# You Only Look Once: Unified, Real-Time Object Detection

Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi

**摘要**

我们展示了一种名为 YOLO 的物体检测新方法。之前物体检测的方法将分类器重新应用与检测。取而代之的是，我们将物体检测建模为空间分离的边界框以及与之关联的物体类别回归问题。一个单独的网络在一次评估中直接从全图像预测边界框和类别概率。由于整个检测管道是一个单一网络，所以它可以对检测性能端到端直接优化。

我们的统一架构极其快。我们的 base YOLO 模型以 45 帧每秒的速度实时处理图片。名为 Fast YOLO 的更小网络版本，以惊人的 155 帧每秒的速度处理，同时仍然达到了其他实时检测器两倍的 mAP。相比于 sota 检测系统，YOLO 会产生更多定位错误但是在背景产生误报的可能性更小。最后，YOLO 习得了对于物体十分通用的表征。当从自然图像推广到例如艺术品的其他领域时，它的性能由于包括 DPM 和 R-CNN 在内的其他检测方法。

## 1 Introduction

人们只要撇图片一眼就可以立即知道什么物体在图片中、它们在哪里以及它们是如何交互的。人类的视觉系统是快速且准确的，这允许我们在不那么小心的情况下去进行例如开车的复杂任务。快速且准确的检测算法将会允许计算机在没有专门传感器的情况下驾驶汽车、使得辅助设备可以传输实时场景信息给人类用户、并释放通用响应机器人系统的潜力。

目前的检测系统通过复用分类器来进行检测。为了检测一个物体，这些系统为该物体使用一个分类器，并在测试图像中的不同位置和不同尺度对其进行评估。可变部件模型（DPM）等系统使用滑动窗口方法，其中分类器在整个图像上的均匀间隔的位置运行 [1]。

最近的方法如 R-CNN 通过区域候选的方法首先在图像中生成潜在的边界框，然后在这些候选框上运行分类器。分类后，通过后处理来微调边界框、消除重复检测，并根据场景中的其他物体对框重新评分 [3]。因为必须单独训练复杂管道中的每个单独组件，所以这些管道速度缓慢且难以优化。

我们将目标检测重新定义为一个单一的回归问题，直接从图像像素到边界框坐标和类别概率。使用我们的系统，只需查看图像一次 (YOLO) 即可预测存在哪些物体以及它们在哪里。

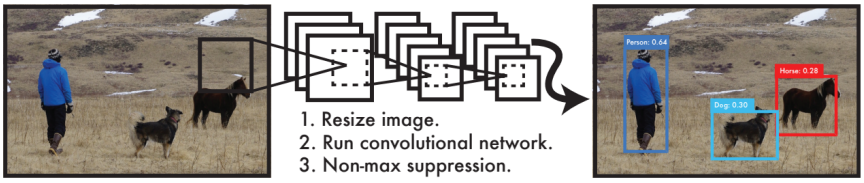


图 1: **YOLO 检测系统**。使用 YOLO 处理图像简单明了。我们的系统 (1) 将输入图像的大小调整为 $448 \times 448$，(2) 在图像上运行单个卷积网络，以及 (3) 通过模型的置信度对结果检测进行阈值处理。

YOLO 非常简单：见图1。单个卷积网络同时预测多个边界框和这些框的类别概率。YOLO 在完整图像上训练并直接优化检测性能。与传统的物体检测方法相比，这种统一模型有多个优点。

首先，YOLO 非常快。由于我们将检测视为回归问题，因此我们不需要复杂的管道。我们只是在测试时在新图像上运行我们的神经网络来预测检测。在 Titan X GPU 上且没有批处理的情况下，我们的基础网络达到了每秒 45 帧的运行速度，快速版本达到了超过 150 fps 的运行速度。这意味着我们可以以不到 25 毫秒的延迟实时处理流视频。此外，YOLO 的平均精度是其他实时系统平均精度的两倍以上。有关我们系统在网络摄像头上实时运行的演示，请参阅我们的项目网页：http://pjreddie.com/yolo/。

其次，YOLO 在进行预测时会对图像进行全局推理。与基于滑动窗口和区域候选的技术不同，YOLO 在训练和测试期间看到整个图像，因此它隐式编码了关于类及其外观的上下文信息。Fast R-CNN 是一种顶级检测方法 [5]，由于无法看到更大的上下文，因此会将图像中的背景块误认为是物体。与 Fast R-CNN 相比，YOLO 的背景误报数量不到一半。

第三，YOLO 学习物体的可泛化表示。在对自然图像进行训练并在艺

术品上进行测试时，YOLO 的性能大大优于 DPM 和 R-CNN 等顶级检测方法。由于 YOLO 具有高度泛化性，因此在应用于新领域或意外输入时不太可能崩溃。

YOLO 在准确性方面仍然落后于最先进的检测系统。虽然它可以快速识别图像中的物体，但它很难精确定位一些物体，尤其是小物体。我们在实验中进一步研究了这些权衡。

我们所有的训练和测试代码都是开源的。还可以下载各种预训练模型。

# 2 Unified Detection

我们将物体检测的独立组件统一到一个神经网络中。我们的网络使用来自整个图像的特征来预测每个边界框。它还同时预测图像的所有类别的所有边界框。这意味着我们的网络对完整图像和图像中的所有物体进行全局推理。YOLO 设计支持端到端训练和实时速度，同时保持高平均精度。

我们的系统将输入图像划分为 $S \times S$ 网格。如果物体的中心落入网格单元中，则该网格单元负责检测该物体。

每个网格单元预测 $B$ 个边界框和这些框的置信度。这些置信度反映了模型对框包含物体的信心程度，以及它认为盒子预测的准确度。形式上，我们将置信度定义为 $\Pr(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}$。如果该单元格中不存在物体，则置信度分数应为零。否则，我们希望置信度得分等于预测框和 ground truth 之间的交集（IOU）。

每个边界框由 5 个预测组成：$x, y, w, h$ 和置信度。$(x, y)$ 坐标表示相对于网格单元边界的框中心。宽度和高度是相对于整个图像预测的。最后，置信度预测表示预测框和任何 ground truth 框之间的 IOU。

每个网格单元还预测 $C$ 个条件类概率，$\Pr(\text{Class}_i|\text{Object})$。这些概率以网格单元包含物体的为条件。我们只为每个网格单元预测一组类概率，而不管框 $B$ 的数量。

在测试时，我们将条件类概率和单个框置信度预测相乘，

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

这为我们提供了每个框特定类的置信度分数。这些分数编码了该类出现在框中的概率以及预测的框与物体的匹配程度。

为了在 PASCAL VOC 上评估 YOLO，我们使用 $S = 7$，$B = 2$。

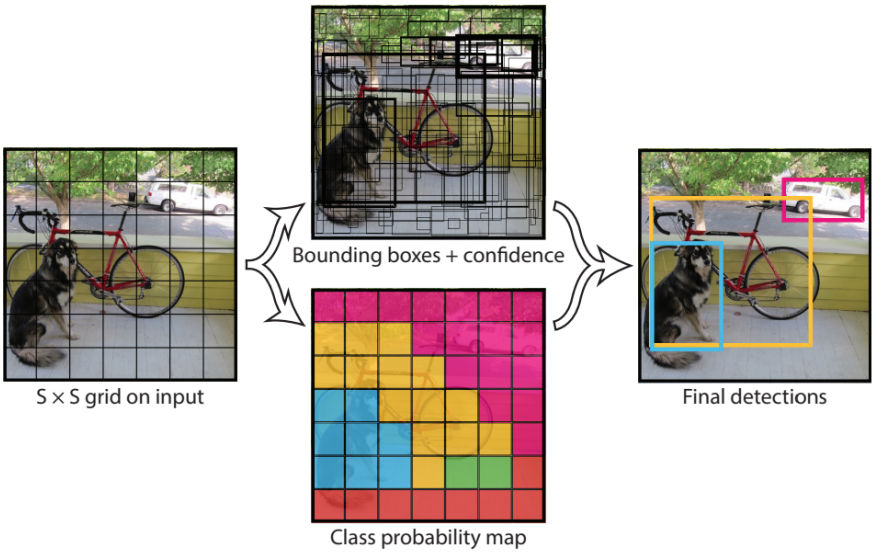


图 2: **模型**。我们的系统将检测建模为一个回归问题。它将图片划分为 $S \times S$ 个网格，并为每个网格预测 $B$ 个边界框、这些边界框的置信度和 $C$ 个类别概率。这些预测被编码为一个 $S \times S \times (B * 5 + c)$ 的张量。

PASCAL VOC 有 20 个标记类别，因此 $C = 20$。我们的最终预测是一个 $7 \times 7 \times 30$ 张量。

## 2.1 网络设计

我们将此模型实现为卷积神经网络，并在 PASCAL VOC 检测数据集 [4] 上对其进行评估。网络的初始卷积层从图像中提取特征，而全连接层预测输出概率和坐标。

我们的网络架构受到用于图像分类的 GoogLeNet 模型的启发 [6]。我们的网络有 24 个卷积层，后跟 2 个全连接层。我们不使用 GoogLeNet 使用的 inception 模块，而是简单地使用 $1 \times 1$ 缩减层和 $3 \times 3$ 卷积层，类似于 Lin 等人 [2]。完整的网络如图3所示。

我们还训练了一个快速版本的 YOLO，旨在突破快速目标检测的边界。Fast YOLO 使用较少卷积层（9 个而不是 24 个）并在这些层中使用较少的卷积核。除了网络的大小之外，YOLO 和 Fast YOLO 的所有训练和测试参

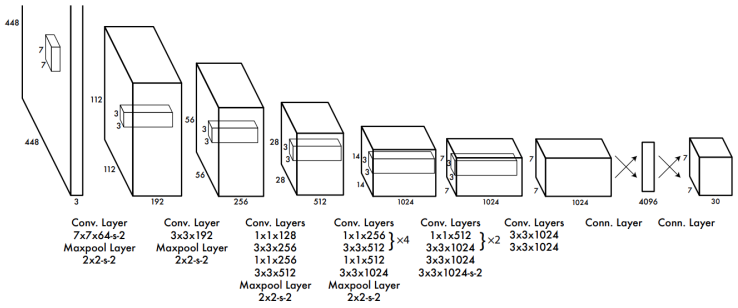


图 3: **网络结构**。我们的检测网络有 24 个卷积层，后跟 2 个全连接层。交替的 $1 \times 1$ 卷积层减少了前一层的特征空间。我们在 ImageNet 分类任务上以一半的分辨率（$224 \times 224$ 输入图像）预训练卷积层，然后将分辨率提高一倍以进行检测。

数都是相同的。

我们网络的最终输出是 $7 \times 7 \times 30$ 的预测张量。

## 2.2 训练

我们在 ImageNet 1000 类竞赛数据集 [30] 上预训练我们的卷积层。对于预训练，我们使用图3中的前 20 个卷积层，然后是平均池化层和全连接层。我们对该网络进行了大约一周的训练，并在 ImageNet 2012 验证集上实现了 88% 的单次裁剪 top-5 准确率，与 Caffe 的 Model Zoo [24] 中的 GoogLeNet 模型相当。我们使用 darknet 框架进行所有训练和推理 [26]。

然后我们转换模型以执行检测。Ren 等人表明将卷积层和连接层添加到预训练网络可以提高性能 [29]。按照他们的例子，我们添加了具有随机初始化权重的四个卷积层和两个全连接层。检测通常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 增加到 448×448。

我们的最后一层预测类别概率和边界框坐标。我们通过图像的宽度和高度对边界框的宽度和高度进行归一化，使它们落在 0 和 1 之间。我们将边界框的 $x$ 和 $y$ 坐标参数化为特定网格单元位置的偏移量，因此它们也被限制在 0 和 1 之间.

我们对最后一层使用线性激活函数，所有其他层使用以下 leaky 校正线

性激活：

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

我们针对模型输出中的平方和误差进行了优化。我们使用平方和误差是因为它很容易优化，但是它并不完全符合我们最大化平均精度的目标。它将定位误差与分类误差同等加权，这可能并不理想。此外，在每个图像中，许多网格单元不包含任何物体。这会将这些单元格的“置信度”分数推向零，通常会压倒包含物体的单元格的梯度。这可能会导致模型不稳定，从而导致训练早期出现发散。

为了解决这个问题，我们增加了边界框坐标预测的损失，并减少了不包含物体的框的置信度预测的损失。我们使用两个参数 $\lambda_{\text{coord}}$ 和 $\lambda_{\text{noobj}}$ 来实现这一点。我们设置 $\lambda_{\text{coord}} = 5$ 和 $\lambda_{\text{noobj}} = 0.5$。

平方和误差也同样加权大框和小框的错误。我们的误差度量应该反映大盒子中的小偏差比小盒子中的小偏差重要性小。为了部分解决这个问题，我们预测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 为每个网格单元预测多个边界框。在训练时，对于每个物体，我们只希望一个边界框预测器对它负责。我们根据哪个预测与 ground truth 具有最高的 IOU 来指定哪一个预测器为此 ground truth “负责”。这导致边界框预测器之间的专业化。每个预测器在预测特定大小、长宽比或物体类别方面都变得更好，从而提高了整体召回率。

在训练期间，我们优化了以下多部分损失函数：

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

其中 $\mathbb{1}_i^{\text{obj}}$ 表示物体是否在单元 $i$ 出现，$\mathbb{1}_{ij}^{\text{obj}}$ 表示单元 $i$ 的第 $j$ 个边界框预

测器对这个检测“负责”。

请注意，仅当该网格单元中存在物体时（因此是前面讨论的条件类概率），损失函数才会惩罚分类错误。仅当该预测器对 ground truth 框“负责”（即具有该网格单元中任何预测器的最高 IOU）时，它才会惩罚边界框坐标错误。

我们在来自 PASCAL VOC 2007 和 2012 的训练和验证数据集上训练网络约 135 个 epoch。在 2012 测试时，我们在训练时还使用了 VOC 2007 的测试数据。在整个训练过程中，我们使用 64 的批量大小、0.9 的动量和 0.0005 的衰减。

我们的学习率计划如下：对于第一个 epoch，我们慢慢地将学习率从 $10^{-3}$ 提高到 $10^{-2}$。如果我们以高学习率开始，我们的模型通常会因梯度不稳定而发散。我们继续用 $10^{-2}$ 训练 75 个 epoch，然后用 $10^{-3}$ 训练 30 个 epoch，最后用 $10^{-4}$ 训练 30 个 epoch。

为了避免过拟合，我们使用 dropout 和广泛的数据增强。在第一个连接层之后具有 rate = .5 的 dropout 层来防止层之间的协同适应 [18]。对于数据增强，我们引入了最多原始图像大小 20% 的随机缩放和平移。我们还在 HSV 色彩空间中随机调整图像的曝光和饱和度，最高达 1.5。

## 2.3 推理

就像在训练中一样，预测测试图像的检测只需要一次网络评估。在 PASCAL VOC 上，网络预测每个图像的 98 个边界框和每个框的类别概率。YOLO 在测试时非常快，因为它只需要一次网络评估，这与基于分类器的方法不同。

网格设计在边界框预测中强制执行空间多样性。通常很清楚一个物体属于哪个网格单元，并且网络只为每个物体预测一个框。但是，一些大型物体或靠近多个单元格边界的物体可以被多个单元格很好地定位。非最大抑制可用于修复这些重复检测。虽然对 R-CNN 或 DPM 的性能并不重要，但非最大抑制在 mAP 中增加了 2-3%。

## 2.4 YOLO 的局限

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预测两个框并且只能有一个类。这种空间约束限制了我们的模型可以预测的

附近物体的数量。我们的模型在处理成群出现的小物体时遇到了困难，例如成群的鸟。

由于我们的模型学习从数据中预测边界框，因此它很难泛化到具有新的或不寻常的长宽比或配置的物体。我们的模型还使用相对粗略的特征来预测边界框，因为我们的架构包含对输入图像的多个下采样层。

最后，当我们训练近似检测性能的损失函数时，我们的损失函数将小边界框与大边界框的错误处理相同。大框的小错误通常是良性的，但小框的小错误对 IOU 的影响要大得多。我们的主要错误来源是不正确的定位。

# References

[1] Pedro F Felzenszwalb et al. “Object detection with discriminatively trained part-based models”. In: *IEEE transactions on pattern analysis and machine intelligence* 32.9 (2009), pp. 1627–1645.

[2] Min Lin, Qiang Chen, and Shuicheng Yan. “Network in network”. In: *arXiv preprint arXiv:1312.4400* (2013).

[3] Ross Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[4] Mark Everingham et al. “The pascal visual object classes challenge: A retrospective”. In: *International journal of computer vision* 111.1 (2015), pp. 98–136.

[5] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[6] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.