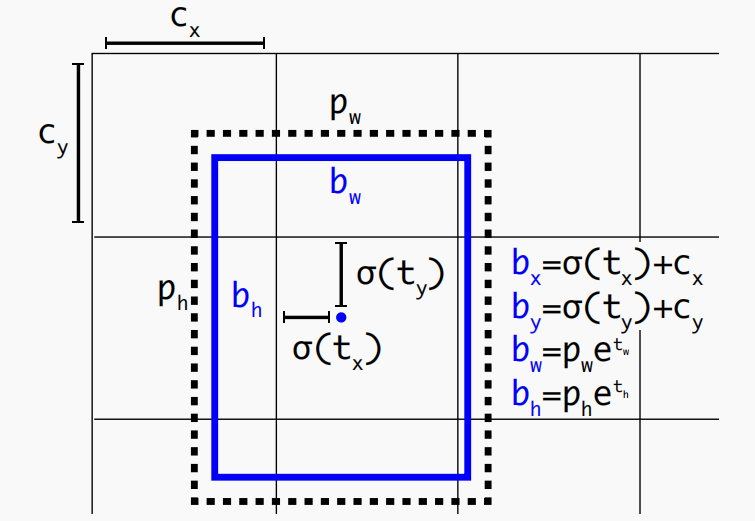
* **预测过程：**

对于一张图片，不管是什么尺寸的，首先reshape到我们自己定义的尺寸，比如416×416，这个在yolo.py程序下的 "input\_shape" : [416, 416],下定义。然后送入YOLOv3网络中，会输出3个特征图，比如13×13×255,26×26×255,52×52×255。

对于其中一个特征图，比如13×13×255，一共13×13个Grid cell，每一个grid cell都会产生3个anchor，中心都在grid cell的像素中心，大小分别为[116,90],[156,198],[373,326]缩放32倍之后的（因为k-means聚类产生的anchor尺寸是相对于原图的）。我们现在只看一个grid cell，他有3个anchor，看其中一个anchor，他预测85个值(tx,ty,tw,th,p,c1,c2,c3,......,c80)。假设这个anchor属于grid cell\_A，grid cell\_A左上角的gird cell我们记作grid cell\_B，那么预测框是怎么来的呢？

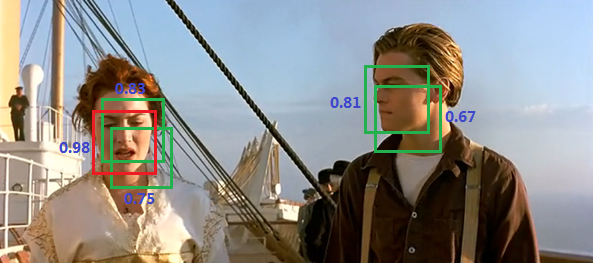


如图，虚线框是anchor，蓝色框是预测框。Cx和Cy是grid cell的宽高，那么该anchor预测出来的tx和ty就能确定预测框的中心点坐标。pw和ph是该anchor的宽高，那么该anchor预测出来的tw和th就能确定预测框的宽高。那么整个预测框就被确定了。这是一个grid cell的一个anchor产生的预测框，同时该预测框被赋予了置信度p和类别置信度ci（i=1,2,...,80）。然后我们设置一个置信度阈值，对p值低于这个阈值的预测框进行过滤扔到，得到剩下的预测框，剩余的预测框的p分别乘以ci就得到每个类别的概率，我们取到最大类别概率及其对应的索引，根据索引会返回对应类别。那么这个预测框就被赋予两个属性：预测的某个类别的概率，也叫confidence，以及类别。再将预测框根据下采样倍数，返回到原图，得到原图上的预测结果。

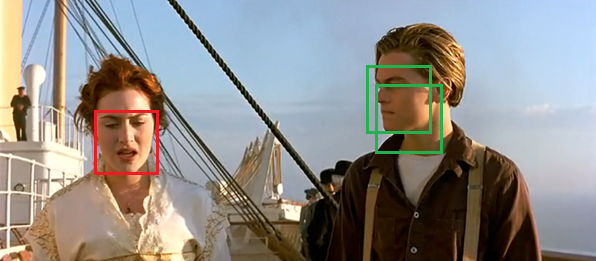
上面只是13\*13这个维度的，那么对于其他维度，也是同样的道理。

最后我们通过NMS消除各个类别重叠较大的预测框，得到最终的预测结果：

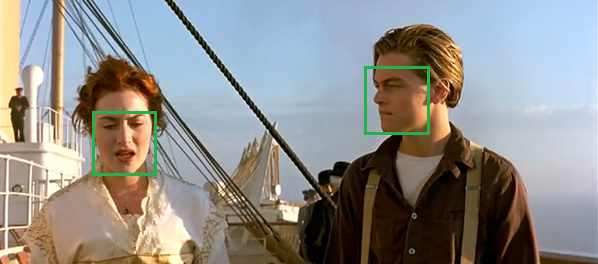
1. 首先对于图片上的所有预测框，将所有框的得分排序，选中最高分及其对应的框；



1. 遍历其余的框，如果和当前最高分框的IOU大于一定阈值，我们就将框删除。



1. 从未处理的框中继续选一个得分最高的，重复(1)(2)步。



* **训练过程：**

首先对一张带标签的训练样本（可以是任意尺寸），进行resize，假定为416\*416，对于一个batch可以进行数据增强，比如随机水平翻转等。送入到YOLOv3网络，得到3个特征图。以其中一个shape为13×13×255的特征图为例：

训练样本的标签根据下采样倍数可以等比例映射到特征图上，比如称为ground truth，然后看该ground truth的中心落在哪个grid cell内，比如grid cell\_A内，grid cell\_A会产生3个anchor。Anchor的中心在A的中心，具有不同的宽高。假定ground truth的中心与anchor的中心重合，计算iou值（也有博文说直接计算iou的），得到3个iou值。同理在其他两个特征图上同样计算3个iou值，对一个ground truth就会得到9个iou值。与ground truth的iou最大的anchor为正样本positive，负责去拟合该ground truth。iou值小于某个阈值（通常为0.5）的anchor为负样本negative。Iou大于阈值但不是最大的anchor是忽略样本，不对损失函数产生贡献。

正样本的置信度标签置为1，负样本的置信度标签置为0.在三个尺度上计算损失函数再进行反向传播即可。

* **如何resize：**

