YOLOv1原文翻译 You Only Look Once Unified, Real-Time Object Detection

Abstract摘要

我提出了YOLO：一种新的物体检测方法。YOLO之前的物体检测方法主要是通过region proposal产生大量的可能包含待检测物体的 potential bounding box，再用分类器去判断每个 bounding box里是否包含有物体，以及物体所属类别的 probability或者 confidence，如R-CNN,Fast-R-CNN,Faster-R-CNN等。YOLO不同于这些物体检测方法，它**将物体检测任务当做一个regression（回归）问题来处理**。YOLO从输入的图像，仅使用一个神经网络，直接从一整张图像来预测出bounding box 的坐标、box中包含物体的置信度和物体的probabilities（概率）。因为YOLO的物体检测流程是在一个神经网络里完成的，所以可以end to end（端对端：指的是输入原始数据，输出的是最后结果，应用在特征学习融入算法，无需单独处理）来优化物体检测性能。

YOLO检测物体的速度很快，标准版本的YOLO在Titan X 的 GPU 上能达到45 FPS。网络较小的版本Fast YOLO在保持mAP是之前的其他实时物体检测器的两倍的同时，检测速度可以达到155 FPS。相较于其他的state-of-the-art 物体检测系统，YOLO在物体定位时更容易出错，但是在背景上预测出不存在的物体（false positives）的情况会少一些。而且，YOLO比DPM、R-CNN等物体检测系统能够学到更加抽象的物体的特征，这使得YOLO可以从真实图像领域迁移到其他领域，如艺术。

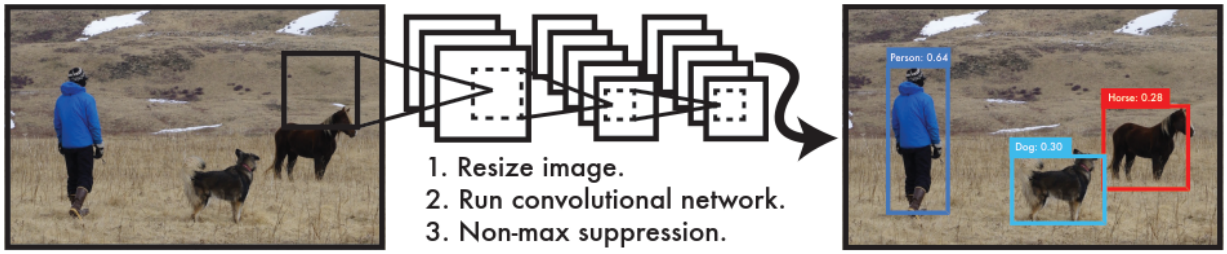
1.Introduction（介绍）

人们瞥视图像，立即知道图像中的物体，它们在哪里以及它们如何相互作用。 人类的视觉系统是快速和准确的，使我们能够执行复杂的任务，例如驾驶时几乎没有意识的想法。 快速，准确的目标检测算法可以让计算机在没有专门传感器的情况下驾驶汽车，使辅助设备能够向人类用户传达实时的场景信息，并释放通用目标响应式机器人系统的潜力。

当前的物体检测系统使用分类器来完成物体检测任务。为了检测一个物体，这些物体检测系统要在一张测试图的不同位置和不同尺寸的bounding box上使用该物体的分类器去评估是否有该物体。如DPM系统，要使用一个滑窗（sliding window）在整张图像上均匀滑动，用分类器评估是否有物体。

在DPM之后提出的其他方法，如R-CNN方法使用region proposal来生成整张图像中可能包含待检测物体的potential bounding boxes，然后用分类器来评估这些boxes，接着通过post-processing来改善bounding boxes，消除重复的检测目标，

并基于整个场景中的其他物体重新对boxes进行打分。整个流程执行下来很慢，而且因为这些环节都是分开训练的，检测性能很难进行优化。



> 图1：YOLO检测系统。 用YOLO处理图像简单而直接。
> 我们的系统（1）将输入图像的大小调整为448×448，（2）在图像上运行单个卷积网络，（3）通过模型的置信度对结果检测进行阈值。

本文提出的YOLO（you only look once），将物体检测任务当做回归问题（regression problem）来处理，直接通过整张图片的所有像素得到bounding box的坐标、box中包含物体的置信度和class probabilities。通过YOLO，每张图像只需要输入到神经网络就能得出图像中都有哪些物体和这些物体的位置。

YOLO非常简单：参见图1.单个卷积网络可同时预测多个边界框和这些框的类概率，YOLO训练全图像并直接优化检测性能。 这种统一的模型与传统的物体检测方法相比有许多优点。

YOLO模型相对于之前的物体检测方法有多个优点：
1、**YOLO检测物体非常快。**
因为没有复杂的检测流程，只需要将图像输入到神经网络就可以得到检测结果，YOLO可以非常快的完成物体检测任务。标准版本的YOLO在Titan X 的 GPU 上能达到45 FPS。更快的Fast YOLO检测速度可以达到155 FPS。而且，YOLO的mAP是之前其他实时物体检测系统的两倍以上。
2、**YOLO可以很好的避免背景错误，产生false positives（背景误检率低）。**
不像其他物体检测系统使用了滑窗或region proposal，分类器只能得到图像的局部信息。YOLO在训练和测试时都能够看到一整张图像的信息，因此YOLO在检测物体时能很好的利用上下文信息，从而不容易在背景上预测出错误的物体信息。和Fast-R-CNN相比，YOLO的背景错误不到Fast-R-CNN的一半。
3、**YOLO可以学到物体的泛化特征。**
当YOLO在自然图像上做训练，在艺术作品上做测试时，YOLO表现的性能比DPM、R-CNN等之前的物体检测系统要好很多。因为YOLO可以学习到高度泛化的特征，从而迁移到其他领域。

尽管YOLO有这些优点，它也有一些**缺点**：
1、YOLO的物体检测精度低于其他state-of-the-art的物体检测系统。
2、YOLO容易产生物体的定位错误。
3、YOLO对小物体的检测效果不好（尤其是密集的小物体，因为一个栅格只能预测2个物体）。

2.Unified Detection（单一检测）

我们将目标检测单独集成到单个神经网络中。 我们的网络使用整个图像的特征来预测每个边界框。 它还同时预测所有类的所有边界框。 这意味着我们的网络能够在全球范围内全面了解图像中的全部图像和图像中的所有对象YOLO设计可实现端到端训练和实时速度，同时保持较高的平均精度。

YOLO将输入图像划分为S*S的栅格，每个栅格负责检测中心落在该栅格中的物体。
每一个栅格预测B个bounding boxes，以及这些bounding boxes的confidence scores。这个 confidence scores反映了模型对于这个栅格的预测：该栅格是否含有物体，以及这个box的坐标预测的有多准。公式定义如下：

$$\Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

如果这个栅格中不存在一个object，则confidence score应该为0。否则的话，confidence score则为predicted bounding box与 ground truth box之间的 IOU（intersection over union）。

YOLO对每个bounding box有5个predictions：x, y, w, h和 confidence。
坐标x,y代表了预测的bounding box的中心与栅格边界的相对值。
坐标w,h代表了预测的bounding box的width、height相对于整幅图像width,height的比例。
confidence就是预测的bounding box和ground truth box的IOU值。

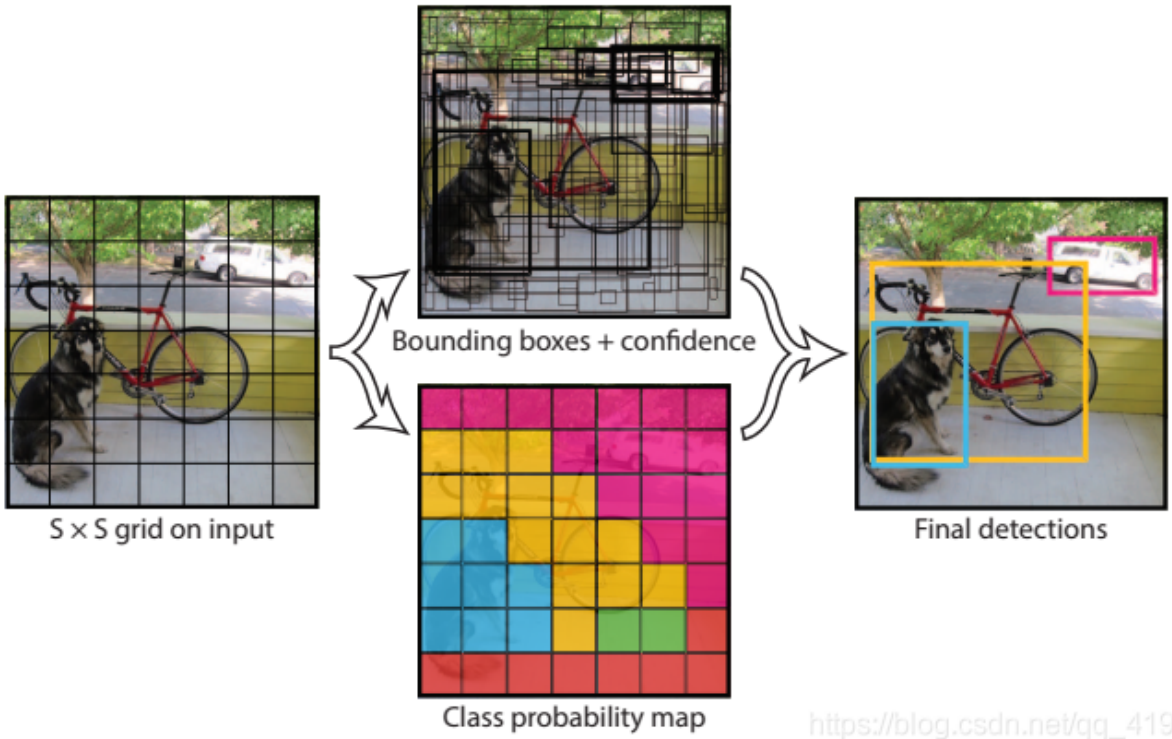
每一个栅格还要预测C个conditional class probability（条件类别概率）：Pr(Classi|Object)。即在一个栅格包含一个Object的前提下，它属于某个类的概率。我们只为每个栅格预测一组（C个）类概率，而不考虑框B的数量。

在测试时，我们将条件类概率和单个框的置信预测相乘，

$$\Pr(\text{Class}_i|\text{Object}) * \Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}} \quad (1)$$

这给了我们每个框（box）的特定类别的信心分数。这些分数编码了类出现在框中的概率以及预测框与对象的匹配程度。

整个*yolo*算法的流程如图*2.*



> 图2： 我们的YOLO系统将检测模型化为回归问题。 它将图像划分为S×S网格，并且每个网格单元预测B个边界框，对这些框的置信度以及C类概率。 这些预测值被编码为S×S×（B * 5 + C）张量。

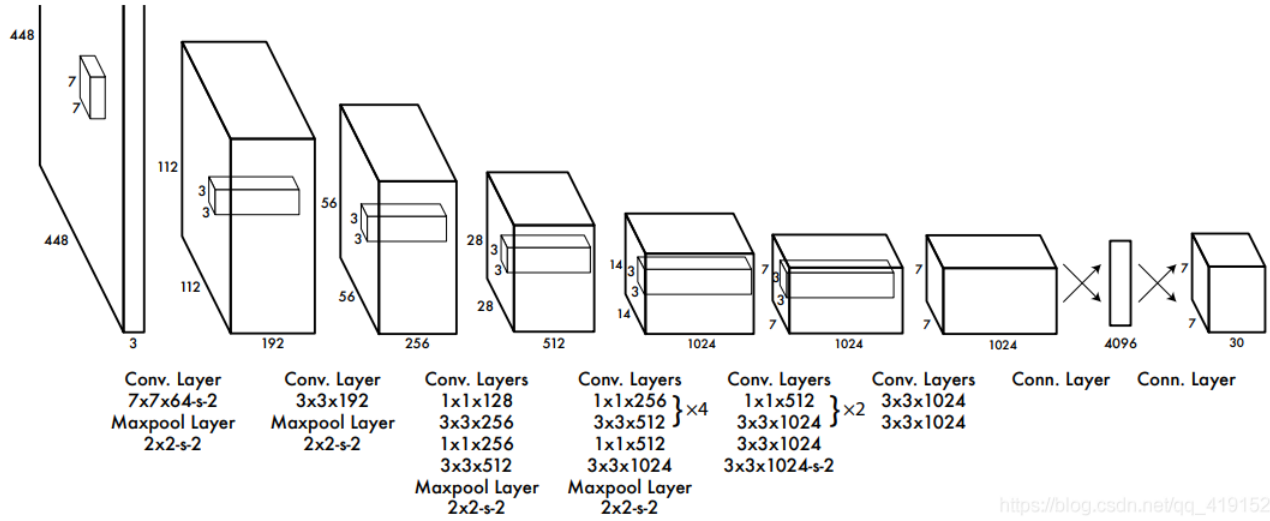
为了评估PASCAL VOC上的YOLO，我们使用S = 7，B = 2。PASCAL VOC有20个标记类，因此C = 20。我们的最终预测是7×7×30张量。

2.1 Network Design（网络设计）

我们将此模型作为卷积神经网络实施并在PASCAL VOC检测数据集上进行评估。网络的初始卷积层从图像中提取特征，而全连接的层预测输出概率和坐标。

YOLO网络借鉴了GoogLeNet的图片分类网络结构。网络有24个卷积层和2个完全连接层。不同的是，YOLO未使用GoogLeNet所使用的inception module，而是使用1x1卷积层（此处1x1卷积层的存在是为了跨通道信息整合）+3x3卷积层简单替

代，类似Lin等人。完整的网络结构如图3所示。



> 图3：**网络结构**。我们的检测网络有24个卷积层和2个完全连接层。交替的1×1卷积层减少了前几层的特征空间。在图像网络分类任务中，我们以一半的分辨率（224×224输入图像）对卷积层进行预处理，然后将检测分辨率提高一倍。

最终的输出结果是一个7730的张量。

2.2 Training（训练）

首先利用ImageNet 1000-class的分类任务数据集预训练卷积层。使用上述网络中的前20 个卷积层，加上一个 average-pooling layer（平均池化层），最后加一个全连接层，作为 Pretrain（预训练） 的网络。训练大约一周的时间，使得在ImageNet 2012的验证数据集Top-5的精度达到 88%，这个结果跟Caffe’s Model Zoo 中的 GoogleNet 的效果相当。

然后我们将模型转换为执行检测。Ren等人提出在预训练网络中同时加入卷积层和连接层可以提高网络性能。以它们为榜样，我们加入4个卷积层和2个具有随机初始权重的全连接层。检测往往需要细粒度的视觉信息，因此我们将网络的输入分辨率从224×224提高到448×448。（直译）

> 将Pretrain的结果的前20层卷积层应用到Detection中，并加入剩下的4个卷积层及2个全连接。同时为了获取更精细化的结果，将输入图像的分辨率由
> 224* 224 提升到 448* 448。（意译）（我发现意译的效果要好很多，更通顺也符合国人对文章的理解）

最后一层class probabilities（预测类概率）和bounding box coordinates（边界框坐标）。我们将所有的预测结果都归一化到 0~1。

最后一层使用线性激活函数，其他所有层都使用 Leaky RELU ， 公式如下：

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases} \qquad (2)$$

> Leaky RELU可以解决RELU的梯度消失问题。

我们对模型输出的误差平方和进行了优化。我们使用误差平方和，因为它很容易优化，但它并不完全符合我们的目标：最大化平均精度。它将定位误差与分类误差平均加权，分类误差不一定很理想。还有，在每个图像中，许多网格单元不包含任何对象。这会将这些细胞的“信心”分数推向零，通常会压倒包含对象的单元格的梯度。这可能会导致模型不稳定，导致训练提前发散。

为了解决这个问题，我们增加了边界框坐标预测的损失，并减少了不包含对象的框的置信预测的损失。我们使用两个参数，λcoord和λnoobj来实现这一点。我们设置了λcoord=5和λnoobj=0.5。

误差平方和在大boxes（框）和小boxes中的权重相等。我们的误差度量应该反映出大boxes的小偏差比小boxes里的小偏差更重要。为了部分解决这个问题，我们预测了边界框宽度和高度的平方根，而不是直接预测宽度和高度。

YOLO预测每个网格有多个边界框单元格。在训练时间我们只希望一个边界框预测器负责每个对象。我们指定一个预测器来“负责”预测一个对象，根据这个对象，预测具有最高的当前IOU和ground truth。这将导致边界框预测器之间的特殊化。每个预测器在预测特定大小、纵横比或对象类别方面都会变得更好，从而提高整体回忆能力。

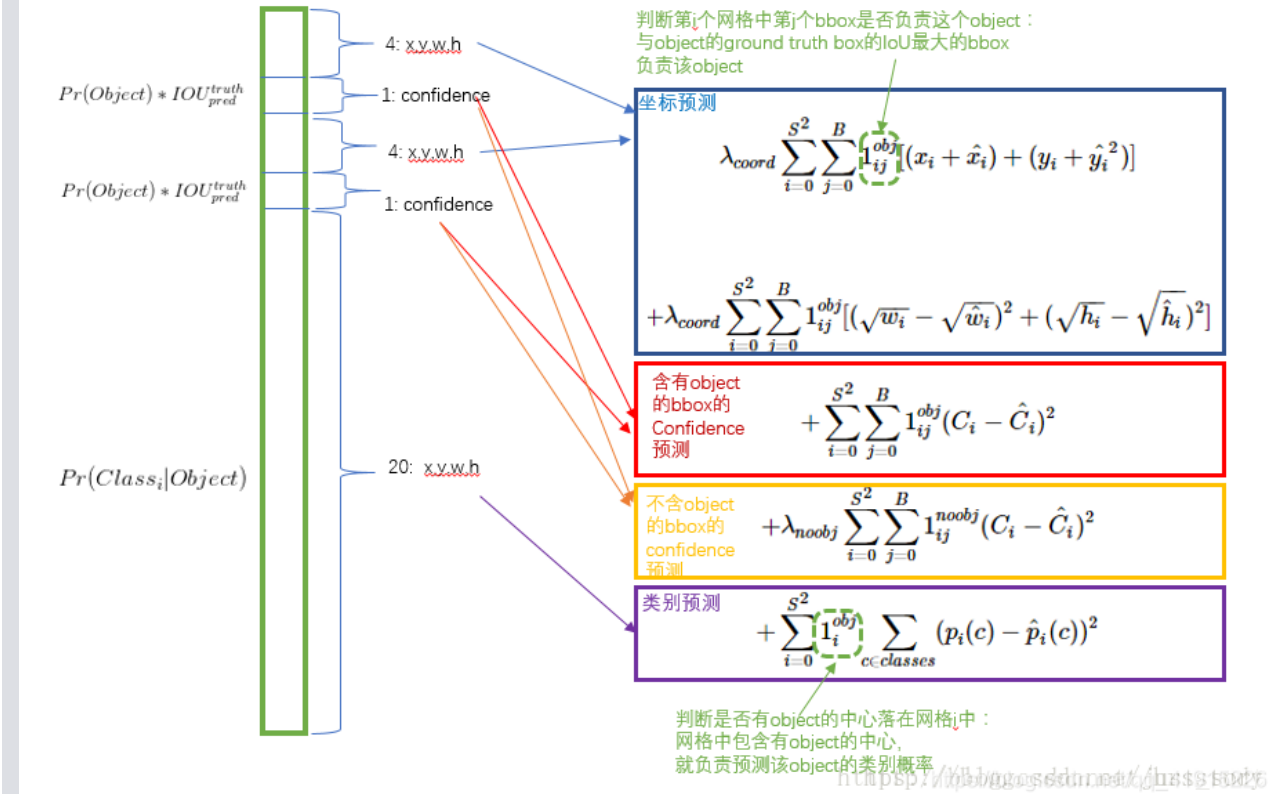
> 在 YOLO中，每个栅格预测多个bounding box，但在网络模型的训练中，希望每一个物体最后由一个bounding box
> predictor来负责预测。因此，当前哪一个predictor预测的bounding box与ground truth box的IOU最大，这个 predictor就负责 predict object。
> 这会使得每个predictor可以专门的负责特定的物体检测。随着训练的进行，每一个predictor对特定的物体尺寸、长宽比的物体的类别的预测会越来越好。

在训练期间，我们优化了以下多部分损失函数：

$$
\begin{aligned}
&\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2 \qquad (3)
\end{aligned}
$$

$\mathbb{1}_{i}^{\text{obj}}$ $\mathbb{1}_{ij}^{\text{obj}}$

式中，$\mathbb{1}_{i}^{\text{obj}}$ 表示对象是否出现在单元格i中，$\mathbb{1}_{ij}^{\text{obj}}$ 表示单元格i中的第j个边界框预测器“负责”该预测。



注意，如果网格单元中存在对象，则loss函数只惩罚分类错误（因此前面讨论的条件类概率）。它也只惩罚边界框坐标误差，如果该预测器是“负责”的地面真相框（即，有最高的IOU任何预测器在该网格单元）。

我们在PASCAL VOC 2007和2012的培训和验证数据集上对网络进行了大约135个阶段的培训。在2012年测试时，我们还包括了用于培训的VOC 2007测试数据。在整个训练过程中，我们使用64个批次，动量为0.9，衰减为0.0005。

我们的学习率计划如下：在第一个阶段，我们将学习率从10^{-3}缓慢提高到10^{-2}。如果我们从一个高学习率开始，我们的模型经常因为不稳定的梯度而发散。我们继续训练，75个阶段10^{-2}，然后30个阶段10^{-3}，最后30个阶段10^{-4}。

为了避免过度拟合，我们使用了dropout(指在深度学习网络的训练过程中，按照一定的概率将一部分神经网络单元暂时从网络中丢弃，相当于从原始的网络中找到一个更瘦的网络)和广泛的数据增加。在第一连接层之后用速率为0.5的dropout层来prevents co-adaptation between layers（防止层之间的协同适应/共同作用）。对于数据增强，我们引入了高达原始图像大小20%的随机缩放和平移。我们还可以在HSV颜色空间中随机调整图像的曝光和饱和度，调整系数高达1.5。

2.3 Inference（推论）

就像在训练中一样，图像的检测只需要一个网络评估。在PASCAL VOC上，网络预测每个图像的98个边界框和每个框的类概率。**YOLO在测试时速度非常快**，因为它只需要一个网络预测，而不像基于分类器的方法。

网格设计在边界框预测中增强了空间多样性。通常很清楚对象属于哪个网格单元，并且网络只为每个对象预测一个框。但是，一些大型对象或靠近多个单元格边界的对象可以被多个单元格很好地定位。非最大抑制可以用来修正这些多重检测。虽然对于R-CNN或DPM的性能来说并不重要，但非最大抑制在mAP中增加了2-3%

2.4 Limitations of YOLO（YOLO的局限性）

YOLO对边界框预测施加了很强的空间约束，因为每个网格单元只能预测两个框，并且只能有一个类。这个空间约束限制了我们的模型可以预测的邻近对象的数量。我们的模型与成群出现的小物体作斗争，例如成群的鸟。

> 每个 grid cell 只预测一个 类别的 Bounding Boxes，而且最后只取置信度最大的那个 Box。这就导致如果多个不同物体(或者同类物体的不同实体)的中心落在同一个网格中，会造成漏检。

由于我们的模型学习从数据中预测边界框，所以它很难推广到新的或不寻常的宽高比或配置中的对象。我们的模型还使用相对粗糙的特征来预测边界框，因为我们的架构有来自输入图像的多个下采样层。

> 预测的 Box 对于尺度的变化比较敏感，在尺度上的泛化能力比较差。

最后，当我们训练一个接近检测性能的损失函数时，我们的损失函数对待小边界框和大边界框中的错误是一样的。大框里的小错误通常是良性的，但是一个小框里的小错误对IOU的影响要大得多，我们的错误的主要来源是不正确的本地化。

> 识别物体位置精准性差。
> 召回率（recall）低

3. Comparison to Other Detection Systems（和其他的目标检测算法的对比）

目标检测是计算机视觉的核心问题。检测通道通常首先从输入图像中提取一组鲁棒特征（Haar、SIFT、HOG、卷积特征）。然后，使用分类器或定位器来识别特征空间中的对象。这些分类器或定位器要么在整个图像上以滑动窗口的方式运行，要

么在图像中的某些区域子集上运行。我们将YOLO检测系统与几个顶级的检测框架进行了比较，突出了关键的相同点和不同点。

可变形部件模型DPM。可变形部件模型（DPM）使用滑动窗口方法进行目标检测。DPM使用一个不相交的管道来提取静态特征、分类区域、预测高分区域的边界框等，我们的系统用一个卷积神经网络来代替所有这些不同的部分。该网络同时执行特征提取、边界框预测、非最大值抑制和上下文推理。网络不是静态特征，而是在线训练特征，并针对检测任务对其进行优化。与DPM相比，我们的统一架构带来了更快、更准确的模型。

R-CNN。R-CNN及其变体使用候选区域（ region proposals）而不是滑动窗口来查找图像中的对象。选择性搜索生成潜在的边界框，卷积网络提取特征，支持向量机对框打分，线性模型调整边界框，非最大抑制消除重复检测。这个复杂通道的每个阶段都必须独立地进行精确的调整，结果系统非常慢，在测试时每张图像需要40秒以上的时间。

YOLO和R-CNN有一些相似之处。每个网格单元提出潜在的边界框，并使用卷积特征对这些边界框进行评分。然而，我们的系统在网格单元建议上设置空间约束，这有助于减少对同一对象的多次检测。我们的系统也提出了更少的边界框，只有98个图像相比，约2000从选择性搜索。最后，我们的系统将这些单独的组件组合成一个单独的、联合优化的模型。

其他快速检测器。Fast和Faster R-CNN关注于通过共享计算和使用神经网络提出区域而不是选择性搜索来加速R-CNN框架。虽然它们比R-CNN提供了速度和准确性的改进，但都还不能达到实时性能。

许多研究工作集中在加速DPM管道。它们加速HOG计算，使用级联，并将计算推送到gpu。然而，只有30Hz DPM能够实时运行。

YOLO没有试图优化大型检测管道的单个组件，而是完全抛出管道，并且设计得很快。

像人脸或人这样的单个类的检测器可以高度优化，因为它们必须处理更少的变化。YOLO是一种通用的探测器，它可以学习同时检测各种对象。

深层多重框(Deep MultiBox)。与R-CNN不同，Szegedy等人训练卷积神经网络来预测感兴趣的区域，而不是使用选择性搜索。MultiBox还可以通过将置信度预测替换为单类预测来执行单目标检测。然而，MultiBox不能进行一般的目标检测，仍然只是一个较大的检测管道中的一部分，需要进一步的图像补丁分类。YOLO和MultiBox都使用
卷积网络预测图像中的边界框，而YOLO是一个完整的检测系统。

OverFeat。Sermanet等人训练卷积神经网络来执行定位，并使该定位器适应执行检测。OverFeat有效地执行滑动窗口检测，但它仍然是一个不相交的系统。OverFeat优化定位，而不是检测表现。比如DPM，定位程序只在进行预测时看到本地信息。OverFeat不能解释全局上下文，因此需要大量的后处理来产生连贯的检测。

多重抓取（MultiGrasp）。我们的工作在设计上与Redmon等人的抓取检测工作相似。我们的边界框预测的网格方法是基于多抓取系统的回归抓取。然而，抓取检测比目标检测简单得多。对于包含一个对象的图像，多重抓取只需要预测一个可抓取区域。它不需要估计物体的大小、位置或边界，也不需要预测其类别，只需要找到一个适合抓取的区域。YOLO预测图像中多个类的多个对象的边界框和类概率。

4. Experiments（实验）

首先，我们比较了YOLO和其他实时检测系统在PASCAL VOC 2007上的性能。为了了解YOLO和R-CNN变体之间的差异，我们探讨了YOLO和Fast R-CNN在VOC 2007上的错误，后者是R-CNN性能最高的版本之一。基于不同的误差分布，我们证明YOLO可以用来重新存储快速的R-CNN检测，减少背景误报带来的误差，从而显著提高性能。我们还展示了VOC 2012的结果，并将mAP与当前最先进的方法进行了比较。最后，我们证明YOLO在两个图形数据集上比其他检测器更好地推广到新域。

4.1. Comparison to Other Real-Time Systems（与其他实时系统的比较）

目标检测的许多研究工作都集中在快速建立标准检测管道上。然而，只有Sadeghi等人实际发明一个实时运行的检测系统（每秒30帧或更好）。我们将YOLO与他们的DPM的GPU实现进行了比较，DPM的运行频率可以是30Hz或100Hz。虽然其他人的努力没有达到实时检测的要求。我们还比较了它们的相对mAP和速度，以检查物体检测系统的准确性和性能之间的权衡。

Fast YOLO是在PASCAL上最快的物体检测方法，而且据我们所知它也是目前最快的物体检测方法。它达到了52.7%的mAP，这比以前的实时检测系统的准确率高出一倍以上。YOLO在保持实时性能的同时将mAP提高到63.4%。

我们也用VGG-16来训练YOLO。这个模型比YOLO准确率更高但是速度降低很多。它与依赖于VGG-16的其他检测系统相比是更有效的，但由于它达不到实时系统速度要求，所以本文的其他部分将重点放在我们的这个更快的模型上。

最快的DPM可以在不牺牲太多mAP的情况下有效加速DPM，但仍然会将实时性能降低2倍。与神经网络方法相比，它还受到DPM检测精度相对较低的限制。

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

> 表1： **PASCAL VOC 2007上的实时系统。**比较快速检测器的性能和速度。Fast YOLO是PASCAL VOC 上速度最快的检测器，而且检测精度是其他系统的两倍。YOLO比Fast YOLO的mAP高10，而且速度远高于实时系统的速度要求。

R-CNN减去R用静态边界框提议取代选择性搜索。虽然它的速度比R-CNN速度快得多，但是它还还达不到实时的要求， 而且因为没有很好的建议框所以精度很受影响。

Fast R-CNN加速了R-CNN的分类阶段，但仍然依赖于选择性搜索，每个图像大约需要2秒才能生成建议边界框。所以虽然它的mAP很高，但是速度只有0.5 fps达不到实时速度要求。

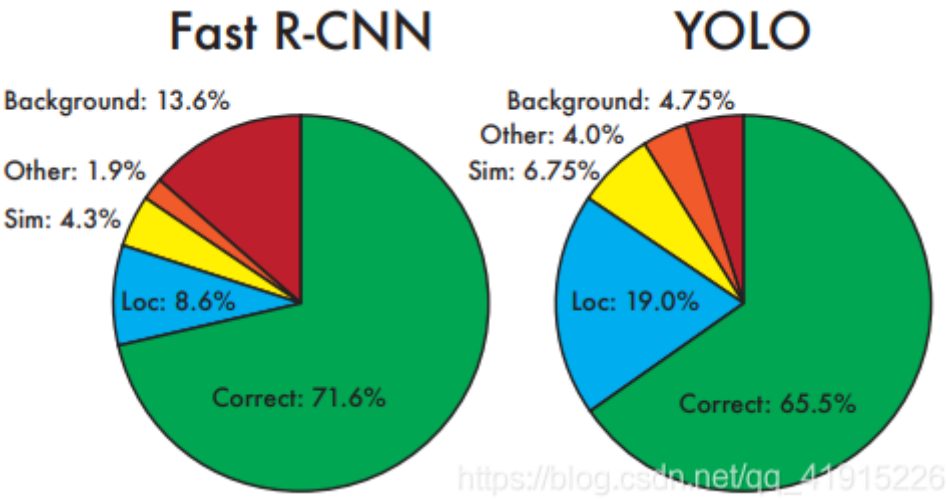
目前的Faster R-CNN使用一个神经网络替代选择性搜索来生成建议边界框。比如：Szegedy等人在我们的测试中，他们最精确的模型速度达到7 fps，而较小的，不太精确的模型以速度达到18 fps。VGG-16版本的Faster R-CNN比YOLO的mAP高10，但是速度比YOLO慢6倍。Zeiler-Fergus Faster R-CNN仅比YOLO慢2.5倍，但是精度还是不及YOLO。

4.2. VOC 2007 Error Analysis（误差分析）

为了进一步研究YOLO和最先进的检测器之间的差异，我们将详细分析在VOC 2007上的检测结果。我们将YOLO与Fast R-CNN进行比较，因为Fast R-CNN是PASCAL上性能最高的检测器之一，它的检测是公开的。

我们使用Hoiem等人的方法和工具。对于测试时的每个类别，我们查看该类别的前N个预测。 每个预测都是正确的，或者根据错误类型进行如下分类：

正确：正确类别 并且 IOU>0.5
定位：正确类别 并且 0.1<IOU<0.5
相似：相似的类别 并且 IOU>0.1
其他：类别错误 并且IOU>0.1
背景：所有类别上IOU<0.1



> 图片4.**错误分析 Fast R-CNN vs. YOLO**
> 这些图表反映了在各个类别的得分最高的前N个预测中定位错误和背景错误的比例。（在该类别中N = # 个目标）

图4显示了所有20个类中平均每种错误类型的细分。YOLO努力的去准确定位物体。YOLO中的定位错误比其他所有类型错误之和还多。Fast R-CNN的定位错误更少但是背景错误更多，它最好的检测结果中有13.6%是假阳（本来不含有物体误报为有物体）。Fast R-CNN对背景的误报错误是YOLO的三倍。

4.3Combining Fast R-CNN and YOLO(Fast R-CNN和YOLO相结合)

与Fast R-CNN相比，YOLO的背景误报错误要少得多。 通过使用YOLO减小Fast R-CNN的背景误报错误，我们可以显着提升性能。对于R-CNN预测的每个边界框，我们检查YOLO是否预测了一个类似的框。如果确实如此，我们会根据YOLO预测的概率和两个框之间的重叠来提高该预测得分。

最好的Fast R-CNN模型在VOC 2007测试集上获得了71.8%的mAP。当与YOLO结合使用时，其mAP增加了3.2%达到75.0%。 我们还尝试将最好的Fast R-CNN模型与其他几个版本的Fast R-CNN相结合。 这些结合使mAP小幅增加0.3%和0.6%之

间，详见表2。

	mAP	Combined	Gain
Fast R-CNN	71.8	-	-
Fast R-CNN (2007 data)	**66.9**	72.4	.6
Fast R-CNN (VGG-M)	59.2	72.4	.6
Fast R-CNN (CaffeNet)	57.1	72.1	.3
YOLO	63.4	**75.0**	**3.2**

> 表2：**VOC 2007的模型组合实验。**我们研究了将各种模型与最佳版本的Fast R-CNN相结合的效果。其他模型和Fast R-CNN结合仅带来了较小的性能提升，而和YOLO结合则带来显著的性能提升。

YOLO带来的性能提升不是模型集成的结果，因为集成不同版本的Fast R-CNN几乎没有什么性能提升。相反，正是因为YOLO在测试中犯了各种各样的错误，导致它能很有效地提升Fast R-CNN的表现。

不幸的是因为我们是分别训练各个模型然后结合结果，所以系统没有从YOLO的快速性上受益，速度没有什么提高。但是，因为YOLO速度很快，所以相对单独的Fast R-CNN，结合YOLO之后不会增加多少计算时间。

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [27]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [28]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [32]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

> 表3：**PASCAL VOC 2012排行榜。**
> 截至2015年11月6日，YOLO与完整comp4（允许外部数据）公共排行榜相比。针对各种检测方法显示了平均精度和每个类平均精度。YOLO是唯一的实时检测器。
> Fast R-CNN + YOLO是得分第四高的方法，比Fast R-CNN提高2.3%。

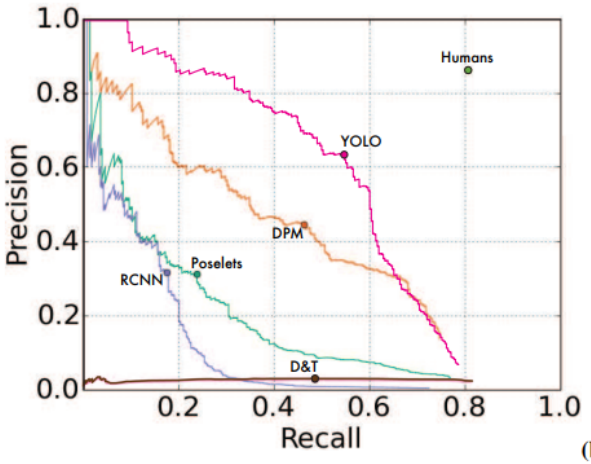
4.4 VOC 2012 Results（VOC 2012结果）

在VOC 2012测试集中，YOLO的mAP分数为57.9%。这低于现有技术水平，更接近使用VGG-16的原始R-CNN，参见表3。与最接近的竞争对手相比，我们的系统在小物体检测时有物体间竞争。在瓶子，羊，电视/监视器等类别上，YOLO得分比R-CNN或Feature Edit低8-10%。然而，在其他类别如猫和火车上，YOLO实现了更高的性能。

我们的Fast R-CNN + YOLO组合模型是性能最高的检测方法之一。 Fast R-CNN从与YOLO的组合中获得了2.3%的提升，使其在公共排行榜上提升了5位。

4.5. Generalizability: Person Detection in Artwork（抽象性：艺术作品中的人物检测）

用于对象检测的学术数据集是从同一分布中提取训练和测试数据。在实际应用中，很难预测所有可能的用例，测试数据可能与系统之前的情况不同。我们将YOLO与其他检测系统在毕加索数据集和人物艺术数据集上进行了比较，这两个数据集是用来测试艺术品中的人员检测。



(a) Picasso Dataset precision-recall curves.

	VOC 2007	Picasso		People-Art
	AP	AP	Best F_1	AP
YOLO	**59.2**	**53.3**	**0.590**	**45**
R-CNN	54.2	10.4	0.226	26
DPM	43.2	37.8	0.458	32
Poselets [2]	36.5	17.8	0.271	
D&T [4]	-	1.9	0.051	

(b) Quantitative results on the VOC 2007, Picasso, and People-Art Datasets. The Picasso Dataset evaluates on both AP and best F_1 score.

> 图5：Picasso和People-Art数据集上的结果。

图5展示了YOLO和其他系统的性能比较。作为参考，我们提供了只在VOC2007上训练的模型的人员检测AP。 Picasso模型在VOC 2012上训练，而People-Art 在VOC2010上训练。

R-CNN在VOC 2007上有较高的AP，但是在艺术品领域性能就急剧下降。R-CNN使用选择性搜索来调整自然图像的建议边界框。 R-CNN中的分类器步骤只能看到小区域，所以需要很好的建议边界框。

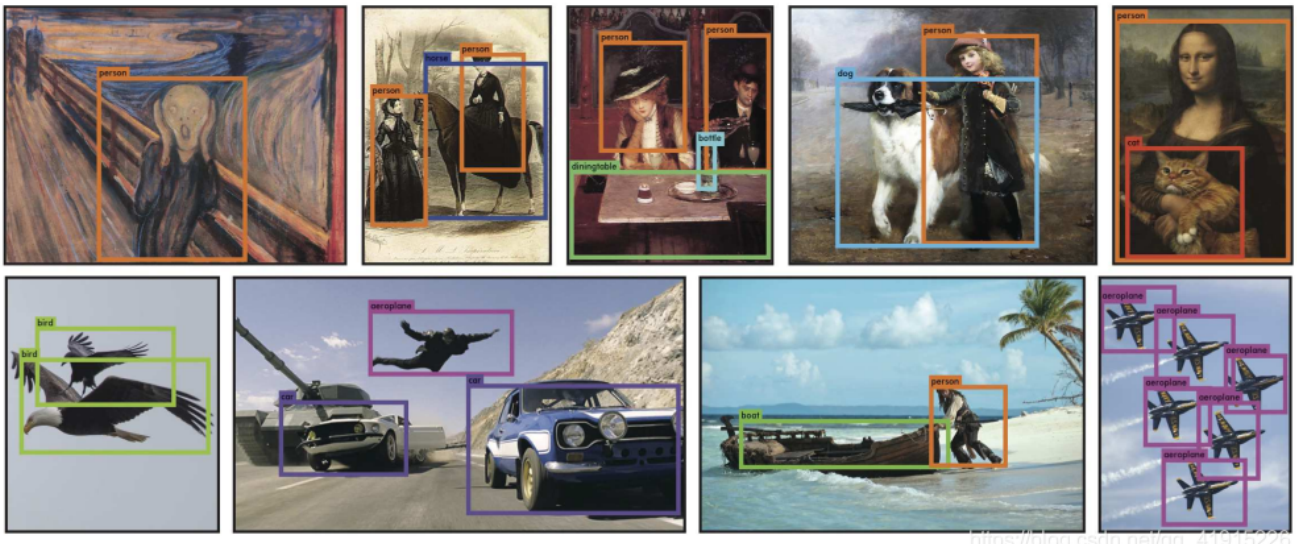
DPM在应用于艺术品时可以很好的保持它的AP。之前的工作认为DPM表现良好是因为它具有物体的形状和布局的空间模型。虽然DPM不会像R-CNN那样退化，但是它的起始AP比较低。

YOLO在VOC 2007上有很好的性能，其AP在应用于艺术品时的降解率低于其他方法。与DPM一样，YOLO对对象的大小和形状、对象之间的关系以及对象通常出现的位置进行建模。艺术品和自然图像在像素级别上有很大的不同，但它们在物体的大小和形状上是相似的，因此YOLO仍然可以预测良好的边界框和检测。

5. Real-Time Detection In The Wild（实地场景的实时检测）

YOLO是一款快速而准确的检测器，非常适合应用在计算机视觉领域。我们将YOLO连接到网络摄像头，并验证它是否保持实时性能，计算时间包括从摄像头获取图像并显示检测结果的时间。

由此生成的系统是交互式且迷人的。虽然YOLO可以单独处理图像，但是当它和网络摄像头连接起来时就像一个追踪系统，在物体运动或者变化的时候实时检测系统。系统演示和源代码可以在我们的项目网站上找到：
http：//pjreddie.com/yolo/。



> 图片6：**定性结果。**YOLO在检测线上的艺术品图片和自然图片的表现。虽然它将一个人识别成飞机但是准确性还是很高的。

6.Conclusion（结论）

我们介绍了一款一体化（端到端）的物体检测系统YOLO。我们的模型结构很简单，可以在整个图像上进行训练。与基于分类器的方法不同，YOLO针对与检测性能直接相关的损失函数来训练，而且整个模型是联合训练的。

Fast YOLO是目前文献中最快的通用物体检测系统，YOLO引领目前最先进的实时物体检测技术。YOLO还可以很好的迁移到新的领域，这使它成为需要快速高效的物体检测系统的应用的理想选择。

> YOLOv1最大的开创性贡献在于将物体检测作为一个回归问题进行求解，输入图像经过一次inference，便能得到图像中所有物体的位置和其所属类别及相应的置信概率。而rcnn/fast rcnn/faster rcnn将检测结果分为两部分求解：物体类别（分类问题），物体位置即bounding box（回归问题），所以YOLO的目标检测速度很快。