这里，我不具体列出参数，只说明参数的物理意义。

1 Bottom Up

1. 第0层：生成model和对应的instance。
2. 第1 ~ L层（第1步）：组合。组合下层的instance，生成本层instance的list。
3. 第1 ~ L层（第2步）：生成本层的model。由第一步的list进行聚类产生。
4. 第1 ~ L层（第3步）：inference。对生成的model做inference。
5. 第1 ~ L层（第4步）：prune。去除支持少的model。
6. 第1 ~ L层（第5步）：Competitive Exclusion。

2 Bottom Up: 第0层

1. 生成4个model（$ori = 0, \pi/4, \pi/2, \pi * 3/4,$）。
2. 对图像抽canny edge并做edge grouping，生成所有pixel（也是第0层的instances）
3. 对上一步生成的instance进行聚类。算法是DBScan，没有排序之类的东西。要求：同图、位置近和梯度方向相似。
4. 对聚类后的instance找对应的model。1个instance可对应多个model。要求：model方向和instance梯度方向相似。

3 Bottom Up: 第l层（第1步）

1. for $p1$ in 第l层所有的instances，寻找它周围区域中的两个instance $p2, p3$进行组合生成新的instance p。
2. 如果满足下面要求，则生成一个新的instance p，加入list。
   - $p1$ 与$p2, p3$ 的距离要小于($p1$的半径*2)。
   - $p1, p2, p3$的score要大于一定值（即取下一层中分数较高的instance）
   - $p2, p3$ 必须在$p1$最近的K近邻中。（这意味着不考虑重复最多生成$N * K * K$个proposal，N是下一层instances的个数)
   - p 和$p1, p2, p3$ 分别的overlap要小于一定值。（意义：p比它的孩子而言，描述了更大的区域）

4 Bottom Up: 第1 ~ L层（第2步）

1. 用DBScan算法对list中instance进行聚类，其中要判断instance $p1$和$p2$是否相似。生成的每个类就是一个model，均值取所有元素均值，方差固定(0.12)。
2. instance $p1$和$p2$相似条件如下:
   - $p1$和$p2$它们3个子instance对应的model必须一致。
   - 将$p1$转化为一个高斯分布，均值是$p1$的triplet的feature，方差是固定值(0.12)。要求$p2$在$p1$生成的model中的概率大于一定阈值。
3. prune掉一些类，出于计算上考虑。
   - 类中instance小于5个去除。
   - 只保留最大的1000个类。
   - (保留下来的类中所有instance的数量) < (list中instance的数量*一个比例)

5 Bottom Up: 第1 ~ L层（第3步）

1. 对生成的model做inference，需要参数是prune的条件和NMS的条件。
2. prune的条件:
   - 处理Missing的code被注释了。原因未知，9成可能是code有问题。
   - 3个孩子中，$p1$ 与$p2, p3$ 的距离要小于($p1$的半径*3)。
   - 3个孩子$p1, p2, p3$之间scale和rotate不能相差太大。
3. NMS的条件：此处无新意，和以前做法比。

6 Bottom Up: 第1 ~ L层（第4步）

1. model的score：该model在所有图像中取score最高的instance，然后对这些instances的前$P\%$（按照score高低排序）求score平均值。该平均值就是model的score。如果图像中没有对应的instance，则用一个惩罚性的低score代替。
2. prune的条件:
   - 在score最高的1000个model中。
   - score大于一定阈值。

7 Bottom Up: 第1 ~ L层（第5步）

1. 描述相同区域的model只留下最好的一个。算法DBScan，要先对model按照它们的score排序。

2. model $m1$ 和$m2$ 相似的条件（$m1$ 分数大于$m2$）：$m2$中的$s1\%$的instance能够在$m1$中找到有overlap大于一定阈值的instance。

3. instace的overlap的计算：它们孩子间的bounding box的交除以并。

8 Top Down

1. 从高层Level L到底层Level 2，进行Top Down。

2. 利用Top Down得后的model对应的instance重新计算model的feature的均值。换个角度，我们已经有了model和它在training image上的parse results，如果假设parse results是对的(groundtruth)，那么可以通过计算model的参数。再一个不恰当的比喻，如EM算法，E步估计了隐变量（object的位置，也就是parse results），这里就是M步，重新估计参数。

9 Top Down: 第L ~ 2层

1. 添加第l层model的孩子。Top Down的过程是和Bottom Up非常类似的过程，并不是在model直接添加孩子，还要通过instance组合来生成。

2. 这里举一个例子，m_l是l层的model（假设3个孩子），我们想把$m_(l-1)$这个l-1层的model添加为m_l需要经历以下步骤:

   (a) m_l在所有图像中有N_i个instance（3个孩子）；$m_(l-1)$在所有图像中有N_j个instance。对它们进行组合，生成新的instance（4个孩子，3个来自m_l，1个来自$m_(l-1)$）。

   (b) 新生成的instance的量级是$K*(N_j/K)*(N_i/K)$, k是training图像个数。

   (c) 如何从新生成的instances得到model，和bottom up的过程就是一致的（思想上也是一致的）。

   (d) 和bottom up的区别在：Competitive Exclusion。bottom up是去除描述相同区域的model，top down是保证新model描述比原来更大的区域。

   (e) 最最极端的理解（只是操作上）：就是把l层和$l-1$层一起做了一次bottom up (只是bottom up中一层的操作）

3. 生成第l层所有model所对应instance的bounding box R_l.

4. for model $m_l i$ of level l; for model $m_{(l-1)j}$ of level $l-1$;

(a) 将model $m_l i$对应的instance和$m_{(l-1)j}$的instance进行组合，生成list。（类似bottomup的第1步，组合的两个instances中心在图像上距离小于一定值）

(b) 对list进行聚类生成model，同bottomup的第2步。（同bottomup比，没有inference步骤，model对应的instance就是聚类时类里的instance）

(c) 对model打分并做prune，同bottomup的第4步。（计算score时，只用最好的1/3的图像）

(d) 计算model $m_{(l-1)j}$与R_l的overlap，即有$p\%$的$m_{(l-1)j}$的instance的bounding box的$s\%$在R_l可以找到。如果$p\%$大于一定比例，则$m_{(l-1)j}$不被添加到$m_l i$。

(e) 选择score最高的$m_{(l-1)j}$加入$m_l i$中。如果$m_l i$的孩子还小于5个，重新继续循环$m_{(l-1)j}$，进行添加。

5. 进行下一层的topdown