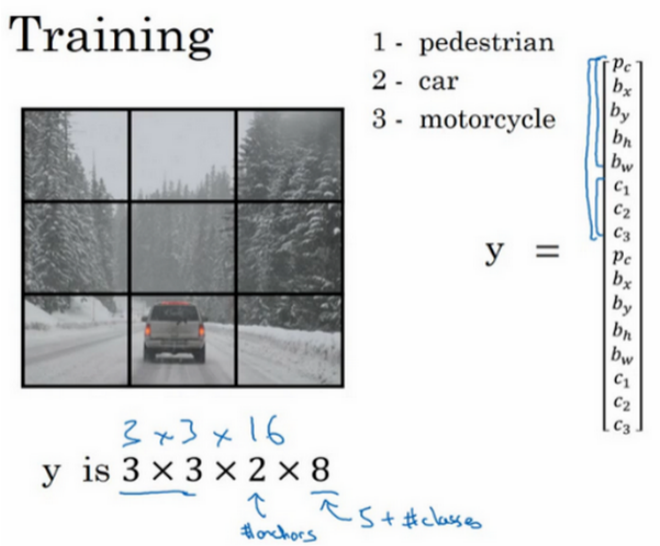
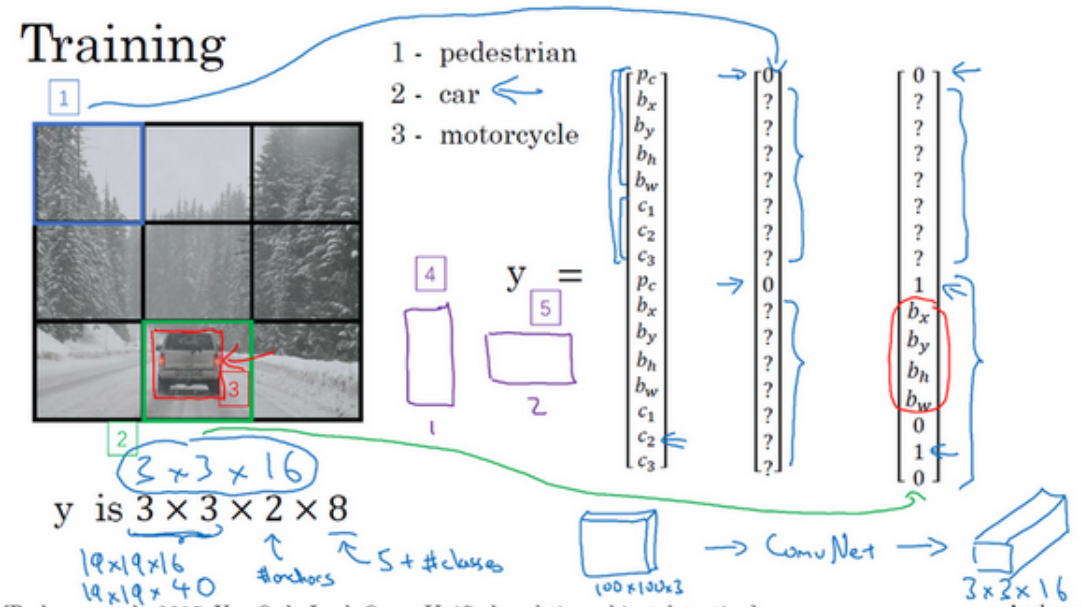
3.9 YOLO 算法（Putting it together: YOLO algorithm）

你们已经学到对象检测算法的大部分组件了，在这个视频里，我们会把所有组件组装在一起构成 **YOLO** 对象检测算法。

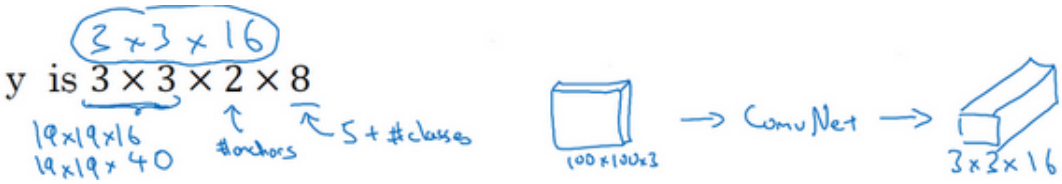


我们先看看如何构造你的训练集，假设你要训练一个算法去检测三种对象，行人、汽车和摩托车，你还需要显式指定完整的背景类别。这里有 3 个类别标签，如果你要用两个 **anchor box**，那么输出 y 就是 **3×3×2×8**，其中 3×3 表示 3×3 个网格，2 是 **anchor box** 的数量，8 是向量维度，8 实际上先是 5（p_c, b_x, b_y, b_h, b_w）再加上类别的数量（c_1, c_2, c_3）。你可以将它看成是 3×3×2×8，或者 3×3×16。要构造训练集，你需要遍历 9 个格子，然后构成对应的目标向量y。



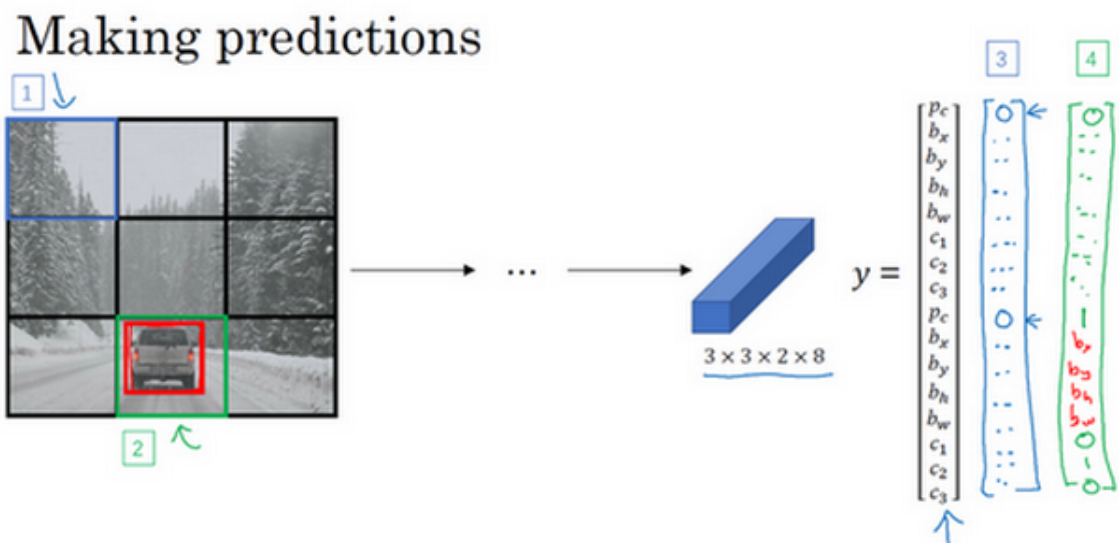
所以先看看第一个格子（编号 1），里面没什么有价值的东西，行人、车子和摩托车，三个类别都没有出现在左上格子中，所以对应那个格子目标y就是这样的，$y = [0 \quad ? \quad ? \quad ? \quad ? \quad ? \quad ? \quad ? \quad 0 \quad ? \quad ? \quad ? \quad ? \quad ? \quad ? \quad ?]^T$，第一个 **anchor box** 的 p_c 是 0，因为没什么和第一个 **anchor box** 有关的，第二个 **anchor box** 的 p_c 也是 0，剩下这些值是 **don't care-s**。

现在网格中大多数格子都是空的，但那里的格子（编号 2）会有这个目标向量y，$y = [0 \quad ? \quad ? \quad ? \quad ? \quad ? \quad ? \quad ? \quad 1 \quad b_x \quad b_y \quad b_h \quad b_w \quad 0 \quad 1 \quad 0]^T$，所以假设你的训练集中，对于车子有这样一个边界框（编号 3），水平方向更长一点。所以如果这是你的 **anchor box**，这是 **anchor box 1**（编号 4），这是 **anchor box 2**（编号 5），然后红框和 **anchor box 2** 的交并比更高，那么车子就和向量的下半部分相关。要注意，这里和 **anchor box 1** 有关的 p_c 是 0，剩下这些分量都是 **don't care-s**，然后你的第二个 $p_c = 1$，然后你要用这些（b_x, b_y, b_h, b_w）来指定红边界框的位置，然后指定它的正确类别是 2($c_1 = 0, c_2 = 1, c_3 = 0$)，对吧，这是一辆汽车。

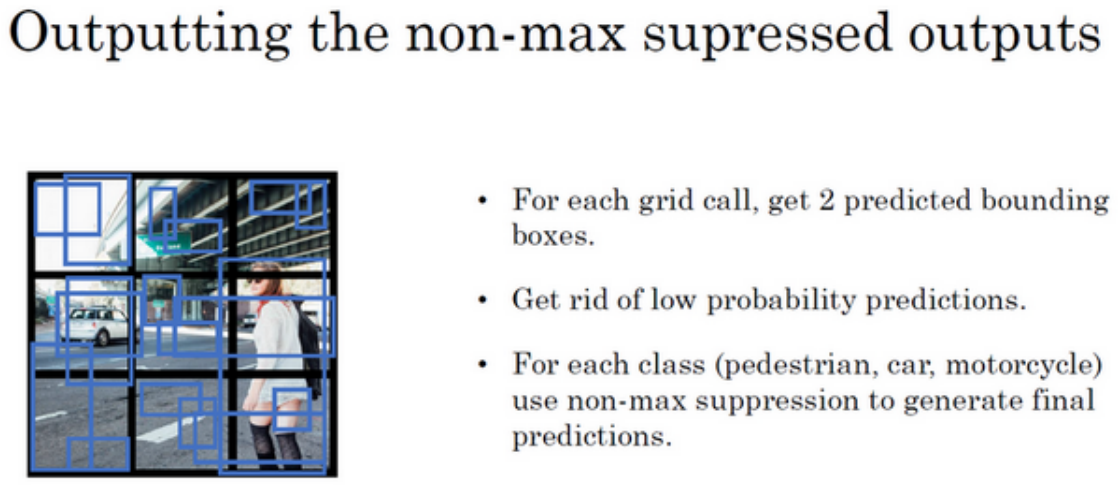


所以你这样遍历 9 个格子，遍历 3×3 网格的所有位置，你会得到这样一个向量，得到一个 16 维向量，所以最终输出尺寸就是 3×3×16。和之前一样，简单起见，我在这里用的是 3×3 网格，实践中用的可能是 19×19×16，或者需要用到更多的 **anchor box**，可能是 19×19×5×8，即 19×19×40，用了 5 个 **anchor box**。这就是训练集，然后你训练一个卷积网络，输入是图片，可能是 100×100×3，然后你的卷积网络最后输出尺寸是，在我们例子中是 3×3×16 或者 3×3×2×8。

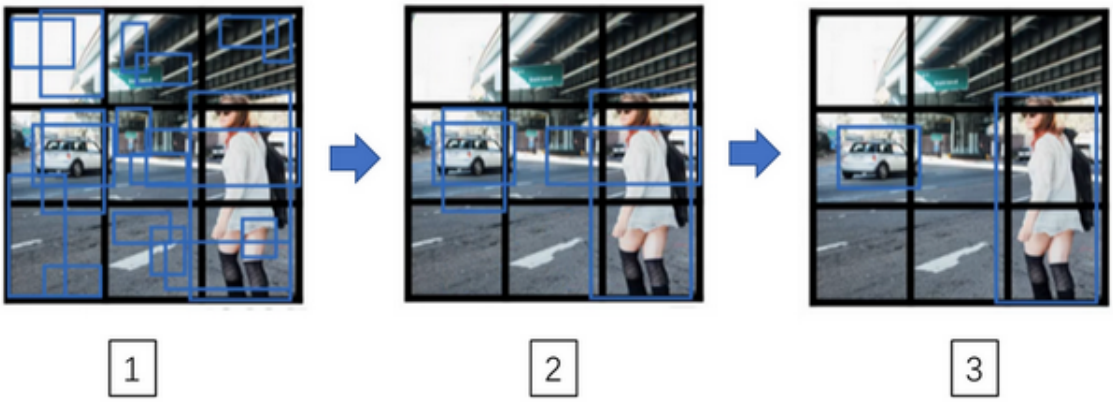
接下来我们看看你的算法是怎样做出预测的，输入图像，你的神经网络的输出尺寸是这个 3×3×2×8，对于 9 个格子，每个都有对应的向量。对于左上的格子（编号 1），那里没有任何对象，那么我们希望你的神经网络在那里（第一个p_c）输出的是 0，这里（第二个p_c）是 0，然后我们输出一些值，你的神经网络不能输出问号，不能输出 **don't care-s**，剩下的我输入一些数字，但这些数字基本上会被忽略，因为神经网络告诉你，那里没有任何东西，所以输出是不是对应一个类别的边界框无关紧要，所以基本上是一组数字，多多少少都是噪音（输出 y 如编号 3 所示）。



和这里的边界框不大一样，希望y的值，那个左下格子（编号 2）的输出y（编号 4 所示），形式是，对于边界框 1 来说（p_c）是 0，然后就是一组数字，就是噪音（**anchor box 1** 对应行人，此格子中无行人，$p_c = 0, b_x = ?, b_y = ?, b_h = ?, b_w = ?, c_1 = ? c_2 = ?, c_3 = ?$）。希望你的算法能输出一些数字，可以对车子指定一个相当准确的边界框（**anchor box 2** 对应汽车，此格子中有车，$p_c = 1, b_x, b_y, b_h, b_w, c_1 = 0, c_2 = 1, c_3 = 0$），这就是神经网络做出预测的过程。



最后你要运行一下这个非极大值抑制，为了让内容更有趣一些，我们看看一张新的测试图像，这就是运行非极大值抑制的过程。如果你使用两个 **anchor box**，那么对于 9 个格子中任何一个都会有两个预测的边界框，其中一个的概率p_c很低。但 9 个格子中，每个都有两个预测的边界框，比如说我们得到的边界框是是这样的，注意有一些边界框可以超出所在格子的高度和宽度（编号 1 所示）。接下来你抛弃概率很低的预测，去掉这些连神经网络都说，这里很可能什么都没有，所以你需要抛弃这些（编号 2 所示）。



1 2 3

最后，如果你有三个对象检测类别，你希望检测行人，汽车和摩托车，那么你要做的是，**对于每个类别单独运行非极大值抑制，处理预测结果所属类别的边界框，用非极大值抑制来处理行人类别，用非极大值抑制处理车子类别，然后对摩托车类别进行非极大值抑制，运行三次来得到最终的预测结果。**所以算法的输出最好能够检测出图像里所有的车子，还有所有的行人（编号 3 所示）。

这就是 **YOLO** 对象检测算法，这实际上是最有效的对象检测算法之一，包含了整个计算机视觉对象检测领域文献中很多最精妙的思路。