我们把问题体型分类问题划分成了两个大的重点子问题来解决——人体关键点检测以及聚类算法。通过将若干张图片进行人体关键点数据的识别和提取，再利用聚类算法求出目标3类体型的大概质心，就可以实现将体型分类为3种（当然也可以是更多种）。下面，先来说说人体关键点检测。

人体关键点检测（Human Keypoints Detection）又称为人体姿态估计，是计算机视觉中一个相对基础的任务，是人体动作识别、行为分析、人机交互等的前置任务。一般情况下可以将人体关键点检测细分为单人/多人关键点检测、2D/3D关键点检测，同时有算法在完成关键点检测之后还会进行关键点的跟踪，也被称为人体姿态跟踪。

目前COCO keypoint track是人体关键点检测的权威公开比赛之一，COCO数据集中把人体关键点表示为17个关节，分别是鼻子，左右眼，左右耳，左右肩，左右肘，左右腕，左右臀，左右膝，左右脚踝。而人体关键点检测的任务就是从输入的图片中检测到人体及对应的关键点位置。

我们也考虑过要用到与人体关键点检测很接近的另外一个领域的模型——人体姿态估计。关键点检测通过对人体主要骨骼点（关键点）的位置预测，实现人体姿态估计 (Human Pose Estimation, HPE)。HPE 是计算机视觉领域的基础任务之一，可广泛应用于动作识别、行人跟踪、自动驾驶、人机交互等领域。HPE 问题存在几大难点，包括：

（1）人体的非刚性结构导致任何一个部位发生变化都会产生新的姿态，尤其是自遮挡和复杂姿态问题。

（2）多样的服饰和不同人体结构的高度相似性，给关键点的精确定位带来诸多问题。

（3）环境干扰、前景遮挡、视野截断、人群遮挡和角度多变的拍摄视角，带来一些不确定性。

考虑到由于目标女性图片存在不同的背景以及各种各样的衣服的干扰，人体姿态估计模型的可靠性远远地不如最为基本的人体关键点检测模型。

我们所使用的人体关键点检测模型参考于以下地址：<https://blog.csdn.net/csdnliwenqi/article/details/121694973>。人体关键点检测的核心在于pytorch提供的提供的keypointrcnn\_resnet50\_fpn()网络模型，可以对17个人体关键点进行检测，分别是对人体部位的鼻子，左眼，右眼，左耳，右耳，左肩，右肩，左胳膊肘，右胳膊肘，左手腕，右手腕，左臀，右臀，左膝，右膝，左脚踝，右脚踝这17个关键点进行检测。

。它是通过使用 ResNet-50-FPN 主干构造 Keypoint R-CNN 模型。模型的输入应该是一个张量列表，每个形状为 [C, H, W] ，每个图像一个，并且应该在 0-1 范围内，不同的图像可以有不同的尺寸。由于我们使用的是预训练模型，所以该模型在处理图片时，直接进行推理。在推理过程中，模型只需要输入张量，并将后处理的预测作为List[Dict[Tensor]] 返回，每个输入图像一个。Dict的字段如下，其中N是检测到的实例数：

1. 框(FloatTensor[N, 4])：[x1, y1, x2, y2]格式的预测框，有0 <= x1 < x2 <= W和0 <= y1 < y2 <= H。
2. 标签(Int64Tensor[N])：每个实例的预测标签
3. 分数(Tensor[N])：分数或每个实例
4. 关键点(FloatTensor[N, K, 3])：预测关键点的位置，格式为[x, y, v]。

关键点 R-CNN 可导出到 ONNX 以用于固定批量大小，输入图像大小固定。

而其他代码，则是辅助keypointrcnn\_resnet50\_fpn()网络模型，包括将图片进行0~255到0~1的变换，对图片物体的定位、识别和分类，以及筛选出类别为“person”的标签进行关键点检测。

我们对代码的修改主要体现在以下：

1. 由于我们在后续的处理中主要需要用到左右肩和左右臀，所以计算了肩宽、臀宽以及肩臀距离，并在实验过程中，将每个人体数据保存到txt文件中，以方便承接后续聚类算法。
2. 原代码中对于检测人体模型，会先通过对于图片里的物品进行一次完全地检测和定位，选取概率阈值大于0.5的物品进行定位。但由于本身是人体关键点检测，所以关键点的检测和定位只对标签为“人”有效，所以结果图片中只会出现对标签“人”的框选。
3. 我们发现，在照片不清晰或者人物体型不确切的情况下，关键点检测和类别框选的准确度会下降，导致我们在利用50张单人女性照片进行实验的时候，实验结果出现了更多的情况（五十几个数据）。于是我们对代码进行了完善，将原本对图片的关键点检测的循环操作改成了单次操作，同时将优先级选定为最高概率对应的物品（优先定为“人”），使得最后每张图片只选取对应的女性进行人体关键点检测。此外，针对后续聚类算法的输入精度问题，我们只保存人体关键点检测的输出数据的前六位小数。

k-means算法属于无监督学习的一种聚类算法，其目的为：在不知数据所属类别及类别数量的前提下，依据数据自身所暗含的特点对数据进行聚类。对于聚类过程中类别数量k的选取，需要一定的先验知识，也可根据“类内间距小，类间间距大“（一种聚类算法的理想情况）为目标进行实现。

k-means算法以数据间的距离作为数据对象相似性度量的标准，因此选择计算数据间距离的计算方式对最后的聚类效果有显著的影响，常用计算距离的方式有：余弦距离、欧式距离、曼哈顿距离等。我们选用的是欧式距离。

欧式距离公式：

例子：若数据为其计算欧式距离如下(可理解为D表示维度，i,j表示行数)：

通过公式（1）可计算出每对数据对象间的距离，根据距离的远近进行聚类成指定的类别数K。对每一类中的数据初步选取类心，取的方式有多种如：

1.该类所有数据的均值；

2.随机取k个数据作为类心；

3.选取距离最远的k个点作为类心等。

以上方法均需要对初步的类心进行迭代，当类心变化缓慢时便可认为收敛，此时该点便为最终的类型。我们选取的方法1：

表示第k类，表示第k类中数据对象的个数。类心迭代过程如下：

通常迭代终止的条件有两种：1）达到指定的迭代次数T；2）类心不再发生明显的变化，即收敛。

我们在最初阶段的试验中选择的数据量不多，数据集选取clothing-co-parsing-master的25张图片和肥胖体型图片的25张图片。对比图片和结果，两者差距比较大，实验结果精度偏低。而后，我们尝试进行对更多数据的测试，从3个大的数据集中，提取出了2802张相对以肉眼分类比较清晰地女性体型图片。通过对这2802张图的实验，实验的结果相比上次有了很大的提升，准确度远超小数据集的结果。通过观察肥胖数据集的分离情况来评估聚类结果来看，我们将绝大部分肥胖数据集已经分离，说明结果比较符合预期。并且通过这次结果的可行性，我们完善了对后续聚类算法的优化。

结果：



