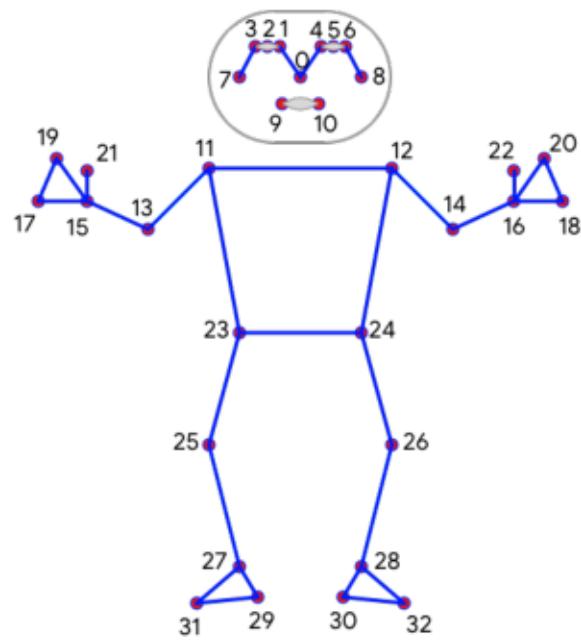


图 3-6 33 个关键点

第 4 章 基于 BlazePose 的训练、姿态识别与模仿

4.1 训练

如图 3-2 所示，我们将两种主流的方法，即基于热图和基于回归相结合。热力图所在的网络层只在训练过程中出现，不包含回归，即只用右塔来训练中塔和左塔。当训练完成时，热力图相对应的输出层将会被删除，即移除右塔，从而降低模型的复杂度，提升模型的轻量化。

训练步骤主要分如下几步：

- 训练

- 下载 LSP 数据集。

手动下载并解压缩数据集至 ./dataset 中。

- 预训练热图分支。

将 config.py 中的 train_mode 设置 train_mode = 0，然后运行 python train.py。

- 微调联合回归分支。

将 config.py 中的 train_mode 设置为 train_mode = 1，并且设置一个合适的 best_pre_train 值，best_pre_train 是训练损失下降但测试准确的最佳时期数。然后运行 python train.py。

- 测试

将 config.py 中的 epoch_to_test 设置为想要测试的 epoch，然后运行 python test.py。

如图 4-1 所示，最后训练的模型在 LSP 数据集上的正确率可以达到 82.5%。

4.2 性能比较

同时，本项目还与官方模型以及其它模型做了比较，其中，我们对模型在 LSP 数据集上做了 PCK@0.2 的结果对比，见表 4-1 和图 4-2。此外，为了验证模型的轻量化，我们还在 CPU 型号是 AMD Ryzen 7 5800H with RadeonGraphics 3.20 GHz 上做了帧数对比，见表 4-2 和图 4-3。

从表中数据我们可以看出，虽然我们根据官方 Full 版本复现的模型在准确度方面与原版有着差距，但是也已经达到了可以应用的程度，并且在轻量化方面也做到了媲美原版。

图 4-1 模型在 LSP 数据集上的正确率

表 4-1 LSP 数据集上一些模型的 PCK

Model	LSP Dataset(PCK@0.2)
官方模型 (Heavy)	97.5
官方模型 (Full)	95.7
官方模型 (Lite)	93.5
AlphaPose ResNet50	96.0
Apple Vision	88.6
Ours	82.5

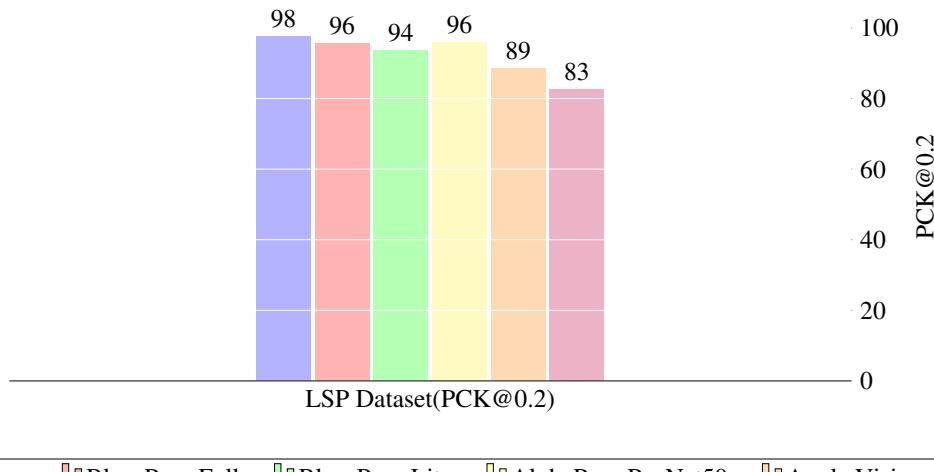


图 4-2 模型在 LSP 数据集上 PCK 值的柱状图

表 4-2 一些模型对于视频检测的帧数

Model	FPS(AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz)
官方模型 (Heavy)	38
官方模型 (Full)	32
官方模型 (Lite)	25
OpenPose	6
Ours	29

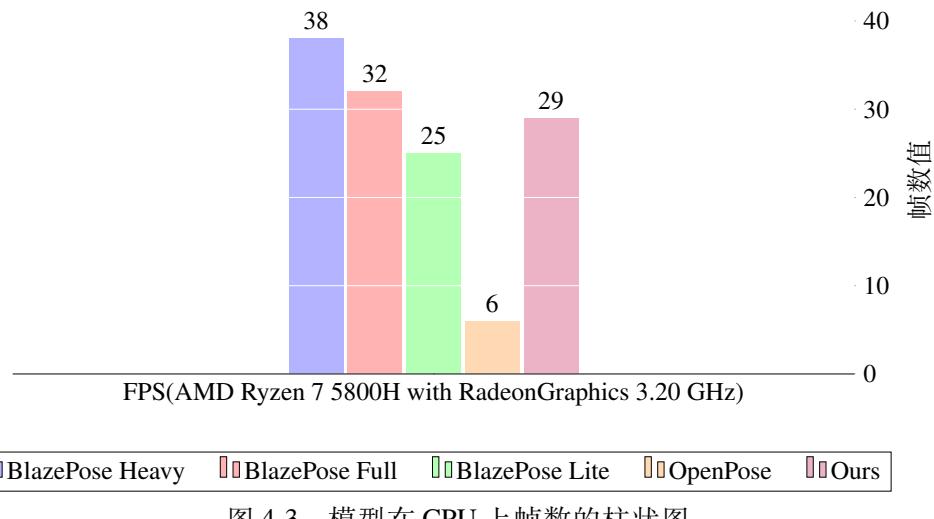


图 4-3 模型在 CPU 上帧数的柱状图

4.3 API

对于模型的部署，本项目没有设计函数，而是将我们的模型替换官方模型，再用 MediaPipe 库调用我们的模型，所以在部署模型之前，我们有必要介绍一下 API。

1. STATIC_IMAGE_MODE

默认值是 `false`，可以选择 `true`。如果选择 `false`，合适于检测视频，如图 3-1 所示，检测器只会在第一次出现人脸之前运行，当检测到人脸之后，检测器定位图像的姿态 ROI，然后传给跟踪器，预测出 33 个关键点的坐标。对于检测到人脸之后，会从前一帧的 33 个关键点中预测出 ROI，而不会再次启用检测器；对于 `true`，就是每一帧都会启用检测器，这会大大加大 CPU 的负担。

2. MODEL_COMPLEXITY

可选值是 0, 1, 2 分别对应着官方模型的 Lite、Full 和 Heavy 版本。由于我们复现的是官方的 Full 模型，所以在此我们只能选择 1，值得说明的是，该参数的默认值就是 1。

3. UPPER_BODY_ONLY

设置为 true 表示识别出全身 33 个关键点；设置为 false 表示识别出上半身的 25 个关键点。

4. SMOOTH_LANDMARKS

设置为 true，将会平滑关键点，从而减少抖动。但如果检测单张图像，即 STATIC_IMAGE_MODE 也设置为 true，则忽略该参数。

5. MIN_DETECTION_CONFIDENCE

置信度阈值，范围是 [0.0, 1.0]，默认设置为 0.5。在人体检测器中，如果该像素点的置信度超过了设定的阈值，则认为该像素点为人体。

6. MIN_TRACKING_CONFIDENCE

追踪阈值，范围是 [0.0, 1.0]，默认设置为 0.5。在追踪器中，如果超过该阈值，则表示成功检测到 33 个关键点，否则将会在下一帧中调用人体检测器。但如果 STATIC_IMAGE_MODE 也设置为 true，即检测单张图像，则忽略该参数。

4.4 基于 BlazePose 的姿态识别

4.4.1 电脑端的姿态识别

这里对于在电脑端部署模型，本项目并没有另外设计函数，本项目采取的方法是将训练好的.h5 模型用 tensorflow 工具转换成.tflite 模型，再将其替换谷歌官方的 MediaPipe 库中的模型，从而可以直接调用谷歌 MediaPipe 库的方法来测试我们的模型。

4.4.1.1 检测人物并蒙版抠图

该部分主要将图像中的人物检测出来并蒙版抠图，主要是为后续的关键点检测提供 ROI。

值得注意的一点是，我们需要使用 opencv 库提供的 cvtColor 颜色空间转换函数，将读入的 BGR 格式转换成 matplotlib 可输出的 RGB 格式。

实现方法主要是使用了 Selfie Segmentation 工具包，该工具包会给出每个像素点是人体一部分的概率，我们设定一个阈值（0.5），如果概率小于阈值，我们不输出该像素点；如果概率大于该阈值，我们则认为该像素点是人体的一部分，输出该像素点。

效果如图 4-4 所示。

4.4.1.2 单张图片检测

本部分完成的是对于单张图片的 BlazePose 关键点检测。函数主要调用了 MediaPipe 库。



图 4-4 左上为原图，后上是蒙版左下是前景人像，右下为抠掉前景人像的背景

同时由于单张图片的检测会使得模型第一次检测到人体，因此一定会调用人检测器定位 ROI。

我们会获得 33 个关键点的各自像素坐标，然后将该关键点染色并连接
对于各个关键点用不同的颜色加以区分。

主要代码如下：

```

for i in range(33):
    h, w = img.shape[0], img.shape[1]

    cx = int(results.pose_landmarks.landmark[i].x * w)
    cy = int(results.pose_landmarks.landmark[i].y * h)
    cz = results.pose_landmarks.landmark[i].z

    radius = 10

    if i == 0: # 鼻
        img = cv2.circle(img, (cx, cy), radius, (0, 0, 255), -1)
    elif i in [11, 12]: # 肩
        img = cv2.circle(img, (cx, cy), radius, (223, 155, 6), -1)
    elif i in [23, 24]: # 臼
        img = cv2.circle(img, (cx, cy), radius, (1, 240, 255), -1)
    elif i in [13, 14]: # 肘
        img = cv2.circle(img, (cx, cy), radius, (140, 47, 240), -1)
    
```

```

elif i in [25,26]: # 膝
    img =cv2.circle(img,(cx,cy), radius, (0,0,255), -1)
elif i in [15,16,27,28]: # 手腕、脚腕
    img =cv2.circle(img,(cx,cy), radius, (223,155,60), -1)
elif i in [17,19,21]: # 左手
    img =cv2.circle(img,(cx,cy), radius, (94,218,121), -1)
elif i in [18,20,22]: # 右手
    img =cv2.circle(img,(cx,cy), radius, (16,144,247), -1)
elif i in [27,29,31]: # 左脚
    img =cv2.circle(img,(cx,cy), radius, (29,123,243), -1)
elif i in [28,30,32]: # 右脚
    img =cv2.circle(img,(cx,cy), radius, (193,182,255), -1)
elif i in [9,10]: # 嘴
    img =cv2.circle(img,(cx,cy), radius, (205,235,255), -1)
elif i in [1,2,3,4,5,6,7,8]: # 眼、脸颊
    img =cv2.circle(img,(cx,cy), radius, (94,218,121), -1)
else: # 其它关键点
    img =cv2.circle(img,(cx,cy), radius, (0,255,0), -1)

look_img(img)

```

结果如图 4-5 所示。

4.4.1.3 在线摄像头检测

本部分主要是使用 BlazePose 模型对摄像头实时检测。

主要代码和单张图片类似，不过输入换成了摄像头。

另外值得说明的是，对于连续视频流，我们只会在第一个人体出来的帧之前帧调用人检测器，此后的帧只会在上一帧的 33 个关键点中预测出当前帧的 ROI。从而使得模型更加轻量化，更加实时化。

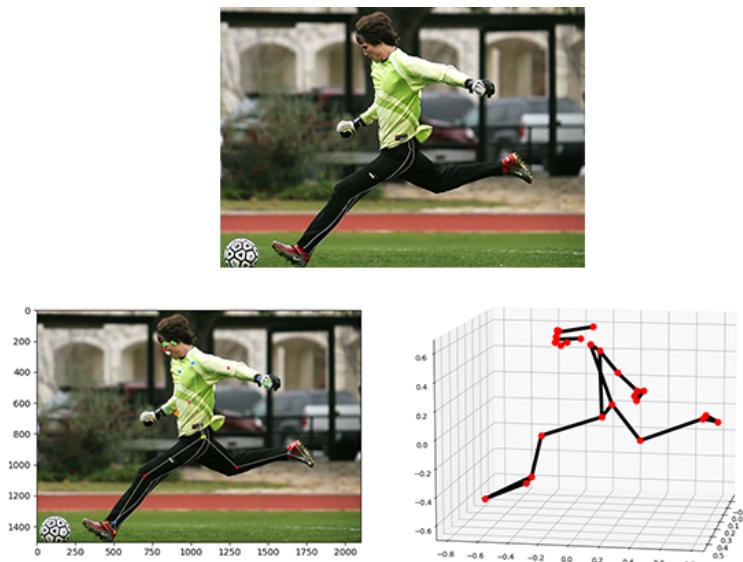


图 4-5 上方为原图，左下为关键点骨架图，右下为三维图

结果如图 4-6 所示。

4.4.1.4 离线视频检测

本部分主要是使用 BlazePose 模型对离线视频检测。

本部分核心代码和实时检测摄像头相同，主要加了一个生成视频的函数。这样使得输入输出均为视频。

结果如图 4-7 所示。

4.4.2 移动端的姿态识别

对于移动端的部署，本项目采用腾讯的 TNN^①轻量化框架，其提供了各种框架模型(如 TensorFlow、PyTorch、Caffe 等)转化为 TNN 模型文件的脚本，还有丰富的 demo 可供学习。其架构图如图 4-8 所示。

首先我们使用 TNN 官方的转换脚本^②将我们的.tflite 模型转换成.tnn 模型

然后编译 TNN 引擎，我们选择了 arm 和 OpenCL 这两个平台。对于这两个平台，TNN 都有极其方便的脚本^③。

最后，我们构建了一个调用编译好的 TNN 引擎进行推理的应用程序，从而在手机端部署了我们的模型。

移动端 app 对于摄像头实时检测的结果截图如图 4-9 所示。

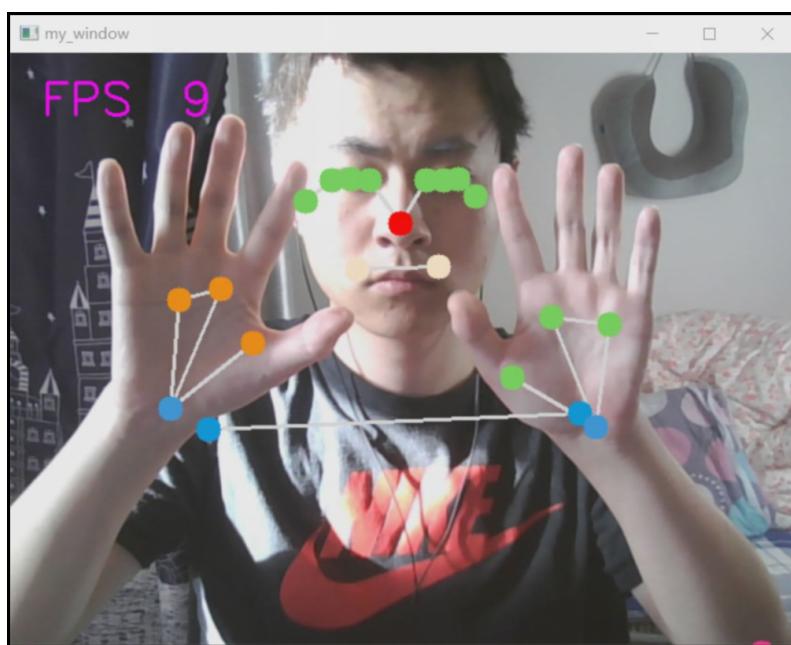
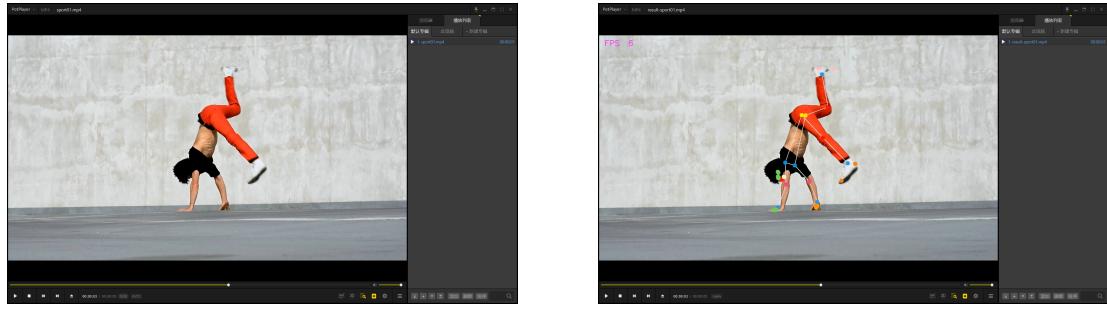


图 4-6 摄像头实时检测截图

① <https://github.com/Tencent/TNN>

② <https://github.com/Tencent/TNN/blob/master/doc/cn/user/convert.md>

③ <https://github.com/Tencent/TNN/blob/master/doc/cn/user/compile.md>



(a) 原视频截图

(b) 输出视频截图

图 4-7 离线视频检测

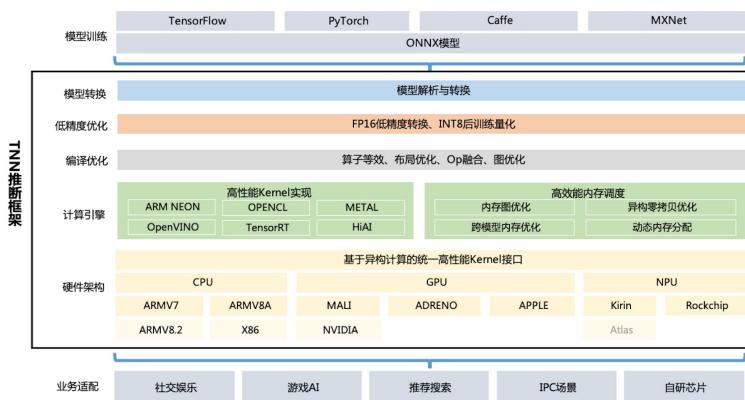


图 4-8 TNN 架构图

4.5 基于 BlazePose 的姿态模仿

4.5.1 虚拟机器人的姿态模仿

本部分使用了 unity3d 创建了一个拥有 33 个关键点的虚拟火柴人。

再用 python 实时检测摄像头或者视频输入。获得每个点的相对位置和角度角度。使用 UDP，创建了在本机“127.0.0.1”上的 5052 端口，在此端口上进行数据传输，将每个点的位置传输到 unity 中，从而使得机器人能够姿态模仿。

对于 unity3d 的虚拟火柴人进行姿态模仿结果如图 4-10 所示。

4.5.2 真实机器人的姿态模仿

对于真实机器人，我们使用 arduino 开发板和 SG90 舵机来完成实验，arduino 开发板指导 SG90 舵机调整旋转角度，从而达到完成姿态模仿的效果。至于数据传输，我们使用 USB 串联，主要库为 serial。

以右肘部姿态模仿为例，如图 3-6 所示，我们只需计算 12-14-16 的夹角即可。

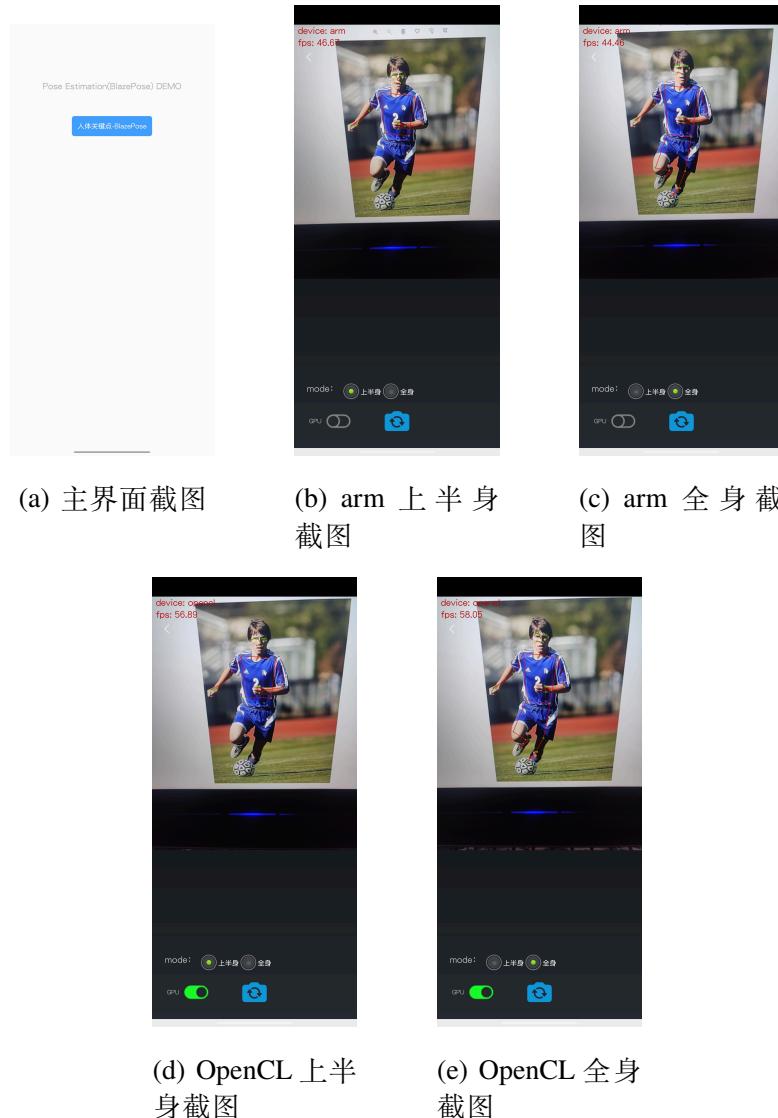


图 4-9 移动端 app 截图

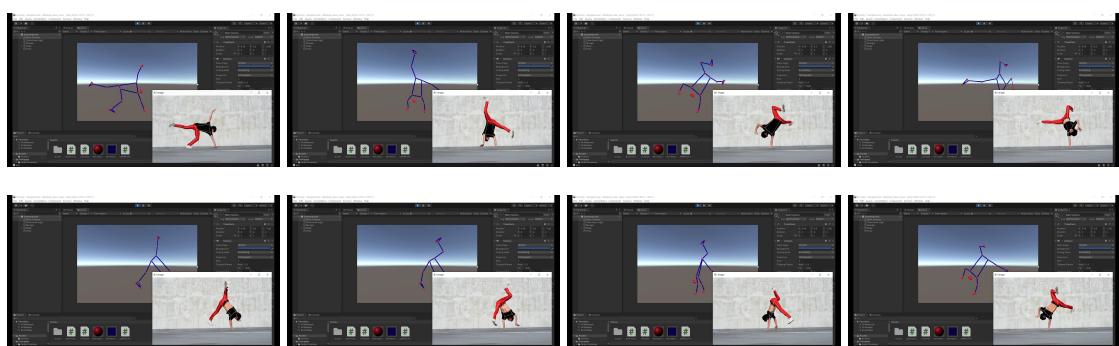


图 4-10 unity 虚拟火柴人姿态模仿截图

记关键点 12 为 P_1 , 关键点 14 为 P_2 , 关键点 16 为 P_3 , 如图 4-11 所示, 我们使用 `math.atan2` 函数, 分别计算出线 P_3P_2 和 X 轴的夹角 $\angle P_3P_2X$ 和线 P_1P_2 和

X 轴的夹角 $\angle P_1P_2X$ ，然后再用 $\angle P_3P_2X - \angle P_1P_2X$ 即可得到 $\angle P_1P_2P_3$ ，即右肘部对应的舵机应该旋转的角度。

计算角度的关键代码如下：

```
def findAngle(self, img, p1, p2, p3):
    # Get the landmarks
    x1, y1 = self.lmList[p1][1:3]
    x2, y2 = self.lmList[p2][1:3]
    x3, y3 = self.lmList[p3][1:3]

    # Calculate the Angle
    angle = math.degrees(math.atan2(y3 - y2, x3 - x2) -
                         math.atan2(y1 - y2, x1 - x2))
    if angle < 0:
        angle += 360

    return angle
```

由于经费有限，不失一般性，在本次实验中我们仅以右肘部单节点姿态模仿为例。对于全节点的机器人，我们只需按照单节点同样原理计算出多节点的角度并传到 arduino 开发板中即可。

最终效果如图 4-12 所示

从时间同步性以及动作准确度的角度来说，当摄像头读取的人体姿态发生变换的同时，机器人也会基本同步完成姿态转变。体现了本模型及时性和轻量化。进一步说明该模型已经达到实用的程度。

4.6 本章小结

本章是实验章节，主要是复现了 BlazePose 算法，并且将复现的模型与原版模型以及一些主流的模型进行了性能对比，在得到一个可以实用的模型后，本

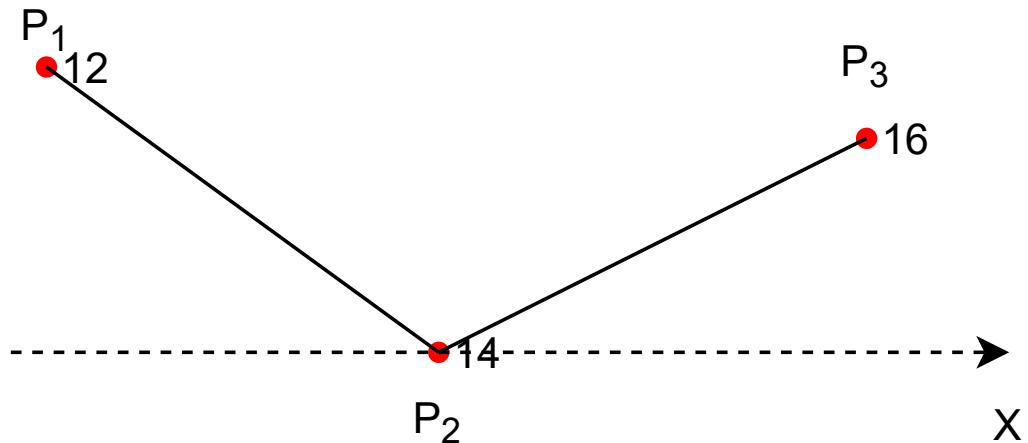


图 4-11 角度计算图示

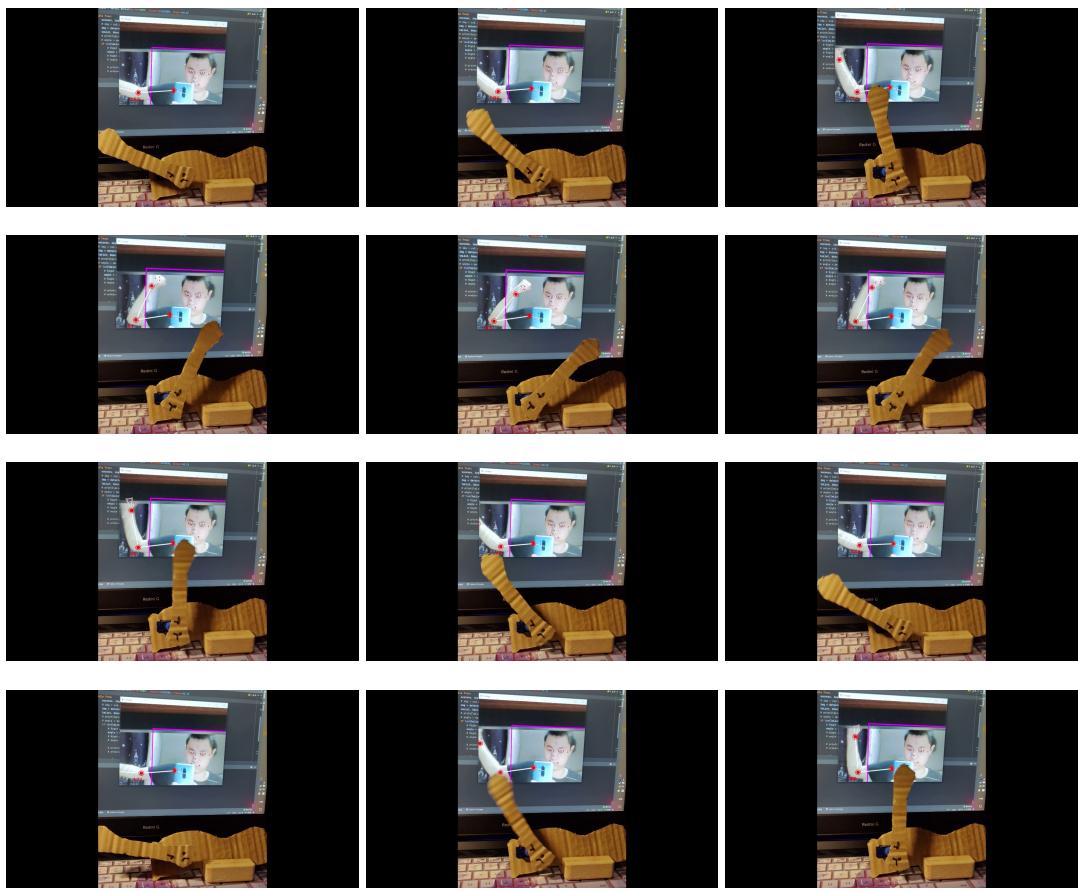


图 4-12 真实机器人单右肘部姿态模仿截图

章又将其部署到了电脑端，移动端，虚拟机器人和真实机器人上。最后，得出结论：从机器人模仿的准确度和实时性而言，本项目以达到了预期的效果。

结 论

在本项目的结论中，我们首先概括论文的主要内容和贡献，然后对存在的不足之处提出展望。

单人姿态估计是视觉领域难度极大的研究方向之一。但在应用场景上，单人姿态估计表现出了极大的需求，虚拟试衣、智能安防、体育健身、VR 技术等场景都亟需该算法的落地应用。

人机交互的主要目的是使机器人能够学习和了解人，能领会和模仿人的语言和行为。为了使人机交互自然，必须引入类似于人与人之间的沟通方式，即依赖语音与视觉。在这种背景下，人体姿态估计在人机交互方面有着举足轻重的作用。本文就是对此提出了基于人体姿态估计的机器人姿态跟踪算法，系统地完成了足够轻量化的人体姿态估计算法，并落实到了移动端，机器人端。具体的研究内容和研究成果如下：

1. 系统全面地介绍了人体姿态估计所涉及领域的主要内容，包括卷积神经网络的组成、反向传播算法、激活函数以及正则化。并且介绍了国内外对于单人姿态估计算法的发展历程以及各自的优势，为后续的 BlazePose 算法研究打下基础。

2. 研究了 BlazePose 算法，将目前最主流的基于热图的姿态估计和基于回归的姿态估计相结合。使用编码器-解码器网络架构来预测所有关节的热图。同时后面跟着另一个编码器，直接回归到所有关节的坐标。值得注意的是，热图分支可以在推理过程中被丢弃，使得网络足够轻量化，使得其可以架构在移动端和机器人端。

3. 在目标检测的后处理步骤中，不使用 NMS 算法，而是假设人脸这一相对刚性的目标一直会出现。因为多个不明确的框满足 NMS 算法的交并集 (IoU) 阈值，而人脸检测器则打破了这一限制。

4. 同时我们拓展了现在比较基础的 17 的关键点，增加至 33 个关键点，使得模型所表达的语义信息更加丰富，以便于可以用到更复杂的应用场景。

5. 然后，我们还对比了重构的模型与官方模型以及其它模型的性能，结果显示，重构模型的准确率能达到 82.5%，并且在帧数上也远超 OpenPose 接近官方模型。

6. 最后，我们一步步将模型部署到电脑端，移动端，虚拟机器人以及真实

机器人上。从时间同步性以及动作准确度的角度来说，该模型已经达到实用的程度。

本次研究复现的模型虽然达到了一定的效果，但依然存在一些局限性，结合市场需求以及国内外目前研究内容，对该模型作出如下展望：

1. 进一步提高模型的轻量化和性能，使得模型能在部署在性能较差的机器人上，并且使得机器人处理更加复杂的动作，改善交互体验。
2. 该模型是 2.5D 模型，虽然拥有着 Z 轴的分量，但是语义信息的损失极大，如何在保证模型轻量化的同时提高 Z 轴分量信息的准确度是一大难题。
3. 人体动作关键帧的准确提取。机器人通过自身的视觉系统捕获人体动作序列帧时，不用对每一帧图像进行处理，只需处理关键帧，其余的帧通过插值的方法估计出即可。如何得到这些动作关键帧需要进一步研究。

参考文献

- [1] 杨川. 基于深度学习的人体姿态估计技术研究 [D]. [出版地不详]: 哈尔滨工业大学, 2019.
- [2] CUI A, MCKEE D, LAZEBNIK S. Dressing in order: Recurrent person image generation for pose transfer, virtual try-on and outfit editing[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. [S.l.: s.n.], 2021: 14638-14647.
- [3] 戴汉彬. 基于深度学习的人体姿态估计技术研究 [D]. [出版地不详]: 电子科技大学, 2021.
- [4] ZHANG Z. Microsoft kinect sensor and its effect[J]. IEEE multimedia, 2012, 19(2): 4-10.
- [5] FELZENZWALB P F, HUTTENLOCHER D P. Pictorial structures for object recognition[J]. International journal of computer vision, 2005, 61(1): 55-79.
- [6] PISHCHULIN L, ANDRILUKA M, GEHLER P, et al. Strong appearance and expressive spatial models for human pose estimation[C]//Proceedings of the IEEE international conference on Computer Vision. [S.l.: s.n.], 2013: 3487-3494.
- [7] TOMPSON J J, JAIN A, LECUN Y, et al. Joint training of a convolutional network and a graphical model for human pose estimation[J]. Advances in neural information processing systems, 2014, 27.
- [8] CARREIRA J, AGRAWAL P, FRAGKIADAKI K, et al. Human pose estimation with iterative error feedback[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016: 4733-4742.
- [9] TOMPSON J, GOROSHIN R, JAIN A, et al. Efficient object localization using convolutional networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2015: 648-656.
- [10] HU P, RAMANAN D. Bottom-up and top-down reasoning with hierarchical rectified gaussians[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2016: 5600-5609.
- [11] PISHCHULIN L, INSAFUTDINOV E, TANG S, et al. [C]//[S.l.: s.n.].

- [12] LIFSHITZ I, FETAYA E, ULLMAN S. Human pose estimation using deep consensus voting[C]//European Conference on Computer Vision. [S.l.]: Springer, 2016: 246-260.
- [13] GKIOXARI G, TOSHEV A, JAITLEY N. Chained predictions using convolutional neural networks[C]//arXiv preprint arXiv:1605.02346. [S.l.: s.n.], 2016.
- [14] RAFI U, LEIBE B, GALL J, et al. An efficient convolutional network for human pose estimation.[C]//BMVC: volume 1. [S.l.: s.n.], 2016: 2.
- [15] BELAGIANNIS V, ZISSERMAN A. Recurrent human pose estimation[C]//2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017). [S.l.]: IEEE, 2017: 468-475.
- [16] INSAFUTDINOV E, PISHCHULIN L, ANDRES B, et al. [C]//[S.l.: s.n.].
- [17] WEISE E, RAMAKRISHNA V, KANADE T, et al. Convolutional pose machines[C]// Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2016: 4724-4732.
- [18] BULAT A, TZIMIROPOULOS G. Human pose estimation via convolutional part heatmap regression[C]//ECCV. [S.l.: s.n.], 2016.
- [19] NEWELL A, YANG K, DENG J. Stacked hourglass networks for human pose estimation[C]//European conference on computer vision. [S.l.]: Springer, 2016: 483-499.
- [20] TANG Z, PENG X, GENG S, et al. Quantized densely connected u-nets for efficient landmark localization[C]//Proceedings of the European Conference on Computer Vision (ECCV). [S.l.: s.n.], 2018: 339-354.
- [21] NING G, ZHANG Z, HE Z. Knowledge-guided deep fractal neural networks for human pose estimation[J]. IEEE Transactions on Multimedia, 2017, 20(5): 1246-1259.
- [22] LUVIZON D C, TABIA H, PICARD D. Human pose regression by combining indirect part detection and contextual information[J/OL]. CoRR, 2017, abs/1710.02322. <http://arxiv.org/abs/1710.02322>.
- [23] CHU X, YANG W, OUYANG W, et al. Multi-context attention for human pose estimation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2017: 1831-1840.
- [24] CHOU C, CHIEN J, CHEN H. Self adversarial training for human pose estimation[J/OL]. CoRR, 2017, abs/1707.02439. <http://arxiv.org/abs/1707.02439>.

- [25] CHEN Y, SHEN C, WEI X S, et al. Adversarial posenet: A structure-aware convolutional network for human pose estimation[C]//Proceedings of the IEEE International Conference on Computer Vision. [S.l.: s.n.], 2017: 1212-1221.
- [26] YANG W, LI S, OUYANG W, et al. Learning feature pyramids for human pose estimation[C]//proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2017: 1281-1290.
- [27] KE L, CHANG M C, QI H, et al. Multi-scale structure-aware network for human pose estimation[C]//Proceedings of the European Conference on Computer Vision (ECCV). [S.l.: s.n.], 2018.
- [28] TANG W, YU P, WU Y. Deeply learned compositional models for human pose estimation[C]//Proceedings of the European Conference on Computer Vision (ECCV). [S.l.: s.n.], 2018.
- [29] ZHANG H, OUYANG H, LIU S, et al. Human pose estimation with spatial contextual information[J]. arXiv preprint arXiv:1901.01760, 2019.
- [30] SU Z, YE M, ZHANG G, et al. Cascade feature aggregation for human pose estimation[J]. arXiv preprint arXiv:1902.07837, 2019.
- [31] BULAT A, KOSSAIFI J, TZIMIROPOULOS G, et al. Toward fast and accurate human pose estimation via soft-gated skip connections[C]//2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020). [S.l.]: IEEE, 2020: 8-15.
- [32] PFISTER T, CHARLES J, ZISSERMAN A. Flowing convnets for human pose estimation in videos[C]//Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2015: 1913-1921.
- [33] CHU X, OUYANG W, LI H, et al. Structured feature learning for pose estimation[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2016: 4715-4723.
- [34] BAZAREVSKY V, KARTYNNIK Y, VAKUNOV A, et al. Blazeface: Sub-millisecond neural face detection on mobile gpus[J]. arXiv preprint arXiv:1907.05047, 2019.
- [35] Cao Z, Hidalgo Martinez G, Simon T, et al. Openpose: Realtime multi-person 2d pose estimation using part affinity fields[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019.

- [36] XU H, BAZAVAN E G, ZANFIR A, et al. Ghum & ghuml: Generative 3d human shape and articulated pose models[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2020: 6184-6193.
- [37] ZANFIR A, BAZAVAN E G, XU H, et al. Weakly supervised 3d human pose and shape reconstruction with normalizing flows[C]//Computer Vision – ECCV 2020. [S.l.: s.n.], 2020: 465-481.

哈尔滨工业大学本科毕业设计（论文）原创性声明

本人郑重声明：在哈尔滨工业大学攻读学士学位期间，所提交的毕业设计（论文）《基于 BlazePose 算法的机器人人体姿势识别与模仿》，是本人在导师指导下独立进行研究工作所取得的成果。对本文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明，其它未注明部分不包含他人已发表或撰写的成果，不存在购买、由他人代写、剽窃和伪造数据等作假行为。

本人愿为此声明承担法律责任。

作者签名：

陶冶

日期：2022年06月10日

致 谢

光阴似箭，转眼已在哈尔滨工业大学度过了四年本科生涯。在这段时间里，我通过自己的努力学习，以及老师同学的帮助，完成了这篇毕业设计，为自己的本科生涯交了一份美好的答卷。

衷心感谢导师傅忠传老师对本人的精心指导。感谢他的包容，给予我足够的自由来选择感兴趣的研究话题。也感谢他在我研究困难的时候给予我耐心，有效的指导，同时也感谢傅老师在时间安排方面，论文撰写方面给予我的宝贵意见。千言万语道不尽，只能在此对他道以最真诚的感谢。

此外，也要感谢哈工大 L^AT_EX 论文模板 HiTHESIS，为我提供了简便的模板文件，让我更加专注于论文的内容。

另外，还要感谢 BlazePose 的合作者，来自 OPPO 研究院的 Fan Zhang 研究员，他在我构建模型困难的时候给予了我宝贵的意见，使我能够按时完整地构建出与原模型媲美的模型。

同时也要感谢我的室友已经同届的同学们，感谢你们陪我度过了欢乐的四年时光，在我学业压力很大的时候，也感谢你们陪我放松玩耍，鼓励鞭策着我。

最后，特别感谢我的家人，谢谢你们一直以来无条件地支持我做自己想做的事情，为我的生活排忧解难，让我感受到亲情与温暖。

附录 1 BlazePose 各关键点的名称

表 1-1 BlazePose 各关键点的名称

关键点序号	关键点名称
0	鼻子
1	左眼内侧
2	左眼
3	左眼外侧
4	右眼内侧
5	右眼
6	右眼外侧
7	左耳
8	右耳
9	嘴巴左部
10	嘴巴右部
11	左肩
12	右肩
13	左肘
14	右肘
15	左手腕
16	右手腕
17	左小指
18	右小指
19	左手
20	右手
21	左拇指
22	右拇指
23	左髋关节
24	右髋关节
25	左膝
26	右膝
27	左脚踝
28	右脚踝
29	左脚后跟
30	右脚后跟
31	左脚
32	右脚