

2021 年易方达资产杯“人工智能+”大学生创新 技能挑战大赛暨 2022 年江苏省大学生计算机设计大 赛校内选拔赛

人工智能实践赛作品报告

作品名称： 基于人体骨骼关键点识别的摔倒检测技术

作 者： 一发密苏里

填写说明：

- 1、 本文档适用于人工智能实践赛；
- 2、 正文一律用小四号宋体，1.3 倍行距，0.5 行段后距；一级标题为二号黑体，其他级别标题如有需要，可根据需要设置，标题格式为阿拉伯数字，如第 1 章的标题写为 1.1，1.1.1 等；
- 3、 本文档应结构清晰，突出重点，适当配合图表，描述准确，不易冗长拖沓；
- 4、 提交文档时，命名为“人工智能实践赛作品报告-队伍名称”以 PDF 格式提交；
- 5、 本文档内容是正式参赛内容组成部分，务必真实填写。如不属实，将导致奖项等级降低甚至终止本作品参加比赛。

填写日期： 2021. 10. 16

目 录

第 1 章 作品概述.....	1
第 2 章 问题描述.....	2
第 3 章 技术方案.....	4
第 4 章 系统实现.....	6
第 5 章 分析验证.....	9
第 6 章 作品总结.....	11
参考文献.....	12

第 1 章 作品概述

【填写说明：重点介绍本作品的主题创意来源，产生背景，作品的用户群体、主要功能与特色、应用价值、推广前景等。如果有同类竞品，建议从多个维度对本作品与竞品进行比较，建议不超过 2 页】

1.1 设计背景

根据联合国人口署 2020 年公布的数据中国人口年龄中位数已达到 38.4，并且预计在 2050 年达到 48。未来 30 年国内最大的投资主题应该是老龄化带来健康和医疗服务机遇，而对老年人健康最大的威胁就是独居时的意外摔倒。糖尿病，心血管疾病，骨骼疾病引发的摔倒或晕厥很可能由于得不到及时的处理而恶化。

市场上上已经出现多款家用智能监控，小米等对家居安全深入的企业已经推出了支持双向语音通话，24h2k 高清红外摄像机，同时搭配简单人形检测 AI 的家用摄像头。但这些固定的摄像头普遍有着覆盖范围小，需操作者查看，不能做到实时响应危险等问题，不能提供广泛存在的独生子女家庭需要的服务。

1.2 作品功能

本作品基于以上不足与潜在需求，我们使用了 CMU 开源的人体姿态模型与识别技术 openpose^[1]，以及 MSCOCO 数据库和深度神经网络。期望完成以下工作构想：

- ① 从可移动的摄像头接受输入
- ② 以高帧率处理图像，识别图像中人物
- ③ 解析目标人体姿态以及骨骼关节点，保存数据
- ④ 利用时段内关节点信息重建模型判断目标状况
- ⑤ 目标处于摔倒状态超过阈值发出警报（紧急联系人）

整体工作流程可以参考下图



1.3 突出特点

面对上面提出的已存在产品的不足，我们期望打造产品的主要优势和攻关方向有以下几点：

速度与相应时效，基于现有的技术加以改进取舍，在不影响识别正确率的基础上降低系统流程的反应延迟，做到即时检测；

可靠性，依靠嵌入式开发，将系统集成到类似树莓派的硬件之上，搭配可移动平台做到全方位监控同时保证隐私；

舒适性，避免误报产生的打扰，将在训练和检测数据中加入模糊数据重采样，同时对一个时段内的数据集中分析。

1.4 应用前景

未来面向高龄人群的家居与健康服务市场潜力相当之大，肯定会有配套的系列产品出现。加之许多企业提出了智能家居的理念构想，我们的作品完全可以嵌入式部署在扫地机器人之类的产品上，配套监测心率健康手环，以实现更准确的摔倒判断以及定位目标位置等功能，成为家居物联网中不可缺少的一部分。而且这套技术应用范围不仅限于此，在同一套硬件设备的基础上，还可以附加婴幼儿，宠物监护，防盗防火等功能，以供用户个性化选择。

第 2 章 问题描述

【填写说明：详细描述作品拟解决的实际问题，作品的功能和性能需求；使用的数据集，包括数据格式，数据来源，数据获取方式，数据特点，数据规模等，并给出具体的数据样例。所提出的指标点必须等在第 5 章得到印证】

3.1 样本不均衡与属性缺失

模型一人体姿态分析使用的是开源的 MSCOCO 数据^[3]（url=cocodataset.org 图[2]）进行训练和测试，该数据集提供了超过两万张包含所有人体关键点信息标注的图片，可以直接通过官网打包下载图片与标签数据，能完全满足 openpose 模型对于一般环境之下的正确率的需求^[1]。但从二级输入以及应用背景的角度来看，仅使用该数据集是不够的。

其一是因为本系统的最终目标是检测摔倒，而该数据集中此类标签之下的样例太少，交给模型二的数据正反比例太过悬殊，摔倒这一行为太难准确地进行采

样；

其二性是由于环境遮挡，图片截取等多方面的因素，模型一的输出结果（关键点信息）常存在属值缺失的情况，其中有些样例进行填充之后可继续使用（如左耳），而有些关键的属性丢失会导致整个样例失效。必须对模型一的输出加以判断和处理。对此我们的策略是：

- (1) 扩大数据集，遇到类别不均衡问题时，可以再增加数据更多的数据，更多的数据往往能够得到更多的分布信息，新增加的样本尽量调和原有的比例，即多增加小类的样例。具体来说，我们是在不同环境下多人额外录制视频按照 10 帧/s 的频率采样，其中反例的比率较大。
- (2) 数据集重采样，可以使用一些策略减轻数据的不平衡程度。对小类的数据样本进行过采样（over-sampling）来增加小类的数据样本个数，对大类的数据样本进行欠采样（under-sampling）来减少该类数据样本的个数。
- (3) 数据增强，对较少的正例样本图片进行适当的剪切，平移，旋转得到更多的少量样本，从而实现正负类样本的均衡。
- (4) 数据预处理，分析不同关键点的重要权值，对于缺失较少的样例可用均值或者特征填充，缺失较多的样例直接舍弃。

3.2 快速实时响应

在设想的应用环境里，系统响应时长和处理速度都至关重要。在出现意外后能够及时做出相应，以及处理运行的速度应与视频输入的速度保持一致，是衡量系统性能的重要指标。为了减少整个系统的运行用时，以达到即时对视频输入作出反应，我们仅对图片中置信度（confidence）最高的部分所组成的人体进行判断分析。同时减少相关性较低的 face feature 和 hand feature 的模型关键点匹配。从原有的 22(face) + 12(hand) + 25(body)减少为 25 关键点位。同时降低输出的帧速率以提高模型二的吞吐量。

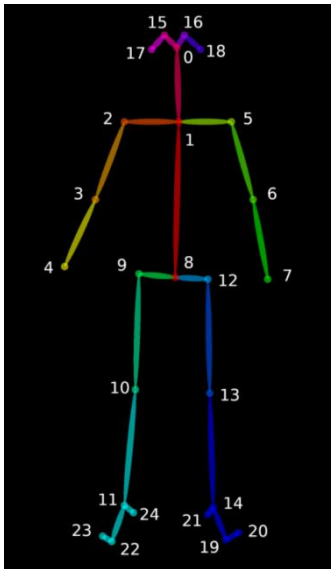
3.1 假反例率高

不仅样本中正例（正常）反例（摔倒）比例过大，正反例的先验概率之比也很大。直观来想，生活中真正摔倒的情况是比较罕见的，而且存在很多容易被误判为摔倒的动作：躺下，睡觉，坐下等。对于这些敏感易混淆的数据应重点分析处理。不然就会频繁出现虚假警报的情况，降低真正危险受到的注意程度。同时也不能降低整体反例的预测阈值——误报固然会形成骚扰，但漏报会影响产品整体可靠性的评估。最好是能从多方面测量提高模型的准确度。

第 3 章 技术方案

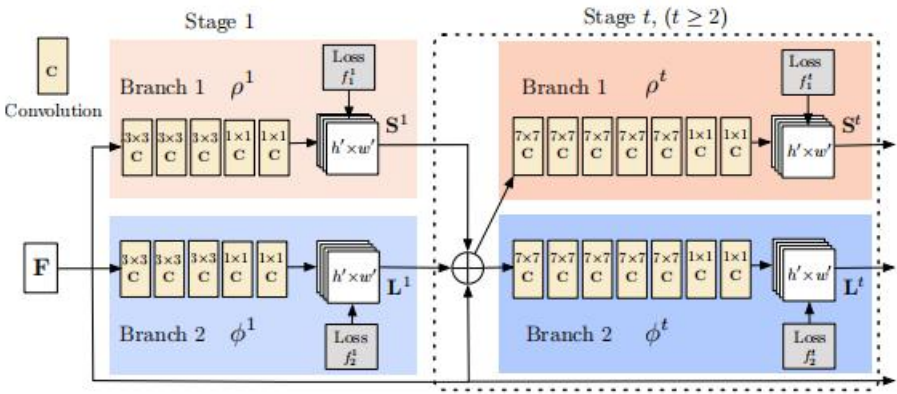
【填写说明：从原理层面，详细介绍系统所采用的技术方案，先总体介绍，给出技术路线框架图，然后分模块详细介绍。**着重介绍解决问题的思路，以及所涉及的模型、协议、算法等，以及对可能的对算法的改进；**原创工作详述，非原创工作简述，并尽可能标注引用文献】

3.1 openpose 关键点检测.



我们使用了基于 CMU 团队开发的 openpose 检测架构的 bottom-up 算法实现模型一检测人体关键点的功能 [1] 。其继承了已有的 Convolutional Pose Machines[2] 与传统 Pose Machines[4] 的优势, 使用 mscoco 数据集提供的 body 匹配模型与关键点信息。输入指定尺寸的图片, 输出检测到的关键点的位置信息。

模型整体由两部分卷积神经网络构成, 分别是匹配不同关键点的 belief map 与将不同关键点之间连接起来构成完整人体的 limb vector map , 依次经过卷积池化线性激发等层提取特征。其工作原理如下:



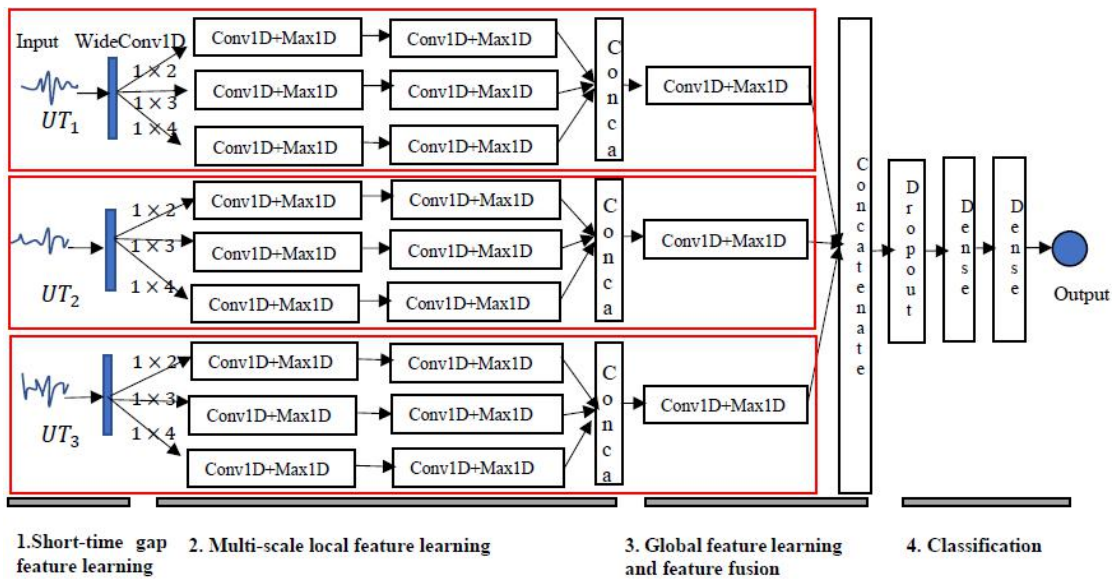
整个训练过程会被分成多个效果渐强的 stage, 在最初的 stage, 原图作为输入由两个网络做出初步的分析, 其结果会与原图的信息一起作为下一阶段的输入。网络会不断提取相关特征, 在限制时间内以及损失函数值给出最终的检测信息。



训练数据全部含有标记，使用传统的监督学习方法，损失函数为生成的 map 与 ground truth 之间的距离差。

3.2 跌倒时间序列检测

通过 3.1 的模型，我们能够在视频输入的每一帧中提取人体关键点（共 25 个）的坐标。我们每次选取从 30 帧中提取的数据作为一个时间序列作为输入，这是一个多变量时间序列分类问题 (Multivariate Time Series Classification, MTSC). 由于每个时间切片本身没有类别标签，类比文本分类问题，我们采用一维卷积神经网络的方法。然而，传统的基于 CNN 的 MTSC 算法存在着无法适应较小数据集的问题。同时，对于该问题，若人体关键点坐标和置信度以不同通道输入到模型中时，存在互相干扰中和的问题。釜山大学 Xiaorui Shao, Chang Soo Kim 提出了一种多规模特征提取的 TSC 模型[5], 该模型将时间序列的不同变量分开，分别送入本地的卷积神经网络，该网络拥有 GoogLeNet[6] 的 Inception 结构，多个拥有不同大小的卷积核的卷积层能够同时识别不同长度的特征。最后将这些本地网络输出的特征向量拼接，再送入一个卷积层进行特征提取。



Xiaorui Shao, Chang Soo Kim 的工作将时间序列的不同变量完全分开提取特征再连接，然而考虑到本问题的特殊性，我们对算法做出了一些改进。不同关键点的坐标若完全分开送入卷积神经网络提取特征可能导致丢失与关键点之间相对位置有关的信息，导致卷积核过拟合。所以，在我们的模型中，我们将关键点的横坐标，纵坐标和置信度作为三个变量分别送入三个本地卷积神经网络，在本地神经网络中，第一层卷积层我们用三种大小的核来识别不同长度的特征（对应于摔倒的时长），同时将步长设计为一帧的数据量。此外，考虑到三种变量时

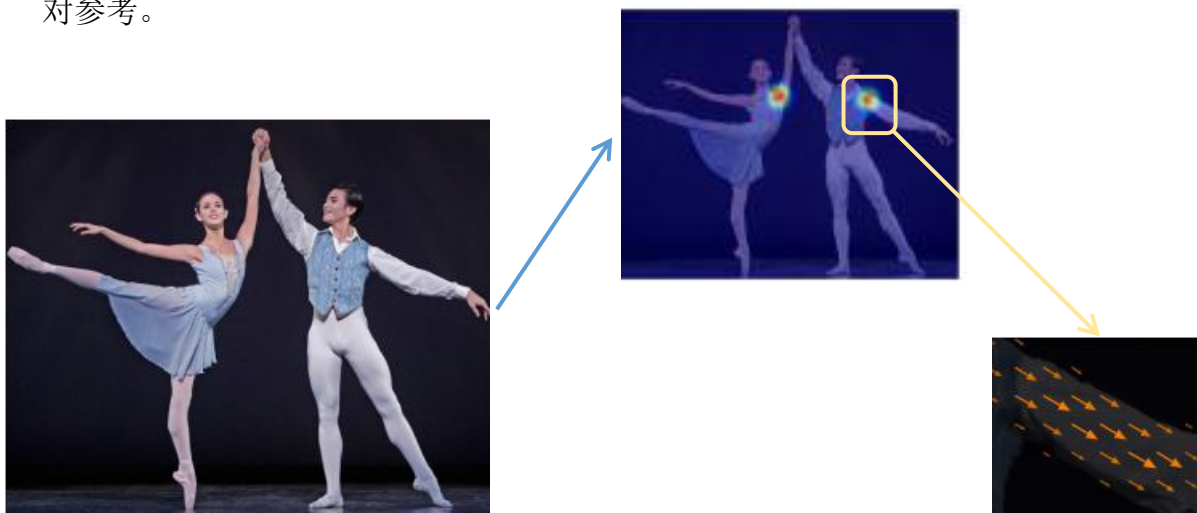
间序列的联合分布与类别较强的相关性，我们在连接三个本地神经网络输出特征图之后，再采用一层卷积层进行特征提取。

第 4 章 系统实现

【填写说明：从工程实现的角度，详细阐述第 3 章提出的技术方案的具体实现过程，包括且不限于软件设计实现，用户界面，数据来源，数据训练，改进过程，以及系统部署方法等，以及其中所遇到的困难，解决的方法等】

4.1 数据采集与标注

对于模型 1，可以使用已有的带有标注的 MSCOCO 图片数据集进行训练，鉴于数据集过于庞大，我们使用随机采样选取部分数据集，并根据图片上像素点位的标记分别生成最大值高斯分布的 map 作为两部分神经网络的 groundtruth 损失相对参考。



而对于模型二，需要网络上较少的视频数据集，而且网络上几乎没有摔倒相关的数据集，因此除了搜集补充数据集之外我们还需要手动进行标记，我们除了使用，通过网络上搜索收集、自行拍摄等方法补充了部分样例（主要是较少的正例）并把我们认为比较典型的样例做了变换之后添加入数据集作为加权重处理。例如，我们使用了 University of Rzeszow 公开的动作数据集^[5]，但该数据集只含有 70 段 10s 左右的视频参考，我们在此基础上对这些视频处理后产生的中间数据

4.2 关键点检测模型参数调整与训练

关键点识别模型默认会使用最大运行设备 GPU 默认最大输出帧速率，为了能够以稳定的速率产生输出，同时保证数据动作的连贯性以及后续模型的处理响应速度，我们固定了 frame 的频数，同时在进行 bottom-up 的时仅保留构建出的完整个体中置信度最大的一个（默认使用环境中只会出现一人），最大化提高处理速度。

4.3 数据的加工与预处理

模型 1 的输出结果是 pattern 中每一关键点相对于图像起点的二维偏移坐标，以浮点数格式表示，未识别到的部分会以空值代替，为了避免人物所处的位置这一无关特征以及缺失值的影响，将对数据做以下处理：

缺失值填充：有时可能会由于遮挡，画面虚化，人物离开摄像范围导致部分关键点信息缺失，对此我们将试图读取该 frame 前后其他时刻的采样值。由于视频中人物的动作应该是连续的，我们可以使用一段时间内的该关键点的均值来填补缺失值，当然对于重要关键点的缺失（如躯干、肩、腿、髋），我们会判别为该帧信息无效。

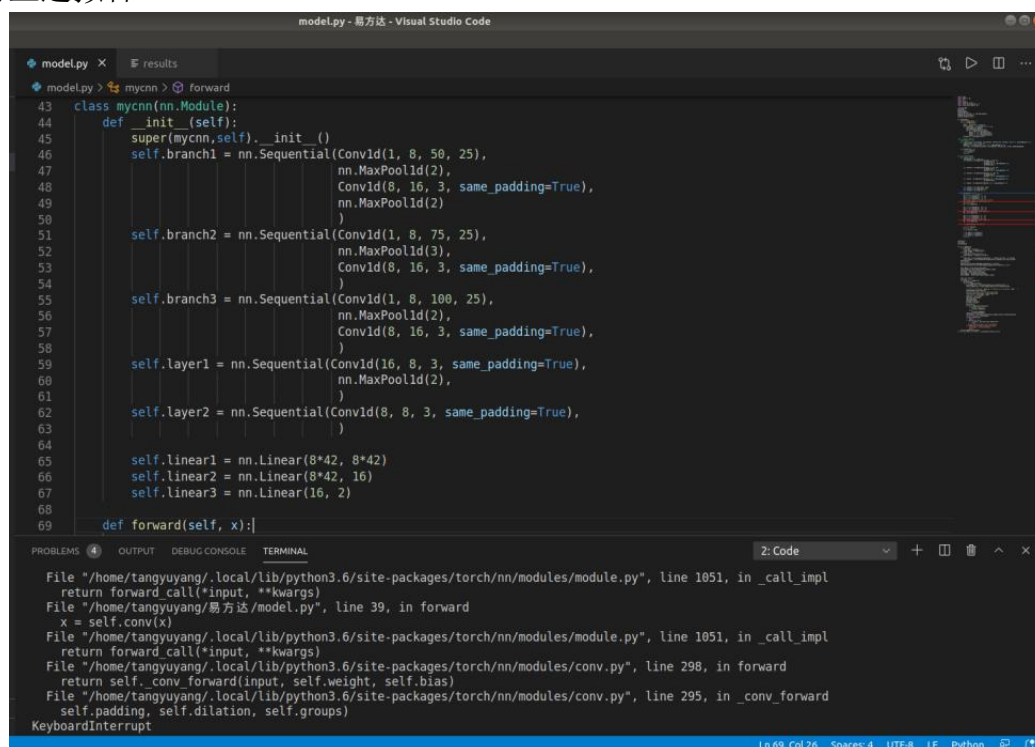
样本中心化：最大程度提炼数据中的姿态信息，将对 json 文件中关键点信息进行修正，基于躯干关键点 (point=1)，重新修改各关键点位置为相对于躯干位置偏移。

将输出的连续帧 json 数据按 0.5s 一个样例切割聚合，并标注相应类别标签，作为下一步的训练和测试数据。

4.4 模型的训练和评估

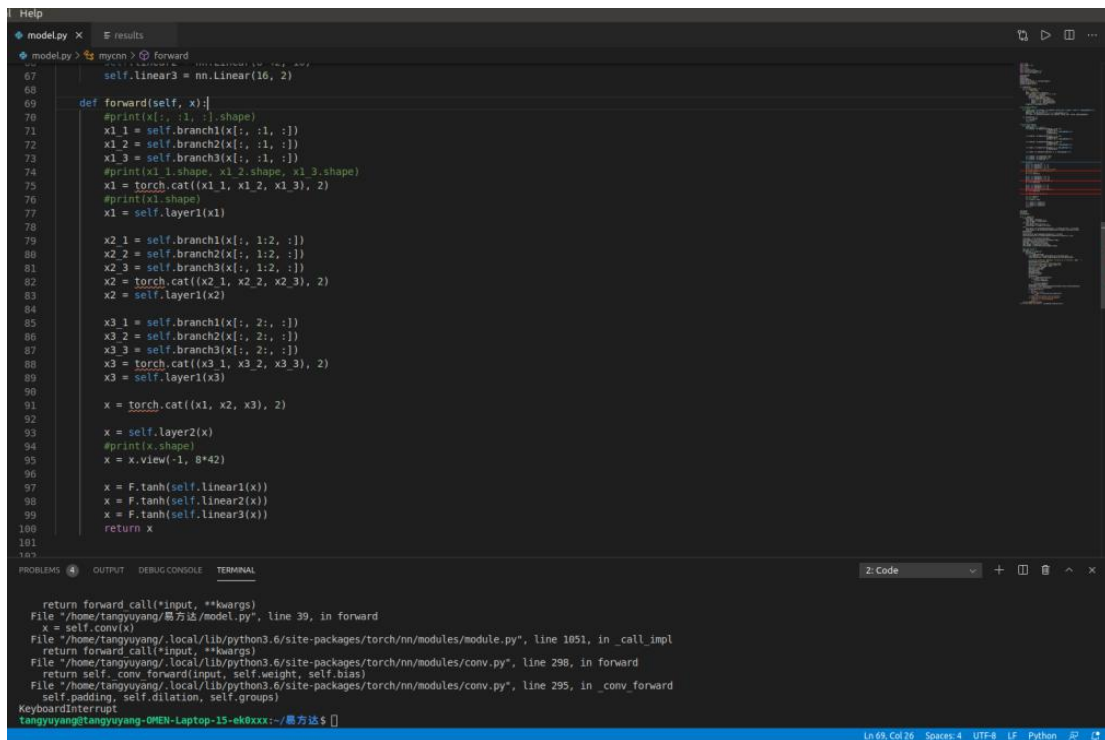
我们利用 Pytorch 库对 3.2 所述模型进行了搭建。在本地神经网络中我们采用 50, 75 和 100 作为第一层卷积核大小得到三个分支。在一层池化层和第二层卷积层之后将三个特征向量连接，再通过一个卷积层 layer1 得到此变量的时间序列特征向量。

在所有变量的特征向量提取完毕后，我们将它们连接并用一个新的卷积层提取与它们联合分布相关的信息，最后通过全连接层收敛并输出一个一维的类别向量。对于该二分类问题，我们采用交叉熵作为损失函数，SGD 优化器并用早停法防止过拟合。



```
43 class mycnn(nn.Module):
44     def __init__(self):
45         super(mycnn, self).__init__()
46         self.branch1 = nn.Sequential(Conv1d(1, 8, 50, 25),
47                                     nn.MaxPool1d(2),
48                                     Conv1d(8, 16, 3, same_padding=True),
49                                     nn.MaxPool1d(2))
50
51         self.branch2 = nn.Sequential(Conv1d(1, 8, 75, 25),
52                                     nn.MaxPool1d(3),
53                                     Conv1d(8, 16, 3, same_padding=True),
54                                     nn.MaxPool1d(2))
55
56         self.branch3 = nn.Sequential(Conv1d(1, 8, 100, 25),
57                                     nn.MaxPool1d(2),
58                                     Conv1d(8, 16, 3, same_padding=True),
59                                     nn.MaxPool1d(2))
60
61         self.layer1 = nn.Sequential(Conv1d(16, 8, 3, same_padding=True),
62                                     nn.MaxPool1d(2))
63
64         self.layer2 = nn.Sequential(Conv1d(8, 8, 3, same_padding=True),
65                                     nn.MaxPool1d(2))
66
67         self.linear1 = nn.Linear(8*42, 8*42)
68         self.linear2 = nn.Linear(8*42, 16)
69         self.linear3 = nn.Linear(16, 2)
70
71     def forward(self, x):
72         x = self.conv(x)
73         x1, x2, x3 = self.branch1(x), self.branch2(x), self.branch3(x)
74         x = torch.cat((x1, x2, x3), 1)
75         x = self.layer1(x)
76         x = self.layer2(x)
77         x = self.linear1(x)
78         x = self.linear2(x)
79         x = self.linear3(x)
80         return x
```

File "/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/modules/module.py", line 1051, in _call_impl
return forward_call(*input, **kwargs)
File "/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/modules/module.py", line 39, in forward
x = self.conv(x)
File "/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/modules/module.py", line 1051, in _call_impl
return forward_call(*input, **kwargs)
File "/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/modules/conv.py", line 298, in forward
return self._conv_forward(input, self.weight, self.bias)
File "/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/modules/conv.py", line 295, in _conv_forward
self.padding, self.dilation, self.groups)
KeyboardInterrupt



```
67 self.linear3 = nn.Linear(16, 2)
68
69 def forward(self, x):
70     #print(x[:, :1, :].shape)
71     x1 = self.branch1(x[:, :1, :])
72     x2 = self.branch2(x[:, :1, :])
73     x3 = self.branch3(x[:, :1, :])
74     #print(x1.shape, x2.shape, x3.shape)
75     x1 = torch.cat((x1, x2, x3), 2)
76     #print(x1.shape)
77     x1 = self.layer1(x1)
78
79     x2 = self.branch1(x[:, 1:2, :])
80     x2 = self.branch2(x[:, 1:2, :])
81     x2 = self.branch3(x[:, 1:2, :])
82     x2 = torch.cat((x2, x2, x2), 2)
83     x2 = self.layer1(x2)
84
85     x3 = self.branch1(x[:, 2:, :])
86     x3 = self.branch2(x[:, 2:, :])
87     x3 = self.branch3(x[:, 2:, :])
88     x3 = torch.cat((x3, x3, x3), 2)
89     x3 = self.layer1(x3)
90
91     x = torch.cat((x1, x2, x3), 2)
92
93     x = self.layer2(x)
94     #print(x.shape)
95     x = x.view(-1, 8*42)
96
97     x = F.tanh(self.linear1(x))
98     x = F.tanh(self.linear2(x))
99     x = F.tanh(self.linear3(x))
100     return x
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2
```

+800 万像素的索尼 IMX219 的套件，期望将整个系统移植部署到这个移动系统上。但在实现过程中发现 Nvidia 未发布相应的硬件驱动，使用 cudatoolkit 的模型 1 无法在 raspberry 上运行，于是改变计划仅将摄像头远程连接到笔记本电脑上，通过 ssh 链接仍在电脑上使用模型进行判别和输出。但这样没有利用到树莓派本身的编程和计算能力，我们在上面配置了简单的目标检测功能，以脚本的形式控制摄像头的开关。下一步可能会应用相对简单目标识别以及路径规划的功能来辅助系统提高运行效果。

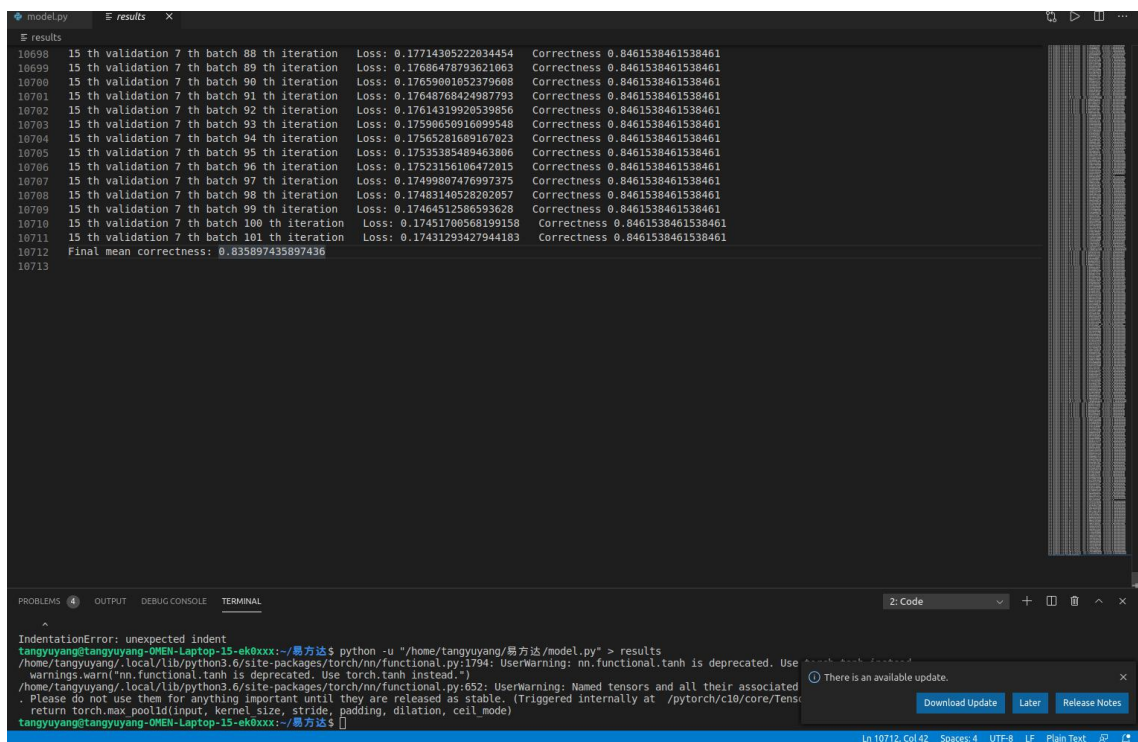
第 5 章 分析验证

【填写说明：通过测试与对比，论证系统的有效性，可包括验证数据的来源与规模、测试过程、分析与结论等等。各参赛队务必重视数据测试，所有对自己作品准确性、有效性、稳定性，甚至作品受欢迎的程度的宣称，都应该得到数据结果或对比实验的支持，否则评审人有理由怀疑其真实性】

5.1 模型准确率

针对所研究的时间序列，用将三个通道的信息同时放进同一个卷积神经网络，效果不佳，准确率仅能达到约 69%。采用三个信号分别输入三个网络，且用多种卷积核同时识别特征的方法后，准确率上升到约 77%。

在这基础上，我们在整个网络最后加入一层卷积层，并采用 mini-batch 梯度下降的方法，调整 batch 大小，同时，运用拉伸，镜像等方法扩充训练集，最终将精度提高到约 87%。（以上精度均由 k 折交叉验证产生）



```
model.py E results x
E results
10698 15 th validation 7 th batch 88 th iteration Loss: 0.17714305222034454 Correctness 0.8461538461538461
10699 15 th validation 7 th batch 89 th iteration Loss: 0.17686478793621063 Correctness 0.8461538461538461
10700 15 th validation 7 th batch 90 th iteration Loss: 0.17659801052379688 Correctness 0.8461538461538461
10701 15 th validation 7 th batch 91 th iteration Loss: 0.17648768424987793 Correctness 0.8461538461538461
10702 15 th validation 7 th batch 92 th iteration Loss: 0.17614318920530856 Correctness 0.8461538461538461
10703 15 th validation 7 th batch 93 th iteration Loss: 0.17590650916099548 Correctness 0.8461538461538461
10704 15 th validation 7 th batch 94 th iteration Loss: 0.17565281689167023 Correctness 0.8461538461538461
10705 15 th validation 7 th batch 95 th iteration Loss: 0.17535385489463806 Correctness 0.8461538461538461
10706 15 th validation 7 th batch 96 th iteration Loss: 0.17523156106472015 Correctness 0.8461538461538461
10707 15 th validation 7 th batch 97 th iteration Loss: 0.17499807476997375 Correctness 0.8461538461538461
10708 15 th validation 7 th batch 98 th iteration Loss: 0.17483140528202057 Correctness 0.8461538461538461
10709 15 th validation 7 th batch 99 th iteration Loss: 0.17464512560593826 Correctness 0.8461538461538461
10710 15 th validation 7 th batch 100 th iteration Loss: 0.17451700268109158 Correctness 0.8461538461538461
10711 15 th validation 7 th batch 101 th iteration Loss: 0.17431293427944183 Correctness 0.8461538461538461
10712 Final mean correctness: 0.835897435897436
10713
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

IndentationError: unexpected indent
tangyuyang@tangyuyang-OMEN-Laptop-15-ek0xxx:~/易方达\$ python -u "/home/tangyuyang/易方达/model.py" > results
/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/functional.py:1794: UserWarning: nn.functional.tanh is deprecated. Use torch.tanh instead.
/home/tangyuyang/.local/lib/python3.6/site-packages/torch/nn/functional.py:652: UserWarning: Named tensors and all their associated
Please do not use them for anything important until they are released as stable. (Triggered internally at /pytorch/c10/core/Ten
return torch.max_pool1d(input, kernel_size, stride, padding, dilation, ceil_mode)
tangyuyang@tangyuyang-OMEN-Laptop-15-ek0xxx:~/易方达\$

There is an available update.

Download Update Later Release Notes

Ln 10712, Col 42 Spaces: 4 UTF-8 LF Plain Text

由于在进行处理会首先将输入流拉伸处理到默认尺寸比例 (w:400h:300) 所以视频的尺寸影响不大。默认最大 fps 会在 10 左右之间波动, 同时每秒产生 100 个 frame 的 json 文件, 如果要兼顾后续的处理和判断, 必须对 frames/s 进行调整。

另外, 我们通过树莓派摄像头使用 Vlc 向计算机推流, 在现阶段使用的局域网存在大概 1~2s 波动的画面延迟。

综合考虑以上, 我们固定接受的 fps 为 10, 处理 frame 的步长为 3, 拍摄 1280x720 像素的画面, 系统整体响应延迟能够控制在 5s 以内

第 6 章 作品总结

【填写说明: 从创意、技术路线、工作量、数据和测试效果等方面对作品进行自我评价和总结, 并对作品的进一步提升和应用拓展提出展望】

在本作品中, 我们通过不断地尝试和调整, 基本完成了预期的目标需求, 还在开发过程中加入了许多新的创新点。我们围绕预定的应用环境在现有的成熟模型与技术的基础上进行改进和调整。我们首先完成了快速识别人体主要关键点的模型 I。接下来对于问题一开始存在的缺少数据以及类别不平衡, 我们通过各种渠道重新收集了大量数据, 并对原有数据中不同类别的样例进行了一些处理。同时对视频通过模型 I 的结果进行中心化、填补、切割打包等操作, 作为模型 II 的训练和测试数据。

最后将全部的模型部署为一个系统, 进行初步的产品开发尝试。以树莓派避障车上的摄像头作为视频源输入, 在电脑上进行处理以及结果的预测。

从最终测试结果来看, 我们的系统还存在反应速率比较慢, 仍有一定的错误率, 处理过程过于复杂等问题。我们下一步改进的工作重心是继续完善相应数据集的多样性, 以提高模型泛化能力, 同时进行一些更专业的市场需求调研, 在系统上部署被需要的功能, 完善系统提高整体用户体验。

我们设计的初衷是让技术更好地服务于社会, 解决实际的民生问题。希望我们的作品能够在未来真正进入家庭, 为人们排忧解难!

参考文献

【请按照标准参考文件格式填写】

- [1] Zhe Cao, Tomas Simon. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. The Robotics Institute of CMU, In CVPR, 2016.
- [2] Shih-En Wei, Varun Ramakrishna, Yaser Sheikh. Convolutional Pose Machines. The Robotics Institute of CMU, In CVPR, 2016.
- [3] MSCOCO keypoint evaluation metric. <http://mscoco.org/dataset/#keypoints-eval>.
- [4] V. Ramakrishna, D. Munoz, M. Hebert, J. Bagnell, and Y. Sheikh. Pose Machines: Articulated Pose Estimation via Inference Machines. In ECCV, 2014.
- [5] <http://fenix.univ.rzeszow.pl/~mkepski/ds/uf.html>