# TensorFlow.js浏览器中的实时人体姿态估计

发布者：，谷歌创意实验室的自由创意技术人员，纽约大学ITP的研究生。编辑和插图：，谷歌创意实验室的创意技术人员和自由平面设计师

与Google Creative Lab合作，我很高兴宣布发布了一个版本的机器学习模型，它允许在浏览器中进行实时人体姿势估计。试试现场演示。

那么什么是姿势估计呢？姿势估计是指计算机视觉技术，它可以检测图像和视频中的人物，以便确定，例如，某人的肘部在图像中出现的位置。明确地说，这项技术没有识别出图像中的人，没有与姿势检测相关的个人识别信息。算法只是简单地估计关键身体关节的位置。

好吧，为什么一开始就让人兴奋？姿势估计有很多用途，从那个到到，等等。我们希望这个模型的可访问性能够激励更多的开发者和制作者在他们自己独特的项目中进行姿势检测的实验和应用。虽然许多替代姿态检测系统已经存在，但都需要专门的硬件和/或摄像机，以及相当多的系统设置。随着PoseNet运行在一个有一个像样的网络摄像头配备桌面或电话可以体验这项技术从一个网络浏览器。由于我们已经开放了该模型的源代码，Javascript开发人员只需几行代码就可以修补并使用这项技术。此外，这实际上有助于保护用户隐私。由于TensorFlow.js上的PoseNet在浏览器中运行，因此任何姿势数据都不会离开用户的计算机。

在我们深入研究如何使用这个模型的细节之前，先向所有使这个项目成为可能的人大喊大叫：还有，论文背后的谷歌研究人员，以及图书馆背后的谷歌大脑团队的工程师。

### PoseNet入门

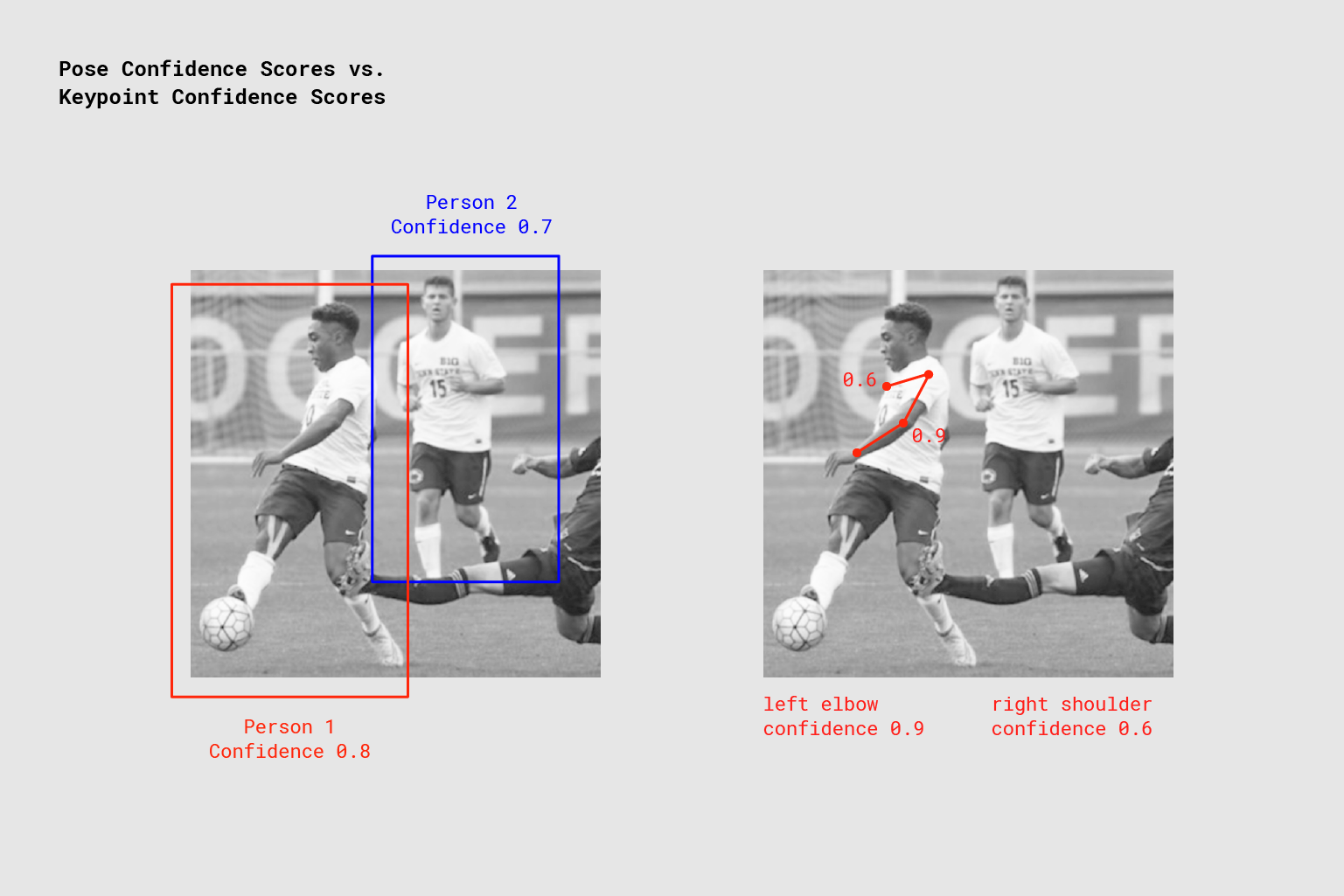
可用于估计单个姿势或多个姿势，这意味着该算法的一个版本只能检测图像/视频中的一个人，而另一个版本可以检测图像/视频中的多个人。为什么有两个版本？单人姿势检测器更快更简单，但只需要图像中的一个对象（稍后会详细介绍）。我们先介绍一下单体式，因为它更容易跟随。

在高层次上，姿态估计分为两个阶段：

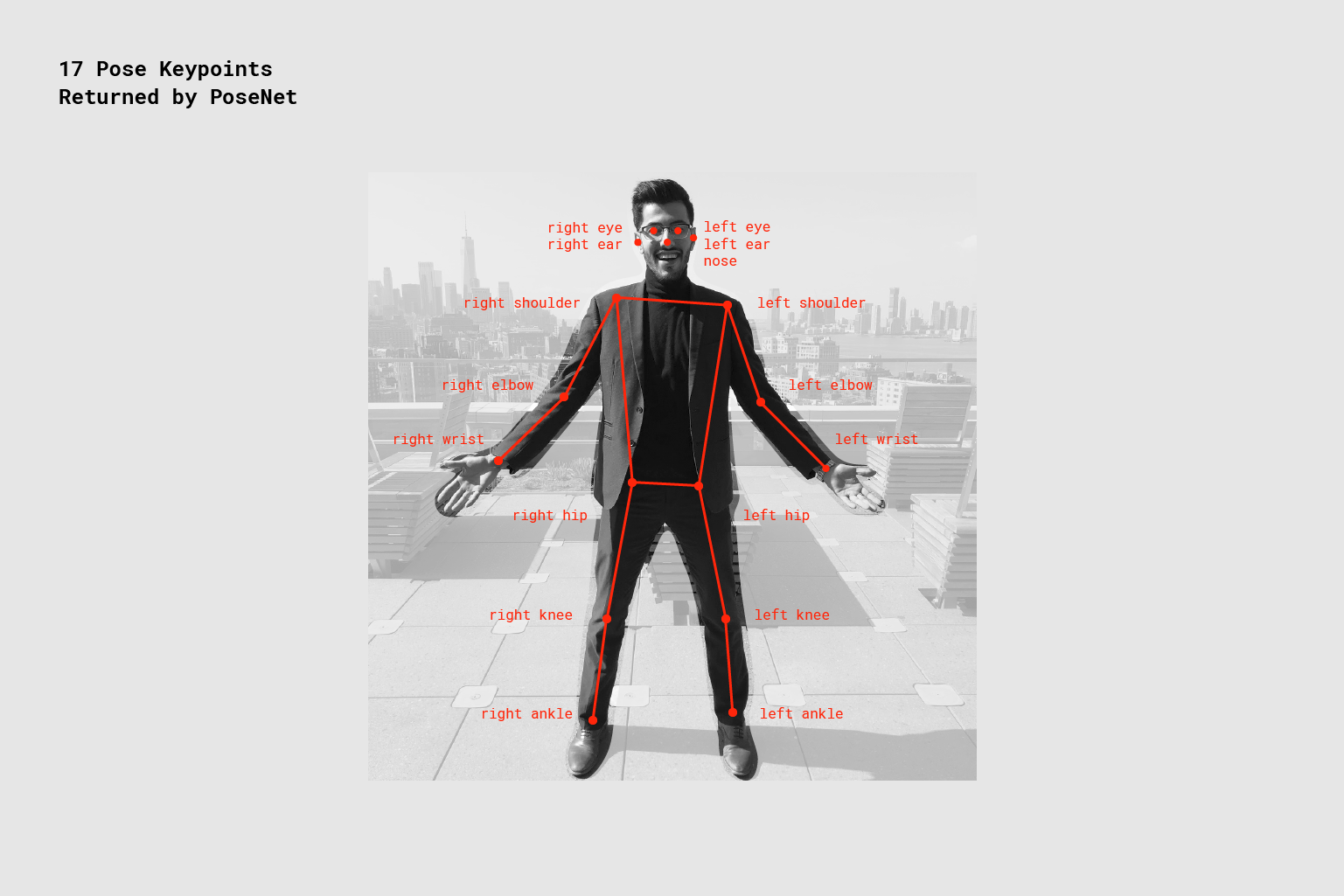
1. 通过卷积神经网络输入RGB图像。
2. 使用单姿态或多姿态解码算法对模型输出的姿态、姿态置信度得分、关键点位置、\*\*\*\*和关键点置信度得分进行解码。

但是等等，这些关键词都是什么意思？让我们回顾一下最重要的：

* Pose - 在最高级别，PoseNet将返回一个Pose对象，该对象包含每个检测到的人的关键点列表和实例级置信度分数。



* 姿势置信度 - 这决定了姿势估计的总体置信度。范围在0.0到1.0之间。它可以用来隐藏被认为不够强壮的姿势。
* Keypoint - 估计的人体姿势的一部分，如鼻子、右耳、左膝、右脚等。它包含一个位置和一个Keypoint置信度得分。PoseNet目前检测到17个关键点，如下图所示：



* 关键点信心得分- 这决定了估计的关键点位置是否准确的信心。范围在0.0到1.0之间。它可以用来隐藏被认为不够强大的关键点。
* 在检测到关键点的原始输入图像中的关键点位置 - 2D x和y坐标。

#### 第1部分：导入TensorFlow.js和PoseNet库

大量的工作都是为了抽象模型的复杂性，并将功能封装成易于使用的方法。让我们回顾一下如何设置PoseNet项目的基础知识。

可以使用npm安装库：

并使用es6模块导入：

或者通过页面中的捆绑：

#### 第2a部分：单人姿势估计



如前所述，单姿态估计算法是两种算法中比较简单和快速的。它的理想用例是当一个输入图像或视频中只有一个人的时候。缺点是，如果图像中有多个人，两个人的关键点很可能被估计为同一个单一姿势的一部分，这意味着，例如，该人的左臂和右膝可能被该算法合并为属于同一姿势。如果输入图像可能包含多人，则应使用多姿态估计算法。

让我们回顾一下单姿态估计算法的输入：

* Input image element— 包含要预测姿势的图像的html元素，例如视频或图像标记。重要的是，输入的图像或视频元素应该是方形的。
* 图像比例因子- 介于0.2和1之间的数字。默认为0.50。在通过网络传送图像之前，通过什么来缩放图像。将此数字设置得更低，以缩小图像并在通过网络馈送时提高速度，但会降低精度。
* 水平翻转- 默认为false。如果姿势应该水平翻转/镜像。对于默认水平翻转视频（即网络摄像头）的视频，这应该设置为true，并且您希望以正确的方向返回姿势。
* 输出跨距 - 必须为32、16或8。默认为16。在内部，该参数影响神经网络中层的高度和宽度。在较高的水平上，它会影响姿态估计的精度和速度。输出跨距值越小，精度越高，但速度越慢；输出跨距值越高，速度越快，但精度越低。查看输出跨距对输出质量影响的最佳方法是使用

现在让我们回顾一下单姿态估计算法的输出：

* 一种姿势，包含一个姿势置信度得分和一个由17个关键点组成的数组。
* 每个关键点包含一个关键点位置和一个关键点信心得分。同样，所有的关键点位置在输入图像空间中都有x和y坐标，并且可以直接映射到图像上。

此短代码块显示了如何使用单姿态估计算法：

示例输出姿势如下所示：

#### 第2b部分：多人姿势估计



多人姿态估计算法可以估计图像中的多个姿态/人。它比单一姿势算法更复杂，速度稍慢，但它的优点是，如果一张图片中出现多个人，他们检测到的关键点就不太可能与错误的姿势相关联。因此，即使用例是为了检测一个人的姿势，这个算法可能更可取。

此外，该算法的一个吸引人的特性是性能不受输入图像中的人数的影响。无论是15人检测还是5人检测，计算时间都是一样的。

让我们回顾一下输入：

* 输入图像元素- 与单姿态估计相同
* 图像比例因子 - 与单姿态估计相同
* 水平翻转 - 与单姿态估计相同
* 输出步幅 - 与单姿态估计相同
* 最大姿态检测-整数。默认为5。要检测的最大姿态数。
* 姿势信心得分阈值 - 0.0至1.0。默认为0.5。在高水平上，这将控制返回姿势的最小置信分数。
* 非最大抑制（NMS）半径α-α的像素数。在高级别上，这控制返回的姿势之间的最小距离。这个值默认为20，这在大多数情况下可能是好的。它应该增加/减少，作为一种过滤不太准确的姿势的方法，但只有在调整姿势信心分数不够好的情况下。

查看这些参数有何影响的最佳方法是使用。

让我们回顾一下输出：

* 用一系列姿势来解决的承诺。
* 每个姿势包含的信息与单人估计算法中描述的相同。

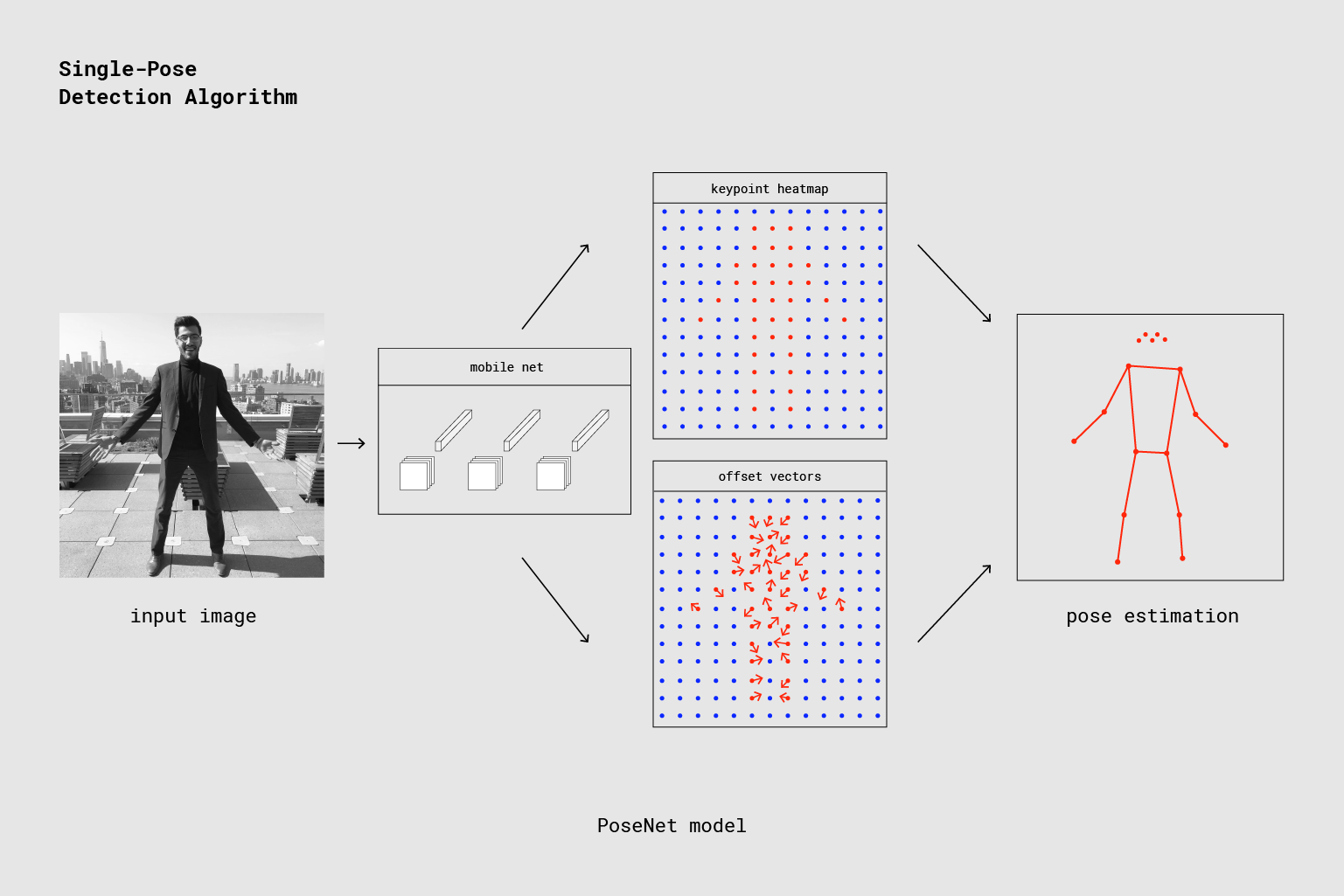
此短代码块演示如何使用多姿态估计算法：

姿势的示例输出数组如下所示：

如果你读了这么多，你知道的足够多了，可以开始使用PoseNet了。这可能是一个很好的停止点。如果您想了解更多关于模型和实现的技术细节，我们邀请您继续阅读下面的内容。

### 对于好奇的人来说：技术上的深度潜水

在这一节中，我们将讨论关于单姿态估计算法的更多技术细节。在较高层次上，流程如下：



需要注意的一个重要细节是，研究人员同时训练了PoseNet的a和a模型。ResNet模型具有较高的精度，但由于其规模大、层次多，使得页面加载时间和推理时间都不适合任何实时应用。我们采用了MobileNet模型，因为它是为在移动设备上运行而设计的。

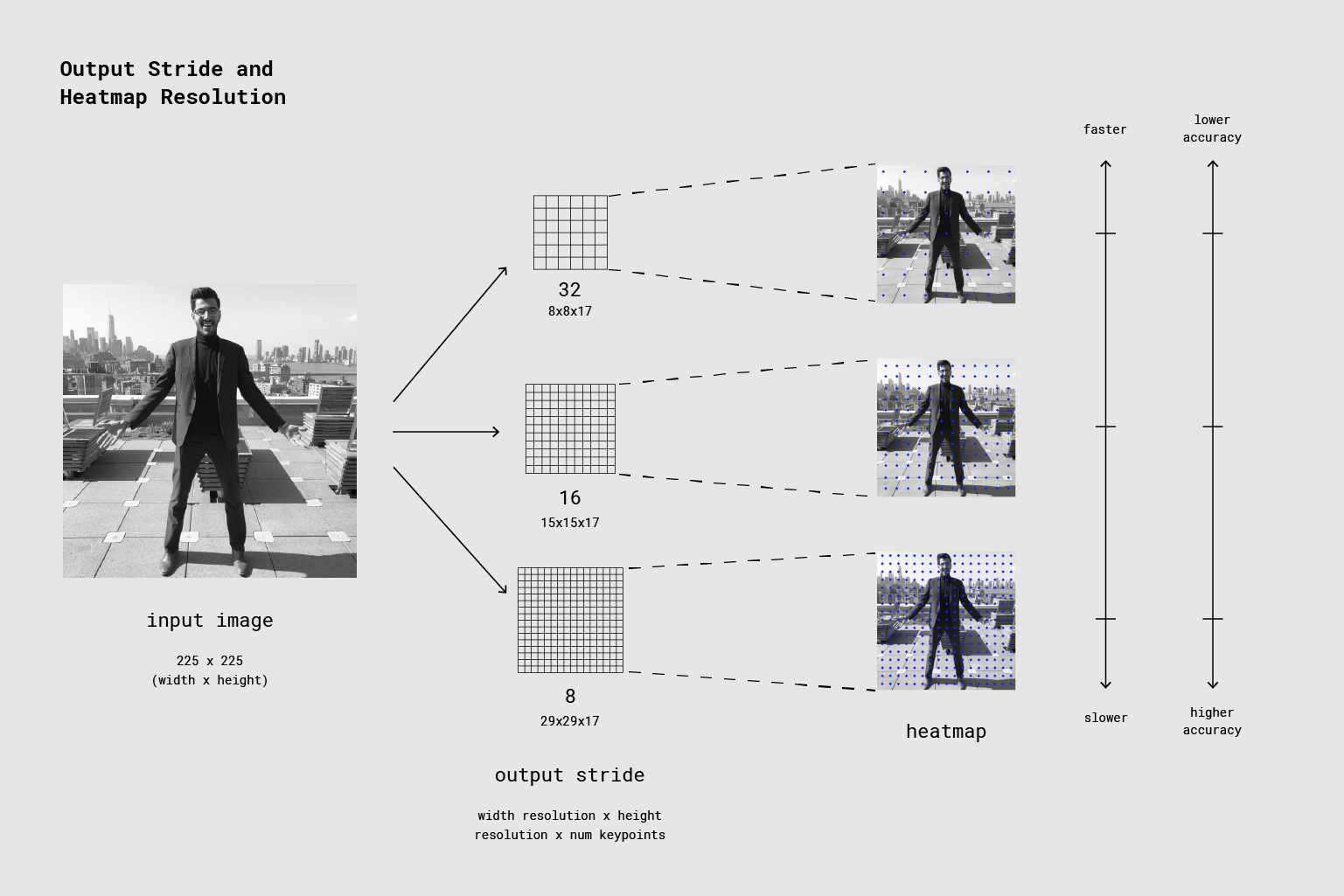
#### 单姿态估计算法再探讨

处理模型输入：输出跨步的解释

首先，我们将讨论如何通过输出跨步获得PoseNet模型输出（主要是热图和偏移向量）。

方便的是，PoseNet模型是图像大小不变的，这意味着无论图像是否缩小，它都可以在与原始图像相同的尺度上预测姿态位置。这意味着PoseNet可以通过在运行时设置上面提到的输出步幅来配置为以牺牲性能为代价具有更高的精度。

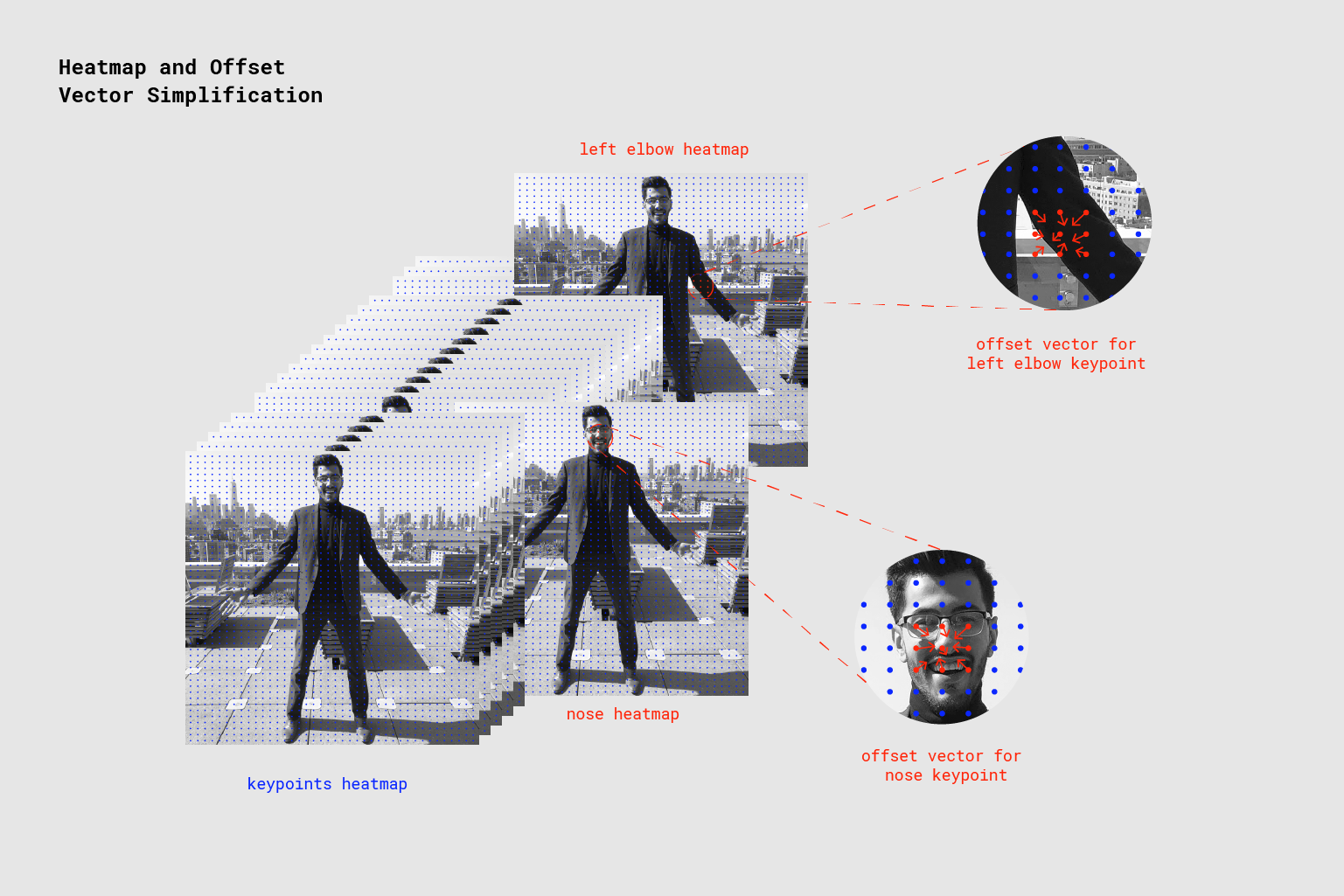
输出跨距决定了我们相对于输入图像大小缩小输出的程度。它影响图层的大小和模型输出。输出步长越大，网络层和输出层的分辨率就越小，相应的精度也就越低。在这个实现中，输出跨距的值可以是8、16或32。换句话说，32的输出步幅将导致最快的性能，但精度最低；8的输出步幅将导致最高的精度，但性能最低。我们建议从16开始。



在引擎盖下面，当输出跨距设置为8或16时，层中的输入跨距量减少，以创建更大的输出分辨率。然后用于使随后层中的卷积滤波器具有更宽的视野（当输出跨距为32时不应用阿托罗斯卷积）。虽然Tensorflow支持atrus卷积，但Tensorflow.js不支持，因此我们将其包括在内。

模型输出：热图和偏移矢量

当PoseNet处理图像时，实际上返回的是一个热图和偏移向量，可以解码这些向量，以在图像中找到与姿势关键点相对应的高置信度区域。我们将在一分钟内讨论每一个关键点的含义，但现在下面的插图从高层捕捉到每个姿势关键点是如何与一个热图张量和一个偏移向量张量相关联的。



这两个输出都是具有高度和宽度的三维张量，我们将其称为分辨率。根据以下公式，分辨率由输入图像大小和输出步幅确定：

热图

由于17是PoseNet检测到的关键点数量，因此每个热图都是一个大小分辨率x分辨率x 17的3D张量。例如，如果图像大小为225，输出跨距为16，则为15x15x17。三维（共17个）中的每个切片对应于特定关键点的热图。该热图中的每个位置都具有置信度得分，这是该点类型的一部分存在于该位置的概率。它可以被认为是原始图像被分解成15x15网格，其中热图得分提供了每个网格方格中每个键点存在的可能性的分类。

偏移矢量

每个偏移向量是尺寸分辨率x分辨率x 34的3D张量，其中34是关键点的数目\*2。图像大小为225，输出跨距为16，这将是15x15x34。由于HEATMAPS是关键点所在的近似，偏移矢量在位置上对应于热图点，并且用于通过沿着相应的热图点沿着矢量行进来预测关键点的确切位置。偏移向量的前17个切片包含向量的x，后17个切片包含向量的y。偏移向量大小与原始图像的比例相同。

根据模型输出估计姿态

在图像通过模型后，我们进行一些计算来估计输出的姿态。例如，单姿态估计算法返回一个姿态置信度得分，它本身包含一个关键点数组（由部件ID索引），每个关键点都有一个置信度得分和x，y位置。

要获取姿势的关键点：

1. 在热图上激活乙状结肠以获得分数。分数=热图。乙状结肠（）
2. 在关键点置信分数上进行ARGMAX2D，得到热图中的X和Y指数，每个部分的得分最高，这是最有可能存在的部分。这将产生17x2大小的张量，其中每一行是热图中的y和x索引，每个部分的得分最高。热图位置=分数。argmax（y，x）
3. 通过从对应于该零件的热图中的x和y索引的偏移量中获取x和y来检索每个零件的偏移矢量。这会产生17x2大小的张量，每一行是对应关键点的偏移向量。例如，对于索引k处的零件，当热图位置为y和d时，偏移向量为：offset vector=[偏移量.get（y，x，k），偏移量.get（y，x，17+k）]
4. 为了得到关键点，每个部分的热图x和y乘以输出跨距，然后添加到它们对应的偏移矢量中，偏移矢量与原始图像的比例相同。keypointPositions=heatmapppositions\*output stride+offset vector s
5. 最后，每个关键点的置信度得分是其热图位置的置信度得分。姿势自信得分是关键点得分的平均值。

#### 多人姿态估计

多姿态估计算法的细节不在本文的讨论范围之内。该算法的主要区别在于，它使用贪心过程，通过沿着基于零件的图形跟踪位移向量，将关键点分组为姿势。具体来说，采用了本文研究的快速贪婪译码算法。有关多姿态算法的更多信息，请阅读完整的研究论文或查看。

It’s our hope that as more models are ported to TensorFlow.js, the world of machine learning becomes more accessible, welcoming, and fun to new coders and makers. PoseNet on TensorFlow.js is a small attempt at making that possible. We’d love to see what you make — and don’t forget to share your awesome projects using #tensorflowjs and #posenet!