重新思考基于Transformer的集合预测在目标检测中的应用

孙志清* 曹胜操* 杨一鸣 北山真人 卡内基梅隆大学
{zhiqings,shengcao,yiming,kkitani}@cs.cmu.edu

摘要

DETR is a recently proposed Transformer-based method which views object detection as a set prediction problem and achieves state-of-the-art performance but demands extra-long training time to converge. In this paper, we investigate the causes of the optimization difficulty in the training of DETR. Our examinations reveal several factors contributing to the slow convergence of DETR, primarily the issues with the Hungarian loss and the Transformer cross-attention mechanism. To overcome these issues we propose two solutions, namely, TSP-FCOS (Transformer-based Set Prediction with FCOS) and TSP-RCNN (Transformer-based Set Prediction with RCNN). Experimental results show that the proposed methods not only converge much faster than the original DETR, but also significantly outperform DETR and other baselines in terms of detection accuracy. Code is released at https://github.com/Edward-Sun/TSP-Detection.

1. 引言

目标检测旨在找出图像中所有感兴趣的目标，并预测其类别标签和边界框，这本质上是一个集合预测问题，因为无需对预测目标进行排序。大多数前沿的神经检测器[25, 29, 23, 30, 43, 31, 14]采用"检测-合并"的开发范式——这些方法并非以端到端方式直接优化预测集合，而是先对一组区域提议或滑动窗口进行预测，再通过后处理步骤（如"非极大值抑制"NMS）合并可能属于同一目标的不同提议或窗口中的检测结果。由于检测模型的训练与合并步骤无关，这类目标检测器的模型优化并非端到端，且存在次优性。

DEtection TRansformer (DETR) [3] 是近期提出的首个完全端到端目标检测器。它采用Transformer [37]能够直接输出最终的预测集合，无需进一步后处理。然而，其训练收敛需要超长的时间。例如，流行的Faster RCNN模型[31]仅需约30个训练周期即可收敛，而DETR则需要500个周期，在8块V100 GPU上至少耗时10天。如此高昂的训练成本在大规模应用中实际上是不可行的。因此，如何以何种方式加速类似DETR的基于Transformer的检测器的训练过程以实现快速收敛，是一个具有挑战性的研究问题，也是本文的主要关注点。

为分析DETR优化困难的成因，我们进行了大量实验，发现Transformer解码器通过交叉注意力模块从图像中获取目标信息，该模块是导致收敛缓慢的主要原因。为追求更快收敛，我们进一步研究了移除交叉注意力模块后的纯编码器版DETR。研究发现，纯编码器DETR对小目标检测效果提升显著，但对大目标检测性能欠佳。此外，分析表明DETR匈牙利损失中二分匹配的不稳定性也是造成收敛缓慢的原因之一。

基于上述分析，我们提出了两种显著加速基于Transformer的集合预测方法训练过程的模型，二者均可视为带特征金字塔的纯编码器DETR[22]的改进版本。具体而言，我们提出了受经典一阶段检测器FCOS[35]（全卷积单阶段目标检测器）启发的TSP-FCOS（基于Transformer的FCOS集合预测）和受经典两阶段检测器Faster RCNN[31]启发的TSP-RCNN（基于Transformer的RCNN集合预测）。TSP-FCOS中开发了新颖的感兴趣特征（FoI）选择机制，以帮助Transformer编码器处理多层级特征。针对匈牙利损失中二分图匹配的不稳定性问题，我们还为两个模型分别设计了新的二分图匹配方案以加速训练收敛。在COCO 2017检测基准[24]上的评估中

*indicates equal contribution.

所提出的方法不仅收敛速度远超原始DETR，而且在检测精度方面也显著优于DETR及其他基线模型。

2. 背景

2.1. 单阶段与双阶段目标检测器

现代大多数目标检测方法可分为两大类：单阶段检测器与双阶段检测器。典型的单阶段检测器[25,29,23]直接基于图像中提取的特征图和（可变尺寸的）滑动窗口位置进行预测，而双阶段检测器[31,14]首先生成基于滑动窗口位置的区域提议，随后对每个提议区域进行检测优化。一般而言，双阶段检测器精度更高，但计算成本也显著高于单阶段检测器。然而，这两类检测器均采用"检测-合并"的开发范式*i.e*，即需要通过后处理步骤确保每个被检测对象仅对应一个检测区域，而非多个重叠区域作为检测结果。换言之，许多前沿目标检测方法并未建立针对集合预测的端到端训练目标。

2.2. 采用端到端目标的DETR

与上述流行的目标检测器不同，DEtection TRansformer (DETR) [3] 首次提出了一种针对集合预测的端到端优化目标方法。具体而言，它通过二分图匹配机制构建损失函数。设真实物体集合为$y = \{y_i\}_{u=1}^{M}$，预测集合为$\hat{y} = \{\hat{y}_i\}_{u=1}^{N}$。通常我们拥有$M < N$，因此用$\emptyset$ (无物体)将y填充至N大小，并记作$\bar{y}$。该损失函数，即匈牙利损失，定义为:

$$\mathcal{L}_{\text{Hungarian}}(\bar{y}, \hat{y}) = \sum_{i=1}^{N} \left[\mathcal{L}_{class}^{i,\hat{\sigma}(i)} + \mathbb{1}_{\{\bar{y}_i \neq \emptyset\}} \mathcal{L}_{box}^{i,\hat{\sigma}(i)} \right] \quad (1)$$

其中$\mathcal{L}_{class}^{i,\hat{\sigma}(i)}$和$\mathcal{L}_{box}^{i,\hat{\sigma}(i)}$分别表示$i^{th}$真实标注与$\hat{\sigma}(i)^{th}$预测结果之间的分类损失和边界框回归损失。而$\hat{\sigma}$则是填充后的真实标注集$\bar{y}$与预测集$\hat{y}$之间具有最低匹配成本的最优二分匹配:

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i=1}^{N} \mathcal{L}_{\text{match}}(\bar{y}_i, \hat{y}_{\sigma(i)}) \quad (2)$$

其中$\mathfrak{S}_N$是所有N排列的集合，$\mathcal{L}_{\text{match}}$是两两匹配成本。

DETR [3]采用了基于CNN骨干网络的编码器-解码器Transformer [37]框架。该Transformer编码器部分处理来自CNN主干的扁平化深度特征[1]。随后，非自回归解码器部分以编码器的输出和一组学习得到的目标查询向量作为输入，预测类别标签和边界框作为检测输出。解码器中的交叉注意力模块通过为不同目标查询关注图像中的不同位置发挥关键作用。对Transformer概念不熟悉的读者可参阅附录。DETR中的注意力机制消除了对NMS后处理的需求，因为自注意力组件能够学会去除重复检测*i.e*，其匈牙利损失（公式1）在二分图匹配中促使每个目标对应一个预测结果。

与我们的工作同期，一些DETR的变体被提出以提升其训练效率和准确性。可变形DETR[47]提出整合可变形卷积与注意力模块的概念，在多层次特征图上实现稀疏注意力机制。UP-DETR[10]则利用一种名为随机查询块检测的无监督预训练任务，来增强DETR在下游任务微调时的表现。相较于这些工作，我们探索通过仅使用编码器的Transformer结构进一步简化检测头的设计。

2.3. 优化真实值分配

DETR中的匈牙利损失可视为一种端到端的方式，用于将真实标签分配给系统预测。在DETR之前，已有研究尝试采用启发式规则完成此任务[12,31,29]。另有若干前期工作致力于改进启发式真值分配规则。[44]提出了一种最大似然估计(MLE)流程来学习滑动窗口与真实目标间的匹配关系。[32]则提出广义交并比(generalized IoU)以提供更优的度量标准。然而这些方法均未直接优化基于集合的目标函数，且仍需依赖非极大值抑制(NMS)后处理步骤。

2.4. 基于注意力的目标检测

基于注意力的建模已成为自然语言处理（NLP）领域当前的主流方法[37, 11]，并在近期目标检测研究中日益流行。在DETR问世之前，[16]提出了一个基于注意力的模块来建模物体间关系，该模块可嵌入现有检测器中，从而实现更好的识别效果并减少重复检测。[28]采用空间注意力模块对特征图进行重加权，使前景特征更为突出。[5]则利用类Transformer的注意力模块来桥接不同形式的表征。

[1] In this paper, we use “feature points” and “features” interchangeably.

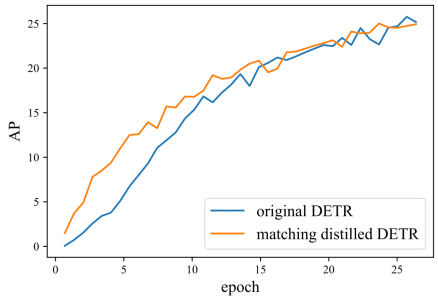

图1. COCO验证集上的AP结果：原始DETR对比匹配蒸馏DETR。可以看出，匹配蒸馏在最初几个训练周期内加速了DETR的训练。

但是，这些方法中没有一个尝试过端到端的集合预测目标。

3. DETR收敛缓慢的原因是什么?

为了确定主要因素，我们使用基于ResNet-50骨干网络构建的DETR及其变体进行了一系列实验，并在COCO 2017验证集上进行了评估。

3.1. 二分匹配的不稳定性是否会影响收敛?

作为DETR中一个独特的组成部分，基于二分图匹配的匈牙利损失（第2.2节）可能因以下原因而不稳定：

- 二分匹配的初始化本质上是随机的；
- 匹配不稳定性可能由不同训练周期中的噪声条件引起。

为了探究这些因素的影响，我们为DETR提出了一种新的训练策略——匹配蒸馏。具体而言，我们采用一个经过良好预训练的DETR作为教师模型，其预测的双边匹配结果被视作学生模型的真实标签分配。教师模型中所有随机性模块（*i.e*，如dropout[34]和批量归一化[17]）均被关闭，以确保提供的匹配具有确定性，从而消除双边匹配及匈牙利损失中的随机性与不稳定性。

我们评估了原始DETR与匹配蒸馏后的DETR。图1展示了前25个epoch的结果。可以看出，匹配蒸馏策略确实在最初几个epoch中促进了DETR的收敛。然而，这种效果在大约15个epoch后变得不再显著。这表明DETR中二分匹配组件的不稳定性仅部分导致了收敛缓慢（尤其在训练初期），而不一定是主要原因。

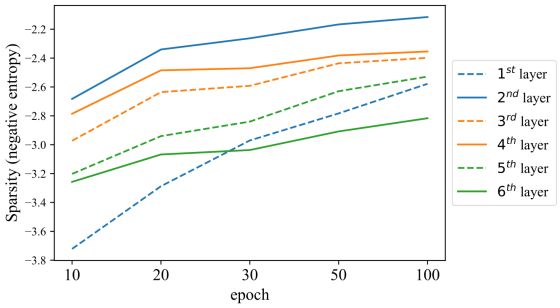

图2. Transformer交叉注意力在各层的稀疏性（负熵），通过在COCO验证数据上的评估得出。不同线型代表不同层。可以看出，稀疏性持续增加，尤其是编码器与解码器之间的1^{st}交叉注意力层。

3.2. 注意力模块是主要原因吗?

与其他现代目标检测器相比，DETR另一个显著特点是其采用了Transformer模块。在初始化阶段，Transformer的注意力图几乎呈均匀分布，但随着训练过程向收敛推进，这些注意力图会逐渐变得越来越稀疏。先前的研究[18]表明，在BERT[11]中用更稀疏的模块（{v*}，如卷积）替换部分注意力头，能显著加速训练。因此，我们很自然地会思考：DETR中Transformer注意力模块的稀疏性动态变化，对其收敛缓慢现象的影响程度究竟有多大。

在分析DETR注意力模块对其优化收敛性的影响时，我们特别关注交叉注意力部分的稀疏性动态，因为交叉注意力模块是解码器中对象查询从编码器获取对象信息的关键组件。不精确（未充分优化）的交叉注意力可能导致解码器无法从图像中提取准确的上下文信息，从而造成定位效果不佳，尤其是对小物体而言。

我们在评估DETR模型不同训练阶段时，收集了交叉注意力机制的注意力图。由于注意力图可被视作概率分布，我们采用负熵作为稀疏性的直观度量指标。具体而言，给定一个$n \times m$注意力图$\mathbf{a}$，首先通过$\frac{1}{m}\sum_{j=1}^{m} P(a_{i,j}) \log P(a_{i,j})$计算每个源位置$i \in [n]$的稀疏度，其中$a_{i,j}$表示从源位置$i$到目标位置$j$的注意力得分。随后我们对每层中所有注意力头及所有源位置的稀疏度取平均值。计算稀疏度时，掩码位置[3]不予考虑。

图2展示了不同训练周期下多个层的稀疏性情况。可以看出，交叉注意力机制的稀疏性持续上升，即便在训练超过100个周期后仍未达到稳定状态。这意味着

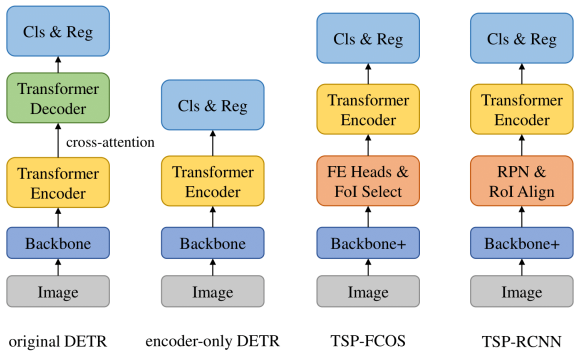

图3. 原始DETR、仅编码器DETR、TSP-FCOS和TSP-RCNN的示意图，其中Backbone+、FE Heads、RPN、Cls & Reg分别代表“Backbone + FPN”、“特征提取头（子网络）”、“区域提议网络”、“分类与回归”。TSP-FCOS和TSP-RCNN的更详细图示可参见图5。

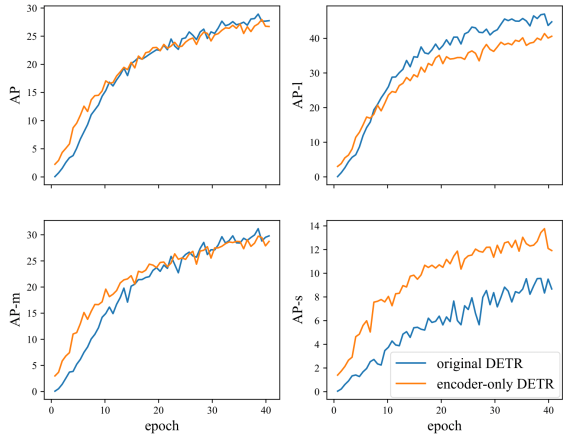

图4. COCO验证集上的AP、AP-l、AP-m和AP-s结果：原始DETR对比仅编码器DETR。可以看出，仅编码器DETR显著加快了小物体检测的训练速度。

与之前讨论的早期二分匹配不稳定性因素相比，DETR中的交叉注意力部分对收敛速度缓慢的影响更为显著。

3.3. DETR真的需要交叉注意力吗?

我们的下一个问题是：能否从DETR中移除交叉注意力模块以加速收敛，同时不牺牲其在目标检测中的预测能力？我们通过设计一个仅含编码器版本的DETR，并将其收敛曲线与原版DETR进行对比，来回答这一问题。

在原始的DETR中，解码器负责为每个对象查询生成检测结果（类别标签和边界框）。相比之下，我们提出的仅编码器版本DETR直接利用Transformer编码器的输出进行对象预测。具体而言，对于具有$\frac{H}{32} \times \frac{W}{32}$ Transformer编码器特征图的$H \times W$图像，每个特征会被送入检测头以预测检测结果。由于编码器自注意力机制本质上与非自回归解码器中的自注意力相同，因此集合预测训练对仅编码器DETR仍然可行。更多关于仅编码器DETR的细节可在附录中找到。图3对比了原始DETR、仅编码器DETR以及我们新提出的两种模型（TSP-FCOS和TSP-RCNN），后两者将在下一节详述。

图4展示了原始DETR与仅编码器DETR的平均精度（AP）曲线，包括整体AP曲线（标记为AP）以及分别针对大（AP-l）、中（AP-m）、小（AP-s）物体[2]的曲线。总体曲线（左上角）显示，仅编码器DETR的表现与原始DETR相当。这意味着我们可以从DETR中移除交叉注意力部分而不会显著影响性能——生成，这是一个积极的结果。从剩余的曲线可以看出，仅编码器的DETR在小物体上显著优于原始DETR，在中型物体上也有部分优势，但在大物体上表现则相对逊色。我们认为，一个可能的解释是大物体可能包含过多潜在可匹配的特征点，这对于仅编码器DETR中的滑动点方案来说难以处理。另一个可能的原因是，编码器处理的单一特征图对于预测不同尺度的物体不够鲁棒[22]。

4. 所提出的方法

根据我们在前一节的分析，为了加速DETR的收敛，需要同时解决DETR中二分图匹配部分的不稳定性问题以及Transformer模块中的交叉注意力问题。具体而言，为了充分发挥仅编码器DETR的加速潜力，必须克服其在处理不同尺度物体时的不足。近期，FCOS[35]（全卷积单阶段目标检测器）表明，结合特征金字塔网络（FPN）[22]的多层级预测是解决这一问题的有效方案。受此启发，我们提出了首个模型——基于Transformer的FCOS集合预测（TSP-FCOS）。随后，在TSP-FCOS基础上进一步引入两阶段优化，由此衍生出第二个模型——基于Transformer的RCNN集合预测（TSP-RCNN）。

4.1. TSP-FCOS

TSP-FCOS融合了FCOS与仅编码器DETR的优势，引入了一个创新组件——Fea-

[2]We follow the definitions of small, medium, and large objects in [24].

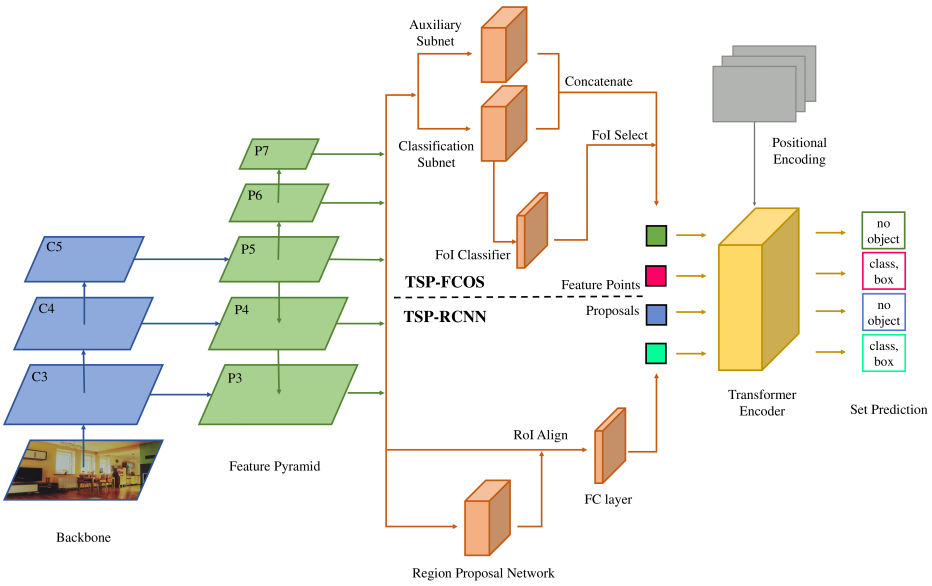

图5. TSP-FCOS与TSP-RCNN的网络架构，其中C_3至C_5表示骨干网络的特征图，P_3至P_7为特征金字塔网络（FPN）的特征图。TSP-FCOS（上图）和TSP-RCNN（下图）均配备Transformer编码器，并通过集合预测损失进行训练。两者的区别在于：TSP-FCOS中的FoI分类器仅预测各特征的目标性（*i.e*，即兴趣特征），而TSP-RCNN中的区域提议网络（RPN）则同时预测边界框及其目标性作为兴趣区域（RoI），即提议框。

兴趣目标（FoI）选择机制，使Transformer编码器能够处理多层次特征，以及一种新的二分匹配方案以加速集合预测训练。图5（上部）展示了TSP-FCOS的网络架构，包含以下组件：

主干网络与FPN　我们遵循FCOS[35]的设计方案构建主干网络和特征金字塔网络(FPN)[22]。在流程起始阶段，主干CNN网络负责从输入图像中提取特征。基于主干网络输出的特征图，我们构建了FPN模块，该模块能生成多尺度特征，帮助仅含编码器的DETR检测不同尺寸的目标。公式标记{v*}保持原样。

特征提取子网络　为了与其他单阶段检测器（如RetinaNet、FCOS和RetinaNet）进行公平比较，我们遵循其设计，采用两个共享于不同特征金字塔层级的特征提取头。其中一个称为分类子网络（头），用于兴趣区域分类；另一个称为辅助子网络（头）。两者的输出经拼接后，由兴趣区域分类器进行筛选。

兴趣特征（FoI）分类器　在Transformer的自注意力模块中，计算复杂度与序列长度呈平方关系，这阻碍了直接使用特征金字塔上的所有特征。为了提高自注意力的效率，我们设计了一个二元分类器来筛选有限数量的特征，并将其称为兴趣特征（FoI）。该二元FoI分类器经过训练采用FCOS的真实标注分配规则[3]。在兴趣区域分类后，得分最高的特征被选为兴趣区域并输入到Transformer编码器中。

Transformer编码器　在完成兴趣区域（FoI）选择步骤后，Transformer编码器的输入是一组FoI及其对应的位置编码。在Transformer编码器的每一层内部，通过自注意力机制来聚合不同FoI的信息。编码器的输出经过一个共享的前馈神经网络，该网络为每个FoI预测类别标签（包括“无物体”）和边界框。

位置编码 遵循DETR的做法，我们将Transformer[37]的位置编码推广至二维图像场景。具体而言，对于一个归一化位置为$(x, y) \in [0, 1]^2$的特征点，其位置编码定义为$[PE(x) : PE(y)]$，其中[:]表示拼接操作，函数PE的定义如下：

$$\begin{aligned} PE(x)_{2i} &= \sin(x/10000^{2i/\mathrm{d}_{\text{model}}}) \\ PE(x)_{2i+1} &= \cos(x/10000^{2i/\mathrm{d}_{\text{model}}}) \end{aligned} \tag{3}$$

其中$\mathrm{d}_{\text{model}}$是FoIs的维度。

更快的集合预测训练　如第3.2节所述，目标检测任务可视为集合预测问题。给定检测结果集合与

[3]Please refer to the FCOS paper [35] for more details.

真实目标物体，集合预测损失将它们联系在一起，并为模型提供了优化的目标。但正如我们在第3.1节所示，匈牙利二分图匹配损失在训练初期可能导致收敛速度较慢。因此，我们设计了一种新的二分图匹配方案，以加速TSP-FCOS的集合预测训练。具体而言，只有当特征点位于目标物体的边界框内且处于适当的特征金字塔层级时，该点才能被分配给一个真实目标。这一设计灵感来源于FCOS[35]的真实目标分配规则。接着，执行一个基于受限成本的匹配过程（公式2），以在匈牙利损失（公式1）中确定检测结果与真实目标之间的最优匹配。

4.2. TSP-RCNN

基于TSP-FCOS与Faster RCNN的设计理念，我们可以融合二者的优势，采用两阶段边界框精细化作为集合预测，虽然需要更多计算资源，但能实现更精确的目标检测。这一思路催生了TSP-RCNN（基于Transformer与RCNN的集合预测模型）。图5（下半部分）展示了我们提出的TSP-RCNN网络架构。TSP-FCOS与TSP-RCNN的主要区别如下：

区域提议网络 在TSP-RCNN中，我们并未采用双特征提取头与FoI分类器来获取Transformer编码器的输入，而是遵循Faster RCNN[31]的设计，利用区域提议网络（RPN）生成一组待优化的感兴趣区域（RoIs）。与TSP-FCOS中的FoIs不同，TSP-RCNN中的每个RoI不仅包含目标性得分，还包含预测边界框。我们采用RoIAlign[14]从多层级特征图中提取RoIs信息，随后将提取的特征展平并通过全连接网络输入至Transformer编码器。

位置编码 一个RoI（提案）的位置信息由四个量（cx, cy, w, h）定义，其中（cx, cy）$\in [0, 1]^2$表示归一化的中心坐标，（w, h）$\in [0, 1]^2$表示归一化的高度和宽度。我们使用[$PE(cx) : PE(cy) : PE(w) : PE(h)$]作为该提案的位置编码，其中$PE$和[:]的定义方式与TSP-FCOS相同。

更快的集合预测训练 TSP-RCNN同样采用集合预测损失进行训练。与TSP-FCOS不同，我们借鉴了Faster RCNN的真实标注分配规则，以加速TSP-RCNN的集合预测训练。具体而言，当且仅当提案框与真实标注框的交并比（IoU）得分大于0.5时，该提案框才会被分配给对应的真实标注对象。

5. 实验

5.1. 数据集与评估指标

我们在COCO[24]目标检测数据集上评估了我们的方法，该数据集包含80个目标类别。遵循常规做法[23,35]，我们使用trainval35k分割中的所有115k张图像进行训练，并使用minival分割中的所有5k张图像进行验证。测试结果通过将test-dev分割的结果提交至评估服务器获得。为与其他方法进行比较，我们主要关注平均精度（AP）——这是COCO使用的主要挑战指标，以及衡量计算开销的浮点运算次数（FLOPs）。

5.2. 实现细节

我们简要描述了我们实现的默认设置。更详细的设置可在附录中找到。

TSP-FCOS 遵循FCOS[35]的设计，分类子网络与辅助子网络均采用四层3×3卷积层，每层通道数为256，并应用了分组归一化[38]。在兴趣区域选择阶段，我们从兴趣区域分类器中选取得分最高的700个特征位置作为Transformer编码器的输入。

TSP-RCNN 与原始Faster RCNN不同，我们在P_3-P_7上应用了两个不共享参数的卷积子网络，作为RPN的分类和回归头，并采用了RetinaNet[23]风格的锚框生成方案。我们发现这种做法能以更少的计算开销提升RPN的性能。在候选区域选择阶段，我们从RPN中筛选出得分最高的700个特征。通过RoI Align操作[14]和全连接层，从候选区域中提取出提案特征。

Transformer编码器 由于TSP-FCOS和TSP-RNN仅包含Transformer编码器，而DETR同时具备Transformer编码器和解码器，为了在FLOPs上与DETR-DC5具有可比性，我们采用了宽度为512、8个注意力头的6层Transformer编码器。Transformer中前馈网络（FFN）的隐藏层大小设置为2048。训练过程中，我们随机丢弃70%的Transformer编码器输入以提升集合预测的鲁棒性。

训练 我们遵循Detectron2[39]的默认设置，采用36个周期（3×）的多尺度训练时增强调度方案。

5.3. 主要结果

表1展示了我们在COCO 2017验证集上的主要结果。我们将TSP-FCOS和TSP-RCNN与FCOS[35]、Faster RCNN[31]以及DETR[3]进行了比较。同时，我们还对比了同期关于改进DETR的研究工作：可变形DETR[47]和UP-DETR[10]。由表中可见，

Model	Backbone	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	FLOPs	FPS
FCOS†	ResNet-50	36	41.0	59.8	44.1	26.2	44.6	52.2	177G	17
Faster RCNN-FPN	ResNet-50	36	40.2	61.0	43.8	24.2	43.5	52.0	180G	19
Faster RCNN-FPN+	ResNet-50	108	42.0	62.1	45.5	26.6	45.4	53.4	180G	19
DETR+	ResNet-50	500	42.0	62.4	44.2	20.5	45.8	61.1	86G	21
DETR-DC5+	ResNet-50	500	43.3	63.1	45.9	22.5	47.3	61.1	187G	7
Deformable DETR*	ResNet-50	50	43.8	62.6	47.7	26.4	47.1	58.0	173G	-
UP-DETR	ResNet-50	300	42.8	63.0	45.3	20.8	47.1	**61.7**	86G	21
TSP-FCOS	ResNet-50	36	43.1	62.3	47.0	26.6	46.8	55.9	189G	15
TSP-RCNN	ResNet-50	36	43.8	63.3	48.3	28.6	46.9	55.7	188G	11
TSP-RCNN+	ResNet-50	96	**45.0**	**64.5**	**49.6**	**29.7**	**47.7**	58.0	188G	11
FCOS†	ResNet-101	36	42.5	61.3	45.9	26.0	46.5	53.6	243G	13
Faster RCNN-FPN	ResNet-101	36	42.0	62.5	45.9	25.2	45.6	54.6	246G	15
Faster RCNN-FPN+	ResNet-101	108	44.0	63.9	47.8	27.2	48.1	56.0	246G	15
DETR+	ResNet-101	500	43.5	63.8	46.4	21.9	48.0	61.8	152G	15
DETR-DC5+	ResNet-101	500	44.9	64.7	47.7	23.7	49.5	**62.3**	253G	6
TSP-FCOS	ResNet-101	36	44.4	63.8	48.2	27.7	48.6	57.3	255G	12
TSP-RCNN	ResNet-101	36	44.8	63.8	49.2	29.0	47.9	57.1	254G	9
TSP-RCNN+	ResNet-101	96	**46.5**	**66.0**	**51.2**	**29.9**	**49.7**	59.2	254G	9

表1. COCO 2017验证集上的评估结果。† 代表我们的复现结果。+表示模型采用随机裁剪增强和更长训练周期进行训练。我们使用Detectron2包测量FLOPs和FPS指标，并采用单块Nvidia GeForce RTX 2080 Ti GPU测算推理延迟。∗代表未使用迭代优化的版本。附录提供了TSP-RCNN与Deformable DETR在同等迭代优化条件下的公平对比。

Model	AP	AP_S	AP_M	AP_L
TSP-RCNN-R50	**43.8**	**28.6**	**46.9**	55.7
w/o set prediction loss	42.7	27.6	45.5	**56.2**
w/o positional encoding	43.4	28.4	46.3	55.0
TSP-RCNN-R101	**44.8**	**29.0**	**47.9**	**57.1**
w/o set prediction loss	44.0	27.6	47.2	**57.1**
w/o positional encoding	44.4	28.2	47.7	56.7

表2. 在COCO 2017验证集上关于集合预测损失与位置编码消融研究的评估结果。

Model	AP	AP_S	AP_M	AP_L
FCOS	45.3	28.1	49.0	59.3
TSP-FCOS	**46.1**	**28.5**	**49.7**	**60.2**
Faster-RCNN	44.1	26.4	47.6	58.1
TSP-RCNN	**45.8**	**29.4**	**49.2**	**58.4**

表3. 使用ResNet-101-DCN骨干网络在COCO 2017验证集上的评估结果。

可以看出，我们的TSP-FCOS和TSP-RCNN显著优于原始FCOS与Faster RCNN。此外，TSP-RCNN在整体性能和小物体检测方面优于TSP-FCOS，但在推理延迟方面稍逊一筹。

为了与最先进的DETR模型进行比较，我们采用了DETR[3]中类似的训练策略，即使用96个epoch的(8×)的训练计划并应用了随机裁剪增强。我们将增强版的TSP-RCNN记为TSP-RCNN+。同时，我们从[3]中复制了增强版Faster RCNN（*i.e*，即Faster RCNN+）的结果。通过比较这些模型，可以发现我们的TSP-RCNN以更短的训练周期取得了最先进的结果。我们还注意到，TSP-RCNN+在大物体检测上仍逊色于DETR-DC5+。我们认为这是由于DETR采用的编码器-解码器结构所带来的归纳偏置及其更长的训练周期所致。

5.4. 模型分析

在模型分析中，我们评估了在默认设置*i.e*下训练的多个模型，采用36周期（3×）的训练计划，且未使用随机裁剪增强。

5.4.1 消融研究

我们对集合预测损失和位置编码进行了消融研究，这两者是我们模型中的两个关键组成部分。表2展示了基于ResNet-50和ResNet-101骨干网络的TSP-RCNN消融实验结果。从表中可以看出，集合预测损失和位置编码对于TSP机制的成功都非常重要，而集合预测损失对TSP-RCNN性能提升的贡献大于位置编码。

Model	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet [23]	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2
FSAF [45]	ResNet-101	40.9	61.5	44.0	24.0	44.2	51.3
FCOS [35]	ResNet-101	41.5	60.7	45.0	24.4	44.8	51.6
MAL [19]	ResNet-101	43.6	62.8	47.1	25.0	46.9	55.8
RepPoints [40]	ResNet-101-DCN	45.0	66.1	49.0	26.6	48.6	57.5
ATSS [43]	ResNet-101	43.6	62.1	47.4	26.1	47.0	53.6
ATSS [43]	ResNet-101-DCN	46.3	64.7	50.4	27.7	49.8	58.4
Fitness NMS [36]	ResNet-101	41.8	60.9	44.9	21.5	45.0	57.5
Libra RCNN [27]	ResNet-101	41.1	62.1	44.7	23.4	43.7	52.5
Cascade RCNN [2]	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2
TridentNet [21]	ResNet-101-DCN	46.8	**67.6**	51.5	28.0	**51.2**	**60.5**
TSD [33]	ResNet-101	43.2	64.0	46.9	24.0	46.3	55.8
Dynamic RCNN [42]	ResNet-101	44.7	63.6	49.1	26.0	47.4	57.2
Dynamic RCNN [42]	ResNet-101-DCN	46.9	65.9	51.3	28.1	49.6	60.0
TSP-RCNN	ResNet-101	46.6	66.2	51.3	28.4	49.0	58.5
TSP-RCNN	ResNet-101-DCN	**47.4**	66.7	**51.9**	**29.0**	49.7	59.1

表4. 在COCO 2017测试集上与最先进模型的比较（单模型和单尺度结果）。下划线和加粗数字分别代表以ResNet-101和ResNet-101-DCN为骨干网络的最佳模型。

5.4.2 与可变形卷积的兼容性

人们可能会好奇Transformer编码器与可变形卷积[9,46]是否能够兼容，因为两者都能利用物体间的长程关系。在表3中，我们将TSP-FCOS和TSP-RCNN与采用可变形ResNet-101作为骨干网络的FCOS及Faster RCNN进行了对比。结果表明，TSP机制与可变形卷积具有很好的互补性。

5.5. 与最先进技术的比较

我们将TSP-RCNN与多种单阶段和两阶段目标检测模型[31, 36, 2, 33, 23, 45, 35, 4, 20, 40, 43]进行比较，这些模型同样采用ResNet-101主干网络或其可变形卷积网络(DCN)[46]变体，结果如表4所示。实验采用8×训练周期和随机裁剪数据增强策略，性能指标基于COCO 2017测试集的单模型单尺度检测结果进行评估。在两种主干网络配置下，我们的模型均实现了所有检测器中最高的平均精度(AP)得分。

6. 收敛性分析

在图6的上半部分，我们比较了更快的集合预测训练与DETR原始集合预测训练的收敛速度。可以看出，我们提出的快速训练技术持续加速了TSP-FCOS和TSP-RCNN的收敛过程。

我们还在图6的下半部分绘制了TSP-FCOS、TSP-RCNN和DETR-DC5的收敛曲线，从中可以发现，我们提出的模型不仅收敛速度更快，而且实现了更优的检测性能。

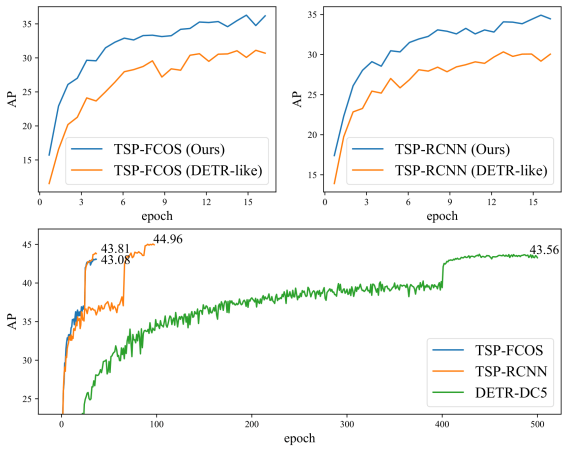

图6. 上方两幅图比较了我们提出的更快集合预测训练损失与类似DETR损失在TSP-FCOS和TSP-RCNN上的收敛速度。底部图展示了TSP-FCOS、TSP-RCNN和DETR-DC5的收敛曲线。

7. 结论

为了加速DETR的训练收敛并提升目标检测的预测能力，我们通过大量实验探究了其收敛缓慢的原因，并提出了两种创新解决方案——TSP-FCOS与TSP-RCNN。这两种方法大幅减少了训练时间，同时实现了最先进的检测性能。在未来的工作中，我们将探索如何成功运用稀疏注意力机制{v*}，直接建模多层级特征间的关联关系。

致谢

我们感谢评审们提出的宝贵意见。本工作部分由美国能源部通过布鲁克海文国家实验室根据合同PO 0000384608提供支持。

参考文献

[1] Jimmy Lei Ba、Jamie Ryan Kiros和Geoffrey E Hinton。层归一化。*arXiv preprint arXiv:1607.06450*，2016年。11 [2] Zhaowei Cai与Nuno Vasconcelos。Cascade R-CNN：深入高质量目标检测。载于*Proceedings of the IEEE conference on computer vision and pattern recogni-tion*，页码6154–6162，2018年。8, 13, 14 [3] Nicolas Carion、Francisco Massa、Gabriel Synnaeve、Nicolas Usunier、Alexander Kirillov及Sergey Zagoruyko。基于Transformers的端到端目标检测。*arXiv preprintarXiv:2005.12872*，2020年。1, 2, 3, 6, 7, 11, 12 [4] Yuntao Chen、Chenxia Han、Naiyan Wang与Zhaoxiang Zhang。重新审视单阶段目标检测中的特征对齐。*arXiv preprint arXiv:1908.01570*，2019年。8, 14 [5] Cheng Chi、Fangyun Wei和Han Hu。RelationNet++：通过Transformer解码器桥接目标检测的视觉表示。*Advances in Neural Information ProcessingSystems*，33，2020年。2 [6] Rewon Child、Scott Gray、Alec Radford及Ilya Sutskever。使用稀疏Transformers生成长序列。*arXiv preprint arXiv:1904.10509*，2019年。8 [7] Kyunghyun Cho、Bart Van Merriënboer、Caglar Gulcehre、Dzmitry Bahdanau、Fethi Bougares、Holger Schwenk与Yoshua Bengio。利用RNN编码器-解码器学习短语表示以进行统计机器翻译。*arXiv preprint arXiv:1406.1078*，2014年。11 [8] Gonçalo M Correia、Vlad Niculae及André FT Martins。自适应稀疏Transformers。*arXiv preprintarXiv:1909.00015*，2019年。8 [9] Jifeng Dai、Haozhi Qi、Yuwen Xiong、Yi Li、Guodong Zhang、Han Hu和Yichen Wei。可变形卷积网络。载于*Proceedings of the IEEE international conference on computer vision*，页码764–773，2017年。8 [10] Zhigang Dai、Bolun Cai、Yugeng Lin与Junying Chen。UP-DETR：基于Transformers的目标检测无监督预训练。*arXiv preprint arXiv:2011.09094*，2020年。2, 6 [11] Jacob Devlin、Ming-Wei Chang、Kenton Lee及Kristina Toutanova。BERT：面向语言理解的深度双向Transformer预训练。*arXiv preprintarXiv:1810.04805*，2018年。2, 3 [12] Ross Girshick。Fast R-CNN。载于*Proceedings of the IEEE international conference on computer vision*，页码1440–1448，2015年。2, 11 [13] Ross Girshick、Jeff Donahue、Trevor Darrell和Jitendra Malik。用于精确目标检测与语义分割的丰富特征层次结构。载于*Proceedings of the IEEE conference on computer vision and pattern recognition*，页码580–587，2014年。11 [14] 何恺明、Georgia Gkioxari、Piotr Dollár和Ross Girshick。Mask R-CNN。载于*Proceedings of the IEEE international conference on computer vision*，第2961–2969页，2017年。1, 2, 6, 12 [15] 何恺明、张翔宇、任少卿、孙剑。深度残差学习用于图像识别。载于*Proceedings of the IEEE conference on computer vision and pattern recognition*，第770–778页，2016年。11, 12 [16] 胡翰、顾家元、张政、代继峰、魏亦宸。用于目标检测的关系网络。载于*Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*，第3588–3597页，2018年。2 [17] Sergey Ioffe和Christian Szegedy。批量归一化：通过减少内部协变量偏移加速深度网络训练。*arXiv preprint arXiv:1502.03167*，2015年。3 [18] 姜子航、余炜浩、周大全、陈云鹏、冯佳时、颜水成。ConvBERT：利用基于跨度的动态卷积改进BERT。*Advances in NeuralInformation Processing Systems*，33卷，2020年。3 [19] 柯巍、张天良、黄泽毅、叶启翔、刘建柱、黄东。视觉目标检测的多锚点学习。载于*Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*，第10206–10215页，2020年。8, 13 [20] 孔涛、孙富春、刘华平、蒋宇宁、李磊、史建波。FoveaBox：超越基于锚点的目标检测。*IEEE Transactions on Image Processing*，29卷，第7389–7398页，2020年。8, 14 [21] 李阳皓、陈云涛、王乃岩、张兆翔。尺度感知的三叉戟网络用于目标检测。载于*Proceedings of the IEEE international conference on computer vision*，第6054–6063页，2019年。8, 13 [22] 林倞毅、Piotr Dollár、Ross Girshick、何恺明、Bharath Hariharan、Serge Belongie。特征金字塔网络用于目标检测。载于*Proceedings of the IEEE conference on computer vision and pattern recogni-tion*，第2117–2125页，2017年。1, 4, 5, 12 [23] 林倞毅、Priya Goyal、Ross Girshick、何恺明、Piotr Dollár。密集目标检测的焦点损失。载于*Proceedings of the IEEE international conference on computer vision*，第2980–2988页，2017年。1, 2, 6, 8, 12, 13, 14 [24] 林倞毅、Michael Maire、Serge Belongie、James Hays、Pietro Perona、Deva Ramanan、Piotr Dollár、C Lawrence Zitnick。Microsoft COCO：上下文中的常见物体。载于*European conference on computer vision*，第740–755页。Springer，2014年。1, 4, 6 [25] 刘伟、Dragomir Anguelov、Dumitru Erhan、Christian Szegedy、Scott Reed、傅城阳、Alexander C Berg。SSD：单次多框检测器。载于*European conference on computer vision*，第21–37页。Springer，2016年。1, 2 [26] Ilya Loshchilov和Frank Hutter。解耦权重衰减正则化。*arXiv preprint arXiv:1711.05101*，2017年。12 [27] 庞江淼、陈凯、石建平、冯华君、欧阳万里、林达华。Libra R-CNN：迈向均衡学习的物体检测。载于*Proceedings of the IEEE conference on computer vision and pattern recogni-tion*，第821–830页，2019年。8, 13

[28] 郑勤，李泽明，张昭宁，鲍一平，余刚，彭宇星，孙剑。ThunderNet：面向移动设备的实时通用目标检测。于*Proceed-
ings of the IEEE International Conference on Computer Vi- sion*，页码6718–6727，2019年。2 [29] Joseph Redmon，Santosh Divvala，Ross Girshick，Ali Farhadi。You Only Look Once：统一的实时目标检测。于*Proceedings of the IEEE conference on computer
vision and pattern recognition*，页码779–788，2016年。1, 2 [30] Joseph Redmon与Ali Farhadi。YOLO9000：更好、更快、更强。于*Proceedings of the IEEE conference on computer
vision and pattern recognition*，页码7263–7271，2017年。1 [31] 任少卿，何恺明，Ross Girshick，孙剑。Faster R-CNN：利用区域提议网络实现实时目标检测。于*Advances in neural information pro- cessing systems*，页码91–99，2015年。1, 2, 6, 8, 11, 13, 14 [32] Hamid Rezatofighi，Nathan Tsoi，JunYoung Gwak，Amir Sadeghian，Ian Reid，Silvio Savarese。广义交并比：边界框回归的度量与损失函数。于*Proceedings of the IEEE Conference on Com-
puter Vision and Pattern Recognition*，页码658–666，2019年。2, 12 [33] 宋广录，刘宇，王晓刚。重探目标检测中的兄弟头结构。于*Proceedings of
the IEEE/CVF Conference on Computer Vision and Pattern Recognition*，页码11563–11572，2020年。8, 13, 14 [34] Nitish Srivastava，Geoffrey Hinton，Alex Krizhevsky，Ilya Sutskever，Ruslan Salakhutdinov。Dropout：防止神经网络过拟合的简单方法。*The journal of machine learning research*，15(1):1929–1958，2014年。3 [35] 田志，沈春华，陈浩，何通。FCOS：全卷积单阶段目标检测。于*Proceed-
ings of the IEEE international conference on computer vi- sion*，页码9627–9636，2019年。1, 4, 5, 6, 8, 12, 13, 14 [36] Lachlan Tychsen-Smith与Lars Petersson。通过适应性NMS和有界IoU损失改进目标定位。于*Proceedings of the IEEE conference on computer vision and
pattern recognition*，页码6877–6885，2018年。8, 13, 14 [37] Ashish Vaswani等。注意力机制就是你所需要的一切。于*Advances in neural information processing systems*，页码5998–6008，2017年。1, 2, 5 [38] 吴育昕，何恺明。组归一化。于*Pro- ceedings of the European conference on computer vision (ECCV)*，页码3–19，2018年。6 [39] 吴育昕等。Detectron2，2019年。6, 12 [40] 杨泽，刘少辉，胡涵，王立伟，林史蒂芬。RepPoints：目标检测的点集表示方法。于*Proceedings of the IEEE International Conference
on Computer Vision*，页码9657–9666，2019年。8, 13, 14 [41] Manzil Zaheer等。Big Bird：面向长序列的Transformer模型。*Advances in Neu- ral Information Processing Systems*，33卷，2020年。8
[42] 张宏凯、常虹、马冰澎、王乃岩、陈熙霖。动态R-CNN：通过动态训练实现高质量目标检测。*arXiv preprint arXiv:2004.06002*，2020年。8, 13 [43] 张世峰、迟程、姚永强、雷震、李振。通过自适应训练样本选择弥合基于锚点与无锚点检测间的差距。载于*Proceedings of the IEEE/CVF Conference on Computer
Vision and Pattern Recognition*，第9759–9768页，2020年。1, 8, 13, 14 [44] 张晓松、万方、刘畅、纪荣荣、叶启祥。FreeAnchor：学习锚点匹配的视觉目标检测方法。载于*Advances in Neural Information Processing Systems*，第147–155页，2019年。2 [45] 朱晨晨、何一辉、Marios Savvides。面向单阶段目标检测的特征选择性无锚模块。载于*Proceedings of the IEEE Conference on Computer Vision
and Pattern Recognition*，第840–849页，2019年。8, 13, 14 [46] 朱喜洲、胡涵、林史蒂芬、代继峰。可变形卷积网络V2：更强的可变形性，更好的结果。载于*Proceedings of the IEEE Conference on Computer Vision
and Pattern Recognition*，第9308–9316页，2019年。8, 12, 14 [47] 朱喜洲、苏伟杰、陆乐威、李斌、王晓刚、代继峰。可变形DETR：端到端目标检测的可变形Transformer。*arXiv preprint arXiv:2010.04159*，2020年。2, 6, 13

A. 预备知识

A.1. Transformer与检测Transformer

由于本工作旨在改进DEtection TRansformer（DETR）模型[3]，为求完整，我们将更详细地描述其架构。

编码器-解码器框架DETR可在编码器-解码器框架中表述为[7]。DETR的编码器以CNN骨干网络处理后的特征作为输入，生成上下文表示；而DETR的非自回归解码器则以目标查询{v*}为输入，基于上下文条件生成检测结果。

多头注意力 DETR中使用了两种类型的多头注意力机制：多头自注意力和多头交叉注意力。通用的注意力机制可以表述为利用查询向量Q和键向量K对值向量V进行加权求和：

$$\text{Attention}(Q,K,V)=\text{softmax}\left(\frac{QK^T}{\sqrt{d_{\text{model}}}}\right)\cdot V, \quad (4)$$

其中d_{model}代表隐藏表示的维度。在自注意力机制中，Q、K和V是前一层的隐藏表示。对于交叉注意力，Q指代前一层的隐藏表示，而K和V则是来自编码器的上下文向量。注意力机制的多头变体允许模型同时关注来自不同表示子空间的信息，其定义为：

$$\text{Multi-head}(Q,K,V)=\text{Concat}(\text{head}_1,\ldots,\text{head}_H)W^O,$$
$$\text{head}_h=\text{Attention}((P_Q+Q)W_h^Q,(P_K+K)W_h^K,VW_h^V),$$

其中$W_h^Q,W_h^K\in\mathbb{R}^{d_{\text{model}}\times d_k}$、$W_h^V\in\mathbb{R}^{d_{\text{model}}\times d_v}$和$W_h^O\in\mathbb{R}^{Hd_v\times d_{\text{model}}}$是投影矩阵，$H$表示注意力头的数量，$d_k$和$d_v$分别是每个头的查询/键和值的隐藏大小，而$P_Q$和$P_K$则是位置编码。

前馈网络 位置式前馈网络FFN）在编码器和解码器的多头注意力机制之后应用。它包含一个带有ReLU激活的双层线性变换：

$$\text{FFN}(x)=\max(0,xW_1+b_1)W_2+b_2, \quad (5)$$

其中$W_1\in\mathbb{R}^{d_{\text{model}}\times d_{\text{FFN}}}$、$W_2\in\mathbb{R}^{d_{\text{FFN}}\times d_{\text{model}}}$、$b_1\in\mathbb{R}^{d_{\text{FFN}}}$、$b_2\in\mathbb{R}^{d_{\text{model}}}$和$d_{\text{FFN}}$代表FFN的隐藏层大小。

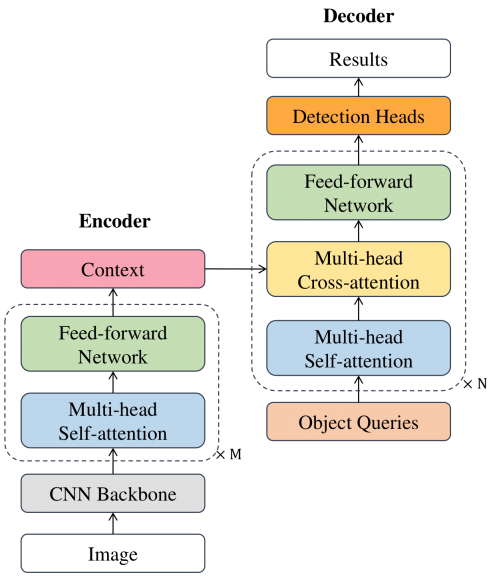

图7. DETR架构的详细图解。残差连接与层归一化部分已省略。

多头注意力机制与前馈网络交替堆叠，构成了编码器和解码器，其间包含残差连接[15]和层归一化[1]。图7详细展示了DETR架构的示意图。

A.2. Faster R-CNN

Faster R-CNN [31] 是一种基于R-CNN [13]和Fast R-CNN [12]先前工作开发的两阶段目标检测模型。通过引入区域提议网络（RPN），Faster R-CNN显著提升了两阶段目标检测的精度与效率。

区域提议网络 Faster R-CNN的第一个模块是一个深度全卷积网络，称为区域提议网络（RPN），用于生成感兴趣区域（RoIs）。RPN以图像的特征图作为输入，输出一组带有物体性得分的矩形物体提议。RPN包含一个共享的3×3卷积层和两个并行的1×1卷积层，分别用于回归和分类。在每个滑动窗口位置，RPN生成k个提议。这些提议相对于称为锚点的k个参考框进行参数化。在Fast R-CNN中，使用了3种尺度和3种长宽比的锚点，因此每个滑动窗口有k =9个锚点。对于每个锚点，回归头输出4个坐标参数$\{t_x,t_y,t_w,t_h\}$，用于编码边界框的位置和大小；分类头输出2个得分$\{p_{\text{pos}},p_{\text{neg}}\}$，用于估计框内存在物体的概率。

Fast R-CNN　第二部分是Fast R-CNN检测器，它利用RPN生成的每个候选区域来优化检测结果。为了减少冗余，会对候选区域应用非极大值抑制（NMS），只有排名靠前的候选区域才会被Fast R-CNN采用。接着，通过RoI池化或RoI对齐[14]技术，从主干特征图中提取给定候选区域的特征，确保每个候选区域输入到Fast R-CNN检测器时具有固定的空间尺寸。在此阶段，Fast R-CNN输出边界框回归参数和分类分数，以进一步精炼区域提议。同样地，需要通过NMS来消除检测结果中的重复项。

A.3. FCOS

全卷积单阶段目标检测（FCOS）[35]是近期提出的一种无锚点、逐像素检测框架，它在一阶段目标检测任务中实现了最先进的性能。

逐像素预测　与基于锚框的目标检测器不同，FCOS以逐像素预测的方式构建任务，即在特征图的每个位置上直接回归目标边界框，无需依赖预定义的锚框。若特征图某位置对应输入图像中的点落入任一真实框内，则该位置被视为正样本。当某位置同时落入多个真实框重叠区域时，将选择面积最小的真实框作为回归目标。实验表明，结合多层级预测和FPN[22]结构时，这种模糊性不会影响整体检测性能。

网络输出　在FCOS中，主干网络提取的特征图后接两个分支。第一个分支包含4个卷积层和两个并列输出层，分别生成C类别的分类分数和"中心度"分数。中心度表示该位置到其负责目标中心的归一化距离，其值域为[0,1]，并通过二元交叉熵损失进行训练。在测试阶段，中心度会与分类分数相乘，使得远离目标中心的低质量边界框在非极大值抑制(NMS)中获得较低权重。第二个分支包含4个卷积层和一个边界框回归层，输出该位置到边界框四边的距离。该预测头在多个特征层级间共享。

B. 详细实验设置

我们提供了关于实现默认设置的更多细节。

主干网络 我们采用ResNet-50和ResNet-101[15]作为主干网络，并在其上构建了特征金字塔网络[22]利用ResNet中的$\{C_3, C_4, C_5\}$特征图生成特征金字塔$\{P_3, P_4, P_5, P_6, P_7\}$。若指定使用DCN（可变形卷积网络），我们会在ResNet的最后三个阶段采用Deformable ConvNets v2[46]。所有特征图均具有256个通道。

数据增强 我们遵循Detectron2[39]的默认设置进行数据增强。具体而言，采用尺度增强方法调整输入图像尺寸，使其短边长度在{640, 672, 704, 736, 768, 800}范围内，长边不超过1333像素。除尺度增强外，我们还对训练图像进行随机水平翻转。

损失函数 我们采用提出的快速集合预测训练损失进行分类任务，回归任务则结合L1损失与广义IoU损失[32]。对于TSP-FCOS和TSP-RCNN，分类任务中正负样本的加权均使用焦点损失[23]。与DETR[3]不同，我们未在每层编码器后添加辅助损失。实验表明，这种端到端的方案能提升模型性能。

优化 我们采用AdamW[26]优化Transformer组件，其他部分则使用动量0.9的SGD进行优化。在36周期（3×）训练计划中，检测器以16的批量大小训练2.7××10^5次迭代。初始学习率设为AdamW 10^{-4}、SGD 10^{-2}，并在1.8××10^5和2.4××10^5次迭代时均乘以0.1进行衰减。前1000次迭代采用线性学习率预热。权重衰减设置为1$0^{-4}$。针对Transformer部分实施梯度裁剪，最大L_2梯度范数为0.1。

更长的训练周期　在论文中，我们还采用96周期（8×）的训练计划。该96周期（8×）计划将从36周期（3×）计划的模型检查点恢复，具体在第24^{th}周期（即1.8×10^5次迭代）处继续训练72周期（即5.4×10^5次迭代）。学习率在4.8×10^5次迭代和6.4×10^5次迭代时分别乘以0.1。在8×计划中，我们将进一步应用随机裁剪增强。我们遵循DETR[3]中的增强策略，即以0.5的概率对训练图像进行随机矩形裁剪，随后将其重新调整至800-1333的尺寸范围。

C. 仅编码器DETR的更多细节

我们仅含编码器的DETR同样采用匈牙利损失进行集合预测训练，但边界框回归过程略有不同。在原版DETR中，边界框回归是无参考的，直接预测归一化的中心坐标$(cx, cy) \in [0, 1]^2$及高度

Model	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN [31]	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2
Fitness NMS [36]	ResNet-101	41.8	60.9	44.9	21.5	45.0	57.5
Libra RCNN [27]	ResNet-101	41.1	62.1	44.7	23.4	43.7	52.5
Cascade RCNN [2]	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2
TridentNet [21]	ResNet-101-DCN	46.8	67.6	51.5	28.0	51.2	60.5
TSD [33]	ResNet-101	43.2	64.0	46.9	24.0	46.3	55.8
Dynamic RCNN [42]	ResNet-101	44.7	63.6	49.1	26.0	47.4	57.2
Dynamic RCNN [42]	ResNet-101-DCN	46.9	65.9	51.3	28.1	49.6	60.0
RetinaNet [23]	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2
FSAF [45]	ResNet-101	40.9	61.5	44.0	24.0	44.2	51.3
FCOS [35]	ResNet-101	41.5	60.7	45.0	24.4	44.8	51.6
MAL [19]	ResNet-101	43.6	62.8	47.1	25.0	46.9	55.8
RepPoints [40]	ResNet-101-DCN	45.0	66.1	49.0	26.6	48.6	57.5
ATSS [43]	ResNet-101	43.6	62.1	47.4	26.1	47.0	53.6
ATSS [43]	ResNet-101-DCN	46.3	64.7	50.4	**27.7**	**49.8**	58.4
TSP-FCOS	ResNet-101	46.1	65.8	50.3	27.3	49.0	58.2
TSP-FCOS	ResNet-101-DCN	**46.8**	**66.4**	**51.0**	27.6	49.5	**59.0**

表5. 在COCO 2017测试集上比较TSP-FCOS与最先进模型（单模型和单尺度结果）。下划线和加粗数字分别代表使用ResNet-101和ResNet-101-DCN骨干网络的最佳单阶段模型。

Model	AP	AP_S	AP_M	AP_L	FLOPs	#Params
FCOS	41.0	26.2	44.6	52.2	177G	36.4M
FCOS-larger	41.5	26.0	45.2	52.3	199G	37.6M
TSP-FCOS	**43.1**	**26.6**	**46.8**	**55.9**	189G	51.5M
Faster RCNN	40.2	24.2	43.5	52.0	180G	41.7M
Faster RCNN-larger	40.9	24.4	44.1	54.1	200G	65.3M
TSP-RCNN	**43.8**	**28.6**	**46.9**	**55.7**	188G	63.6M

表6. 不同FLOPs下模型在COCO 2017验证集上的评估结果，采用ResNet-50作为骨干网络。

Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Deformable DETR	43.8	62.6	47.7	26.4	47.1	58.0
+ iterative refinement	45.4	64.7	49.0	26.8	48.3	61.7
++ two-stage*	**46.2**	**65.2**	**50.0**	28.8	**49.2**	**61.7**
TSP-RCNN	44.4	63.7	49.0	29.0	47.0	56.7
+ iterative refinement	45.4	63.1	49.6	**29.5**	48.5	58.7

表7. TSP-RCNN与带迭代优化的Deformable DETR在COCO 2017验证集上的评估结果。所有模型均以50个训练周期和32的批量大小进行训练。* 关于两阶段Deformable DETR的定义，请参阅原版Deformable DETR论文。

以及框相对于输入图像的宽度（w, h）$\in [0, 1]^2$。在仅编码器的DETR中，由于每个预测都基于Transformer编码器输出的特征点，我们将使用特征点坐标（x_r, y_r）作为参考点

回归的：

$$cx = \sigma(b_1 + \sigma^{-1}(x_r)), cy = \sigma(b_2 + \sigma^{-1}(y_r))$$

其中$\{b_1, b_2\}$来自回归预测的输出。

D. TSP-RCNN与带迭代优化的可变形DETR对比

受可变形DETR[47]启发，我们对TSP-RCNN进行了级联式预测框迭代优化实验[2]。此处采用简化的两级级联方案：全连接检测头的维度从12544-1024-1024缩减为12544-512，Transformer前馈网络从512-2048-512调整为512-1024-512，以保持与原始模型相近的参数量和计算量(FLOPs)。为公平对比，我们同样遵循可变形DETR的实验设置，使用批量大小32进行50个训练周期。

表7展示了TSP-RCNN与可变形DETR在迭代优化后的结果对比。从数据可见，在未进行迭代优化时，采用相同训练配置的TSP-RCNN表现优于可变形DETR。经过迭代优化后，TSP-RCNN的性能可提升1个AP点。值得注意的是，当两者均采用迭代优化时，TSP-RCNN的表现略逊于可变形DETR。我们认为这是由于可变形DETR采用了D =6次解码器优化迭代，而我们仅实验了两次优化迭代。如何高效整合多轮

将细化迭代整合到TSP-RCNN模型中留作未来工作。

E. 相似FLOPs下的比较

与原始的FCOS和Faster RCNN相比，我们的TSP-FCOS和TSP-RCNN引入了一个额外的Transformer编码器模块。因此，自然会质疑这些改进是否源于更多的计算量和参数。表6通过为基线模型应用更强的基准模型回答了这一问题。对于Faster RCNN，我们首先在P_3-P_7上应用两个非共享卷积层作为更强的RPN，然后将原始的12544-1024-1024全连接(fc)检测头改为12544-2048-2048-2048。这使得Faster RCNN模型达到约200 GFLOPs的计算量和65.3M参数。对于FCOS，我们评估了一个约199 GFLOPs的FCOS模型，其中在分类和回归头部分别增加了一个卷积层。从表6可以看出，虽然为基线模型增加计算量和参数能略微提升性能，但这种改进远不如我们的TSP机制显著。

F. TSP-FCOS与最先进技术的比较

为了全面性，我们还将提出的TSP-FCOS模型与其他采用ResNet-101主干网络或其可变形卷积网络（DCN）[46]变体的先进检测模型[31, 36, 2, 33, 23, 45, 35, 4, 20, 40, 43]在表5中进行比较。实验采用8×训练周期和随机裁剪增强策略，性能指标基于COCO 2017测试集的单模型单尺度检测结果进行评估。可以看出，TSP-FCOS在单阶段检测器中以AP分数衡量达到了最先进水平。但对比主论文表4与表5可发现，TSP-FCOS性能略逊于我们提出的TSP-RCNN模型。

G. 特征位置与提议数量消融研究

对于TSP-FCOS，我们在FoI选择阶段从FoI分类器中选取得分最高的700个特征位置作为Transformer编码器的输入；而对于TSP-RCNN，则在RoI选择时从RPN中筛选出得分最高的700个候选框。不过，实验中采用的特征位置与候选框数量并非最优值。我们在表8中针对这一点进行了消融实验，结果表明：即使仅使用半数特征位置，我们的模型仍能保持较高的预测准确率。

表8. R-50 TSP-RCNN提案数量与R-50 TSP-FCOS特征位置数量在验证集上的消融实验结果。

Num. of Proposals	100	300	500	700
TSP-RCNN	40.3	43.3	43.7	43.8
TSP-FCOS	40.0	42.5	42.9	43.1

H. 定性分析

我们在图8中对TSP-RCNN进行了多幅图像的定性分析。选取了一个特定的Transformer注意力头进行分析。所有框均为RPN预测的RoI框，其中虚线框表示与同色实线框对应的前5个关注框。可见Transformer编码器能有效捕捉指向同一实例的RoI框，从而有助于减少预测冗余。

图8. TSP-RCNN在验证集六张图像上的定性分析。所有框均为RPN预测的RoI框，其中虚线框为同色实线框对应的前5个关注框。