POSE ESTIMATION

<https://mp.weixin.qq.com/s/_78TgTkxnKWxbLKiTqElEQ>

## 数据集：

Openpose不是可以检测64个关键点吗，为什么没有64个关键点的数据集，还是openpose的数据集并没有开源？

## 传统算法概述：

在几何先验的基础上基于模板匹配思路来进行。

**Pictorial structure：**单元模版（unary templates）和模版关系（pairwise springs）模版关系：弹簧形变模型

**Mini parts：**将每个肢体结构（part）切分成更小的parts以能够模拟更多的姿态变化，从而提高模版匹配的效果。

## 人体骨骼关键点检测：

### 算法概述：

自上而下：人体检测和担任人体关键点检测。先进行目标检测，在框选基础上实现人体骨骼关键点检测，代表性算法有G-RMI, CFN, RMPE, Mask R-CNN, and CPN

自下而上：关键点检测和关键点聚类。对关键点之间关系进行建模的代表性算法有PAF, Associative Embedding, Part Segmentation, Mid-Range offsets。

### Coordinate、Heatmap和Heatmap + Offsets：

Ground Truth：监督学习的训练集的分类准确性，正确打标记的数据称为Ground Truth。

关键点回归的Ground Truth问题构建：

Coordinate和Heatmap两种思路，Coordinate没有Heatmap好。

Heatmap+Offset的Ground Truth构建方案，关键点一定范围内为1，范围外为offset值。目前还没有得到应用。

### 自上而下的关键点检测算法：

目标检测和单人人体骨骼关键点检测，目标检测算法，略。关键点检测算法，一要考虑较大的感受野区域，二不同的关键点要区别对待，三是关键点定位依赖于算法的proposal，出现检测不准或者检测重复问题。

Convolutional Pose Machine：响应图和特征图一起作为数据在网络中传递。先验分布指导网络的学习。多监督学习。

Cascaded Pyramid Network：不同类别关键点检测难度不同，先检测比较简单的关键点、最后检测更难的或者不可见的关键点。两个stage，GlobalNet和RefineNet。RefineNet中采用了Hard Negative Mining策略。

RMPE：考虑自上而下的关键点检测算法都在目标检测中产生Proposals的过程中，可能会出现检测框定位误差、对同一个物体重复检测等问题。

### 自下而上的人体关键点检测算法：

关键点检测和关键点聚类，关键点检测中会将图片中所有类别的所有关键点检测出来，然后对关键点进行聚类处理，将不同人的不同关键点连接在一块，从而聚类产生不同的个体。关键点聚类方法的探索：

Part Segmentation：

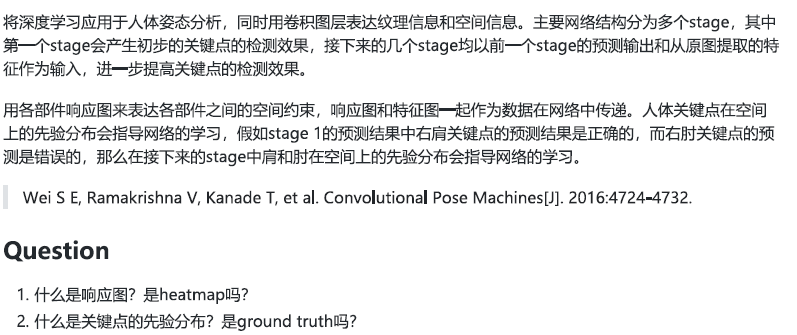
Part Affinity Field：

Associative Embedding：高维空间的向量编码

Mid-Range Offsets：直接回归一个关键点到另一个关键点之间的offset来建模两个关键点之间的关系。

## 问题解决：

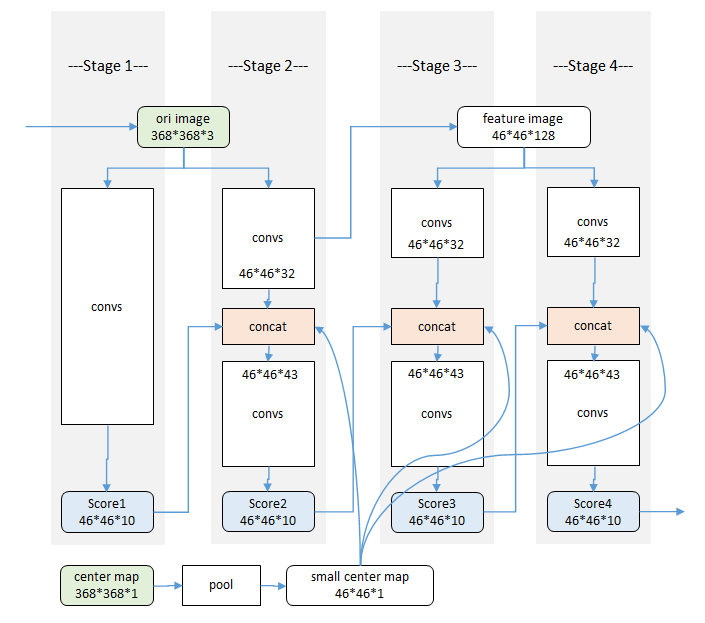
### Convolutional Pose Machines：



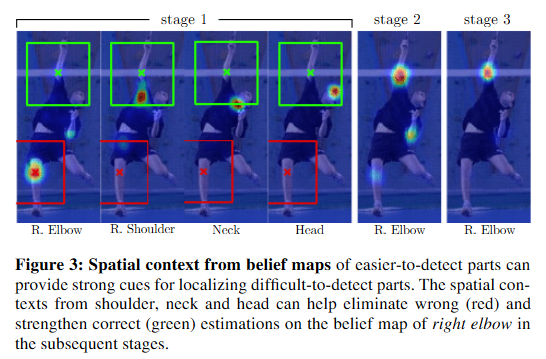
#### Question：

**1.什么是响应图？是heatmap吗？**

在Concolutional Pose Machines中网络结构如下图所示：其训练分为四个阶段，在二三四阶段都将image features and the belief maps produced by the previous stage（即特征图和响应图）在concat串联层一起作为数据传递。其实还加入了46\*46\*1的中心约束。所以响应图就是图中的score1、score2、score3。个人理解应该是对于各个关键点的location的一个belief map，也就是heatmap。（confidence map，belief map，heatmap……真是各种叫法都有……）

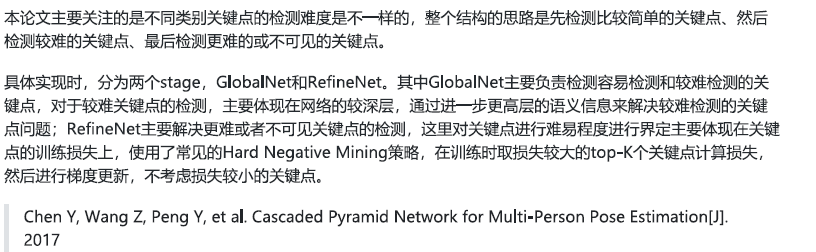


**2.什么是关键点的先验分布？是ground truth吗？**



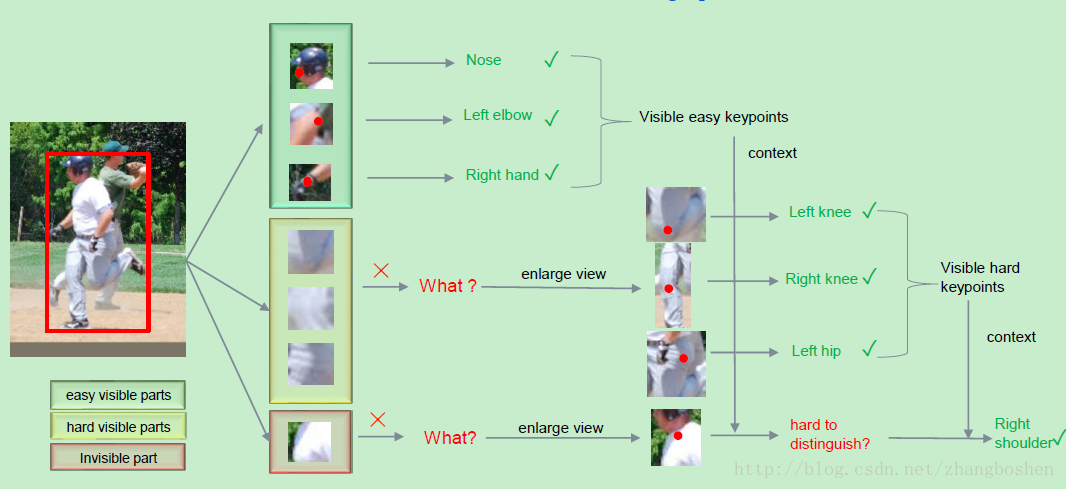
参考figure3下面的注释以及原文中内容，我倾向于认为关键点的先验分布是指关键点空间位置的相互限制，在图片举例中，对于R.Elbow的检测一开始出现错误，但是后面R.Shoulder，neck和head的正确数据会帮助判断红色框选是错误的，而绿色框选是正确的。所以先验分布是指（个人理解）在大数据统计下得到的某关键点附近其他关键点的概率分布。

### Cascaded Pyramid Network:



#### Question：

**1.怎么定义哪些是容易检测，哪些较难检测的点？在训练集中如何做到区分的？**

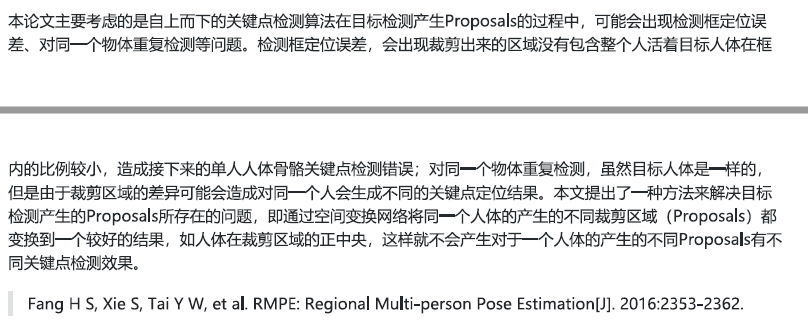


容易检测的点是指在小的感受野中就很容易识别的点，较难检测的点是指需要增大局部区域感受野（通过增加网络层数）才可以检测到的点，并且在较深的网络中会使用容易检测点作为辅助判断。原文中说的是用globalnet检测simple keypoints，用refinenet检测hard keypoints。在训练集中不做预先划分吧。

**2.什么是Hard Negative Mining策略？**

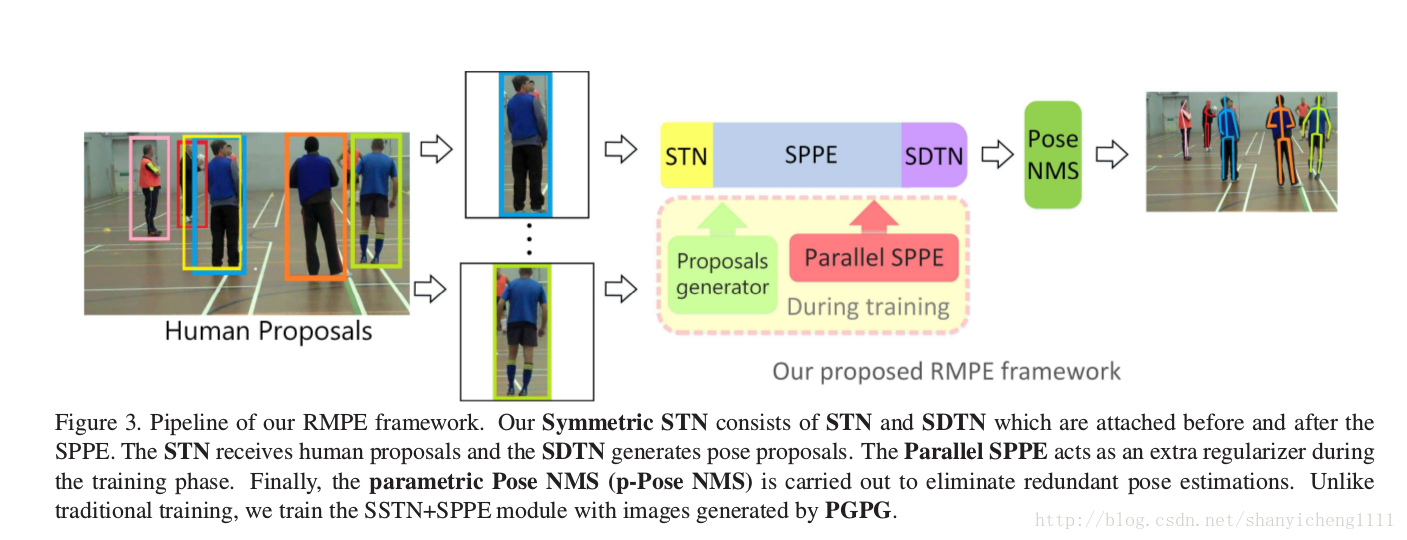
对于目标检测中我们会事先标记处ground truth，然后再算法中会生成一系列proposal，这些proposal有跟标记的ground truth重合的也有没重合的，那么重合度（IOU）超过一定阈值（通常0.5）的则认定为是正样本，以下的则是负样本。然后扔进网络中训练。However，这也许会出现一个问题那就是正样本的数量远远小于负样本，这样训练出来的分类器的效果总是有限的，会出现许多false positive，把其中得分较高的这些false positive当做所谓的Hard negative，既然mining出了这些Hard negative，就把这些扔进网络再训练一次，从而加强分类器判别假阳性的能力。

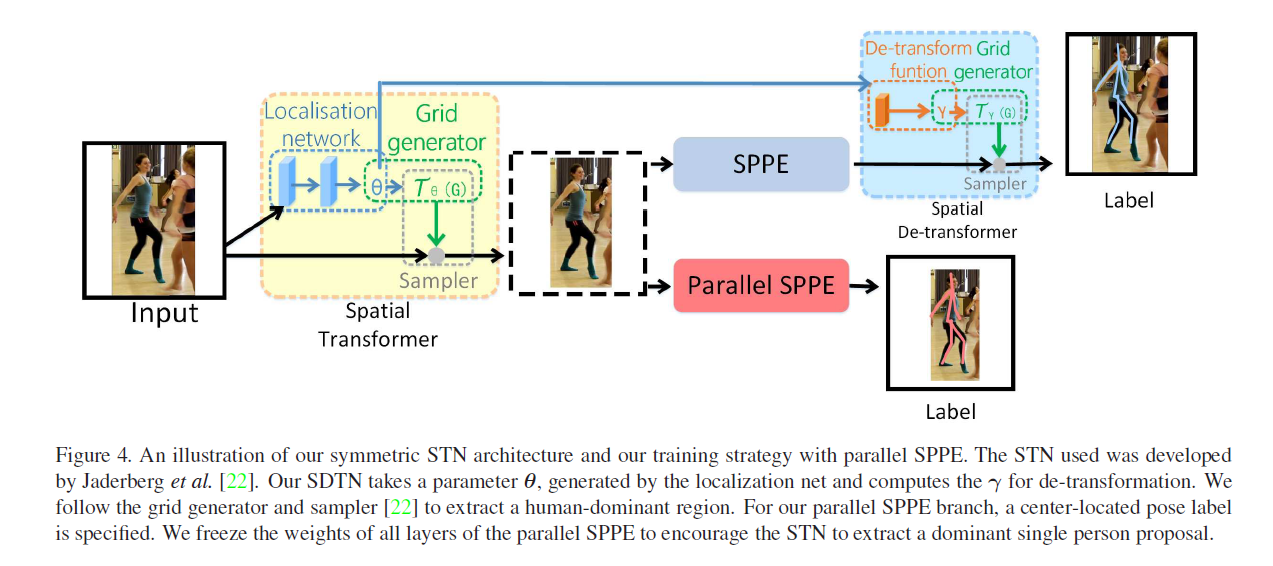
### RMPE:



#### Questions：

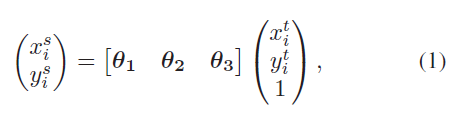
**如何进行空间变化，什么叫较好的结果？**



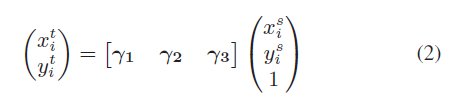


空间变换是指其中的STN(spatial transformer network)，她的作用是从一个不准确的边框中提取一个高质量的单人区域来进行处理。SDTN(spatial de-transformer network)是其对称变换。这样做的原因是SPPE需要单人图像来训练会得到比较好的准确率。较好的结果就是得到了一个高质量的单人区域。

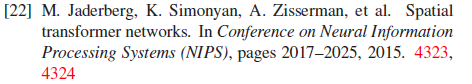
空间变换方法，对于STN，它提供一个2D仿射变换：



对于SDTN，它提供了反变换：



关于为什么这个变换可以得到高质量的单人区域，还需要细读另一篇文章

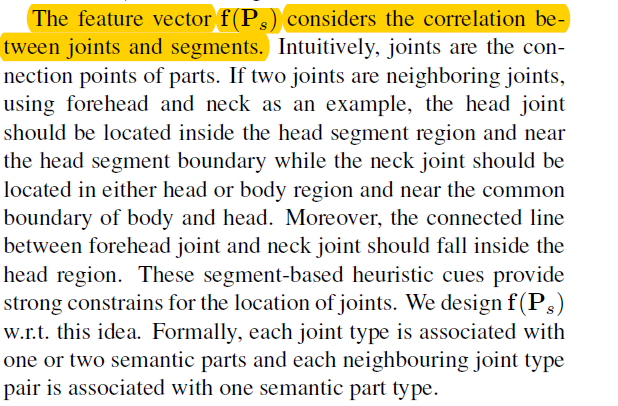


### Part Segmentation:

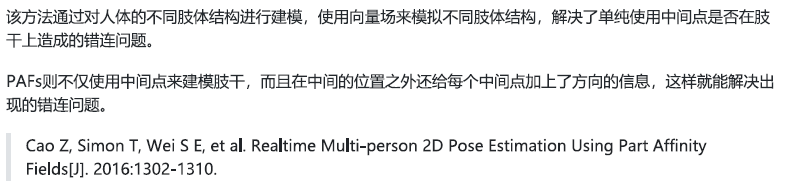


#### Questions：

**什么叫做分割区域的特定位置：**

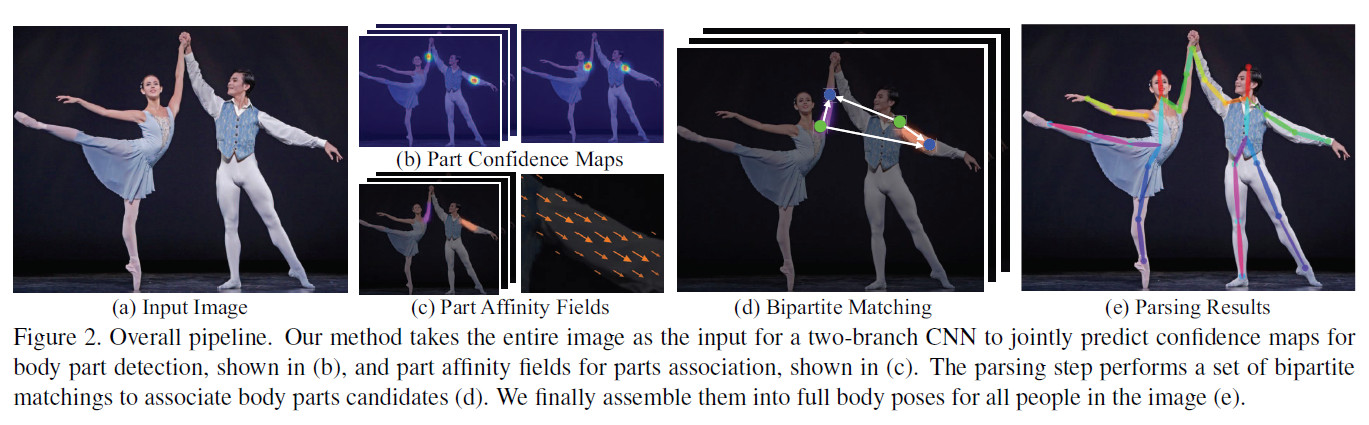
在本文中，先是独立的利用pose FCN和part FCN分别得到pixel-wise joint score map Pj、piwel-wise joint neighbor score map Pn和part score map Ps。然后利用三个map作为输入，设计一个FCRF得到refined pose estimation，然后用part score map Ps和refined pose estimation得到refined poses part segmentation。文章中的一段话详细的解释了问题的FCRF过程。

### Part Affinity Fields：



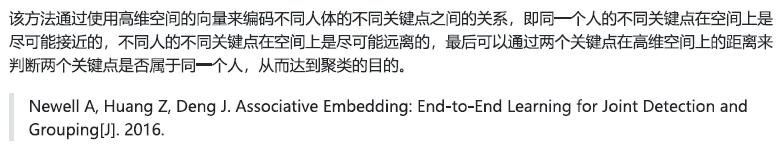
#### Questions：

**方向点是如何获取的？中间点是否就是图片上距离的中间点？**



网络的结构通过上图来说明，首先，网络通过两个互相独立的CNN得到关键点的confidence maps S（即heat maps、belief maps）和关键点之间连接的affinity fields L，S和L都是通过ground truth数据训练的。（感觉带向量的ground truth好高级，但是文中没有说怎么得到的）个人理解中间点就是指关键点，方向点就是指L上的点。

### Associative Embedding：



#### Questions：

**应该是特殊训练集，训练集中就要给出不同人的关键点的向量编码：**

训练集是需要有关键点标签以及几个关键点是否属于同一物体，不需要其他特殊标注，因为在训练中，会同时产生keypoint detection heatmap和human tagging heatmap，对于多人图片，同一个关键点的heatmap会有多个peak，同一个human的各个关键点有相近的tag（但是这些tag不是自己指定的固定值，我们只在意同一个human的tag的difference），根据tag值进行group，