arXiv:2611.30304v1 [cs.CV] 23 Nov 2026

PinGround：将查询固定到点以实现视觉语言模型中的并行定位

Shuxuan Li[1,2] **Jialiang Kang**[2] **YanghaoZhou**[2]**Ziye Zhu**[2,3]
Zhilin Zhao[1*] **Xiao Yang**[2*‡]中山大学 [2]CanvaResearch [3]
PekingUniversity

摘要

视觉语言模型（视觉语言模型）能够通过文本接口进行定位，凭借强大的语义理解能力，对开放词汇对象、复杂指代表达式、文本行和布局区域进行定位。它们将边界框逐个写入同一个序列中，尽管查询的实例与句子中的词语不同，构成的是无序集合，不存在序列依赖。因此，解码步骤的数量会随实例数量增长，使密集场景的处理成本最高。并行解码实例可以避免这种增长，但彼此不可见的解码流会重复检测某些实例，并漏检其他实例。我们提出 *PinGround*，为每个查询写入一个点作为空间锚点，并要求在局部半径内寻找距离该点最近的匹配实例，否则返回 *None*。给定图像后，锚定查询彼此独立，因此可以作为前缀树的叶节点并行解码；该树共享图像和查询，模型也在同一棵树上进行训练。由粗到细的锚点仅细化发现实例的区域，重复抑制确保每个实例只保留一个边界框，集合级强化学习则优化集合 *F1*。改变的只有提示和注意力掩码，架构本身不变。在涵盖检测、光学字符识别、布局和指代的 *12* 个基准测试上，*PinGround* 在三种不同的视觉语言模型中，相比使用相同数据训练的顺序解码，最多低 *5.5* 个 *F1*值；与此同时，其加速比会随实例密度增加，在密集场景中达到 *21.9*×

1. 引言

计算机视觉正从特定任务模型转向统一模型，通过单一接口解决多种任务 [7, 26, 35, 38, 42]。视觉语言模型（VLM）正将这一趋势带入定位领域。给定一张图像和一个文本查询，单个 VLM 即可返回物体的边界框、文本行、布局区域或自由形式表达式所指向的对象，并将每个坐标表示为普通数字

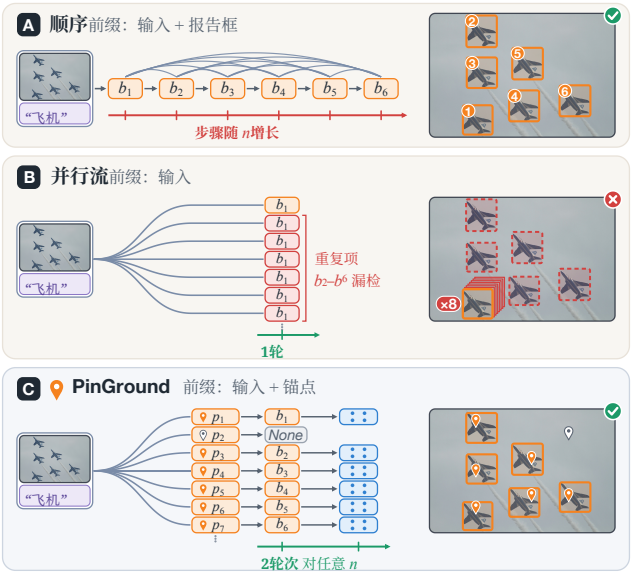


图1。在 LocateAnything-3B 上解码某个查询的所有实例时的实际输出。(A) 顺序解码逐一列出边界框，因此其步骤数会随 n 增长。(B) 输入相同的并行流会重复输出一个飞机，并漏检其余目标。(C) PinGround 为每个流提供一个锚点，并且对任意 n都只需两轮。

文本 [1, 5, 36] 或专用坐标词元 [6, 17, 29]。Grounding DINO [24]等开放词汇检测器也接受文本提示词，并基于固定的一组目标查询并行预测所有边界框；但它们使用相对浅层的文本编码器（如 BERT [11],）对提示进行编码。这类编码器适用于类别名称和短语，却不适合复杂描述 [16, 17]。VLM 使用完整的语言模型读取查询，因此很适合作为通用定位器。

视觉语言模型的语言模型通过下一词元预测生成答案，因此边界框会在单个序列中一个接一个地输出（图 1A）。这种序列形式源自语言接口，而不是定位任务本身的要求。与句子中的词语不同，查询中的实例不存在序列依赖。它们的任何排列

*通讯作者。

都是同样正确的答案，并且每个实例都可以仅根据图像和查询进行定位。然而，解码步骤的数量，即语言模型的序列前向传播次数，仍会随着实例数量增加而增加。Rex-Omni 报告称，生成时间大致随预测框的数量线性增长，对于包含超过 400 个对象的图像，生成时间会超过 16 秒 [17], 因此，人群、航拍图像、文档和场景文字等密集场景需要最长的解码时间。现有的加速方法使每个实例的处理成本降低，但仍保留了列表形式。Rex-Omni 将每个坐标写成一个专用词元，而不是由数字组成的字符串 [17], LocateAnything 通过多令牌预测，在一个步骤中预测边界框的四个坐标 [37], 而 GroundAnything 则通过掩码扩散并行解码固定长度的输出块 [44], 但在这三种方法中，前向传播次数仍会随着实例数量增加而增加。

这表明，应并行解码查询对应的所有实例。然而，在顺序解码中，先前输出的边界框会显示哪些实例已经被报告。LocateAnything 发现，看到这些边界框能让模型避免输出重复框和漏检；而在训练期间直接并行化输出，则可能扰乱这一因果过程 [37]。仅能看到图像和查询的并行流无法获取这一信号，因此会重复输出某些实例，并漏掉其他实例（图 1B）。因此，必须告知并行流要报告哪个实例。查询对应的实例位置各不相同，所以我们用位置来指定实例：将一个点作为锚点写入查询，并要求输出中心距该点最近的匹配实例的边界框；如果锚点所给定半径范围内没有这样的中心，则输出 无（图 1C）。这样，每个锚点都会覆盖一个局部邻域，同时关注整幅图像。给定图像后，各个锚定查询彼此独立，可以并行解码。

锚点分布在粗网格和细网格上，仅对粗粒度锚点发现实例的单元格进行细化，从而使计算适应场景的密度。相邻锚点以及不同查询仍可能报告同一实例，重复抑制则为每个实例保留一个边界框。并行解码依赖于每个锚点坚守各自邻域，从而使各锚点共同尽可能少地漏检和重复实例。已发布的视觉语言模型（如 Qwen3-VL-4B）即使在提示中说明了该规则，也不会遵守它。因此，我们用 47k 个（图像，查询）对进行轻量后训练来适配该视觉语言模型。训练与解码共享前缀树版面：图像为根节点，每个查询为分支，每个锚点为叶节点，仅保存其点及其答案。注意力掩码将每个叶节点限制在其自身路径上，而图像和每个查询只需为其所有锚点编码一次。在该版面上进行的监督微调对每个锚点单独评分，而输出则作为集合进行评估。训练因此以集合级强化学习（强化学习）结束，它对解码器的完整输出进行采样，以集合 F1 对每个输出评分，并按其贡献为每个锚点记得分。

我们将 PinGround 应用于三种不同的视觉语言模型：LocateAnything-3B[37], Qwen3-VL-4B[1] 和 GroundAnything-VLM [44], ，并在涵盖目标检测、光学字符识别、版面分析和指代任务的 12 个基准测试上进行评估。我们将每个基础模型与其官方解码器，以及使用相同数据微调的顺序基线进行比较。在 12 个基准测试上的平均结果中，且各解码器均采用默认模式时，PinGround 在 LocateAnything 上比该基线低 1.3 个 F1 值，在 Qwen3-VL 上高 4.3 个 F1 值，在 GroundAnything 上则低 1.0 个 F1 值。其加速比随实例密度的增加而提升：在 12 个基准测试上的平均加速比分别为 3.1、13.1 和 8.7 倍，而在密集检测基准测试 Dense200 上则分别达到 9.3、15.2 和 21.9 倍。我们的贡献如下。

•我们提出 PinGround，这是一种分层空间锚点策略，可将视觉语言模型定位转化为彼此独立、可并行解码的查询，且无需改动架构。

• 我们通过轻量级后训练，使视觉语言模型适配这一策略：训练和解码共用一棵前缀树，并以集合级强化学习收尾。

• 在三个视觉语言模型和 12 个基准测试上，PinGround 达到与顺序解码相当的 F1值，并且在密集场景中速度提升一个数量级。

2. 相关工作

基于视觉语言模型的定位。 Pix2Seq 将检测视为生成坐标标记序列的任务，并已指出，难以确定物体的排列顺序以及序列的结束位置 [6]。如今，这种语言接口已涵盖指代表达、视觉定位对话和统一视觉任务 [5, 29,38, 41–43], ; Qwen3-VL 等通用视觉语言模型则会在一个回答中列出许多类别和实例 [1, 2, 36]。在检测、光学字符识别、版面和指代数据上进行大规模训练的视觉语言模型定位系统，如 Rex-Omni [17]，（我们采用其评估套件）、LocateAnything [37]和 GroundAnything [44],, 能够在一个模型中处理所有这些任务。为缩短不断增长的回答的解码时间，LocateAnything 在一步中预测边界框的四个坐标，而 GroundAnything 则通过掩码扩散并行解码词元块。与大语言模型解码的通用加速方法类似 [3, 12, 20, 32],, 两者都降低了每个实例的处理成本，但实例仍需在同一列表中依次处理。PinGround 则改为调整查询，使解码步骤数不再随实例数增加。

用于定位的并行查询。 专用检测器并行回归所有边界框，正如 DETR 从一组学习型查询出发，通过一对一匹配实现的那样 [4]。Anchor DETR将每个查询与一个锚点关联起来 [39], , 而 GLIP、OWL-ViT 和 Grounding DINO 等开放集检测器



图 2。PinGround 概览。(1) 一个锚点会请求半径 r 内距离其最近的匹配实例，若不存在则为无。(2) 锚定查询共享一棵前缀树，其中每个叶节点仅关注自身的路径，因此一个轮次中的所有锚点都能并行解码。(3) 粗轮次10×10 找到占用单元格，细轮次对其进行细化，非极大值抑制合并两个轮次的结果。后训练通过对每个叶节点进行监督微调，以及通过强化学习根据每个锚点带来的集合 F1变化给予得分，来学习锚点规则。

使用文本编码器读取提示 [21, 24, 28]。GiT 和 UFO 将这种并行性扩展到文本输出，而 UFO 将其应用于预训练视觉语言模型 [33, 34]。固定网格中的每个点都会接收局部图像特征作为额外输入词元，并在该点输出对象的类别和边界框，或将该点标记为空。因此，每个点都在询问该处有什么，这种方式可根据提示中给出的类别列表进行检测和分割；而指代表达式仍按序解码为单个边界框。PinGround 则询问目标位于每个点附近的何处，并将目标和锚点坐标写入视觉语言模型的常规提示中，因此，同一评估方案可在不同视觉语言模型上并行解码开放词汇检测、光学字符识别、版面分析和多实例指代。PET 使用稀疏网格上的点查询定位人群中的人，并在学习型分割器判定某个区域较密集时，将一个查询拆分为四个更精细的查询 [23]，正如 PinGround 在找到实例时会细化粗粒度锚点。

3. 方法

3.1. 锚定查询和前缀树

给定一幅图像 I 和一个文本查询 c，它可以是类别名称、文本或布局类别，也可以是指代表达式；定位任务要求找出与 c匹配的所有实例所对应的边界框集合 $\mathcal{B} = \{b_1, \ldots, b_n\}$。每个边界框 $b = (x_1,\ y_1,\ x_2,\ y_2)$由其左上角和右下角坐标给出，并根据图像尺寸归一化到 0 到 1000 的范围。顺序式视觉语言模型以列表形式生成 $\mathcal{B}$，因此其解码步骤数随 n增长。

锚定查询。 PinGround 将列表替换为一组锚定查询。锚点是写入查询中的一个点 p；其答案是匹配实例的边界框，该实例的中心距 p 最近且位于半径 r范围内，否则答案为无。令 ctr(·) 表示边界框的中心，$j^\star = \operatorname{argmin}_j \|\mathrm{ctr}(b_j) - p\|$ 表示最近实例，则答案为

$$y(p) = \begin{cases} b_{j^\star} & \text{if } \|\mathrm{ctr}(b_{j^\star}) - p\| \le r, \\ \texttt{None} & \text{otherwise.} \end{cases} \tag{1}$$

每个答案只取决于图像、查询和锚点，因此锚定查询彼此独立，可以并行解码；同一规则也适用于每个锚点，无需考虑实例的顺序或进行匹配步骤。所有锚点共同划分图像，各自负责附近的实例。在归一化坐标中，我们采用两层锚点：间距为 100 的粗粒度 10×10 网格中心点，以及间距为 50 的细粒度 20×20网格中心点。将每一层的半径设为相应的间距后，相邻锚点便会彼此重叠，使靠近单元格边界的实例仍处于覆盖范围内。

前缀树。 一幅图像的锚定查询共享该图像和大部分提示，仅在查询和点上有所不同。因此，我们将它们排列成一棵前缀树（图2）。根节点包含所有查询共有的图像和提示部分。每个查询都是一条分支，其中包含提示的其余部分、 c以及响应的起始词元；对于该查询的所有锚点，这些内容都相同。每个锚点

都是一个叶节点，仅包含其点和答案，因为点是唯一依赖于锚点的输入。注意力掩码使每个词元只能关注从根节点到自身的路径，并且一条分支的所有叶节点都从同一个位置索引开始。因此，每个锚点所看到的上下文，恰好与单独询问它时所看到的相同，而图像和查询则为其所有锚点编码一次。训练使用相同的树结构：每一行包含一个（图像、查询）对，其采样锚点作为同一掩码和位置索引下的叶节点，因此训练和解码计算的是同一个函数。

3.2. 由粗到细解码

每个锚点层级需要进行一轮解码。在对根节点进行一次预填充后，第一轮会对每个查询分支各计算一次，同时解码所有查询共享这些分支的粗粒度锚点。随后，每个锚点输出不为无的粗粒度单元格都会细化为四个细粒度锚点，并在第二轮中基于相同的根节点和分支进行解码。细层级之所以必要，是因为每个锚点最多只能报告一个实例，因此仅靠粗粒度锚点，每个单元格最多只能检出一个实例；对于文本行、人群以及航拍图像中的物体等较小或密集排列的实例，这远远不够。只细化占用单元格，既能在发现实例的区域提高这一上限，又能让空区域的计算开销保持较低，因此计算量会随场景密度自适应变化。每轮的所有锚点都会以基础模型的原生输出格式一并解码，因此顺序前向传播次数不会随实例数量增加。

两个层级的边界框都会被汇集到池中，同一个实例可能在池中出现多次，这是因为相邻锚点会发生重叠，细化单元格的细粒度锚点会再次报告其粗粒度锚点找到的实例，而且模型对公式（1）中规则的遵循也只是近似的。因此，我们在每个查询内应用非极大值抑制（NMS），保留每个实例中置信度最高的边界框，其置信度根据答案的词元概率计算；然后再对图像中的各个查询应用非极大值抑制，因为汽车和厢式货车等不同查询可能返回同一个对象。这两个步骤都使用主要评估指标的 IoU阈值 0。5 ，且不涉及任何学习组件。

3.3. 监督式后训练

PinGround 的并行解码器依赖每个锚点都遵循公式（1）中的规则，而定位本身并不要求这一点。即使直接应用该规则，且遵循
提示，发布的基础模型并不会以这种方式作答，而解码器在 Dense200 上只能达到 3.6 到 48.0 的 F1@0.5，相比之下，后训练后可达到 63.7 到 83.7。因此，我们对视觉语言模型进行后训练以学习该规则：首先进行有监督微调，然后采用第3.4节中的集合级强化学习。

每个训练对由一张图像、一个查询及其

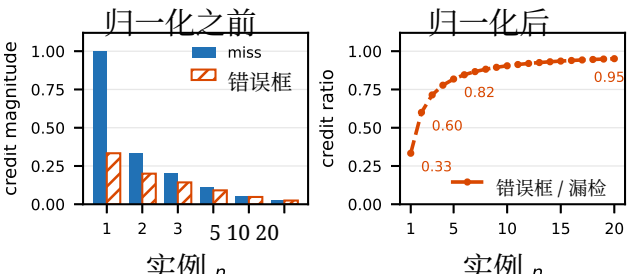


图3。漏检和错误框的得分与实例数量 n 的关系。左：归一化前，随着 n 增大，两种得分都会趋近于零。右：对最大得分为漏检的一对进行归一化后，漏检的得分为 1，而错误框（$2n-1$）/（$2n+1$）的得分介于 $1/3$ 和 1 之间。

标注框组成，而式（1）直接根据标注为两个层级的每个锚点赋予标签。对于每个图像-查询对，我们从两个层级中均匀抽取最多四个带有目标的锚点，并且在每个层级抽取与带有目标的无 锚点数量相同的锚点。粗粒度无 锚点从所有不带目标的粗粒度锚点中抽取，因为解码器会查询整个粗网格；这些锚点覆盖空白背景以及其他物体附近的区域。细粒度无 锚点仅从解码器可能细化的单元格中抽取；这些单元格的粗粒度锚点半径范围内有一个带标注的实例，因此它们位于同一查询或其他查询的实例附近。查询未出现在图像中的图像-查询对会贡献八个粗粒度无 锚点。

给定图像 I、查询 c和锚点 p，模型经过训练后会生成答案 $y(p)$ （式（1）中的 $b_{j^\star}$ 或无，以基础模型的原生输出格式将四个坐标表示为词元，并使用词元级交叉熵

$$\mathcal{L} = -\sum_{p}\sum_{t} \log \pi_\theta\big(y_t(p) \mid I, c, p, y_{<t}(p)\big), \qquad (2)$$

其中，$y_t(p)$ 是 $y(p)$ 的第 t 个词元，求和范围为前缀树中采样的锚点。为保留基础模型的原生接口并缓解灾难性遗忘，训练混合数据还会以原生列表格式重放相同的样本对。

3.4. 集合级强化学习

监督式后训练会独立评判每个锚点答案，而集合 F1 会评判某个查询的所有锚点答案合并后的结果，因此逐锚点损失会以三种方式错误衡量错误代价。首先，多个锚点会在看不到彼此边界框的情况下报告同一个实例；如果一个不精确的重复框与保留框的交并比低于 0.5，且整体上未被抑制，它就会成为假框。因此，每个查询有数百个锚点时，即使这类答案的比例很低，也会增加数个假框。其次，错误的粗粒度空值对其锚点而言只是一个错误词元，但它会剪除对应单元格，

表 1. 锚点答案在匹配输出集中的结果对应的得分，其中 δ_{FP} 和 δ_{FN} 分别表示图像中有 n 个实例时，一个错误框和一次漏检导致的 F1值下降。边界框行适用于两个层级的锚点。

答案	匹配集中的结果	得分
box	与其目标匹配	$+\delta_{\mathrm{FN}}$
	与另一个锚点的目标匹配	0
	因抑制而被移除	0
	错误框	$-\delta_{\mathrm{FP}}$
	错误框，目标漏检	$-\delta_{\mathrm{FP}}-\delta_{\mathrm{FN}}$
细粒度空值	无目标	$+\delta_{\mathrm{FP}}$
	目标漏检	$-\delta_{\mathrm{FN}}$
	由另一个锚点找到的目标	0
粗粒度空值	k 其单元格中的实例漏检	$-\min(k,3)\ \delta_{\mathrm{FN}}$

并从输出中移除其中的所有实例。第三，错误代价取决于图像中的实例数量 n。设 TP 为真框数、FP 为假框数、FN 为漏检实例数，则 F1 $=2\mathrm{TP}/(2\ \mathrm{TP}+\mathrm{FP}+\mathrm{FN})$。从完美输出开始，其中 TP 为 $=n$，FP 为 $=$，FN 为 $=0$，一个假框会使 F1 降至 $2n/(2n+1)$，下降 $\delta_{\mathrm{FP}}=1/(2n+1)$；一次漏检会使其降至 $2(n-1)/(2n-1)$，下降 $\delta_{\mathrm{FN}}=1/(2n-1)$。因此，在稀疏图像中，漏检的代价高于假框；在密集图像中，两者的代价趋于相同；而在空图像中，一个假框就会让 F1 从 1 降为 0。

因此，我们使用强化学习优化集合 F1。对于一对 (I, c)，由粗到细解码器以温度 0.7 进行采样并运行 $G=2$ 次，每次运行都会为每个解码出的锚点 p生成一个答案 $\hat{}y(p)$，以及一个输出集合。GRPO [31] 会为整个输出提供一个奖励，并将其分摊给数百个锚点答案。遵循差分奖励 [40]，（MCR 将其加到序列答案中每个边界框的 GRPO 优势上） [14]，，我们改为根据锚点答案所导致的 F1 变化，为每个锚点计分。锚点的目标是粗粒度半径内最近的实例；其边界框为 $y^{\star}(p)$，若不存在，则为无。与评估时一样，输出集合会按照交并比 0.5 与标注框进行一对一匹配，由此确定每个答案属于真正例边界框、错误边界框，还是目标漏检的无；在完美集合上，每种情况都会获得相应的 F1 变化得分（表 1）。只有当真正例边界框与其锚点的目标匹配时，才能获得 δ_{FN}，因此锚点无法通过代替邻近锚点作答来获益；错误边界框（包括抑制后仍保留的重复框）会扣除 δ_{FP}；粗粒度空值则会针对其单元格内的每个漏检实例扣除 δ_{FN}，最多计三个实例。由于一对中的所有得分都按 $1/n$ 缩放（图 3，左），我们将它们除以绝对值最大的得分，得到 $a_p\in[-1,1]$，使所有图像处于相同尺度，并保持图像内的比例 $\delta_{\mathrm{FP}}/\delta_{\mathrm{FN}}=(2n-1)/(2n+1)\in[1/3,1)$（图 3，右）。

带有 $a_p>0$ 的锚点会强化 $\hat{}y(p)$，而带有 $a_p<0$ 的锚点会降低 $\hat{}y(p)$，并在前缀树新增的一个分支中学习 $y^{\star}(p)$。将 $\pi_\theta(y)$记作 $\pi_\theta(y\mid I,\ c,\ p)$，损失为

$$\mathcal{L}_{\mathrm{RL}}=-\sum_{p}a_p\log\pi_\theta(\hat{y}(p)) \\ -\sum_{a_p<0}|a_p|\log\pi_\theta(y^{\star}(p))+\beta\,\mathrm{KL}(\pi_\theta\,\|\,\pi_{\mathrm{sft}}), \tag{3}$$

其中，求和遍历两个样本的解码锚点，而答案的对数概率按其词元求和，如式（2）所示。第一项是策略梯度，a_p 作为优势；对于 $a_p<0$，其逐词元交叉熵被裁剪在4，当某个词元的概率低于 e^{-4}时，便停止继续压低其概率。第二项利用 $a_p<0$ 目标对应的答案来训练每个锚点；KL项则在每个受监督词元处对完整词汇表求取，$\beta=0.5$使模型保持接近冻结的监督模型 π_{sft}。第3.3节中监督混合数据的行（包括其原生格式重放）按强化学习数据一半的量混入；所有样本对均来自监督训练集，因此这一阶段无需增加标注。

4. 实验

4.1. 设置

基础模型。 我们将 PinGround 应用于三种采用不同坐标格式和解码方案的视觉语言模型。LocateAnything-3B (LA) [37] 将每个坐标表示为以下词元之一 `<0>`，取值范围为 `<1000>`，并提供三种解码模式。快速模式通过一个多词元块，在一步中预测边界框的四个坐标；慢速模式逐词元写出坐标；默认的混合模式则会在块预测存在歧义时，对边界框回退至逐词元解码。Qwen3-VL-4B-Instruct [1]将坐标表示为纯文本整数，范围从 0 到1000；GroundAnything-VLM (GA-VLM) 则 [44] 使用从 `<0>`到 `<999>`的词元。对于锚点和答案，PinGround 均保留各基础模型的原生坐标格式（表 A.1）。在 LA 上，PinGround 的快速、混合和慢速模式分别保留每个锚点的块预测、根据 LA 规则重新解码被标记的边界框，以及重新解码每个边界框答案；除非另有说明，我们均采用慢速模式。重新解码会将所有选定锚点的四个坐标一并逐词元写出，因此无论涉及多少个边界框，每张图像都会增加四次前向传递。

训练。所有后训练均使用来自 COCO 和 LVIS 训练集的相同 47,170 个（图像，查询）对，用于物体 [13, 22], HierText、ICDAR 2013、ICDAR 2015 和 SROIE，用于文本 [15, 18, 19, 25], DocLayNet 和 M6Doc，用于版面 [9, 30], 以及 RefCOCOg 和 HumanRef，用于指代 [16, 27]。监督阶段运行两个周期

表2. 12个框基准测试上的 F1@0.5，以及平均 F1@0.5、F1@mIoU 和每张图像耗时。官方行表示采用官方解码器的已发布模型，-SFT 行表示在我们的配对数据上微调后的这些模型。UFO 在对象集上使用其多任务模型，在指代数据集上使用其指代表达模型；其平均值（*）涵盖这六个数据集。

	对象				Text				版面		指代		Mean		
解码	密集	COCO	VisD.	LVIS	HierT.	TotalT.	SROIE	IC15	DocL.	M6Doc	HRef	RefG	F1	mIoU	秒/图像
UFO（InternVL2.5-8B）															
官方	7.7	80.5	51.1	31.5	–	–	–	–	–	–	59.0	88.7	53.1*	40.3*	2.12*
LocateAnything-3B															
官方，混合	75.7	79.1	61.8	68.7	72.5	76.1	81.6	57.0	90.1	90.6	83.2	88.6	77.1	60.2	2.78
LA-SFT，fast	64.0	77.1	53.2	69.1	70.4	70.6	61.8	40.7	88.0	78.3	81.9	87.7	70.2	55.4	1.89
LA-SFT，混合	84.2	79.7	62.3	71.7	81.5	73.5	88.0	47.9	90.4	88.9	83.9	88.2	78.4	61.8	2.23
LA-SFT，slow	85.6	79.2	64.4	72.5	83.4	74.1	93.0	49.7	92.8	90.4	83.4	87.8	79.7	62.1	6.27
PinGround，fast	70.3	71.7	48.4	63.1	65.6	67.3	72.5	44.5	85.7	81.8	82.0	86.4	69.9	55.2	0.56
PinGround，混合	74.0	72.6	52.0	64.2	71.1	69.5	79.1	47.3	89.7	85.4	82.6	86.7	72.9	57.8	0.66
PinGround，slow	81.0	74.7	62.5	66.7	78.8	71.3	89.6	50.3	91.5	89.0	82.7	86.7	77.1	61.4	0.71
Qwen3-VL-4B-Instruct															
官方	17.0	77.3	38.6	71.0	37.6	63.4	71.1	64.6	52.8	31.6	74.0	88.2	57.3	40.0	46.66
Qwen3-VL-SFT	43.8	76.5	43.8	69.4	66.9	74.0	89.6	56.2	83.7	76.8	84.1	88.4	71.1	53.6	27.04
PinGround	63.7	74.1	56.6	68.0	73.0	74.0	90.3	53.3	90.7	86.5	84.7	89.5	75.4	57.0	2.06
GroundAnything-4B															
GA-VLM，官方	92.4	81.6	71.1	78.3	77.0	74.1	96.1	63.8	96.4	93.7	92.8	94.3	84.3	66.1	8.12
GA-DLM，官方	91.6	82.2	64.7	80.8	73.2	84.8	96.5	83.1	97.2	93.8	93.3	99.6	86.7	70.1	2.08
GA-VLM-SFT	87.1	80.9	64.9	77.7	78.0	74.0	93.5	51.4	96.3	92.4	90.4	93.9	81.7	65.6	7.16
PinGround	83.7	78.3	66.4	74.4	77.5	77.9	90.6	54.8	94.0	91.1	87.9	91.4	80.7	64.5	0.82

学习率为 2×10^{-6}，用于 Qwen3-VL 和 GA-VLM。在 LA 上运行一个周期，学习率为 4×10^{-6}，并且每个答案既按词元逐个监督，也通过 LA 的多词元块进行监督。强化学习阶段以监督阶段的学习率对相同数据对进行一轮训练（第 A 节）。

基线。 对于每个基础模型，我们报告其发布版本，以及配套的官方解码器和提示词。我们还仅使用监督学习在我们的配对数据上对其进行微调，使其以原生格式回答每个查询，并输出其所有边界框的列表；我们将这一顺序基线命名为带有后缀 -SFT 的版本。由于 PinGround 从相同的权重开始，而这些权重是在规模大得多的数据上训练的，因此将其与 -SFT 基础模型进行比较，可以隔离解码协议的影响。LA-SFT 在 LA 的三种模式下进行解码，而发布版 LA 则采用其默认的混合模式。Qwen3-VL-SFT 写出紧凑列表`[[x1,y1,x2,y2],...]`，而不是 JSON，以匹配 PinGround 的答案长度；两个 Qwen3-VL 基线均针对每个类别查询一次。对于 GroundAnything，我们还报告 GA-DLM——随其发布的推测扩散解码器——在其发布版推理引擎上的结果。我们还报告 UFO [33]在 InternVL2.5-8B [8], 上进行评估，该模型并行解码固定点网格（第 2节）。其多任务模型提示词中的类别列表包含所查询的类别；但在 COCO（其训练集）上，列表包含全部 80 个类别，与其自身的评估方式一致。我们保留其预测结果中类别得分至少为 0。5的边界框。其指代表达模型每个表达式解码一个边界框。

基准测试与指标。我们在 Rex-Omni 评估套件的 12个框基准测试上进行评估 [17], , 涵盖 Dense200、COCO [22], 、Vis-Drone [45] 和 LVIS [13], 上的目标检测，HierText [25],、TotalText [10], 、SROIE [15] 和 ICDAR2015 [19], 上的文本检测，DocLayNet [30] 和 M6Doc [9], 上的版面分析，以及 HumanRef [16] 和 RefCOCOg [27]上的指代任务。该套件还包含指向基准，但由于 PinGround 输出的是边界框，而边界框的中心点平均而言落后于原生点解码，因此我们将其排除在外（第 C 节）。每个测试集的构成见表 B.1。一个测试样本是一张图像，查询内容为其中包含的类别，或一个指代表达式。按照发布版评估方案，样本中的边界框通过交并比以贪心方式与真实标注进行一对一匹配；精确率和召回率在样本层面取平均，F1值根据平均后的结果计算。我们报告交并比为 0.5 时的 F1值，以及其在交并比阈值 0.5:0.95 上的平均值（F1@mIoU）。无预测记为无，文本检测评估不包含识别。

延迟。 准确率和延迟数据来自相同的运行。每次仅处理一张图像，批量大小为 1，由单个进程在一块 NVIDIA H100 GPU 上运行；每张图像耗时涵盖预处理、视觉编码、预填充和解码，并通过 CUDA同步进行测量。除 GA-DLM 使用随其发布的、基于 SGLang 的引擎运行，以及 UFO 使用其发布的流水线运行外，所有模型均在 HF Transformers 的即时执行模式下运行。顺序基线和 GA-DLM 对每个基准测试最多生成 num-

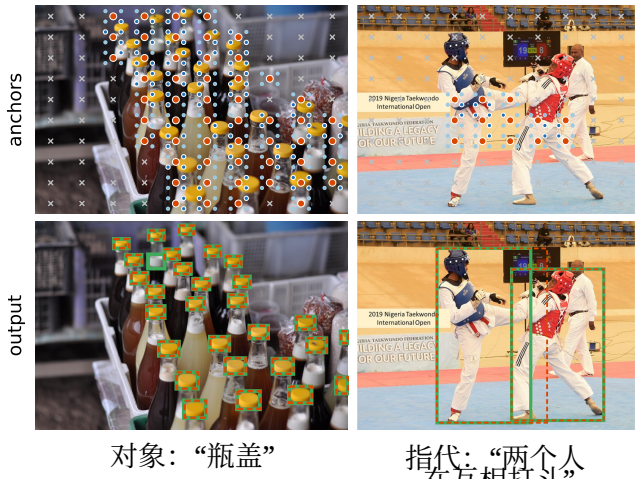


图4. PinGround 在 LA 慢速模式下的示例。上方一行显示报告边界框的锚点（橙色为粗略结果，蓝色为精细结果）或无结果（灰色为粗略结果，浅蓝色为精细结果）；下方一行显示输出边界框（虚线）和真实标注（绿色）。

个词元（第 A节），达到该上限的输出按实际生成的内容进行评估和计时。

4.2. 主要结果

表2将 PinGround 与在相同配对数据上微调的顺序基线（-SFT）进行比较，因此两者的差异仅在于解码评估方案；同时列出采用各自官方解码器的已发布模型以供参考。在 LocateAnything 上，PinGround 默认的慢速模式达到 77.1 F1值和 61.4 F1@mIoU，每张图像耗时 0.71 秒；LA-SFT 默认的混合模式则达到 78.4 和 61.8，每张图像耗时 2.23 秒。在匹配模式下，PinGround 在慢速模式下比 LA-SFT 低 2.6 F1值，在快速模式下低 0.3 F1值，速度分别快 8.8 倍和 3.4 倍。与混合模式相比，差距最大的基准测试是 COCO 和 LVIS；而在 VisDrone、SROIE、ICDAR2015 以及两个布局数据集上，PinGround 表现更佳。图4 展示了其输出示例。在 Qwen3-VL 上，PinGround 比 Qwen3-VL-SFT 高出 4.3 F1值和 3.4 F1@mIoU，且每张图像耗时为 2.06 秒，而非 27.04 秒。性能提升在密集的 Dense200、VisDrone 以及两个布局数据集上最为显著。两个顺序基线都经常陷入重复边界框或缓慢移动边界框的循环，直到达到令牌上限；在一个基准测试的样本中，Qwen3-VL-SFT 平均有 7% 出现这种情况，已发布模型则有 16%；在其余样本上，它们的耗时仍分别是 PinGround 的 7.7 倍和 9.5 倍。

在 GroundAnything 上，PinGround 的 F1值比 GA-VLM-SFT 低 1.0，但每张图像耗时仅 0.82 秒，而非 7.16 秒，快了 8.7 倍。尽管已发布的 GA-DLM 使用自己的推理引擎（该引擎对同一个自回归模型 GA-VLM 进行解码时，每个词元的速度比我们的设置快 1.1 至 1.2 倍），其速度仍比 PinGround 慢 2.5 倍。

表 3。在 LA 上以慢速模式运行 PinGround 时，Dense200 上每张图像和 RefCOCOg 上每个指代表达的平均成本。该表报告了语言模型的前向传播次数（包括预填充，Fwds，四舍五入为整数）、根据处理后的词元和注意力长度计算得出的矩阵乘积浮点运算次数（TFLOPs）、GPU 内存峰值（GiB，Mem）以及耗时（秒，T）。

	Dense200				RefCOCOg			
解码	前向传播次数	TFLOPs	内存	T	前向传播次数	TFLOPs	内存	T
LocateAnything-3B								
LA-SFT，混合	169	25.8	9.1	6.61	5	3.9	7.7	0.23
PinGround，slow	7	41.5	10.3	0.71	7	10.4	8.5	0.36
Qwen3-VL-4B-Instruct								
Qwen3-VL-SFT	885	15.5	8.6	31.07	20	3.5	8.4	0.79
PinGround	41	57.6	10.4	2.04	40	16.9	9.6	1.73
GroundAnything-4B								
GA-VLM-SFT	511	13.4	10.5	19.07	18	4.5	9.5	0.72
PinGround	13	40.8	11.0	0.87	13	13.1	9.6	0.59

GA-DLM 也是平均精度最高的一行。GroundAnything 的训练数据尚未发布；仅使用我们的 47,170 对数据对 GA-VLM 进行微调，就会使其 F1值从 84.3 降至 81.7。因此，与 GA-DLM 的部分差距反映的是我们的训练数据，而非解码协议。

UFO 也会并行解码其网格点，在对象集上的每张图像耗时为 2.5 至 3.1 秒，且几乎不受实例数量影响。其已发布模型是在 COCO 检测和指代表达数据上训练的，因此在 COCO 上达到 80.5 的 F1值，但在类别大多不属于 COCO 的 Dense200 上仅为 7.7，在 LVIS 上为 31.5；它还会将每个指代表达式解码为单个边界框，在多实例 HumanRef 上达到 59.0 的 F1值。对于它未曾训练过的文本与布局集，它无法找到任何边界框，因此我们不将这些数据计入其平均值。这些差距既反映了其训练数据，也反映了其评估方案，我们将 UFO 列出以供参考。

4.3. 延迟与成本

图5按每个样本的实例数细分了耗时。从 1 个实例到约 170 个实例，PinGround 的中位耗时最多增长 2.2 倍，而顺序解码器（包括 GA-DLM）则增长 19 至 182 倍，因为每增加一个边界框都会延长其输出；相比之下，PinGround 的一轮中的所有锚点都会在每次前向传播中同步推进。在指代数据集这类只有一两个实例的情况下，顺序解码器所需的前向传播次数与我们的两轮相当。此时，PinGround 比 GA-VLM 解码器快 1.2 至 1.4 倍；后者会在边界框之前重复查询。PinGround 在 LocateAnything 上最多慢 1.7 倍，比 GA-DLM 慢 1.9 倍，在 Qwen3-VL 上慢 2.2 倍，因为其中每个数字都是一个单独的词元。

表3 统计了在最密集的基准和单目标基准上每张图像的计算开销。在 OnDense200 上，PinGround

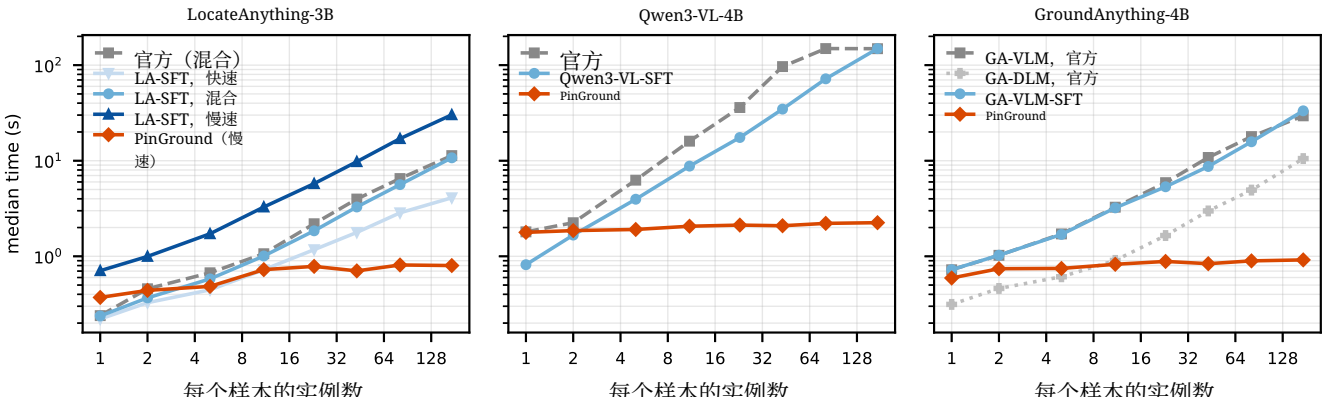


图 5. 12个框基准测试中每个样本的中位时间与每个样本实例数的关系。样本按实例数以 2 的幂分组，每个点表示至少包含 10 个样本的一组的中位数，两个坐标轴均采用对数刻度。

表4. PinGround 在慢速模式下于 LocateAnything-3B 上的解码器组件，取 12个框基准测试的平均值。

解码	F1	P	R	F1@mIoU	秒/图像
仅粗网格	73.7	75.3	75.9	58.9	0.55
仅细网格	76.3	74.1	81.0	60.6	1.14
从粗到细	77.1	72.6	84.9	61.4	0.71

表5. 在三个基础模型上对 PinGround 进行后训练，并在 12个框基准测试上取平均。

Base		F1	P	R	F1@mIoU
LocateAnything-3B	SFT	75.3	70.6	84.2	59.9
	+ RL	77.1	72.6	84.9	61.4
Qwen3-VL-4B	SFT	71.2	65.3	82.1	53.7
	+ RL	75.4	72.6	80.7	57.0
GroundAnything-4B	SFT	77.4	73.1	85.6	61.9
	+ RL	80.7	79.4	83.9	64.5

平均每张图像需要对语言模型进行 7 到 41 次前向传播，而非 169 到 885 次，但浮点运算次数是其 1.6 到 3.7 倍，峰值内存最多多出 1.8 GiB，因为每轮都会计算每个锚点，包括许多答案为无的锚点。在 RefCOCOg 上，它需要 2.7 到 4.8 倍的浮点运算次数，前向传播次数则为 0.7 到 2.0 倍。每个锚点只关注共享前缀和自身的少量词元，因此答案的注意力范围保持固定；而列表中的 k 个边界框则会关注它之前的所有边界框。因此，PinGround 以增加计算量换取更少的顺序步骤。在批量大小为 1 时，这种权衡很有优势：顺序解码器需要读取所有权重才能生成单个词元，因此受内存带宽限制 [20]，而 PinGround 的每次前向传播只需读取一次权重，就能同时推进所有实例的边界框，因此实例数量越多，加速效果越明显。

4.4. 消融实验

由粗到细解码。 表 4 分别仅使用粗网格、在所有位置使用细网格，以及采用由粗到细的调度来解码 LocateAnything。仅使用粗网格速度最快，但会漏检实例，召回率低 9.0 个点，因为相邻实例会落入同一个粗粒度单元格，而其锚点只能报告其中一个。全程使用细网格（一种类似 GiT 和 UFO 所用网格的单一固定网格）解码的锚点数量是粗网格的四倍，每张图像耗时 1.14 秒，而非 0.71 秒，但其召回率和 F1值仍低于由粗到细的方法。仅细化占用单元格即可找回密集区域中的实例，相比粗网格每张图像仅增加 0.16 秒耗时。

后训练。 强化学习在 LocateAnything、Qwen3-VL 和 GroundAnything 上分别使 F1值提升 1.8、4.2 和 3.3 个点（表 5）。提升来自精确率，分别上升 2.0、7.3 和 6.3 个点，而召回率的变化不到两个点。正如设计所预期，强化学习主要减少假框；逐锚点监督损失无法识别这类错误，因为它会单独评判每个答案（第 3.4节）。

5. 结论

我们已经证明，视觉语言模型无需逐一生成查询对应的边界框。单个点就足以告知每个并行流应报告哪个实例，因此，在给定图像的情况下，带锚点的查询彼此独立。PinGround 将它们解码为同一棵前缀树的叶节点，无论实例数量多少，都只需两轮。只有找到实例的锚点才会被细化。在同一棵前缀树上进行监督微调，并结合集合级强化学习，就足以教会现有视觉语言模型回答这类查询。在不改变架构的情况下，在三个视觉语言模型上，使用相同数据训练的 PinGround 与顺序解码相比，F1值最多低 5.5；在 Dense200 上，速度最高可提升 21.9 倍。其实例密度越高，加速效果越明显；而对于包含一个或两个

目标的查询，收益很小，甚至会变慢；此外，其第一级锚点构成一个固定网格，也会在空区域上消耗计算资源。未来的工作将针对每幅图像自适应这些锚点，并将评估方案扩展到本文研究的边界框之外。

参考文献

[1] Shuai Bai、Yuxuan Cai、Ruizhe Chen、Keqin Chen、Xionghui Chen、Zesen Cheng、Lianghao Deng、Wei Ding、Chang Gao、Chunjiang Ge等。Qwen3-VL技术报告。arXiv预印本 arXiv:2511.21631，2025年。1，2，5[2] Shuai Bai、Keqin Chen、Xuejing Liu、Jialin Wang、Wenbin Ge、Sibo Song等。Qwen2.5-VL技术报告。arXiv预印本 arXiv:2502.13923，2025年。2[3] Tianle Cai、Yuhong Li、Zhengyang Geng、Hongwu Peng、Jason D. Lee、Deming Chen和Tri Dao。Medusa：具有多个解码头的简单大语言模型推理加速框架。在国际机器学习会议上，第5209–5235页，2024年。2[4] Nicolas Carion、Francisco Massa、Gabriel Synnaeve、Nicolas Usunier、Alexander Kirillov和Sergey Zagoruyko。使用Transformer实现端到端目标检测。在欧洲计算机视觉会议上，第213–229页。Springer，2020年。2[5] Keqin Chen、Zhao Zhang、Weili Zeng、Richong Zhang、Feng Zhu和Rui Zhao。Shikra：释放多模态大语言模型的指代对话魔力。arXiv预印本 arXiv:2306.15195，2023年。1，2[6] Ting Chen、Saurabh Saxena、Lala Li、David J. Fleet和Geoffrey Hinton。Pix2Seq：用于目标检测的语言建模框架。在国际学习表征会议上，2022年。1，2[7] Ting Chen、Saurabh Saxena、Lala Li、Tsung-Yi Lin、David J. Fleet和Geoffrey E. Hinton。用于视觉任务的统一序列接口。《神经信息处理系统进展》，35：31333–31346，2022年。1[8] Zhe Chen、Weiyun Wang、Yue Cao、Yangzhou Liu、Zhangwei Gao、Erfei Cui、Jinguo Zhu、Shenglong Ye、Hao Tian、Zhaoyang Liu等。通过模型、数据和测试时扩展拓展开源多模态模型的性能边界。arXiv预印本 arXiv:2412.05271，2024年。6[9] Hiuyi Cheng、Peirong Zhang、Sihang Wu、Jiaxin Zhang、Qiyuan Zhu、Zecheng Xie、Jing Li、Kai Ding和Lianwen Jin。M^{6}Doc：面向现代文档版面分析的大规模多格式、多类型、多版面、多语言、多标注类别数据集。在2023年IEEE/CVF计算机视觉与模式识别会议（CVPR）上，第15138–15147页。IEEE，2023年。5，6，12[10] Chee Kheng Ch'ng和Chee Seng Chan。Total-Text：用于场景文本检测与识别的综合数据集。在2017年第14届IAPR国际文档分析与识别会议（ICDAR）上，第935–942页。IEEE，2017年。6[11] Jacob Devlin、Ming-Wei Chang、Kenton Lee和Kristina Toutanova。BERT：用于语言理解的深度双向Transformer预训练。在《2019年北美计算语言学协会分会会议：人类语言技术，第1卷（长篇与短篇论文），第4171–4186页，2019年。1[12] Fabian Gloeckle、Badr Youbi Idrissi、Baptiste Rozi`ère、David Lopez-Paz 和 Gabriel Synnaeve。通过多词元预测实现更优、更快速的大语言模型。载于国际机器学习会议，第15706–15734页，2024年。2[13] Agrim Gupta、Piotr Doll´ar 和 Ross Girshick。LVIS：用于大词汇量实例分割的数据集。载于2019年IEEE/CVF计算机视觉与模式识别会议（CVPR），第5351–5359页。IEEE，2019年。5、6、12[14] Xinheng Han、Jianfei Wang、Yu Chen、Xiang Wang、Shuai Li、Weixing Li 和 Feng Pan。将得分归于正确的边界框：结构化视觉感知的边际贡献分配。arXiv预印本 arXiv:2608.01055，2026年。5[15] Zheng Huang、Kai Chen、Jianhua He、Xiang Bai、Dimosthe- nis Karatzas、Shijian Lu 和 C. V. Jawahar。ICDAR2019扫描收据光学字符识别与信息提取竞赛。载于2019年国际文档分析与识别会议（ICDAR），第1516–1520页。IEEE，2019年。5、6、12[16] Qing Jiang、Lin Wu、Zhaoyang Zeng、Tianhe Ren、Yuda Xiong、Yihao Chen、Liu Qin 和 Lei Zhang。指代任意人物。载于2025年IEEE/CVF国际计算机视觉会议（ICCV），第21667–21678页。IEEE，2025年。1、5、6、12、14[17]Qing Jiang、Junan Huo、Xingyu Chen、Yuda Xiong、Zhaoyang Zeng、Yihao Chen、Tianhe Ren、Junzhi Yu 和 Lei Zhang。通过下一点预测检测任意物体。载于IEEE/CVF计算机视觉与模式识别会议论文集，第25472–25483页，2026年。1、2、6、13[18]Dimosthenis Karatzas、Faisal Shafait、Seiichi Uchida、Masakazu Iwamura、Lluis Gomez i Bigorda、Sergi Robles Mestre、Joan Mas、David Fernandez Mota、Jon Almazan Al- mazan 和 Lluis Pere De Las Heras。ICDAR 2013鲁棒阅读竞赛。载于2013年第12届国际文档分析与识别会议，第1484–1493页。IEEE，2013年。5、12[19] Dimosthenis Karatzas、Lluis Gomez-Bigorda、Anguelos Nicolaou、Suman Ghosh、Andrew Bagdanov、Masakazu Iwa- mura、Jiri Matas、Lukas Neumann、Vijay Ramaseshan Chan- drasekhar、Shijian Lu 等。ICDAR 2015鲁棒阅读竞赛。载于2015年第13届国际文档分析与识别会议（ICDAR），第1156–1160页。IEEE，2015年。5、6、12[20] Yaniv Leviathan、Matan Kalman 和 Yossi Matias。通过推测解码加速 Transformer 推理。载于国际机器学习会议，第19274–19286页，2023年。2、8[21] Liunian Harold Li、Pengchuan Zhang、Haotian Zhang、Jian- wei Yang、Chunyuan Li、Yiwu Zhong、Lijuan Wang、Lu Yuan、Lei Zhang、Jenq-Neng Hwang 等。基于定位的语言-图像预训练。载于2022年IEEE/CVF计算机视觉会议

与模式识别会议（CVPR），第10955–10965页。IEEE，2022年。3[22] Tsung-Yi Lin、Michael Maire、Serge Belongie、James Hays、Pietro Perona、Deva Ramanan、Piotr Doll´ar 和 C. Lawrence Zitnick。Microsoft COCO：上下文中的常见物体。载于欧洲计算机视觉会议，第740–755页。Springer，2014年。5、6、12、14[23] Chengxin Liu、Hao Lu、Zhiguo Cao 和 Tongliang Liu。用于人群计数、定位等任务的点查询四叉树。载于2023年IEEE/CVF国际计算机视觉会议（ICCV），第1676–1685页。IEEE，2023年。3[24] Shilong Liu、Zhaoyang Zeng、Tianhe Ren、Feng Li、Hao Zhang、Jie Yang、Qing Jiang、Chunyuan Li、Jianwei Yang、Hang Su 等。Grounding DINO：将 DINO 与接地预训练相结合，用于开放集目标检测。载于欧洲计算机视觉会议，第38–55页。Springer，2024年。1、3[25] Shangbang Long、Siyang Qin、Dmitry Panteleev、Alessandro Bissacco、Yasuhisa Fujii 和 Michalis Raptis。迈向端到端统一的场景文本检测与版面分析。载于2022年IEEE/CVF计算机视觉与模式识别会议（CVPR），第1039–1049页。IEEE，2022年。5、6、12[26] Jiasen Lu、Christopher Clark、Rowan Zellers、Roozbeh Mot- taghi 和 Aniruddha Kembhavi。Unified-IO：用于视觉、语言和多模态任务的统一模型。载于国际学习表征会议，2023年。1[27] Junhua Mao、Jonathan Huang、Alexander Toshev、Oana Camburu、Alan L. Yuille 和 Kevin Murphy。无歧义物体描述的生成与理解。载于IEEE计算机视觉与模式识别会议论文集，第11–20页，2016年。5、6、12、14[28] Matthias Minderer、Alexey Gritsenko、Austin Stone、Maxim Neumann、Dirk Weissenborn、Alexey Dosovitskiy、Aravindh Mahendran、Anurag Arnab、Mostafa Dehghani、Zhuoran Shen 等。简单的开放词汇目标检测。载于欧洲计算机视觉会议，第728–755页。Springer，2022年。3[29] Zhiliang Peng、Wenhui Wang、Li Dong、Yaru Hao、Shaohan Huang、Shuming Ma 和 Furu Wei。Kosmos-2：将多模态大语言模型定位到现实世界。arXiv预印本 arXiv:2306.14824，2023年。1、2[30] Birgit Pfitzmann、Christoph Auer、Michele Dolfi、Ahmed S. Nassar 和 Peter Staar。DocLayNet：用于文档版面分割的大规模人工标注数据集。载于第28届ACM SIGKDD知识发现与数据挖掘会议论文集，第3743–3751页，2022年。5、6、12[31] Zhihong Shao、Peiyi Wang、Qihao Zhu、Runxin Xu、Junxiao Song、Xiao Bi、Haowei Zhang、Mingchuan Zhang、Y. K. Li、Y. Wu 和 Daya Guo。DeepSeekMath：推动开放语言模型数学推理能力的极限。arXiv预印本 arXiv:2402.03300，2024年。5[32] Mitchell Stern、Noam Shazeer 和 Jakob Uszkoreit。深度自回归模型的分块并行解码。《神经信息处理系统进展》，第31卷，2018年。2

[33] Hao Tang、Chen-Wei Xie、Haiyang Wang、Xiaoyi Bao、Tingyu Weng、Pandeng Li、Yun Zheng 和 Liwei Wang。UFO：通过开放式语言接口实现细粒度视觉感知的统一方法。神经信息处理系统进展，38：83761–83791，2026年。3、6、15、16[34] Haiyang Wang、Hao Tang、Li Jiang、Shaoshuai Shi、Muhammad Ferjad Naeem、Hongsheng Li、Bernt Schiele 和 Liwei Wang。GiT：通过通用语言接口迈向通用视觉Transformer。载于欧洲计算机视觉会议，第55–73页。Springer，2024年。3[35] Peng Wang、An Yang、Rui Men、Junyang Lin、Shuai Bai、Zhikang Li、Jianxin Ma、Chang Zhou、Jingren Zhou 和 Hongxia Yang。OFA：通过简单的序列到序列学习框架统一架构、任务和模态。载于国际机器学习会议，第23318–23340页，2022年。1[36] Peng Wang、Shuai Bai、Sinan Tan、Shijie Wang、Zhihao Fan、Jinze Bai、Keqin Chen、Xuejing Liu、Jialin Wang、Wenbin Ge 等。Qwen2-VL：提升视觉语言模型在任意分辨率下对世界的感知能力。arXiv预印本 arXiv:2409.12191，2024年。1、2[37] Shihao Wang、Shilong Liu、Yuanguo Kuang、Xinyu Wei、Yangzhou Liu、Zhiqi Li、Yunze Man、Guo Chen、Andrew Tao、Guilin Liu 等。LocateAnything：通过并行框解码实现快速、高质量的视觉语言定位。载于欧洲计算机视觉会议，第336–357页。Springer，2026年。2、5、12、13、14、15[38] Wenhai Wang、Zhe Chen、Xiaokang Chen、Jiannan Wu、Xizhou Zhu、Gang Zeng、Ping Luo、Tong Lu、Jie Zhou、Yu Qiao 和 Jifeng Dai。VisionLLM：大语言模型也是面向视觉任务的开放式解码器。神经信息处理系统进展，36：61501–61513，2023年。1、2[39] Yingming Wang、Xiangyu Zhang、Tong Yang 和 Ji an Sun。Anchor DETR：基于Transformer的目标检测中的查询设计。载于AAAI人工智能会议，2022年。2[40] David H. Wolpert 和 Kagan Tumer。集体成员的最优收益函数。复杂系统进展，4(2–3)：265–279，2001年。5[41]Jiannan Wu、Muyan Zhong、Sen Xing、Zeqiang Lai、Zhaoyang Liu、Zhe Chen、Wenhai Wang、Xizhou Zhu、Lewei Lu、To ng Lu、Ping Luo、Yu Qiao 和 Jifeng Dai。VisionLLM v2：面向数百种视觉语言任务的端到端通用多模态大语言模型。神经信息处理系统进展，37：69975，2024年。69925 2：–。[42] Bin Xiao、Haiping Wu、Weijian Xu、Xiyang Dai、Houdong Hu、Yumao Lu、Michael Zeng、Ce Liu 和 Lu Yuan。Florence-2：为多种视觉任务推进统一表示。载于2024年IEEE/CVF计算机视觉与模式识别会议（CVPR），第4818–4829页。IEEE，2024年。1[43] Haoxuan You、Haotian Zhang、Zhe Gan、Xianzhi Du、Bowen Zhang、Zirui Wang、Liangliang Cao、Shih-Fu Chang 和 Yinfei Yang。Ferret：在任意位置引用并定位任何内容

在任意粒度下。在 国际学习表征会议，第57153–57180页，2024年。 2[44] Qize Yu、Lianrui Fan、Bowen Ping、Xini Ding、Zetian Song、Junbo Niu、Kaixuan Wang、Tianxing Chen、Yue Chen、Minghua He 等。GroundAnything：以闪电般的速度协调并行解码与精确视觉定位。 *arXiv*预印本 *arXiv:2609.39600*，2026年。 2， 5[45] Pengfei Zhu、Longyin Wen、Dawei Du、Xiao Bian、Heng Fan、Qinghua Hu 和 Haibin Ling。检测与跟踪无人机挑战赛。 *IEEE*模式分析与机器智能汇刊，44(11)：7380–7399，2021年。 6，14

PinGround：将查询固定到点，实现视觉语言模型的并行定位

补充材料

表 A.1. PinGround 在三个基础模型上的锚点和答案格式。每个锚点是一个点（x，y），每个答案则是一个边界框（x_1，y_1，x_2，y_2）或无，采用该基础模型的原生坐标格式。在 LA 上，锚点后接 <ref>c</ref>，以开启对查询 c 的响应。

LA：坐标词元 <0> 至 <1000> 锚点 <x><y> 答案 <box><x1><y1><x2><y2></box>， <box>无</box>
Qwen3-VL：0 到 1000 的纯文本整数 锚点： （x, y）。 回答 [x1, y1, x2, y2],无
GA-VLM：坐标标记 <0>到 <999> 锚点： <边界框_起点|><x><y><边界框_终点|>。 答案 <边界框_起点|><x1><y1><x2><y2><边界框_终点|>, None

A. 实现细节

训练数据。 在这 47,170 对数据中，12,000 对是 COCO 图像上的目标查询，涵盖 COCO 和 LVIS 类别 [13, 22]，10,162 是来自 Hi- erText、ICDAR 2013、ICDAR 2015 和 SR OIE 的文本检测对 [15, 18,19, 25]，13,000是来自 DocLayNet 和 M6Doc 的布局查询 [9, 30]，还有 12,008 对来自 RefCOCOg 和 HumanRef 的指代表达 [16, 27]。文本检测、版面和 HumanRef 数据对取自 LA 发布的训练数据 [37]，而目标和 RefCOCOg 数据对则根据原始标注构建。

格式。 表 A.1列出了三个基础模型上的锚点和答案格式。LA 和 GA-VLM 原生支持 无答案，而 Qwen3-VL 上的无 则通过后训练引入。

监督阶段。 在 Qwen3-VL 和 GA-VLM 上，每对数据都作为一个序列进行训练，该序列包含其锚点的前缀树；同时，原生格式重放也作为一个序列进行训练。在这两个基础模型上，第一个答案词元的损失（它决定输出边界框还是无）权重设为3。权重设为 1 时，F1值相近；为简洁起见，公式（2）省略了这一权重。在 LA 上，每对数据同样构成一棵前缀树，其锚点作为叶节点，位于对应查询的分支下；这些树与原生格式重放序列一起打包成长度不超过 8,192 个词元的序列。LA 既可以逐词元预测边界框，也可以使用多词元块：后者会向输入中追加掩码词元，并在一次前向传播中预测这些位置上的完整答案。每个锚点都通过两条路径进行监督，并在同一答案上同时使用公式（2）的损失和 LA 的块损失。PinGround 的监督阶段在三个基础模型上均采用余弦调度，并设置 50 个预热步数。-SFT 基线在三个基础模型上均以学习率 2×10^{-6} 训练两个轮次；在 LA 上，它们也包含采用其光学字符识别定位格式的文本对，与 PinGround 的重放方式相同。

强化学习阶段。 使用由粗到细解码器，从监督模型生成一次采样轨迹，其中包括抑制查询内的重复项。Qwen3-VL 和 GA-VLM 以温度 0。7对每个答案词元进行采样；LA 则以温度 0。7 进行块采样，并逐词元重新解码每个边界框，同时采用阈值为 0。9的核采样。幅度小于 0。01 的得分会在归一化前丢弃。对于一对得分为正的无答案，最多保留置信度最低的 16 个。每个得分都会对其答案在公式（3）中的对数概率加权；该概率涵盖 Qwen3-VL 和 GA-VLM 答案中的每个词元，以及 LA 的块预测，即坐标、无 和结束边界框词元。学习率采用余弦调度，不设预热。

LA 上的解码模式。在混合模式下，PinGround 会根据已发布的 LocateAnything 代码中的规则，逐词元重新解码被标记的边界框，即块输出格式错误，或某个坐标的最高概率低于 0。9 的边界框；同时，其四个概率最高的坐标标记分布在超过 60 个区间中。LA 论文指出 0。7 以及五个概率最高的词元，其分布范围为 80 [37]，，我们沿用已发布的规则，官方解码器在已发布的 LA 和 LA-SFT 中也使用该规则。慢速模式会重新解码每个边界框答案，而无 答案在所有模式下均保留块的预测结果。官方解码器以温度 0。7进行解码，并采用阈值为 0。9 的核采样和 1。1的重复惩罚，因此官方行和 LA-SFT 行在不同运行之间会有所变化；LA-SFT 在混合模式下运行三次，在 Dense200 上的 F1值分别达到 84.2、82.6 和 81.5，其中第一次运行的结果报告于表 2中。PinGround、顺序式 Qwen3-VL 和 GroundAnything 解码器以及 GA-DLM 均采用贪心解码。

解码限制与引擎。 顺序式 Lo-cateAnything 解码器每次输出最多生成 8192 个新词元，而顺序式 Qwen3-VL、GroundAny-thing 解码器和 GA-DLM 在 Dense200 上最多生成 4096 个，

在 VisDrone、HierText 和 SROIE 上最多生成 4096 个，在其他对象、文本与布局数据集上最多生成 2048 个，在指代数据集上最多生成 1024 个。GA-DLM 在其发布的、基于 SGLang 的引擎上运行；该引擎的调度器、注意力内核和 KV 缓存均针对其解码器进行了适配。按照发布时的配置，它以 BF16 运行，不使用 CUDA 图或重叠调度，并按请求计时。在相同设置下，该引擎在来自五个基准测试的 1,103 个样本上运行自回归 GA-VLM，每生成一个词元耗时 32.0 至 32.6 毫秒；相比之下，我们的设置为 36.7 至 37.5 毫秒。两者均通过请求耗时随生成词元数量变化的斜率测得。

评估详情。随 Rex-Omni 和 LocateAnything 一同发布的评估代码，针对框基准测试提供了两种评估器 [17, 37]。COCO 风格评估器用于评估 COCO 和 LVIS，在各类别内匹配边界框，并对各类别的精确率和召回率取平均。通用定位评估器则用于评估其他基准测试，并对各样本的精确率和召回率取平均。该评估器会先将一个样本中所有查询类别的边界框汇集起来再进行匹配，因此一个边界框也可能与其他类别的真实标注匹配。表 2使用通用评估器对全部 12 个基准测试进行评分（第 4.1 节），从而统一评估方案。

表 A.2 还使用类别感知协议对表2 中的答案进行评分。该协议保留逐样本平均方式，但在对象和版面基准测试中，只将边界框与其自身类别的真实标注进行匹配。PinGround 对每个锚点只查询一个类别，且不写入类别名称；两个 Qwen3-VL 基线则按类别分别查询，因此对于这些模型，边界框的类别由其对应的查询决定。其他模型则将所有类别一并查询，因此边界框的类别就是与其一起写出的名称。名称不属于查询类别的边界框（包括名称被截断或拼写错误的情况）计为错误框，与表2中的处理相同。COCO人群区域不计为目标，其中的边界框会被忽略；对于 LVIS，未穷尽标注或未在图像中验证的类别所对应的未匹配边界框也会被忽略；此外，主要位于难以辨认的 ICDAR2015 区域内的边界框同样会被忽略。每个边界框首先都被视为由两个角点确定的矩形，无论角点的顺序如何；随后剔除零面积边界框，同一类别下坐标相同的重复边界框只计一次。

三个基础模型上的平均值变化了 -0.5 至 $+1.1$ F1值，PinGround 与 -SFT 基础模型之间的差距最多为 0.7 F1值。各个基准测试的变化幅度更大，且方向不一，其中 VisDrone 和 ICDAR2015 的变化最大。类别匹配会降低 VisDrone 的 F1值，因为汽车与厢式货车、行人与人群等相近类别的边界框不再彼此匹配；而忽略无法辨认的区域会提高所有模型在 ICDAR2015 上的 F1值。PinGround 在 VisDrone 上的 F1值降幅大于 -SFT 基线。LocateAnything 和 GroundAnything 模型会将所有查询类别的边界框按一个序列输出，因此，它们会从所有查询类别中为每个边界框选择类别，并能看到已输出的边界框。PinGround 的每个查询只询问一个类别，并与其他查询并行解码，彼此看不到对方的答案，因此相近类别的查询经常会返回同一个对象，而非极大值抑制只保留置信度较高的查询所对应的边界框（第 3.2 节）；其类别不一定总是标注类别。Qwen3-VL 基线也会通过单独的查询回答每个类别，但不进行非极大值抑制，因此，当两个类别的查询返回同一个对象时，两个边界框都会保留，标注类别对应的边界框仍能匹配。

计算成本。 表 3 统计语言模型和视觉编码器对每个已处理词元和图像块的权重乘积，以及语言模型对每个计算出的查询-键值对的注意力乘积，其中包括被前缀树掩码隐藏的键值对。不统计输出层、视觉编码器内部的注意力，以及归一化和激活函数。计入这些计算，并为预测词元计算输出层后，PinGround 在 Dense200 上所需的浮点运算次数是顺序解码器的 1.4 到 3.6 倍，在 RefCOCOg 上则是 2.7 到 4.8 倍。内存是处理一个样本时分配的峰值 GPU内存，其中包括权重。

B. 基准测试

完整使用指代数据集和 VisDrone 的 1610 张测试图像；COCO、DocLayNet 和 M6Doc 则各随机抽取 1000 张图像，LVIS 使用来自 COCO val2017 且不属于我们训练数据的前 1000 张验证图像，其他数据集则按发布时的形式使用（表 B.1）。每张图像仅使用其中包含的类别进行查询，因此每个查询至少对应一个实例。从回答中解析边界框，并忽略其他文本，因此不含边界框的回复计为未预测。

在 HierText 和 SROIE 上，每张文本图像仅以类别文本行查询一次；在 TotalText 和 IC- DAR2015 上，则仅以类别单词 查询一次，并将回答作为文本检测结果评分。发布的评估仅将文本边界框与转录文本相同的真实标注进行匹配，因此我们将图像中的所有文本边界框合并到这一类别中，并仅按交并比进行匹配。我们不纳入转录文本，因为一轮中的锚定回答会一起解码，因此一轮的耗时取决于最长回答。对于所有基础模型，边界框都是长度几乎固定的简短回答，而单词和文本行的转录文本长度则差异很大。用单独的识别器读取每个检测到的边界框会更适合 PinGround，但这需要额外引入一个专用模型；而视觉语言模型定位旨在用单个模型解决这些任务。

表 A.2。在类别感知评估方案下，表 2 中各答案的 F1@0.5，最后一列为表 2 中的平均 F1@0.5。UFO 的平均值涵盖表 2 中所示的六个数据集。

	对象				Text				版面		指代		Mean	
解码	Dense	COCO	VisD.	LVIS	HierT.	TotalT.	SROIE	IC15	DocL.	M6Doc	HRef	RefG	F1	表 2
UFO（InternVL2.5-8B）														
官方	7.7	81.4	42.3	31.7	–	–	–	–	–	–	59.0	88.7	51.8*	53.1*
LocateAnything-3B														
官方，混合	75.8	79.6	53.2	69.5	72.6	76.1	81.7	60.6	88.4	89.5	83.2	88.6	76.6	77.1
LA-SFT ，fast	64.3	77.9	46.8	70.6	70.7	70.6	62.5	48.8	87.0	77.5	81.9	87.7	70.5	70.2
LA-SFT，混合	84.3	80.3	53.5	73.2	81.6	73.5	88.0	60.5	88.9	87.7	83.9	88.2	78.6	78.4
LA-SFT ，slow	85.6	79.9	54.5	73.9	83.4	74.1	93.1	62.8	91.0	89.0	83.4	87.8	79.9	79.7
PinGround ，fast	70.4	72.5	39.9	65.6	66.0	67.4	72.9	55.9	84.5	81.1	82.0	86.4	70.4	69.9
PinGround，混合	74.1	73.3	42.7	66.5	71.4	69.6	79.4	60.5	88.1	84.3	82.6	86.7	73.3	72.9
PinGround ，slow	81.1	75.1	50.6	68.6	78.8	71.3	89.6	65.2	89.9	87.3	82.7	86.7	77.2	77.1
Qwen3-VL-4B-Instruct														
官方	19.5	78.9	41.0	72.9	41.8	64.6	71.2	71.5	47.4	29.1	74.1	88.2	58.4	57.3
Qwen3-VL-SFT	44.0	77.5	39.6	71.7	67.5	74.1	90.0	68.8	82.1	75.1	84.1	88.4	71.9	71.1
PinGround	63.7	74.6	46.0	70.0	73.0	74.0	90.3	68.3	88.5	83.7	84.7	89.5	75.5	75.4
GroundAnything-4B														
GA-VLM ，官方	92.6	82.3	63.1	80.1	77.3	74.1	96.1	66.9	95.9	92.7	92.8	94.3	84.0	84.3
GA-DLM ，官方	91.6	83.2	56.9	83.0	76.2	85.0	96.6	90.3	96.6	93.0	93.3	99.6	87.1	86.7
GA-VLM-SFT	87.4	81.6	56.2	79.6	78.2	74.0	93.5	64.2	95.5	91.5	90.4	93.9	82.2	81.7
PinGround	83.7	78.5	53.3	75.9	77.5	77.9	90.6	70.0	92.4	89.3	87.9	91.4	80.7	80.7

表 B.1. 12 个基准测试的测试样本：前三组为图像，最后一组为指代表达式。

Task	基准测试	样本	选择
对象	Dense200	200	按发布版本
	COCO	1000	从 4952 个中随机抽取
	VisDrone	1610	完整测试集
	LVIS	1000	COCO val2017，不在训练集中
Text	HierText	1723	按发布版本
	TotalText	300	按发布版本
	SROIE	360	按发布版本
	ICDAR2015	496	按发布时的版本
版面	DocLayNet	1000	从 6480 个中随机抽取
	M6Doc	1000	从 2724 个中随机抽取
指代	HumanRef	4991	完整版本，无重复项
	RefCOCOg	4889	完整验证集

C. 指点基准

按照 LocateAnything [37]，的做法，我们还评估了在 Dense200、VisDrone、 [45]，COCO [22]，Human-Ref [16] 和 RefCOCOg [27] 上的点定位。本节介绍评估方案与结果，并解释 PinGround 为何以边界框作答。

设置。 我们使用 Dense200 的全部 200 张图像，以及 HumanRef 的全部 4,988 个不同指代表达。对于 VisDrone、COCO 和 RefCOCOg，我们分别选取发布文件中的前 500 张图像、1,000 张图像和 1,000 个指代表达；这些数据的每个样本实例数与完整数据集大致相当。LocateAnything 和 GroundAnything 的官方模型与 -SFT 模型以及带点作答的官方 Qwen3-VL。Qwen3-VL- SFT 即使被要求用点作答，也会以边界框作答，因此我们将其边界框和 PinGround 的边界框都取中心点。表 C.1 中的每张图像耗时按表 2。 的方法测量。

我们使用随 LocateAnything 发布的指点评估器对这些点进行评分 [37]，该评估器只会将一个点与其所属类别的实例匹配。PinGround 和两个 Qwen3- VL 基线每次只针对一个类别进行查询，LocateAnything 模型在其发布的指点评估中也是如此，因此点的类别就是其查询的类别。GroundAnything 模型则会同时查询所有类别，并为每个点标注一个名称，该名称决定了点的类别。如果一个点位于某个实例的掩码内，或在少数没有掩码的样本中位于其边界框内，则该点与该实例匹配。每个实例至多匹配一个点，而未匹配的点计为假阳性。与边界框的处理方式相同，精确率和召回率按样本取平均值，F1值则根据这两个平均值计算。在 COCO 上，人群区域不计为实例；其中未匹配任何实例的点也会被忽略，这与第A节一致。

结果。 在三个基础模型上取平均后，PinGround 使用其框中心达到了 68.5 的 F1值，而 -SFT 基线为 70.9（见表C.1）。在 LocateAnything 和 Groun- dAnything 上，-SFT 基线以点作为回答，PinGround 的 F1值分别比 LA-SFT 低 3.5 和比 GA-VLM-SFT 低 7.9。差距主要来自 COCO 和 Ref- COCOg，在这两项上，PinGround 分别比 LA-SFT 低 8.7 和 7.4

表 C.1. 五个数据集上的指向基准 F1值及每张图像平均时间。LocateAny-thing 和 GroundAnything 的官方版与 -SFT 行，以及 Qwen3-VL 的官方版行以点作答；其他行则以边界框作答，并将边界框视为其中心点。

解码	Dense	VisD.	COCO	HRef	RefG	平均值	秒/图像
LocateAnything-3B							
官方版，混合	83.8	58.0	72.8	71.6	76.3	72.5	1.66
LA-SFT，混合	84.1	55.1	74.0	72.1	74.0	71.9	1.74
PinGround ，slow	81.6	57.0	65.3	71.3	66.6	68.4	0.62
Qwen3-VL-4B-Instruct							
官方 36.9		37.1	74.0	57.0	77.4	56.5	53.82
Qwen3-VL-SFT	51.9	44.9	68.1	73.0	68.7	61.3	32.80
PinGround	66.4	53.0	65.7	73.3	69.4	65.6	2.01
GroundAnything-4B							
GA-VLM ，官方	89.9	69.1	84.7	91.3	88.7	84.7	3.97
GA-DLM，官方	88.1	61.2	84.5	91.0	74.2	79.8	1.89
GA-VLM-SFT	75.9	61.1	83.8	88.7	87.6	79.4	7.29
PinGround	82.4	59.3	68.8	75.8	71.2	71.5	0.78

F1值，分别比 GA-VLM-SFT 低 15.0 和 16.4；在 HumanRef 上，它分别低于二者 0.8 和 12.9 F1值。在 Dense200 上，PinGround 比 LA-SFT 低 2.5 F1值，但比 GA-VLM- SFT 高 6.5 F1值；在 VisDrone 上，它比 LA-SFT 高 1.9 F1值，但比 GA-VLM-SFT 低 1.8 F1值。在 Qwen3-VL 上，PinGround 和 Qwen3-VL-SFT 都以框中- 心作为回答，PinGround 比 Qwen3-VL-SFT 高 4.3 F1值，其中在 Dense200 和 Vis- Drone 上的优势最大，分别为 14.5 和 8.1 F1值。这两组数据也是它在边界框上领先 Qwen3- VL-SFT 最多的两组（见表2）。官方 Qwen3-VL 以点作为回答，在 COCO 和 RefCOCOg 上领先于 Qwen3-VL-SFT 和 PinGround；在这些数据集上，框中心经常会漏检其实例，如下文所统计。PinGround 保持了其速度，每张图像耗时 0.62、2.01 和 0.78 秒，而 -SFT 基线则分别耗时 1.74、32.80 和 7.29 秒。GA-VLM-SFT 的耗时远长于 GA-VLM，因为它经常重复之前的点；在 VisDrone 上，它针对 44 个实例每张图像写入 143 个点，而 GA-VLM 为 39 个。

将边界框中心视为点。即使边界框正确，边界框的中心也可能落在其实例之外。我们统计了框基准测试中 1000 张 COCO 图像（表 B.1）以及表 C.1 中 1000 条 RefCOCOg 指代表达中此类漏检的数量。在 LocateAnything 上，PinGround 的边界框与查询对应的真实标注框交并比达到0.5的边界框中，COCO 有 14.1%、RefCOCOg 有 24.4% 的框中心位于实例掩码之外；Qwen3-VL-SFT 的边界框也有相近比例，分别为 15.0% 和 23.1%。这些中心大多属于只占边界框很小部分的实例。对于 PinGround，在 COCO 和 RefCOCOg 上，掩码占边界框面积不足 40% 的实例分别占匹配实例的 18% 和 31%，却分别占掩码外中心的 58% 和 72%。按类别来看，PinGround 在 COCO 上的中心落在匹配对象之外的比例为：餐桌 56%、椅子 29%、人 12%（人是最常见的类别）。因此，对于形状紧凑的实例，边界框中心是可靠的点；但对于形状或遮挡导致边界框内有较大部分未被实例覆盖的实例（如椅子和餐桌），中心往往会落在背景或其他对象上。

以点作为答案。训练每个锚点用其所属实例内的一个点作答，可以避免这些漏检，但很难去除重复项。相邻锚点彼此重叠，且解码时看不到对方的答案，因此同一个实例可能会被多个锚点报告；PinGround 依靠非极大值抑制，仅保留这些重叠边界框中置信度最高的一个（第3.2节）。点只能通过距离进行比较，而同一个大型实例对应的点之间的距离，可能比两个相邻小型实例之间的距离还远。因此，任何固定距离要么会使同一个大型实例上保留多个点，要么会合并相邻的小型实例，两种情况都会降低 F1值。我们在 LocateAnything 上测试了这一点：使用 PinGround 的一个变体，让锚点以点的形式回答指向查询；该变体通过监督微调，在其自身的指向样本对以及来自 LocateAnything 训练数据的 4,000 个指向样本对上进行训练 [37]。与非极大值抑制类似，如果一个点位于同一查询下已保留的某个置信度更高的点的固定距离以内，就会将其丢弃。在表C.1的 COCO 和 Dense200 图像上，归一化坐标中的距离设为 10 时，大型实例上的重复项仍然存在，COCO 上的精确率为 8.8%。距离设为 150 时，Dense200 上的相邻实例会被合并，召回率降至 23.4%，而 COCO 上的精确率仍为 49.2%。我们测试的距离范围为 10 到 150，但没有一个距离适用于这两个数据集。重复项是并行解码的主要难点。点是一个没有面积的单一坐标，因此无法对其应用用于边界框的基于交并比的抑制方法；并行指点需要一种新的去重策略，这超出了本文的范围，将留待未来工作。

D. 与 UFO 的比较

UFO [33]也能在视觉语言模型上并行解码图像的多个点查询（第 2 节）。第 D.1 节详细介绍我们对 UFO 的评估，第D.2 节说明了这两种设计的差异，第 D.3 节则在 PinGround 中测试 UFO 为其点提供的局部图像特征。

D.1. UFO 的评估

我们使用官方代码运行已发布的 UFO-InternVL2.5-8B 模型，所用样本与表 2 中相同。该多任务模型以 1344 的分辨率针对 COCO 检测和分割进行训练，会将每张图像缩放至 1344×1344，并将其划分为九个 448×448 图块，每个图块包含 8×8 个网格点；保留 576 个点中每个点概率最高的三个类别，并对这些候选项中概率最高的 300 个进行解码

表 D.1. 已发布 UFO 模型在交并比 0.5 时的 F1值，与表 2 相同；仅保留得分不低于各列分数阈值的边界框。在网格模式下，所有网格点都会针对提示中的类别列表并行解码；在 COCO 上，该列表包含图像中查询的类别或全部 80 个类别，而在指代数据集上，则仅包含指代表达这一类别。定位头会依次为每个指代表达解码一个边界框，且不产生分数，因此其 F1值不受阈值影响。

分数阈值	0.1	0.2	0.3	0.4	0.5	0.6	0.7
物体，网格模式下的多任务模型							
Dense200	9.0	8.9	8.6	8.3	7.7	7.0	6.1
COCO，查询类别	51.8	62.2	67.9	70.7	72.4	73.3	73.7
COCO，全部 80 个类别	70.5	76.1	78.4	79.8	80.5	80.6	80.8
VisDrone	41.1	46.8	49.5	50.7	51.1	50.5	49.0
LVIS	27.0	29.8	30.8	31.3	31.5	31.2	30.8
指代，网格模式下的指代表达模型							
HumanRef	17.1	16.9	14.6	11.4	7.8	5.1	3.3
RefCOCOg	16.3	16.9	15.7	13.8	11.4	8.9	6.5
指代，其定位头的指代表达模型							
HumanRef	59.0						
RefCOCOg	88.7						

并行处理最多 30 个词元，并在每个类别内按交并比0.5执行软非极大值抑制。边界框的分数是其首个词元（即类别名称）的概率。在 COCO 上，其类别列表包含全部 80 个类别，与其自身评估时相同；我们保留图像查询类别对应的边界框。在其他数据集上，类别列表仅包含查询类别。在 VisDrone 上，若列出全部 10 个 VisDrone 类别，F1值为 50.6；列出查询类别时则为 51.1。其发布配置在每张图像上最多保留 300 个边界框，且不设分数阈值，因为其自身评估中的平均精度会按分数对边界框进行排序。F1值统计固定数量的边界框，因此需要分数阈值；所以表 2 保留分数不低于 0.5的边界框；该阈值在评估前固定。表 D.1报告了分数阈值从 0.1到 0.7时的 F1 值。在文本与布局数据集上，无论分数阈值如何设置，它都不会预测任何查询类别。该指代表达模型在指代和视觉问答数据上以 896 的分辨率进行微调，使用其官方定位头解码每个指代表达对应的一个边界框，不提供分数，也无需阈值。其网格模式将指代表达作为唯一类别，但只能找到少量目标；即使将点上任何非背景答案都视为该指代表达，结果也一样。每个样本都在一块 H100 GPU 上单独计时，包括图像加载、处理流程和后处理。

D.2. 与 PinGround 的区别

两者都在注意力掩码的作用下并行解码图像的点查询，使它们彼此分离；两者中的点都可以回答自己不包含任何对象。点如何进入模型也不是区别，因为 PinGround 在锚点以图像特征形式提供时同样适用，例如 UFO 中的做法（第D.3节）。它们的区别在于起点不同、并行解码的查询不同，以及点与实例的关联方式不同。

新模型还是通用策略。 UFO 添加了网格令牌，每一层都会再次加入其局部图像特征；使图像令牌之间的注意力具备双向性；定义了自己的输出格式，即类别名称和相对于点的边界框；并使用自己的多任务数据训练改动后的模型。PinGround 则是一种适用于通过文本输出进行定位的视觉语言模型的策略，例如表2中的三个基础模型，它们以不同格式书写坐标。PinGround 只修改提示和注意力掩码，保留基础模型的原生输出格式，并使用 47,170 对数据进行后训练；此外，每个基础模型都会与使用相同数据对训练的自身顺序解码版本进行比较。

哪些查询会并行解码。 UFO 的并行路径用于对提示中给出的类别列表执行检测和分割，而指代表达式则交由单独的模型按顺序解码出一个边界框（表D.1）。在 PinGround 中，锚点跟在普通提示中的查询之后，因此基础模型的查询——此处包括物体、文本行、布局区域和指代表达——都能与任意数量的目标一起并行解码（表2）。

点所报告的实例。 UFO 在训练期间通过匈牙利匹配将物体分配给网格点，并为所有类别解码同一组包含 576 个点的固定网格。PinGround 将分配规则写入查询中：匹配实例是中心位于锚点半径范围内且距离最近的实例，或者无。根据此规则，粗粒度锚点的答案决定是否细化其单元格，因此每个查询都会解码 100 个粗粒度锚点，并在每个占用单元格中解码 4 个细粒度锚点，锚点的分布也会随图像中的密度而变化。在 PinGround 中，在整张图像上使用单一细网格，每张图像耗时 1.14 秒，而非 0.71 秒，同时召回率和 F1值也更低（表4）。该规则还为每个锚点指定了各自的目标，使集合级强化学习能够根据每个锚点对集合 F1 的贡献为其分配得分。

D.3. 局部图像特征作为锚点

同一查询的并行流需要一个锚点来区分彼此，而 PinGround 通过位置来区分它们，即在每个查询中写入一个点（第3.1节）。UFO 也采用该点处的局部图像特征作为锚点 [33]，其与 PinGround 的其他差异见第 D.2节。我们在 LA 上进行测试，将每个锚点的两个坐标词元替换为一个占位标记，其输入嵌入为该特征；该特征由输入语言模型的视觉标记经双线性插值得到。与 UFO 不同，我们只在输入时添加该特征，不在每一层重复添加；保留 LA 中图像词元之间的因果注意力，并按照最近中心规则保留每个锚点的标签，答案则使用绝对坐标。只有锚点的形式发生变化，而配对、预

表 D.2。PinGround 在 LA 慢速模式下使用坐标锚点和图像特征锚点时的 F1@0.5；监督微调后和强化学习后，在 12个框基准测试上 F1@0.5 、精确率、召回率和 F1@mIoU 的平均值，以及每张图像平均时间。

锚点	Dense	COCO	VisD.	LVIS	HierT.	TotalT.	SROIE	IC15	DocL.	M6Doc	HRef	RefG	F1	P	R	mIoU	秒/图像
SFT																	
坐标	80.7	73.4	60.1	66.0	78.0	68.6	87.0	47.2	90.6	87.1	79.8	85.4	75.3	70.6	84.2	59.9	0.72
图像特征	71.6	74.5	58.7	67.2	76.2	70.4	81.6	50.2	90.2	88.5	81.2	85.9	74.7	73.4	79.6	59.6	0.70
监督微调 +强化学习																	
坐标	81.0	74.7	62.5	66.7	78.8	71.3	89.6	50.3	91.5	89.0	82.7	86.7	77.1	72.6	84.9	61.4	0.71
图像特征	78.4	75.6	61.6	67.8	79.3	71.5	88.1	50.9	89.3	89.6	81.2	86.6	76.7	75.2	80.9	61.0	0.68

先固定树、网格、训练方案、解码和评估，均与 PinGround 相同。

两种锚点的表现几乎相同（表 D.2）。监督微调后和强化学习后，特征锚点的F1值分别达到74.7和76.7，而坐标锚点分别为75.3和77.1，前者精确率更高、召回率更低。在许多实例彼此靠近的Dense200上，特征锚点的差距最大，−9.1 监督微调后的F1值为 −2.6 强化学习后的F1值。按照表 2 中的计时方式，监督微调后和强化学习后，特征锚点每张图像分别耗时0.70秒和0.68秒，而坐标锚点分别耗时0.72秒和0.71秒。

E. 定性示例

图 E.1 展示了PinGround在五个框基准数据集的图像上，采用慢速模式，在监督微调后和强化学习后的LA边界框。PinGround在Dense200上找到了全部74颗樱桃，在VisDrone上找到了两栋高楼周围街道上的大多数小型车辆，并在LVIS上找到了甜甜圈及其盘子。在HumanRef上，它只框出了坐在桌边的四个人；在TotalText上，它找到了绘制标牌上的每一个字，包括弯曲的文字。监督微调后，更多边界框落在背景上，或落在查询未要求的实例上，例如VisDrone中高楼的窗户，以及HumanRef中站在桌子后面的人；强化学习则移除了其中大多数边界框。

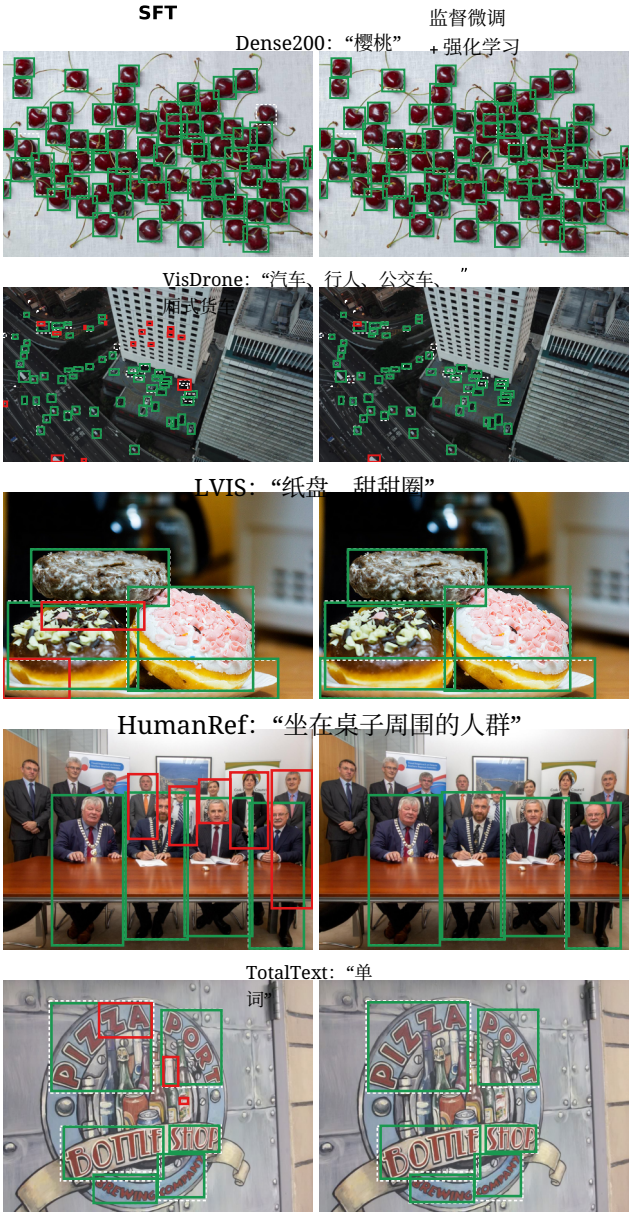


图 E.1。PinGround 在经过 SFT 后（左）和经过 RL 后（右）于 LA 上的边界框。白色虚线框为真实标注，绿色和红色边界框分别表示交并比为 0。5 时的正确预测和错误预测。