**摘要：**

我们提出一种使用单个深层神经网络检测图像中的对象的方法。我们的方法，称为SSD，将边界框的输出空间离散化为不同宽高比的一组默认框和每个特征图位置的缩放。在预测时间，网络为每个默认框中的每个对象类别的存在生成分数，并产生对框的调整以更好地匹配对象形状。此外，该网络将具有不同分辨率的多个特征图的预测结合起来，以自然地处理各种尺寸的对象。SSD相对于需要对象提议的方法而言是简单的，因为它完全消除了提案生成和后续像素或特征重采样阶段，并将所有计算封装在单个网络中。这使得SSD易于训练并直接集成到需要检测组件的系统中。PASCAL VOC，COCO和ILSVRC数据集的实验结果证实，SSD对于利用附加目标提案步骤的方法具有竞争性的准确性，而且速度更快，同时为训练和推理提供统一的框架。对于300×300输入，SSD在VOC2007测试中达到74.3％，在Nvidia Titan X上为59 FPS，512×512输入为SSD，SSD达到76.9％的mAP，超过了相当的最先进的更快的R-CNN模型。与其他单级方法相比，即使使用较小的输入图像尺寸，SSD也具有更好的精度。代码可在：https：//github.com/weiliu89/caffe/tree/ssd。

**1 简介：**

当前最先进的对象检测系统是以下方法的变体：假设边界框，对每个框重新采样像素或特征，并应用高质量分类器。由于选择性搜索工作[1]，目前PASCAL VOC，COCO和ILSVRC检测的主要结果都是基于更快的R-CNN [2]，尽管具有更深入的特征，如[3]，这一流程已经占据了检测基准。虽然这些方法是准确的，但是对于嵌入式系统来说，这些方法对于计算量太大，即使是高端硬件，对于实时应用来说也太慢了。这些方法的检测速度通常以秒为单位（SPF）测量，甚至是最快的高精度检测器，更快的R-CNN，每秒只能工作7帧（FPS）。已经有很多尝试通过修改每个阶段的检测方法来构建更快的检测器（参见第4节的相关工作），但到目前为止，显着增加的速度只是以显着降低的检测精度为代价。

本文介绍了第一个基于深度网络的目标检测器，它不会对边界框假设的像素或特征进行重新采样，并且与所做的方法一样准确。这导致高精度检测的速度显着提高（VOP2007测试为59 FPS，mAP为74.3％，而mAP为73.2％的快速R-CNN 7 FPS或mAP为63.4％的YOLO 45 FPS）。速度的根本改进来自于消除边框提案和随后的像素或特征重采样阶段。我们不是第一个这样做（cf [4,5]），但是通过添加一系列改进，我们设法提高了以前尝试的准确性。我们的改进包括使用小卷积滤波器来预测对象类别和偏移量在边界框位置，对于不同的宽高比检测使用单独的预测变量（过滤器），并将这些过滤器应用于来自网络的后期阶段的多个特征图，以便在多个尺度执行检测。通过这些修改 - 特别是使用多个层在不同尺度的预测 - 我们可以使用相对较低的分辨率输入实现高精度，进一步提高检测速度。虽然这些贡献可能看起来很小，但是我们注意到，所提供的系统可以将PASCAL VOC的实时检测准确度提高，从YOLO的63.4％mAP到SSD的74.3％的mAP。检测精度与非常高的残差网络有非常高的相似度[3]。此外，显着提高高质量检测的速度可以扩大计算机视觉有用的设置范围。

我们的贡献如下：

--我们推出了SSD，一种单一检测器，用于多个类别，比先前最先进的单次检测器（YOLO）更快，并且显着更准确， 事实上，和执行显式区域提议和池化（包括更快的R-CNN）的较慢技术的准确性相当。

--SSD的核心是使用应用于特征图的小卷积滤波器来预测固定的一组默认边界框的类别分数和框偏移量。

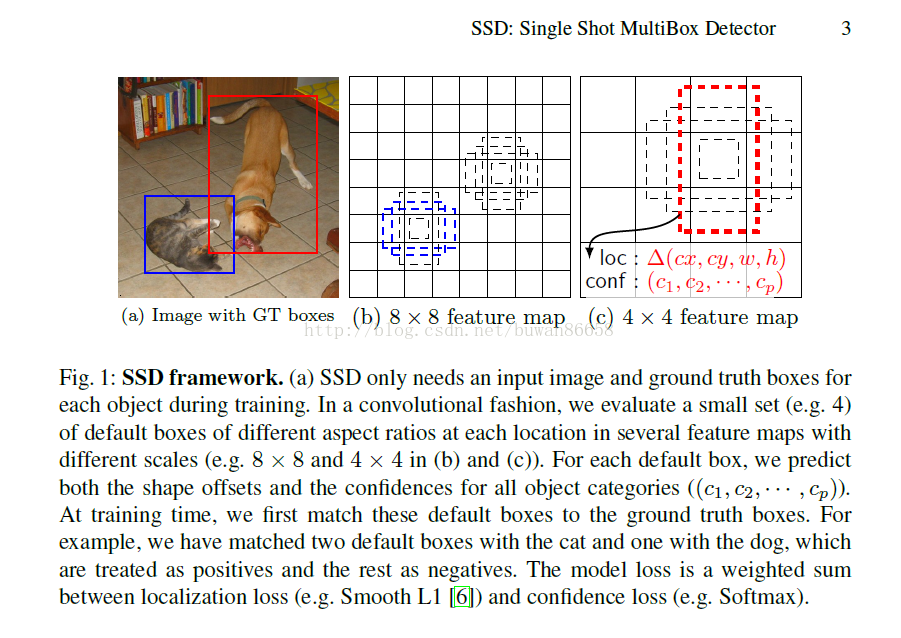
--为了实现高检测精度，我们从不同尺度的特征图中产生不同尺度的预测，并通过纵横比显式分离预测。

--这些设计的特征可实现简单的端到端训练和高精度预测，即使在低分辨率输入图像上也能够达到同样的效果，进一步提高了速度与精度的权衡。

--实验包括对PASCAL VOC，COCO和ILSVRC评估的不同输入大小的模型的实时性和精度分析，并与最近的最新技术方法进行比较。

**2 The Single Shot Detector(SSD)**

本节介绍我们提出的SSD检测框架（第2.1节）和相关的训练方法（第2.2节）。第3节提供数据集特定的模型细节和实验结果。



**2.1 Model**

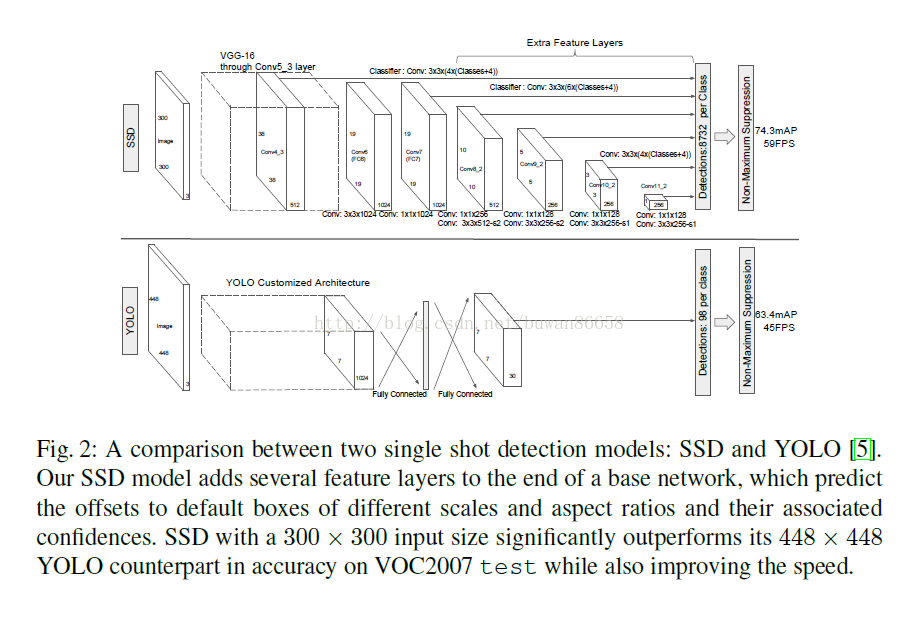
SSD方法基于前馈卷积网络，其产生固定大小的边界框集合和用于对象类的存在的分数  
这些框中的实例，然后是非最大抑制步骤以产生最终检测。 早期网络层基于用于高质量图像分类的标准架构（在任何分类层之前截断），我们称之为基础网络[2]。然后，我们将辅助结构添加到网络中，以产生具有以下主要特征的检测：

**Multi-scale feature maps for detection:**

我们将卷积特征图层添加到截断的基本网络的末尾。这些层逐渐减小，并允许在多个尺度上进行检测的预测。用于预测检测的卷积模型对于在单个尺度特征图上操作的每个特征层（参见Overfeat [4]和YOLO [5]）是不同的。

**Convolutional pridictors for detection:**

每个添加的特征层（或可选地，来自基本网络的现有特征层）可以使用一组卷积滤波器产生一组固定的检测预测。这些在图2中的SSD网络架构的顶部指示。对于具有p个通道的尺寸为m×n的特征层，用于预测潜在检测参数的基本元素是产生类别的分数或相对于默认框的形状偏移的3×3×p小内核坐标。 在应用内核的每个m×n个位置，它产生一个输出值。相对于每个特征图位置（在这一步YOLO[5]使用中间全连接层而不是该步骤的卷积滤波器），相对于默认框位置测量边界框偏移输出值。



**Default boxes and aspect ratios:**

我们将一组默认边界框与每个特征图单元相关联，用于网络顶部的多个特征图。默认框以卷积方式平铺功能图，使每个框相对于其相应单元格的位置是固定的。在每个特征图单元格中，我们预测相对于单元格中默认框形状的偏移量，以及指示每个框中存在类实例的每类分数。具体来说，对于给定位置的k中的每个框，我们计算c类分数和相对于原始默认框形状的4个偏移量。这导致在特征图中的每个位置周围应用的总共（c + 4）k个滤波器，对于m×n特征图产生（c + 4）kmn的输出。有关默认框的说明，请参见图1.我们的默认框类似于Faster R-CNN [2]中使用的主框，但我们将它们应用于不同分辨率的几个特征图。允许在几个特征图中使用不同的默认框形，让我们有效地离散的输出可能的框形状的空间。

**2.2 Training**

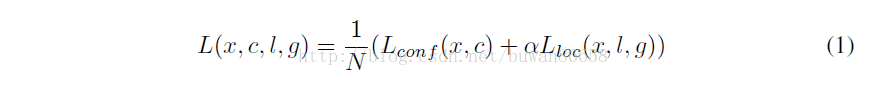
训练SSD和训练使用区域提案的典型检测器之间的关键区别在于，必须将固有的ground-truth分配给固定的检测器输出组中的特定输出。对于YOLO [5]和区域建议的faster R-CNN [2]和MultiBox [7]，也需要这种方式。一旦确定了此分配，则将丢失函数和反向传播应用于端到端。训练还包括选择一组默认的检测框和尺度，以及hard negative mining和数据增广策略。

**匹配策略：**

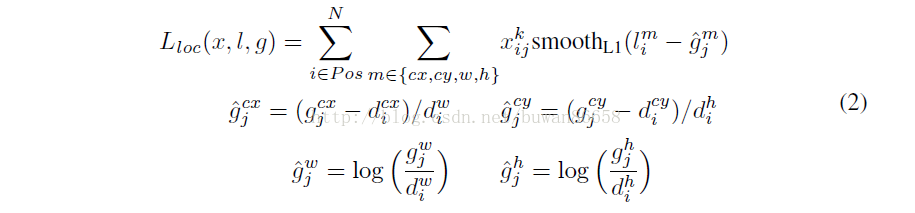
在训练期间，我们需要确定哪些默认框对应ground truth检测，并相应地训练网络。对于每个ground truth框，我们从默认框中选择不同的位置，宽高比和比例。我们首先将每个ground truth框与最佳jaccard重叠的默认框匹配（如MultiBox [7]）。与MultiBox不同，我们将默认框与任何ground truth相匹配，其中jaccard重叠高于阈值（0.5）。这简化了学习问题，允许网络预测多个重叠默认框的高分数，而不是要求它仅选择最大重叠的一个。

**训练目标：**

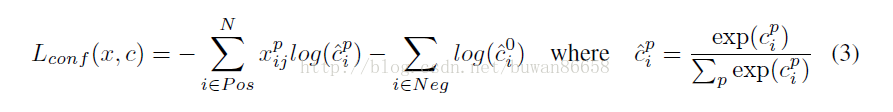
SSD训练目标是从MultiBox目标[7,8]得出的，但扩展到处理多个对象类别。让http://img.blog.csdn.net/20170925142918369?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center成为将第i个默认框与类别p的第j个地面真值框进行匹配的指标。在上述匹配策略中，我们有http://img.blog.csdn.net/20170925143005951?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center.总体目标损失函数是局部性损失（loc）和置信度损失（conf）的加权和：



其中N是匹配的默认框的数量。如果N = 0，我们将损失设置为0.局部损失是预测边框（l）和ground truth边框（g）参数之间的平滑L1损失函数[6]。类似于更快的R-CNN [2]，我们回归到默认边界框（d）的中心（cx; cy）和其宽度（w）和高度（h）的偏移量。



置信度损失是多类别信心的softmax损失（c）。



并且通过交叉验证将权重项α设置为1。

**选择默认框的缩放比例和宽高比：**

为了处理不同的对象尺度，一些方法[4,9]建议处理不同大小的图像，然后结合结果。然而，通过利用单个网络中的几个不同层的特征图进行预测，我们可以模拟相同的效果，同时在所有对象尺度上共享参数。以前的作品[10,11]已经表明，使用较低层的特征图可以提高语义分割质量，因为较低的层捕获输入对象的更精细细节。类似地，[12]表明，添加从特征图集合的全局上下文可以帮助平滑分割结果。通过这些方法，我们使用较低和较高的特征图进行检测。图1显示了在框架中使用的两个示例性特征图（8×8和4×4）。在实践中，我们可以使用更多的小的计算开销。

已知网络中不同层次的特征图具有不同的（经验的）感受野[13]。幸运的是，在SSD框架中，默认框不需要与每个层的实际感受野相对应。我们设计了平铺的默认框，以便特定的特征学习对特定的物体尺度做出反应。假设我们要使用m个特征图进行预测。每个特征图的默认框的比例计算如下：

http://img.blog.csdn.net/20170926082556811?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

其中Smin为0.2，Smax为0.9，意味着最低层的缩放比为0.2，最高层的缩放比为0.9，其间的所有层都是规则间隔的。我们对默认框强加不同的长宽比的默认框，并将它们表示为http://img.blog.csdn.net/20170926083208919?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Centerhttp://img.blog.csdn.net/20170926083216148?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center。我们可以计算每个默认框的宽度http://img.blog.csdn.net/20170926083310510?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center和高度http://img.blog.csdn.net/20170926083319888?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center。对于长宽比为1，我们还添加一个缩放为http://img.blog.csdn.net/20170926083410644?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center的默认框，每个功能图位置有6个默认框。我们将每个默认框的中心设置为（http://img.blog.csdn.net/20170926083529450?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center），其中http://img.blog.csdn.net/20170926083701553?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center是第k个方形特征图的大小http://img.blog.csdn.net/20170926083735835?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center。实际上，还可以设计一个默认框的分布，以适应一个特定的数据集。如何设计最佳平铺是一个悬而未决的问题。

通过对许多特征图的所有位置的不同尺度和宽高比的所有默认框组合预测，我们有一组多样的预测，涵盖各种输入对象的大小和形状。例如，在图1中，狗与4×4特征图中的默认框匹配，但与8×8特征图中的任何默认框匹配。这是因为这些框具有不同的尺度，并且不匹配狗的边框，因此在训练期间被认为是负样本。

**Hard negative mining（难例最小化）：**

在匹配步骤之后，大多数默认框是负样本，特别是当可能的默认框的数量很大时。这导致了正面和负面训练实例之间的很大的不平衡。我们使用每个默认框的最高置信度损失来对它们进行排序，并选择最上面的一个，以使负样本和正样本之间的比例最多为3：1，而不是使用所有的负样本。我们发现这样可以带来更快的优化和更稳定的训练。

**数据增广：**

为了使模型对各种输入对象大小和形状更加鲁棒，每个训练图像都可以通过以下选项之一随机抽样：

---使用整个原始输入图像。

---对块进行采样，以便与对象重叠的最小jaccard为0.1,0.3,0.5,0.7或0.9。

---随机抽样

每个采样块的大小是原始图像尺寸的[0.1,1]，纵横比在1/2和2之间。如果它的中心在采样块中，我们保留ground truth的重叠部分。  
  在上述采样步骤之后，除了应用与[14]中描述的类似的一些光度度失真之外，将每个采样的分块重新调整为固定大小并且以0.5的概率水平翻转。

**3 实验结果**

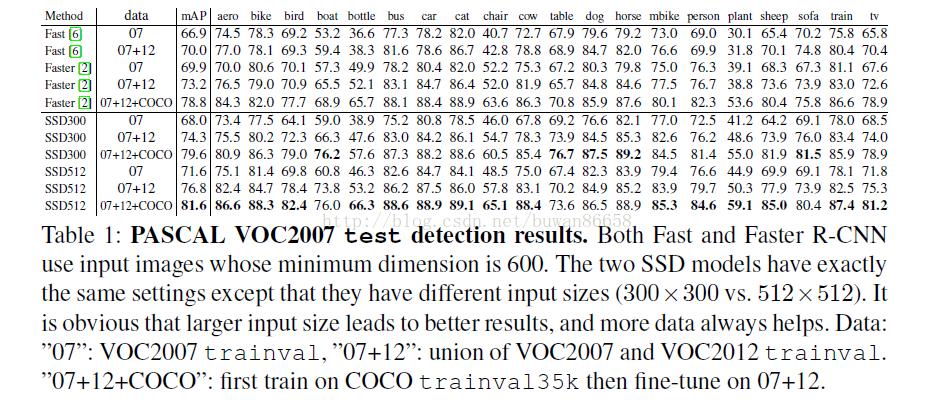
基础网络我们的实验都是基于VGG16 [15]，它是在ILSVRC CLS-LOC数据集上预先训练的[16]。类似于DeepLab-LargeFOV [17]，我们将fc6和fc7转换为卷积层，从fc6和fc7的子样本参数，将pool5从2×2 - s2更改为3×3 - s1，并使用`a trous算法[18]填补“洞”。我们删除所有的dropout层和fc8层。我们使用初始学习率0.001，0.9动量，0.0005重量衰减和批量大小32的SGD微调所得模型。每个数据集的学习速率衰减策略略有不同，稍后将对其进行详细描述。完整的培训和测试代码建立在Caffe [19]上，是开源的：https：//github.com/weiliu89/caffe/tree/ssd。

**3.1 Pascal voc2007**

在这个数据集上，我们比较了VOC2007测试（4952图像）上的Fast R-CNN [6]和Faster R-CNN [2]。所有方法在相同的预训练VGG16网络上进行微调，图2显示了SSD300模型的架构细节。我们使用conv4\_3，conv7（fc7），conv8\_2，conv9\_2，conv10\_2和conv11\_2来预测位置和置信度。我们在conv4上设置了缩放为0.1的默认框。我们用xavier“方法[20]初始化所有新添加的卷积层。对于conv4\_3~~3~~，conv10\_2和conv11\_2，我们仅在每个特征图位置关联4个默认框 - 省略1/3和3的宽高比。对于所有其他图层，我们按照Sec. 2.2。由于如[12]所述，conv4\_3与其他层相比具有不同的特征尺度，我们使用[12]中引入的L2归一化技术将特征图中每个位置的特征规范缩放到20，并在反向传播中使用当前缩放比。我们使用0.001学习率40k次迭代，然后用0.0001和0.00001的学习率继续训练10k次迭代。在VOC2007训练中，表1显示，我们的低分辨率SSD300模型已经比Fast R-CNN更准确。当我们在更大的512×512输入图像上训练SSD时，它甚至更准确，超越了更快的R-CNN 1.7％的mAP。如果我们用更多（即07 + 12）数据来训练SSD，那么我们看到SSD300已经比Faster R-CNN好1.1％，SSD512好3.6％。如果我们采用Sec.3.4中描述的COCO训练35k模型，并使用SSD512对07 + 12数据集进行微调，我们取得最佳结果：81.6％的mAP。

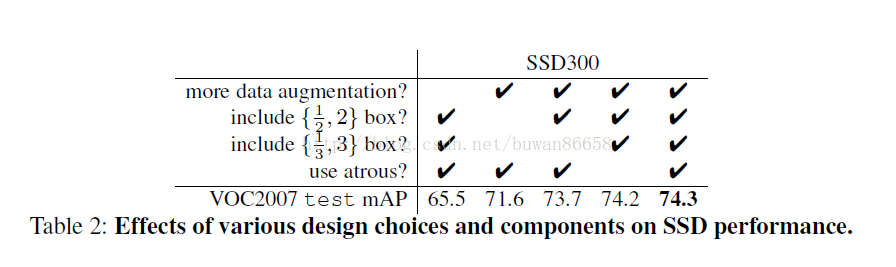
为了更好地了解我们两个SSD模型的性能，我们采用了[21]的检测分析工具。图3显示，SSD可以检测高质量（大白色区域）的各种对象类别。大多数的自信检测是正确的。召回率在85-90％左右，高于“弱”（0.1个 jaccard重叠）标准。与R-CNN [22]相比，SSD具有较少的定位错误，表明SSD可以更好地对对象进行定位，因为它直接学习回归对象形状并对对象类别进行分类，而不是使用两个解耦步骤。然而，SSD与类似的对象类别（特别是动物）有更多的混淆，部分原因是我们分享了多个类别的位置。图4显示了SSD对边框大小非常敏感。换句话说，它在较小的物体上比较大的物体性能要差得多。这并不奇怪，因为这些小物体甚至在顶层甚至没有任何信息。增加输入大小（例如从300×300到512×512）可以帮助改进检测小物体，但还有很大的改进空间。从积极的角度来看，我们可以清楚地看到，SSD在大目标上表现非常出色。并且它对于不同的对象宽高比来说是非常强大的，因为我们使用每个特征图位置的各种宽高比的默认框。

http://img.blog.csdn.net/20170926103638974?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYnV3YW44NjY1OA==/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center



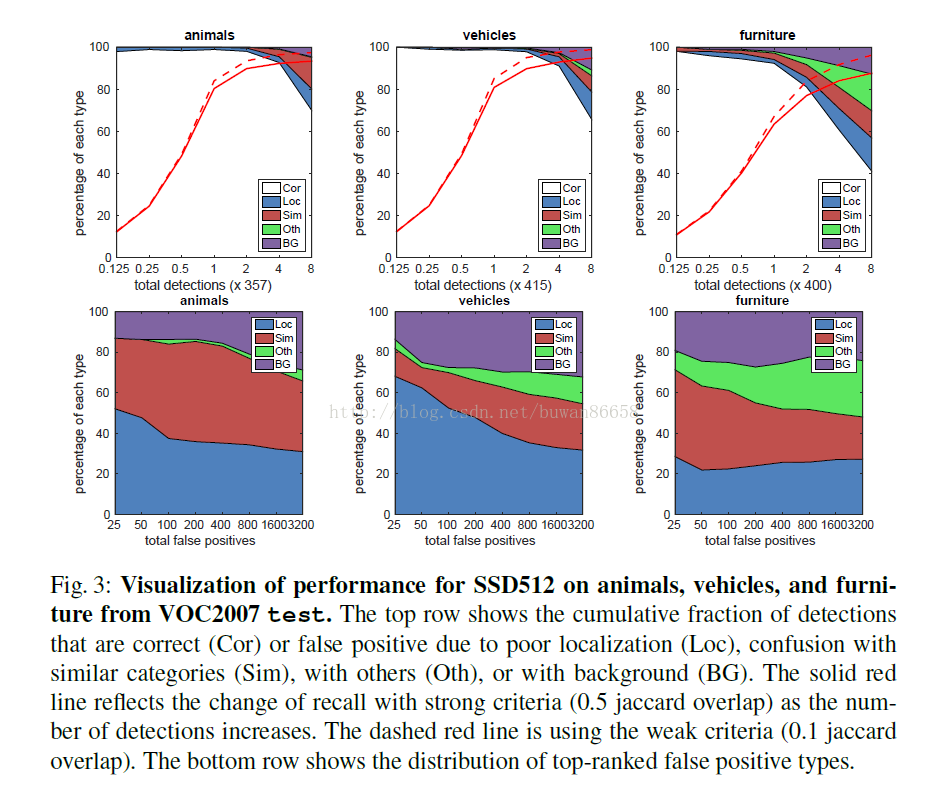
**3.2 模型分析**

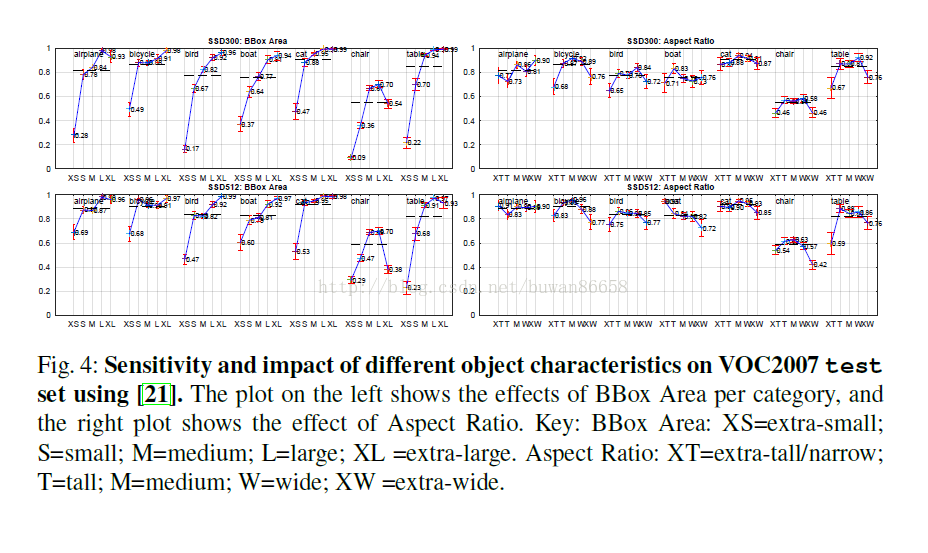
为了更好地了解SSD，我们进行了控制实验来检查每个组件如何影响性能。对于所有实验，我们使用相同的设置和输入大小（300×300），除了对设置或组件的指定更改。



**数据增广至关重要：**

fast 和 faster R-CNN使用原始图像和水平翻转来进行训练。我们采用更广泛的采样策略，类似于YOLO [5]。表2显示，我们可以通过采样策略提高8.8％的mAP。我们不知道我们的采样策略将如何影响快速和快速的R-CNN，但是它们可能会受益更少，因为通过设计强连接的目标转换实现的分类方法使用了特征池化。



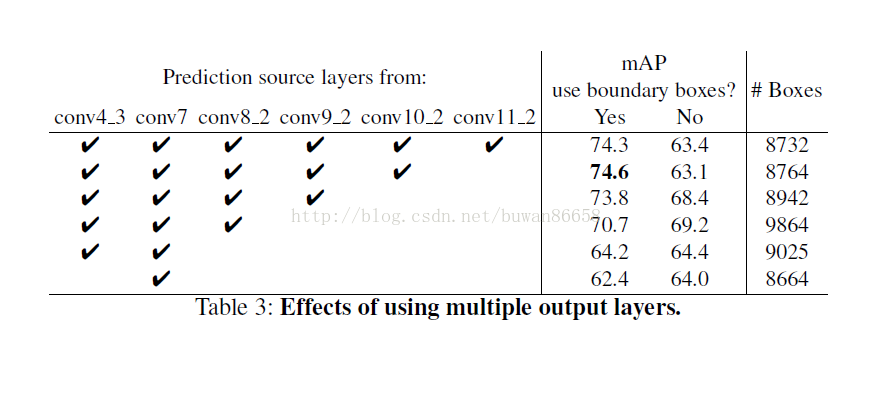


**更多的默认框形状更好：**

如第2.2节所述，默认情况下，每个位置使用6个默认框。如果我们删除1/3和3宽高比的边框，性能下降0.6％。通过进一步去除1/2和2长宽比的边框，性能下降了2.1％。使用各种默认框形似乎使预测框网络的任务更容易。

**Atrous is faster：**

如第3节所述，我们使用了删减的VGG16的atrous版本，遵循DeepLab-LargeFOV [17]。如果我们使用完整的VGG16，将pool5设置为2×2-s2，而不是fc6和fc7的子采样参数，并加上conv5\_3进行预测，结果大致相同，而速度却慢了20％。

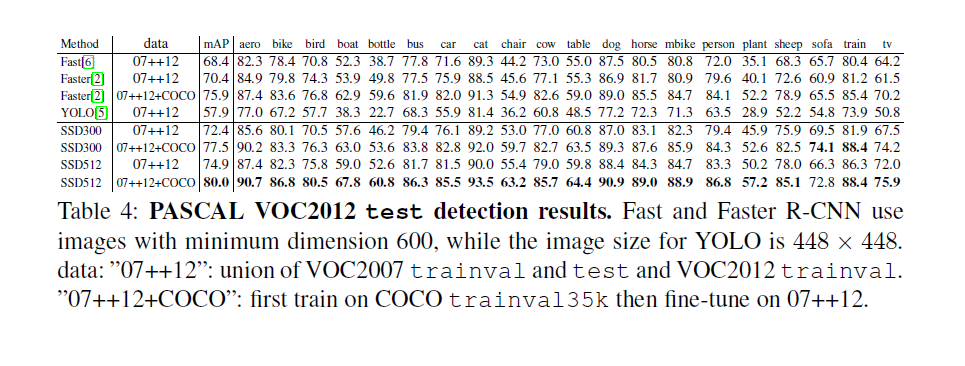


**不同分辨率的多个输出层更好：**

SSD的主要贡献是在不同的输出层上使用不同尺度的默认框。为了衡量所获得的优势，我们逐步删除层并比较结果。为了公平比较，每次我们删除一个层时，我们调整默认的边框，以保持与原始框类似的框总数（8732）。这是通过在其余层上堆叠更多尺寸的盒子并根据需要调整边框尺寸来完成的。我们不会对每个设置进行全面优化。表3显示了较少层的精度下降，单调从74.3下降到62.4。当我们在一个图层上堆叠多尺度的边框时，许多图像在图像边界上，需要仔细处理。我们尝试了在faster R-CNN [2]中使用此策略，忽略边界上的框。我们观察到一些有趣的趋势。例如，如果我们使用非常粗糙的特征图（例如conv11\_2（1×1）或conv10\_2（3×3）），则会大大降低性能。原因可能是修剪后网络中没有足够大的边框覆盖大的物体。当我们的maps主要使用更精细的分辨率时，性能开始再次上升，因为即使在修剪完毕后，仍然有足够数量的大盒子。如果我们只使用conv7进行预测，那么性能是最差的，这就不同层上使用不同尺度边框的关键。此外，由于我们的预测不依赖于ROI池，如[6]中，我们在低分辨率特征图中没有崩溃的问题[23]。SSD架构结合了各种分辨率的功能图的预测，在使用较低分辨率的输入图像的同时，实现了与faster R-CNN的可比的准确度。

**3.3 Padcal voc2012**

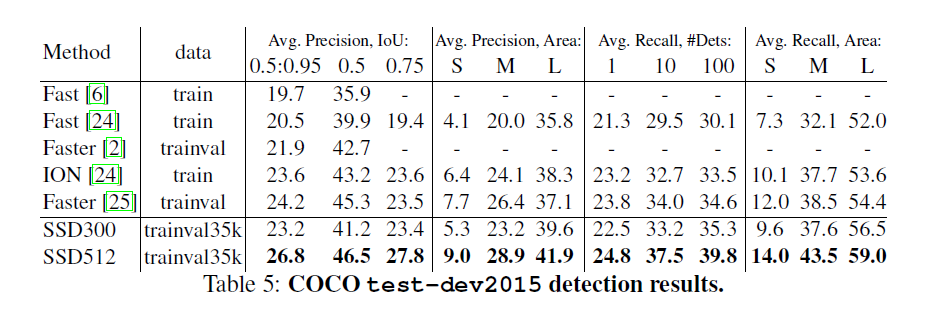
我们使用与上述VOC2007实验基本相同的设置，除了我们使用VOC2012 trainval和VOC2007 trainval和测试（21503images）进行训练，并对VOC2012测试（10991图像）进行测试。我们以0.001学习速率训练模型，进行60k次迭代，然后用0.0001进行20k次迭代。表4显示了我们的SSD300和SSD512 [4]模型试验的结果。我们看到与VOC2007测试相同的性能趋势。我们的SSD300相教于fast/faster的RCNN提高了精度。通过将训练和测试的图片尺寸增加到512×512，我们的精确度比faster R-CNN高4.5％。与YOLO相比，SSD显着更准确，这可能是由于使用来自多个特征图的卷积默认框和在训练期间的所使用的匹配策略。对COCO数据进行fine-tune训练时，我们的SSD512实现了80.0％的mAP，比faster R-CNN高4.1％。



**3.4 COCO**

为了进一步验证SSD框架，我们在COCO数据集上培训了我们的SSD300和SSD512架构。由于COCO中的对象比PASCAL VOC更小，所以我们对所有层使用较小的默认框。我们遵循Sec.2.2中提到的策略，但现在我们最小的默认框的缩放比例为0.15而不是0.2，而conv4\_3的默认框的大小为0.07（对于300×300图像，为21像素）[5]。

我们使用35k [24]的数据进行训练。我们首先用0.001学习速率训练模型，进行160k次迭代，然后继续训练40k次迭代，学习率0.0001和40k次迭代，学习率0.00001。表5显示了test-dev2015的结果。与PASCAL VOC数据集观察到的相似，SSD300在mAP@0.5和mAP @ [0.5：0.95]中均优于Fast R-CNN。 SSD300具有类似的mAP@0.75作为ION [24]和更快的R-CNN [25]，但在mAP@0.5中更差。通过将图像大小增加到512？512，我们的SSD512在两个标准中都优于更快的R-CNN [25]。有趣的是，我们观察到，SSD512在mAP@0.75中为5.3％，但在mAP@0.5中仅为1.2％。我们还观察到，对于大型物体，AP具有更好的AP（4.8％）和AR（4.6％），但小​​物体的AP（1.3％）和AR（2.0％）的改进相对较少。与ION相比，大小物体的AR改善更为相似（5.4％vs. 3.9％）。我们推测，更快的R-CNN在具有SSD的较小对象上更具竞争力，因为它在RPN部分和fast R-CNN部分中执行两个框细化步骤。在图5中，我们使用SSD512模型显示了COCO测试开发的一些检测示例。



**3.5 Preliminary ILSVRC results**

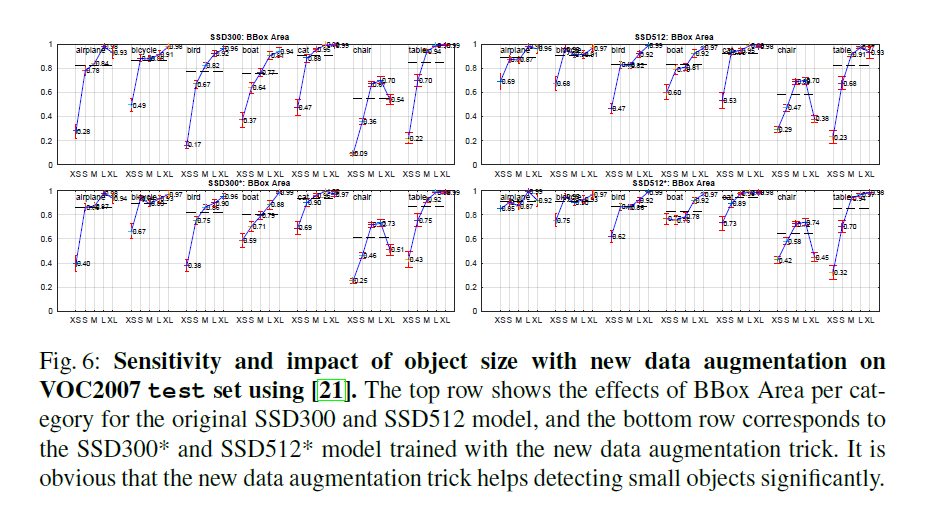
我们使用与COCO相同的网络架构并应用于ILSVRC DET数据集[16]。我们使用ILSVRC2014 DET训练数据来训练一个SSD300模型，如[22]所示。我们首先用0.01学习速率训练模型，进行320k次迭代，然后继续训练80k次迭代，0.001和40k次迭代，0.0001。我们可以在val2集上达到43.4 mAP [22]。同样，它验证了SSD是高质量实时检测的一般框架。

**3.6数据增广对小目标的准确率：**

如果在faster R-CNN中没有后续功能重采样步骤，小目标的分类任务对于SSD来说比较困难，如我们的分析（见图4）所示。 2.2节设计的数据增广策略有助于提高性能，特别是在诸如PASCAL VOC等小型数据集上。随机裁剪操作可以被认为是“放大”操作，可以产生较多的训练样例。通过“缩小”操作，可以获得更多的小的训练样本，我们首先在我们进行任意随机裁剪操作之前随机将图像放置在原始图像尺寸的16倍的画面上并用均值填充。因为我们有更多的训练图像通过引入这个新的“扩展”数据增加技巧，我们必须将训练迭代加倍。我们已经看到跨多个数据集的2％-3％mAP的持续增加，如表6所示。具体来说，图6显示了新的增加技巧显着提高了小对象的性能。该结果强调了数据增加策略对于最终模型精度的重要性。

改进SSD的另一种方法是设计更好的平铺默认框，使其位置和尺度与特征图上每个位置的接受区域更好地对齐。我们把这个工作留在未来。





**3.7 推理时间**

考虑到采用我们方法产生了大量的边框，在推理过程中有必要高效地执行非最大抑制（nms）。通过使用0.01的置信阈值，我们可以过滤掉大多数框。然后，应用每个类别为0.45的jaccard重叠的nms，并保持每个图像的前200个检测框。此步骤对于SSD300和20 VOC类来说，每个图像的成本约为1.7毫秒，这与所有新添加的图层的总时间（2.4毫秒）相当。我们使用Titan X和cuDNN v4与Intel Xeon E5-2667v3@3.20GHz测试batch 为8的的速度作为衡量的标准。

表7显示了SSD，faster R-CNN [2]和YOLO [5]之间的比较。我们的SSD300和SSD512方法在速度和精度方面均胜过faster R-CNN。尽管快速YOLO [5]可以在155 FPS下运行，但是精度降低了近22％的mAP。据我们所知，SSD300是第一个实时方法实现70％以上mAP的检测。请注意，大约80％的前向时间花在基础网络上（在我们的例子中是VGG16）。因此，使用更快的基础网络甚至可以进一步提高速度，这也可能使SSD512成为实时的。

**4 相关工作**

有两种已建立的图像对象检测方法，一种基于滑动窗口，另一种基于区域提案分类。在卷积神经网络出现之前，这两种方法 - 可变形部分模型（DPM）[26]和选择性搜索[1]的最新技术具有可比性能。然而，随着R-CNN [22]的显着改进，结合了选择性搜索区域提案和基于卷积网络的后分类，区域提案对象检测方法变得普遍。

原来的R-CNN方法已经以各种方式得到改进。第一套方法提高了后分类的质量和速度，因为它需要对成千上万的切割图像进行分类，费时费力。SPPnet [9]显着加快了原始的R-CNN方式。它引入了对区域尺寸和缩放更加鲁棒的空间金字塔池化层，并允许分类层使用几个不同图像分辨率下生成的特征图计算的特征。fast R-CNN [6]扩展了SPPnet，它通过获得置信度和边界框回归的最小loss能够在所有层进行端到端的fine-tune，这是MultiBox [7]首先在学习对象中引入的。

第二中方法使用深层神经网络提高了提案生成的质量。在最近的作品如MultiBox [7,8]中，基于低级图像特征的选择性搜索区域提案被直接从单独的深层神经网络生成的提案所替代。这进一步提高了检测精度，但导致了一些复杂的设置，需要训练两个神经网络与它们之间的依赖关系。fasterR-CNN [2]代替了从区域提案网络（RPN）中学习的选择性搜索提案，并介绍了一种通过在这两个网络的finetuning共享卷积层和预测层之间交替来将RPN与Fast R-CNN进行集成的方法。这种方式区域提议用于池化mid-level 特征，最终的分类步骤更容易。我们的SSD非常类似于faster R-CNN中的区域提案网络（RPN），因为我们还使用一组固定的（默认）框进行预测，类似于RPN中的锚点框（anchor boxes）。但是，不是使用这些来池化特征并评估另一个分类器，我们同时为每个框中的每个对象类别生成一个分数。因此，我们的方法避免了将RPN与faster R-CNN合并的复杂性，并且更容易训练，更快速，直接地集成到其他任务中。

与我们的方法直接相关的另一套方法，完全跳过提案步骤，直接预测多个类别的边界框和信任。OverFeat [4]是滑动窗口方法的深度版本，在知道基础对象类别的置信度之后，从最顶层特征图的每个位置直接预测边界框。YOLO [5]使用整个最上面的特征图来预测多个类别和边界框（这些类别共享）的置信度。我们的SSD方法属于此类别，因为我们没有提案步骤，但使用默认框。然而，我们的方法比现有的方法更灵活，因为我们可以在不同尺度的多个特征图的每个特征位置使用不同宽高比的默认框。如果我们只从最上面的特征图中每个位置使用一个默认框，我们的SSD将具有与OverFeat相似的架构[4]; 如果我们使用整个最上面的特征图，并添加一个完全连接的预测层，而不是我们的卷积预测因子，并且不明确考虑多个纵横比，我们可以大致重现YOLO [5]。

**5总结**

本文介绍了SSD，一种用于多种类型的快速单目标检测器。我们的模型的一个关键特征是使用附加在网络顶部的多个特征图的多尺度卷积边界框输出。该表示允许我们有效地模拟可能的边框空间。我们通过实验验证了适当的训练策略，大量精心选择的默认边界框可以提高性能。与现有方法相比，我们建立至少一个数量级的边框预测采样位置，尺度和纵横比的SSD模型[5,7]。我们证明，鉴于相同的VGG-16基础架构，SSD在精度和速度方面与其最先进的目标检测器相比都有优势。我们的SSD512型号在PASCAL VOC和COCO的精度方面显着优于faster R-CNN [2]，而速度提高了3倍。我们的实时SSD300可以达到59 FPS，这比现在的实时YOLO [5]替代方案更快，同时产生显着优异的检测精度。

**参考文献：**

1. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. IJCV (2013)  
2. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS. (2015)  
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR.(2016)  
4. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. In: ICLR. (2014)

5. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR. (2016)  
6. Girshick, R.: Fast R-CNN. In: ICCV. (2015)  
7. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: CVPR. (2014)  
8. Szegedy, C., Reed, S., Erhan, D., Anguelov, D.: Scalable, high-quality object detection.arXiv preprint arXiv:1412.1441 v3 (2015)  
9. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ECCV. (2014)  
10. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation.In: CVPR. (2015)  
11. Hariharan, B., Arbel´aez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: CVPR. (2015)  
12. Liu,W., Rabinovich, A., Berg, A.C.: ParseNet: Looking wider to see better. In: ILCR. (2016)  
13. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Object detectors emerge in deep scene cnns. In: ICLR. (2015)  
14. Howard, A.G.: Some improvements on deep convolutional neural network based image classification. arXiv preprint arXiv:1312.5402 (2013)  
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition.In: NIPS. (2015)  
16. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A.,Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. IJCV (2015)  
17. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR. (2015)  
18. Holschneider, M., Kronland-Martinet, R., Morlet, J., Tchamitchian, P.: A real-time algorithm for signal analysis with the help of the wavelet transform. In: Wavelets. Springer (1990)286–297  
19. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S.,Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: MM. (2014)  
20. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS. (2010)  
21. Hoiem, D., Chodpathumwan, Y., Dai, Q.: Diagnosing error in object detectors. In: ECCV 2012. (2012)  
22. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014)  
23. Zhang, L., Lin, L., Liang, X., He, K.: Is faster r-cnn doing well for pedestrian detection. In:ECCV. (2016)  
24. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.: Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: CVPR. (2016)  
25. COCO: Common Objects in Context. http://mscoco.org/dataset/ #detections-leaderboard (2016) [Online; accessed 25-July-2016].  
26. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR. (2008)