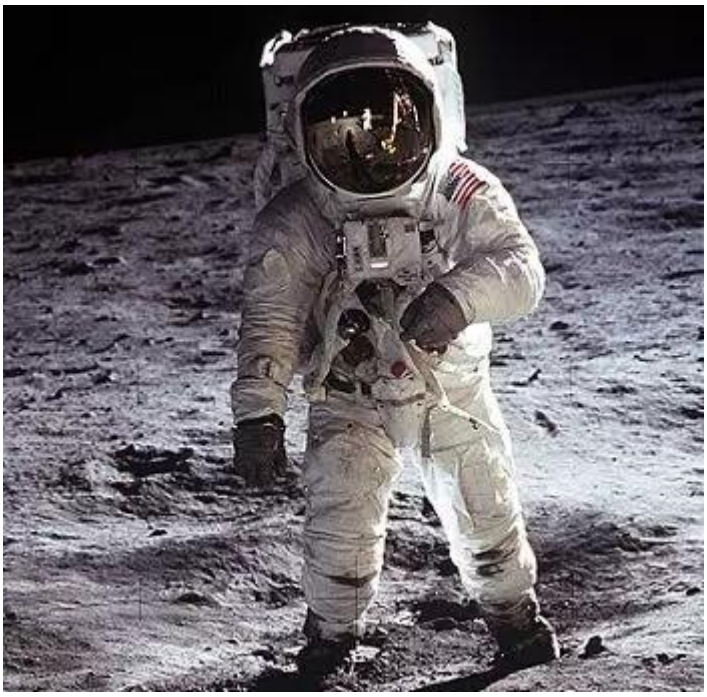
英特尔/人工智能｜宇宙飞船如何使用人工智能？（三）

原创：Justin Shenk英特尔开发人员专区11 月 9 日



第三部分：姿势检测

在之前的文章中，我们实施了 HoG 方法来检测视频帧中的人。HoG 适用于人体直立的情况，但无法检测人群中或处于异常位置的人。在本教程中，您将学习如何使用英特尔® 人工智能 DevCloud 实施深度神经网络，该网络旨在估计图像中多个人的姿势。

HoG 和 SVM 是一个非常强大组合，但我们可以借助预训练的神经网络改进我们的检测。我们将使用卡内基梅隆大学的研究人员开发，并由 Ildoo Kim 面向 TensorFlow* 和 CPU 实施的开源姿势估计器。由于 Kim 选择的是 MobileNet — 一种面向边缘和移动计算设计的神经网

络，因此该软件可以在 CPU 设备上进行实时姿势估计。有关初步实施的更新信息，请访问 GitHub。

用于后期估计的人工智能 DevClou

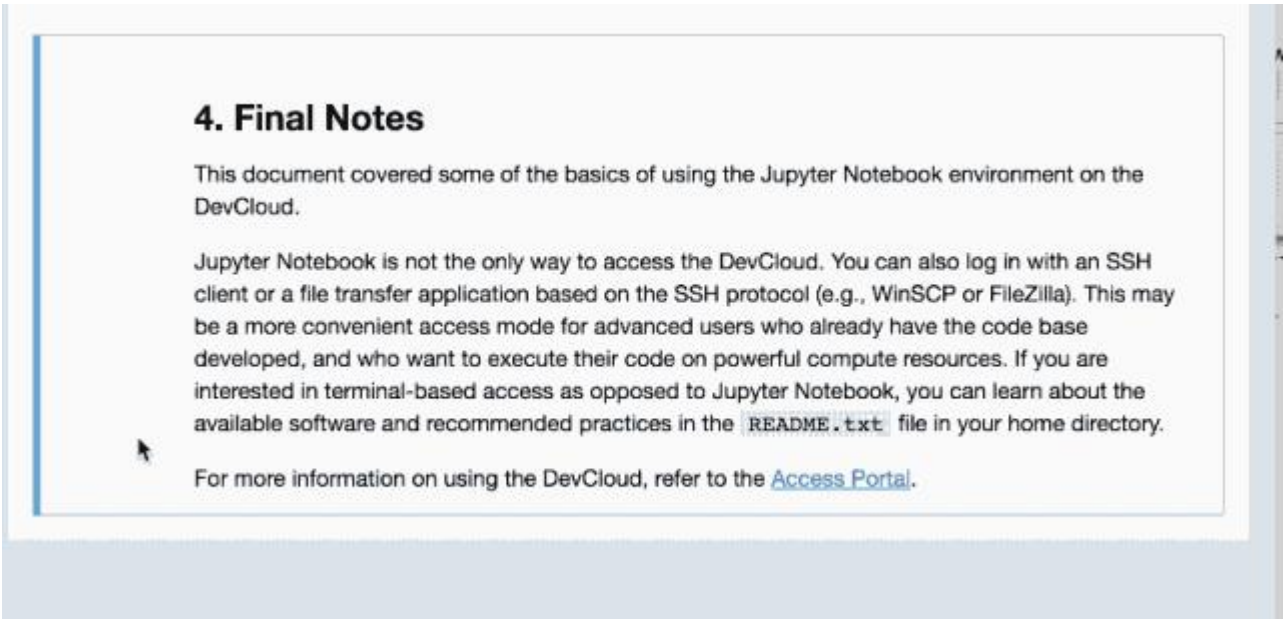
在本教程中，我们将使用英特尔人工智能 DevCloud，不过大家可以随意使用 AWS、Google Cloud 等云服务来应对挑战。英特尔人工智能 DevCloud 支持我们以比仅使用电脑更快的速度生成数据。

- 英特尔® 人工智能 DevCloud 入门
-
- 英特尔人工智能 DevCloud 新用户和将任务提交到计算机集群的用户，请在注册后查看 "Welcome" 笔记本：浏览到网址，将 <userID> 替换为您的用户名：https://hub.colfaxresearch.com/user/<userID>/notebooks/Welcome.ipynb。

这里需要掌握的主要技巧是：

- 用 `qsub` 提交任务
-
- 用 `qstat` 检查任务状态
-
- 用 iPython 魔术命令输出结果
-

- 打开 DevCloud 上的笔记本 ,具体方法是在笔记本底部打开一个单元格并用魔术命令 `!git clone --depth=1 --branch=demo 克隆存储库 https://github.com/JustinShenk/tf-pose-estimation：



然后浏览到笔记本 https://hub.colfaxresearch.com/user/<userID>/notebooks/tf-pose-estimation/pose-estimation.ipynb

第一个单元格使用魔术命令 "%%writefile" 将单元格内容保存为 "pose_job.txt"。重点提示: 请记住向单元格添加尾随换行符，否则任务将无法运行。

```
%%writefile pose_job.txt
#PBS -N DetectHumanoids
#PBS -l nodes=1
cd $PBS_O_WORKDIR
echo "Run pose estimation in conda environment"
conda create -n py3 python=3 -y
source activate py3
conda update setuptools -y
# Clone fork of @ildoonet's CPU-optimized TensorFlow implementation of
@ZheC's
# Realtime Multi-Person Pose Estimation
git clone --depth=1 -b demo https://github.com/JustinShenk/tf-pose-
estimation.git
cd $PBS_O_WORKDIR/tf pose estimation
pip install -r requirements.txt
pip install opencv-python
MPLBACKEND=Agg python src/run.py # prevent matplotlib from displaying
images
source deactivate
echo "End inference"
# leave an empty line at the bottom or else it won't run
```

我们可以在行前加一个感叹号，调用 Jupyter* 笔记本中的系统命令。例如，`!ls` 调用 `ls` 系统命令。而且我们可以将输出返回给变量。这样我们可以检查用 `qsub` 提交的任务状态。

由于我们想在任务完成时接到通知（每个图像只有几分之一秒），我们可以使用辅助函数 `wait_for`：

```
def wait_for(job_id):
    waiting = True
    start = time.time()
    while waiting:
        jobs = !qstat
        if any(job_id in line for line in jobs):
            print('.', end='')
            time.sleep(2) # check every 2 seconds
        else:
            end = time.time()
            duration = (end - start) / 60
            print("\nJob {} completed after {:.2f}
minutes".format(job_id, duration))
            break
```

调用 `wait_for(job_id)` 每隔 2 秒打印一段，直到任务完成：

```
job_files = '*' + job_id + '*'
output = !cat $job_files
```

```
job_id = job[0].split('.')[0]
wait_for(job_id)
```

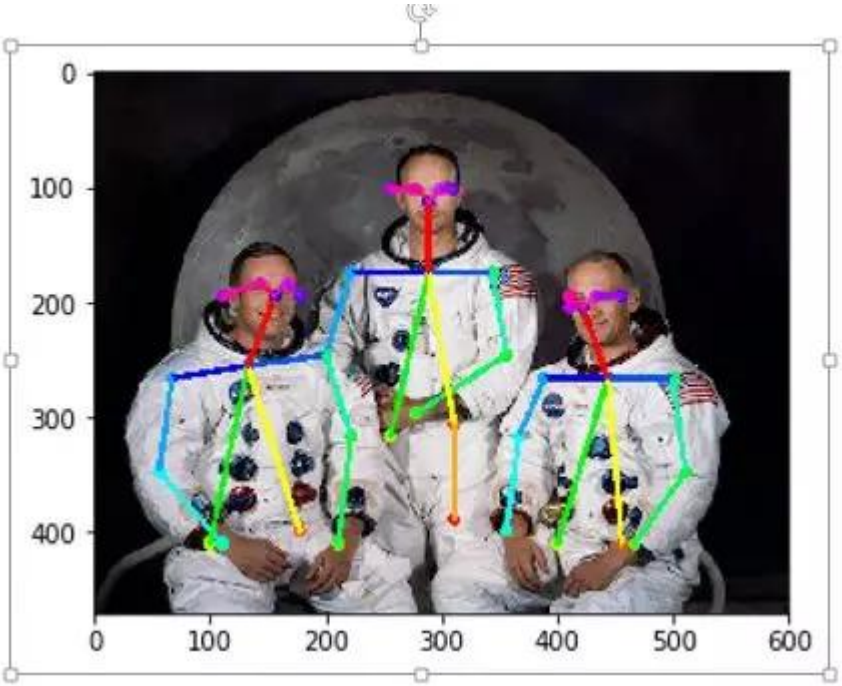
```
.............................
Job 81793 completed after 1.02 minutes
```

然后我们可以检查任务的输出（stderr，然后是 stdout）并看到图像保存在 `output/test.png`：

```
# see the job output (stderr followed by stdout)
output # optional
```

```
Out[10]: ['From https://github.com/JustinShenk/tf-pose-estimation',
 '   8f2dd7c..c3ba192  demo       -> origin/demo',
 "Already on 'demo'",
 '2018-05-13 10:41:57.812668: I tensorflow/core/platform/cpu_feature_guard.cc:140] Your CPU suppor
ts instructions that this TensorFlow binary was not compiled to use: AVX2 AVX512F FMA',
 '[2018-05-13 10:41:58,135] [TfPoseEstimator] [INFO] inference image: ./images/p3.jpg in 0.0883 se
conds.',
 '[2018-05-13 10:41:58,167] [TfPoseEstimator] [INFO] image saved: output/test.png',
 '',
 '######################################################################',
 '#      Date:           Sun May 13 10:41:26 PDT 2018',
 '#    Job ID:           81793.c009',
 '#      User:           u6998',
 '# Resources:           nednodes=1,nodes=1,walltime=06:00:00',
 '######################################################################',
 '',
 'Run pose estimation in conda environment',
 'Updating 8f2dd7c..c3ba192',
 'Fast-forward',
 ' src/run.py | 2 +-',
 ' 1 file changed, 1 insertion(+), 1 deletion(-)',
 'M\ttud-crossing-sequence/pose.dil',
 'End inference',
 '',
 '######################################################################',
 '# End of output for job 81793.c009',
 '# Date: Sun May 13 10:41:58 PDT 2018',
 '######################################################################',
 '']
```

我们可以使用 `plot_dir` 绘制 `output` 的内容，并查看图像：



如果要估计目录中多张图像的姿势，请使用 python src/run_directory.py --folder=<folder> 替换 pose_job.txt 中的 python src/run.py 命令。

注：使用全新英特尔 OpenVINO™ 工具套件，可以在各种平台上针对多种 CPU、FPGA、IPU 和 VPU（比如英特尔® Movidius™ 神经计算棒）对深度神经网络等算法进行优化。

挑战

- 通过手部位置实时提示灯光或音乐（基础）
- 通过手势实时提示灯光或音乐（中级）
- 通过分析身体姿势随时间的变化推荐运动（高级）

后续挑战

- 通过计算光流控制房间内的音乐
- 优化实时姿势检测模型，以使其在英特尔® Movidius™ 神经计算棒 上运行或与 Up Squared* 开发板搭配使用
- 跟踪网络摄像头捕捉的每一帧中的人数，并用它来激活 LED 灯
- 阅读本系列的前两篇文章：
- 计算机视觉简介和面部检测（第一部分）
- 人体检测（第二部分）

关于作者

Justin Shenk

德国奥斯纳布吕克大学硕士研究生和研究助理

Justin 是 Peltarion 的一名 AI Master Thesis 工作者，负责研究深度学习模型自省。他以英特尔软件创新者的身份开发人工智能软件，并在 NIPS、ICML 和 CVPR 召开之际在英特尔展台上演示自己的项目。之前在美国他曾是一名神经科学家。