arXiv:2601.06461v1 [cs.CR] 10 Jan 2026

VIPER Strike：通过结构化视觉–语言推理攻克视觉推理 CAPTCHA*

Minfeng Qi†、Dongyang He†、Qin Wang♭, ✉、Lefeng Zhang†
† 澳门城市大学 ♭ *CSIRO Data61*

摘要

视觉推理 CAPTCHA（VRCs）将视觉场景与自然语言查询相结合，要求对物体、属性和空间关系进行组合推理。它们正日益被部署为抵御自动化机器人的主要防线。现有求解器分为两种范式：视觉中心型方法依赖特定模板检测器，但无法应对新颖布局；而推理中心型方法借助大语言模型，却难以进行细粒度视觉感知。两者都缺乏处理异构 VRC 部署所需的通用性。

我们提出 VIPER，这是一种将结构化多对象视觉感知与自适应大语言模型推理相结合的统一攻击框架。VIPER 解析视觉布局，将属性与问题语义对应起来，并在模块化流程中推断目标坐标。在六个主要 VRC 提供方（VTT、极验、网易、丁香、数美和小盾）上的评估表明，VIPER 的成功率最高达到 93.2%，在多个基准测试中接近人类水平的性能。与以往的求解器 GraphNet（83.2%）、Oedipus（65.8%）和整体方法（89.5%）相比，VIPER 始终优于所有基线方法。该框架在不同的大语言模型骨干网络（GPT、Grok、DeepSeek、Kimi）上也保持了稳健性，准确率持续超过 90%。

为预判防御措施，我们进一步提出模板空间随机化（TSR），这是一种轻量级策略，可在不改变任务语义的情况下扰动语言模板。TSR 能显著降低求解器（即攻击者）的表现。我们提出的设计为打造人类可解、机器难以破解的验证码提供了方向。

1 引言

人机交互证明（HIPs）长期以来一直是抵御互联网自动化滥用的重要防线。在过去二十年中，验证码（*Completely Automated Public Turing test to tell Computers and Humans Apart*）机制已从扭曲文本识别逐渐发展为日益复杂的视觉挑战，以应对迅速发展的机器学习模型，双方展开了一场军备竞赛 [1–6]。

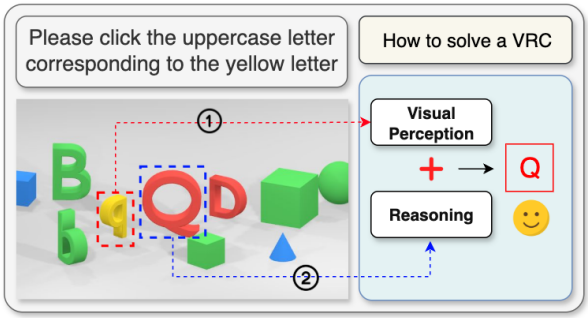


图 1：**视觉推理验证码（VRC）示例**。求解器 1 识别黄色小写字母（感知）；2 将其与对应的大写形式匹配（推理），得出正确答案“Q”。

在此背景下，一类新型视觉推理验证码（VRC）应运而生，成为实践中最具挑战性的基于挑战的防御之一。VRC 会向用户呈现视觉场景，并配以自然语言指令，要求用户针对空间、比较和语义属性进行多步推理 [7–10]。腾讯防水墙 [11] 和极验 [12], 等主要服务商广泛部署了这类验证码，以保护大规模在线服务，每日处理数百万次挑战 [13]。

VRC 带来了巨大挑战。与主要考验单项感知技能的传统验证码不同，视觉推理验证码同时要求解决现代 AI 中最棘手的两个开放问题：细粒度视觉感知 和 语言引导的组合推理。求解器不仅必须在杂乱场景中定位并区分多个物体，还必须理解以自然语言表达的多约束指令，并将其正确对应到视觉证据上。

*已被USENIX Security'26接收； ✉ 供通讯作者使用 or.

如图 1所示，求解器必须先 1 识别黄色小写字母，再 2 推理语义变换（“转换为大写形式”），得出正确答案“Q”。VRC 提供方还会随机变换对象类别、布局和措辞模式 [4, 8], ，使针对单一模板的调优失效。

VRC为何仍然重要。 与此同时，reCAPTCHA [14] 等基于行为的系统已在全互联网部署中占据主导地位，实际上覆盖了约88%的网站 [13]。然而，普及并不意味着鲁棒性。越来越多的既有研究表明，reCAPTCHA式机制日益容易受到基于学习的攻击，在现实威胁模型下，报告的成功率已超过70% [15–17]。

这种反差在我们自己的测量中同样明显。在针对广泛使用的CAPTCHA类型进行的GPT-4o零样本评估中（附录表 7），reCAPTCHA等行为中心挑战的成功率接近50%，而VRC则表现出显著更强的抵抗力，准确率仅降至31%。这些结果表明，相较于许多已广泛部署的替代方案，VRC处于更难的前沿，因此成为理解自动化视觉语言推理真实极限以及基于挑战的防御未来发展的关键目标。

现有范式存在局限。 早期的机器学习（ML）攻击采用模块化流程来攻击视觉推理验证码。Holistic模型 [8] 将 BiLSTM网络与基于规则的推理相结合，并报告称在腾讯VTT上的成功率最高可达 88% [11], 随后通过改进模板匹配加以扩展 [9]。最近，VRC-GraphNet [18] 将物体作为节点、关系作为边，并通过图神经网络进行求解。尽管这两种方法在其目标平台上表现有效，但都存在根本性局限：它们仅编码浅层启发式方法，而非真正的推理，因此无法捕捉查询中的组合语义。

GPT-4o 和 Grok-2 [19, 20] 等大语言模型的兴起，正挑战这一棘手的人工智能问题。这些模型兼具强大的推理能力与视觉感知能力，在视觉问答和图表解读基准测试中表现出色 [21–25]。原则上，这些能力理应自然迁移至视觉推理验证码。事实上，Oedipus [26] 是首个利用大语言模型解决视觉推理验证码的框架，它将查询分解为符号子任务，以便进行思维链推理。然而，尽管这一创新颇具开创性，其性能仍然有限：在腾讯VTT上，其平均成功率仅为 63.5%，远低于人类准确率。由于依赖大语言模型原生编码器，且缺乏稳健的视觉 grounding，Oedipus 容易受到遮挡、杂乱场景以及细粒度属性区分的影响。

这两种范式都无法兼具鲁棒性与通用性-。基于机器学习的求解器往往会对表层特征过拟合，并且在查询措辞变化以及面对异质的视觉推理验证码时表现失常（参见附录B.1）。相比之下，基于大语言模型的求解器缺乏精确的视觉定位能力，且常常误解细粒度的空间信息或属性细节。要弥补这些不足，需要一种集成式框架，将可靠的视觉感知与结构化推理相结合，从而在各种视觉推理验证码中保持稳定表现。

我们的方法。 我们提出 VIPER（***V***视觉***I***通过感知进行推理并借助 ***P***感知与***E***嵌入式 ***R***推理），这是一种通用攻击框架，将结构化视觉感知与符号推理解耦，并通过自适应提示词将二者重新连接。VIPER 使用高精度多对象检测器，从视觉推理挑战图像中盘点候选对象（形状、颜色、方向、坐标）。轻量级解析器从查询中提取语义约束，明确编码属性筛选条件和空间关系。随后，由针对推理类型定制的自适应提示词引导大语言模型处理这一结构化表示。在六个主要的 VRC 提供方中，VIPER 的准确率最高可达 93.2%，且始终优于以往的自动化求解器；在多个基准测试集上，其性能接近人类水平，同时在各种不同的视觉推理验证码中保持了较强的鲁棒性。

我们的贡献如下：

▷ 一种用于破解 *VRC* 的统一框架 (§4)。我们提出 VIPER，这是一种模块化视觉语言框架，将视觉推理挑战求解建模为组合式的感知—推理 任务。通过结合多对象检测与提示词自适应的大语言模型推理，VIPER 无需逐平台重新训练，即可泛化到各种不同的 VRC 模板。

▷ 多平台基准测试(§5.1)。我们整理了一个包含 6,000 个 VRC 挑战的基准测试集，涵盖六个主要平台，每项挑战均标注了标准答案和物体边界框。此外，我们还发布了一个配套数据集，其中包含 1,200 个经过多目标标注的 VRC 场景，用于训练和评估感知模块。这些资源共同为跨平台比较和求解器架构的可复现开发提供了首个标准化基础。

▷ 全面评估(§5.3)。我们从四个维度评估 VIPER：组件消融、基线比较、大语言模型后端替换和响应时间。在所有设置下，我们的框架均取得了最先进的性能，与现有求解器相比，展现出更高的准确率和鲁棒性。

▷ 防御方法。(§6)。我们提出模板空间随机化（TSR），这是一种轻量级防御方法，通过同义词替换、关系改写和迂回表达来扰动问题措辞。针对 600 个定制问题开展的初步实验表明，成功率持续下降，凸显了当前语言—视觉对齐的脆弱性。

我们为何开展这项工作。基于行为的系统（如 reCAPTCHA、视觉推理验证码）已得到广泛部署，用于保护高流量服务。然而，我们表明，与许多替代方案相比，这些系统面临着更难攻克的挑战，但缺乏系统性的安全评估。我们的工作可以记录当前的局限性，引导供应商朝着更稳健的设计迈进，并为人机区分机制提供明确的伦理参照点。

2 技术预热

2.1 基于大语言模型的推理

大语言模型在理解、生成和推理自然语言方面能力出众。近期模型（如 GPT-4o、Gemini-2.5）在需要常识推理、指令遵循和空间语言理解的任务中，表现接近人类水平。

形式化地，令$\mathcal{Q}$表示自然语言查询的空间，令$\mathcal{A}$表示可能答案（如文本标签、数值或坐标）的空间。大语言模型可表示为一个映射：$f_{\text{LLM}} : \mathcal{Q} \to \mathcal{A}$, 其中模型估计：

$$f_{\text{LLM}}(q) = \arg\max_{a \in \mathcal{A}} P(a \mid q; \theta), \tag{1}$$

其中 θ 表示推理过程中固定不变的预训练模型参数。实际应用中，f_{LLM} 通过从提示词q开始的自回归解码来实现，并可选择辅以系统级指令。

多模态大语言模型扩展将这一映射推广至联合处理语言和视觉输入。令I表示图像输入的空间（如原始像素或符号描述）。多模态大语言模型定义为：

$$f_{\text{LLM}} : \mathcal{Q} \times I \to \mathcal{A}, \tag{2}$$

使模型能够对成对的图像文本输入进行推理。当外部视觉模块将视觉信息预处理为结构化符号表示后，问题便简化为仅基于语言对高层视觉事实进行推理，使大语言模型能够利用其预训练的语义和逻辑先验，而无需针对视觉进行再训练。

大语言模型还支持多种推理模式：

- 基于属性的筛选：根据颜色、形状或文本标签等指定属性选择物体。
- 空间与比较推理：解读位置或关系描述（例如“在……左侧”、“在红色正方形上方”、“其中最大”）。
- 逻辑推理：运用布尔组合、否定、排序或多条件约束。

2.2 多目标检测

多目标检测器是我们框架中的感知骨干。它将检测表述为从图像像素到边界框坐标和类别概率的回归问题。给定输入图$I \in \mathbb{R}^{H \times W \times 3}$，检测器将图像划分为$S \times S$网格，并为每个单元预测$B$个候选边界框，每个框由（$x$，$y$，$w$，$h$）参数化以表示框的几何形状，同时还包含目标置信度分数C，以及C_{cls}个类别的类别概率。最终输出张量的维度为$S \times S \times B \times (5 + C_{\text{cls}})$，其中 5 表示（$x$，$y$，$w$，$h$，$C$）。坐标（$x$，$y$）相对于网格单元进行归一化，而（$w$，$h$）则相对于整幅图像进行归一化。

训练过程由复合损失引导：

$$\mathcal{L} = \lambda_{\text{loc}} \mathcal{L}_{\text{bbox}} + \lambda_{\text{obj}} \mathcal{L}_{\text{obj}} + \lambda_{\text{cls}} \mathcal{L}_{\text{cls}}, \tag{3}$$

其中，$\mathcal{L}_{\text{bbox}}$ 用于惩罚边界框回归误差，$\mathcal{L}_{\text{obj}}$ 用于衡量目标置信度质量，而$\mathcal{L}_{\text{cls}}$ 是分类损失。超参数（$\lambda_{\text{loc}}, \lambda_{\text{obj}}, \lambda_{\text{cls}}$）用于平衡这些项。

推理时，检测器会输出带有置信度分数的候选边界框。非极大值抑制（NMS）会筛除冗余预测结果，从而得到最终集合：

$$\mathcal{D} = \{(c_k, s_k, b_k)\}_{k=1}^{K}, \tag{4}$$

其中，c_k 是预测类别，s_k 是置信度分数，而$b_k =$（x_1, y_1,x_2,y_2）是图像空间中的边界框。

3 威胁模型

攻击者（VRC 求解器）。我们假设对手是外部实体，无法特权访问目标服务的内部基础设施、专有 CAPTCHA 生成逻辑或私有训练数据。交互仅通过公开可访问的接口进行，例如基于网络的用户身份验证门户，并遵守对合法用户施加的相同网络和会话限制。其目标是以足以削弱验证码安全目的的速率求解 VRC 挑战，将原本专供人类完成的验证步骤自动化。

然而，对手可以在正常使用限制范围内收集任意数量的 VRC 实例，捕获所有渲染的视觉内容及其配套的自然语言提示词，并使用其控制的任何计算资源进行处理。这包括应用预训练大型多模态模型、集成专用感知模块，以及采用符号推理或程序化推理流程。交互模型严格采用黑盒方式：攻击者会收到关于提交答案正确性的二元反馈，但无法获取中间模型状态、梯度，也无法直接控制挑战构造。

我们不考虑利用软件漏洞、后端入侵或侧信道数据窃取，这些均不属于 VIPER 的研究范围。

防御者（VRC 提供方）。我们将防御者建模为部署视觉推理验证码以防范

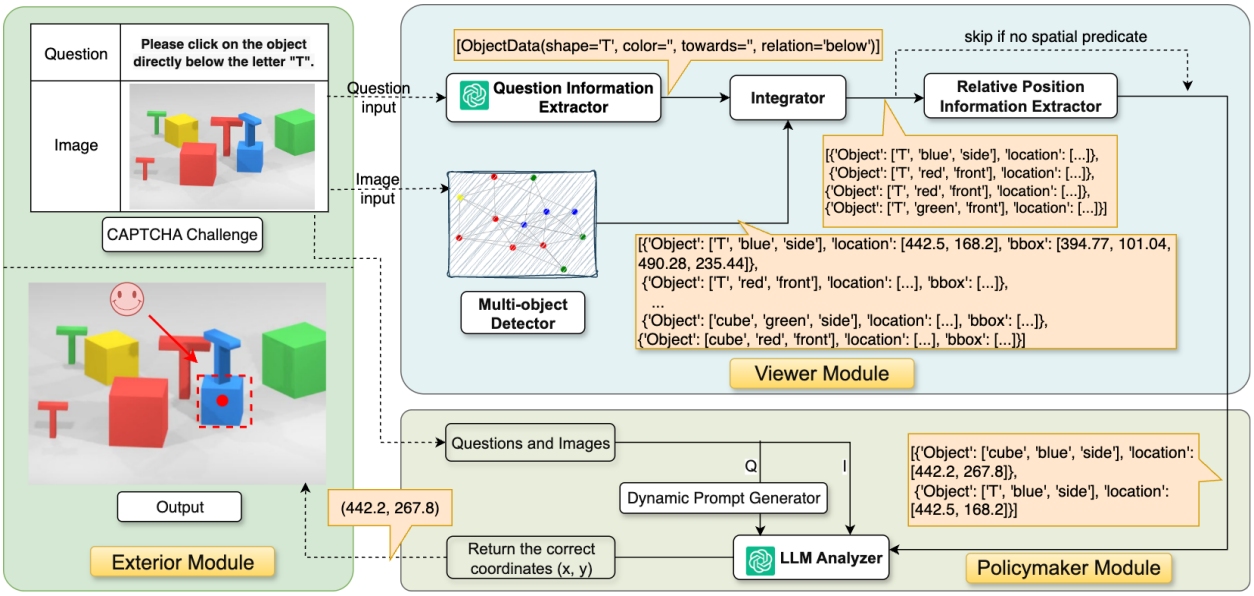


图 2：VIPER 端到端流程。Exterior 接收图像 I 和指令 q。Viewer 执行结构化感知：多目标检测器生成带标签的边界框 V；QIE 将 q 解析为属性槽；Integrator 将这些槽与 V 对齐，得到候选集 V′；如果 q 包含空间短语，RPIE 会通过几何投影解析参照物（§4.2- §4.4）。决策器根据 q 和 V ′组合任务条件提示，向大语言模型发起查询，并输出坐标 (x, y)（§4.5）。

自动化滥用的服务运营方。防御者控制挑战的生成与呈现，实施速率限制，并进行服务器端验证。假设所有挑战内容（包括图像和指令）对用户可见，因此客户端攻击者也能看到。防御者可以从可用的 VRC 模板中进行选择。

防御者的目标是在无需人工干预的情况下，阻止自动化求解器达到人类水平的成功率。我们的研究评估了当前 VRC 部署在对抗情形下对这一目标的实现程度。

4 我们的 VIPER

我们提出了 ViPer，这是一个模块化框架（图2），输入为一张 VRC 图像 I 及其自然语言提示词 q，并返回图像空间中的点击坐标 $(x,\ y)$。我们的设计明确地将感知与推理解耦：视觉模块提取结构化物体及其属性，而查询感知接口仅将与任务相关的表征传递给大语言模型。

4.1 系统架构

VIPER 由三个关键模块组成。

❶**Exterior.** Exterior 负责与挑战环境交互。它接收渲染后的图像及相关问题，将输入规范化为内部数据模式，并格式化最终答案以供提交。此层不实现任何求解器逻辑。

❷ **Viewer.** Viewer 构成感知骨干。它将原始图像转换为包含物体及其属性的结构化清单，并使该清单与提示词的语义意图相匹配。一个经过训练的 多目标检测器(§4.2) 会返回一组带标签的假设，其中每个假设都包含一个复合语义标签（来自表1本体的形状–颜色–方向）、一个置信度分数，以及一个带中心点的像素空间边界框；清单1中展示了一个代表性的序列化示例。

随后，检测器的输出会经过两个步骤的筛选，并在必要时进行补充。

- 首先（§4.3），问题信息提取器 ***(QIE)*** 会将q 解析为结构化查询，列出指令中提及的属性和类别。随后，***Integrator*** 会根据 QIE 约束筛选$\mathcal{D}$，仅保留语义相关的检测结果；查询中未指定的属性视为通配符，而“字母”、“数字”或“三维对象”等粗粒度类别则通过预定义类型映射表进行解析。

- 其次（§4.4），当提示词仅通过与参照物的空间关系来定义目标时（例如，“小灰色立方体左侧的物品”），相对位置信息提取器***(RPIE)*** 会通过从参照物向指定方向投射，并检查检测到的边界框是否包含投射位置，来推断候选目标。结果是一组精简的候选对象 V'，每个对象均由规范化属性元组、边界框和中心坐标表示。

❸**决策器。** 决策器是推理引擎

表 1：**复合标签属性本体**。 形状、朝向和颜色 源自六家 VRC 提供方。

标签	标签值
形状	0–9、a–z、A–Z、球体、立方体、圆锥体、圆柱体、六棱柱 多面体、长方体、倾斜、正四棱锥， 环面、三棱柱、三棱锥、正方形、平行四 边形、圆形、矩形、六边形、梯形、菱形， 三角形、五角星、五边形
朝向前方	, side
颜色	黄色、绿色、灰色、蓝色、红色

(§4.5）。它会对解析后的查询所指示的推理类型（例如空间推理、比较推理或属性匹配）进行分类，构建一个动态提示，其中包含原始指令以及来自 Viewer 的序列化候选集，并调用大语言模型来选择目标。输出被限制为图像中的一个坐标 (x, y)。为减少歧义，提示词会说明预期的推理模式，并且仅提供解决该指令所需的属性。

4.2 结构化感知

感知骨干网络是一个单一的多目标检测器，配置为输出复合类别标签，直接对应 VRC提示中提及的属性。每个标签都编码了三元组 SHAPE×COLOR×ORIENTATION，确保每次检测都包含自然语言指令中最常被查询的属性。本体包含 91 个类别，涵盖字母数字符号和几何基元，并设有五种颜色和两种标准姿势。在标签层级编码属性，避免了事后融合单独的预测结果，并为下游推理提供了稳定的接口。

训练设置。为实现跨平台泛化，我们汇集了来自六家提供方的标注图像，构建统一的训练语料库，并通过将特定于平台的标签映射到复合标签空间（附录 B.1），对异构模式进行归一化。所有图像均调整为 640×640，并采用保持纵横比的填充。检测器通过复合损失进行训练，该损失联合优化定位、目标性和分类；同时采用标准数据增强（马赛克、翻转、颜色抖动），以提升模型在不同视觉风格下的鲁棒性。

推理输出。 在推理时，网络会为每幅图像生成一组假设（公式 4）

$$\mathcal{V} = \{(\ell_k, s_k, b_k)\}_{k=1}^{K},$$

其中 ℓ_k 是复合标签，$s_k \in [0,1]$ 是置信度，$b_k = (x_1, y_1, x_2, y_2)$ 是像素空间中的边界框角点。对于每个保留的检测结果，都会计算中心点 $c_k = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right)$，并将其与（$\ell_k$，$s_k$，$b_k$）一同存储。~~Viewer~~ 将这一结构化输出进行序列化为最简记录格式，并将其传递给 Integra- tor，以进行候选项筛选，并在适用时与后续的关系线索进行对齐。一个真实且具有代表性的序列化示例见清单 1。

```
[
  {"对象": "圆柱体",
   "位置": [366。9, 165。7],
   "边界框": [335.7237854003906, 127.5312576
       2939453, 398.072265625, 203.8830718
       9941406]},
  {"对象": "6",
   "位置": [434。3, 171.2],
   "边界框": [401.0405578613281, 118.4748535
       15625, 467.61608886871875, 223.98490
       90576172]}
]
```

清单 1：**VRC 实例的序列化检测输出**，包括复合标签（对象）、像素空间中心（位置）和边界框（边界框）。

4.3 语义提取与对齐

此阶段将自然语言指令与 Viewer 检测到的物体关联起来。首先将指令解析为结构化查询，然后将其与检测器输出进行对齐，以筛除无关物体。当目标仅通过其与参照物的关系来定义时，此阶段还会编码解析下一步中隐含目标所需的信息。

问题信息提取器。QIE 读取自然语言指令 q，并将其转换为若干条简短记录，用于指明文本中提到的物体。每条记录包含三个字段——形状、颜色和方向；如果指令未指定某个字段，则将该字段留空，之后视为通配符。QIE 还会记录简单的关系短语（例如“下方”和“左侧”），以便下游模块了解何时需要进行空间步骤。该流程如图3所示：指令由 QIE 解析，然后由 Integrator 与检测结果进行对齐；如果存在关系，则通过相对位置步骤进行增强。我们提供了两个问题示例。

($Q1$) 对于“请点击字母‘T’正下方的对象”，QIE 会为参照字母输出一条记录，其形状为=字母 T，颜色/方向为空，并将关系标记为“下方”。Integrator 会汇集所有检测到的字母 T，而相对位置推理模块会逐一检查每个字母 T 的正下方，以找出目标。

（$Q2$）对于“请点击与红色圆锥体朝向一致的数字”，QIE 会输出两条记录：一条对应参照物（形状=圆锥体，颜色=红色），另一条对应候选集（形状=数字，其他字段为空）。Integrator 会将“数字”扩展为具体的

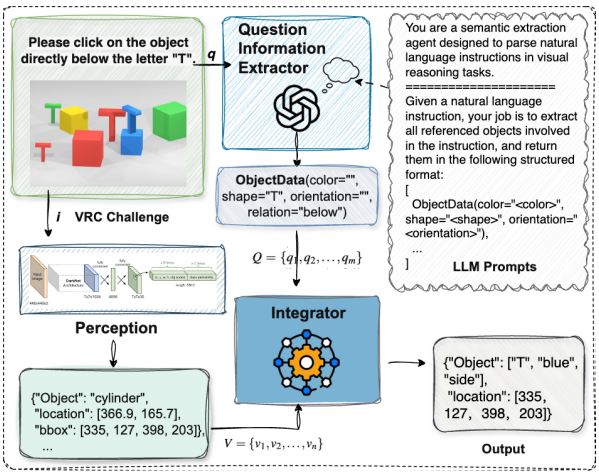


图 3：**语义提取与对齐**。 QIE 将查询解析为槽位；Integrator 将它们与检测器输出进行匹配，得到候选集 V'。

数字标签，随后决策器在选择最终坐标时应用“方向一致”检查。

Integrator。 它充当过滤机制，连接多目标检测器的视觉输出与 QIE 提取的语义意图。其主要功能是根据结构化 QIE 输出所指示的推理任务，从图像中所有检测到的对象里，仅识别出语义相关的对象。Integrator 充当选择器，动态匹配对象级属性，同时抑制无关的视觉信息。

给定检测器检测到的对象集合 $V = \{v_1, v2, \ldots\ldots, v_n\}$，其中每个 v_i 包含对象、位置和边界框等字段，以及来自 QIE 的一组结构化对象查询：$Q = \{q_1, q2, \ldots\ldots, q_m\}$，Integrator 按照以下逻辑进行匹配：

- 对于每个 $q_j \in Q$，Integrator 选择所有满足以下条件的 $v_i \in V$ q_j 中的所有非空字段均与对应的 v_i中的属性匹配。
- q_j 中的字段若为空（例如 形状 = “”），则将其视为通配符，并在匹配时忽略。
- 对于 QIE 返回的一般类别描述（例如，“字母”、“数字”、“三维对象”），Integrator 配备预定义类型映射，以识别相关的视觉标签。

这种部分匹配方案使 Integrator 即使在问题描述不充分或指代抽象类别时也能正常运行。

4.4 相对位置推理

仅靠属性匹配，无法处理目标仅由其与另一对象之间的关系来定义的提示词。在这种情况下，Integrator 会检索参照物（例如字母“T”），但真正的目标（例如“正好位于

算法1 相对位置信息提取器 输入：指令 q；来自 Integrator 的参照集 R；检测结果 V 输出：扩充集 V_{rel} 1: $V_{rel} \leftarrow \emptyset$; r ←提取_空间_关系(q) 2: 如果 r =None 则 3: 返回 R 4: 结束如果 5: 对每个 $r_i \in$R 执行 6: p ← center(r_i.边界框); ˆp ← shift(p,r Δ) 7: 对每个 v j ∈V 执行 8: 如果 ˆp ∈v_j.边界框 则 9: 将 r_i 和 vj 添加到 V_{rel}; 跳出 10: 结束如果 11: 结束循环 12: 结束循环 13: 返回 V_{rel}

当“在字母 T 下方”未出现在候选集中时，相对位置信息提取器（RPIE）会将语言关系转换为针对已检测边界框的几何测试，以弥补这一缺口。

关系检测。RPIE 使用 QIE 标记的关系线索（图3）。当 QIE 指示“左侧”、“右侧””、“上方””或“下方”等空间谓词时，RPIE 会执行几何投影；否则，它会原样返回 Integrator 的输出。如果措辞不规范且缺少关系线索，则会应用轻量级模式匹配作为后备方案，以找回相同的谓词。

几何投影。 给定一个检测到的关系，令R表示参照物 检测结果的集合，令V 表示检测结果的全集。对于每个参照物$r_i \in R$，其边界框为$b_{r_i} = [x_1, y_1, x_2, y_2]$,，RPIE 计算其中心

$$p_{r_i} = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right).$$

根据关系标记r，其中标记r ∈{左侧、 右侧、 上方、下方}，RPIE 推导出一个方向向量 $\vec{d}(r)$，并对边界框应用几何测试。例如，如果r =位于左侧，则目标边界框必须满足$x_{\max}(v_t) < x_{\min}(v_r)$，其中$v_r$ 是 Integrator 返回的参照物。其他三个谓词也适用类似的约束。

$$u(r) = \begin{cases} (-1, 0), & r = left\ of \\ (1, 0), & r = right\ of \\ (0, -1), & r = above \\ (0, 1), & r = below \end{cases}$$

并形成探测点

$$\hat{p}_{r_i} = p_{r_i} + \Delta u(r),$$

其中， $\Delta > 0$ 是以像素为单位测量的固定偏移量。随后，RPIE 按固定的遍历顺序扫描V，并选择第一个边界框包含 ˆp_{r_i} 的检测结果v_j。如果存在这样的v_j，则将其添加为推断目标，并与r_i 关联，以便

下游推理能够明确看到这一对；如果没有边界框包含 $\hat{p}_{r_i}$，则不会为该参照物推断目标。

坐标模型与测试。 边界框以图像空间中的角点表示 $[x_1, y_1, x_2, y_2]$ 左上角像素为原点，x 向右递增，y 向下递增。算法 shift$(p, r\,\Delta)$ 中的算子 1 是更新 $\hat{p} = p + \Delta u(r)$。点在边界框内的谓词为

$$\text{contains}(\hat{p}, [x_1, y_1, x_2, y_2]) \triangleq (x_1 \le \hat{p}_x \le x_2) \wedge (y_1 \le \hat{p}_y \le y_2).$$

如果 $\hat{p}$ 位于图像边界之外，则跳过对该参照物的检查。如果多个边界框都包含 $\hat{p}$，则采用遍历顺序中的第一个匹配项，以保持过程稳定。

输出。 RPIE 会生成一个增强候选集，其中包含原始参照物以及通过投影推断出的任何目标，并附带参考对象—目标对象链接。随后，将该集合与指令一并传递给决策器；决策器利用任务感知提示中的显式配对来选择最终坐标。

4.5 提示条件推理

决策器是最后一个阶段。它接收原始指令以及上游生成的推理线索（例如 QIE 标记的空间谓词和 RPIE 生成的参考对象—目标对象链接），并构造任务感知提示，引导大语言模型返回图像空间中的单个坐标。交互严格限定为仅输出答案：模型必须以像素为单位输出一对 $(x,\ y)$，图像原点位于左上角。

动态提示（图4）。提示词并非采用固定模板，而是根据从指令中推断出的推理类型进行调整。对于空间查询（例如“左侧”、“右侧”、“上方”、“下方”），提示词会指出关系和参照物，并说明答案必须是相应位置物体的坐标。对于比较查询（例如“最大的球体”、“较小的圆锥体”），提示词会明确指出比较属性，并要求模型识别符合条件的唯一物体。对于直接属性查询（例如“红色圆锥体”），提示词会指定属性合取，并要求给出相应坐标。在所有情况下，提示词都会包含明确的答案格式（“仅按以下格式输出一个坐标：(x,y)，单位为像素”），提醒模型只能依据所描述的场景作答（不得使用未呈现的物体或作出假设），并说明坐标约定（整数像素；x 向右递增，y向下递增）。

大语言模型推理（或**Analyzer**）。Analyzer 接收三个输入：任务条件提示p、图像I和剪枝后的候选集V'。它会返回一个像素坐标。根据公式2，我们有

$$f_{\text{LLM}}(p, I, V') \to (x, y).$$

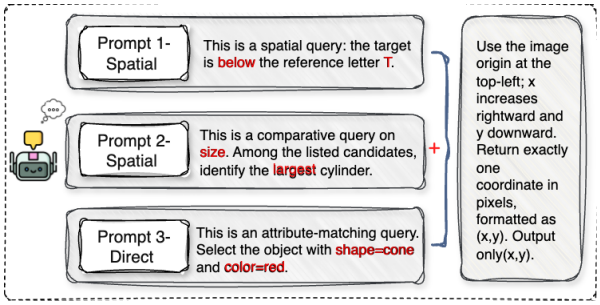


图 4：**决策器的任务条件提示**(§4.5)。空间 提示词会指出关系和参照物；比较提示词会定义属性和目标类别；直接属性提示词会绑定形状和颜色。

在此接口中，p 用于指定推理模式，并将答案格式限定为像素坐标；V' 将决策面约束在一个较小的序列化候选集内，其中每个候选项都包含一个属性三元组（形状–颜色–方向）和一个中心位置。I 在需要视觉定位时提供（例如，有助于利用像素信息的尺寸或姿态线索），但 Analyzer 通常仅凭V'就能做出判断。模型只运行一次；不要求生成思维链。

在此设置中，系统通过明确的决策规则处理不同类型的查询。对于直接属性查询，系统会将指定字段与V'中的属性进行精确匹配，并返回唯一匹配对象的中心点。对于空间查询，系统依赖上游生成的关系和参考对象—目标对象关联：如果存在关联，系统会输出关联目标对象的中心点；如果没有可用关联，Analyzer 会在V'中沿适当的轴比较对象中心，以几何方式执行所述关系。对于比较查询，系统会对V'中的边界框计算所请求的统计量：“最大”通过取最大值来确定

$$\text{area}(b) = (x_2 - x_1) \cdot (y_2 - y_1),$$

而“最左/最右”则通过比较中心点的x坐标来判定，“最上/最下”则通过比较y 坐标来判定。在所有情况下，选定候选对象的中心点（x,y）都会作为最终答案输出。

5 实验

5.1 设置

基准测试集。 我们构建了一个统一的基准测试集，以确保不同系统之间评估的公平性和可复现性。该基准测试集汇集了六家商业 VRC 提供方的公开数据集。我们使用 Selenium 从腾讯官方 VTT 服务中抓取了 12,000 个样本 [11]，并纳入了来自开源 VRC 基准测试集的另外五个数据集 [18]：小盾（12,000 个样本）、极验（12,000 个）、网易（12,000 个）、丁香（12,000 个）、

表2：**数据集概览**：六个真实世界 VRC 平台。

数据集	物体（每张图像）	对象类型	提示词多样性	推理复杂度	典型挑战
VTT [11]	10–20	三维形状、字母、数字	High	High	空间推理、相对位置、抽象
小盾 [27]	12–14	形状、字母、数字	中等	中等–高	物体堆叠、遮挡、属性 + 空间混合
极验 [12]	7–10	规则三维形状	中等	中等	遮挡、空间关系不明确
网易 [28]	5–7	形状、字母、数字	中等	中等	颜色/方向筛选、属性匹配
丁香 [29]	5	形状、字母	Low	低–中等	仅涉及位置的查询、简单关系
数美 [30]	6	规则三维形状	Low	Low	属性筛选（最小的 +颜色 +形状）

注：**对象堆叠** 指多个物体垂直堆叠、需要进行重叠视觉解缠的验证码图像 被裁切或被遮挡的项目s. w).
属性 **+ 空间混合**表示问题同时涉及属性筛选（例如颜色、形状）和空间推理（例如左侧、上方、下方

以及数美（3,000 个）。这些数据集涵盖了多样的视觉风格、问题形式和难度级别，为现实世界中的 VRC 部署提供了广泛覆盖（参见表 2）。我们公开发布了完整的基准测试集，包括原始数据和标注[1]，以促进后续研究。附录 B.1 详细介绍了每个提供方的数据格式和挑战特征。

规范化。我们将所有数据集重构为统一的、以物体为中心的视觉语言格式。原始输入被转换为标准化的兼容检测器的表示，其中每个物体的边界框均标注了颜色、形状和方向。对于缺少边界框标注的数据集，我们使用定制界面手动标注物体，以确保不同来源之间的一致性。为进行评估，我们通过从每个提供方均匀抽取 1,000 张图像，并为每张图像标注一个正确的点击区域，构建了一个“单答案”真实标注集。如果输出坐标落在该区域内，则预测被视为正确。所有模型、基线方法和消融实验均采用此数据格式和评分协议进行评估。

基线方法。我们将 VIPER 与三种代表性的视觉推理验证码求解器进行比较，这些求解器涵盖了主流攻击范式：感知驱动、基于图的推理和大语言模型驱动的流程。为确保一致性，我们在基准测试集上重新实现了所有基线方法，以保证公平性。

- **基于视觉的求解器**。Holistic模型 [8] 通过将检测与基于规则的逻辑相结合，开创了模块化攻击方法，在腾讯VTT等平台上取得了最高88%的成功率。我们重新实现了这一流程及其扩展版本 [9]。与此同时，VRC- GraphNet [18] 将检测结果编码为视觉语言图，其中节点表示物体，边捕捉空间关系或语义关系。随后训练图神经网络以选择目标节点。我们通过对检测结果进行归一化，并在我们的基准测试集上重新训练图神经网络，对这一方法进行了改进。

- **由大语言模型驱动的求解器**。Oedipus [26] 代表推理中心型攻击，将 VRC 查询转换为领域专用语言，并通过大语言模型使用思维链提示。我们重新实现了完整流程并对其进行了评估，同时将骨干模型与 VIPER 对齐，以分离框架层面的差异。

[1] https://zenodo.org/records/17971722

5.2 方法变体消融

我们定义了两种简化的求解器（图5），通过移除特定组件使其区别于 VIPER，同时保持所有其他设置不变。

R1：仅使用大语言模型。指令q 和图像 I 直接提供给多模态大语言模型，不经过任何结构化感知或对齐。提示词内容极为简略，仅要求输出单个图像空间坐标（x，y）。该变体不包含检测器输出，也没有 QIE/Integrator/RPIE 步骤。它完全依赖模型自身的视觉能力来解析场景并解决查询，因此在处理较小或重复的物体、颜色或姿态等细粒度属性，以及仅通过文本表达的空间关系时容易出错。

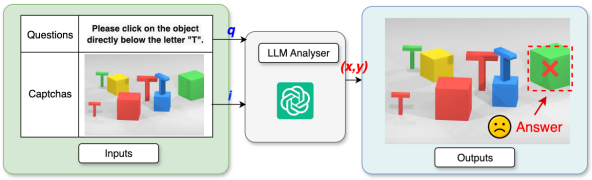
R2：检测器 → 提示词。检测器会列出I 中的所有物体，并为其标注复合标签、边界框和中心点。这个未经筛选的列表会附加到提示词指令中。不进行 QIE 解析、Integrator 剪枝或 RPIE 推理。系统要求大语言模型返回一个（x，y）。虽然这会明确提供候选对象，但提示词中通常包含许多无关或近乎重复的条目，且未明确处理关系线索；当属性仅部分匹配，或目标仅通过相对于参照物的位置来定义时，可能会造成混淆。

5.3 VIPER 评估

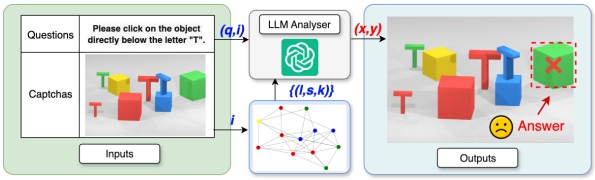
配置。我们的实验在一个一致的公开基准测试集上进行，该基准测试集涵盖六个商用视觉推理与控制平台（§5.1）。实验环境包括一颗 15 核 Intel Xeon Platinum 8474C CPU 和一块 NVIDIA RTX 4090D 图形处理器（24GB），用于模型训练；推理则使用 AMD Ryzen 9 7845HX CPU 和 RTX 4060（8GB）。我们使用 PyTorch v2.5.1 和 CUDA 12.1，所有大语言模型调用均通过公共 API 进行。

我们从四个维度对 VIPER 进行评估：

1 组件消融。为隔离各项设计的贡献，我们比较了三个逐步扩展的变体：R1（仅基础 Viewer）、R2（增加关系推理）和 R3（完整的 VIPER）。使用一致的图像—问题输入，在所有数据集上测量准确率。



(a) **仅使用大语言模型（R1）**：将指令和图像直接传递给多模态大语言模型；不使用结构化感知或对齐。



(b) **检测器→提示词（R2）**：检测物体，并将检测结果（标签、边界框、中心点）注入提示词，但不使用 QIE/Integrator/RPIE。

图5：**消融变体示意图。** R1 仅依赖原生多模态视觉和零样本推理。R2 在提示词中加入检测器输出，但省略语义过滤和相对位置推断。

2 基线比较。 我们在同一基准测试集上，将 VIPER（GPT-4o 后端）与三种先前的验证码求解器进行基准测试：整体模型 [8,9], VRC-GraphNet [18],和 Oedipus [26],。此外，我们报告了人工基线：十名参与者从同一基准测试语料库中随机抽取挑战进行求解。该基线为可达到的准确率提供了实证上限，也为模型延迟提供了参照（参见 §B.2）。 3 大语言模型后端替换。 为评估 VIPER 推理模块的鲁棒性和可移植性，我们使用四种最先进（SOTA）的大语言模型后端进行测试：GPT-4o、DeepSeek-Chat、Kimi-latest 和 Grok-2。每个模型接收相同的动态提示词和 Vie wer 输出，且不启用思维链。 4 响应时间。 我们测量所有大语言模型后端的端到端平均推理时间（每个图像—问题对），以评估延迟权衡。

实验结果。三种测试协议下的所有实验结果汇总于表3中。1组件消融（图6）。我们的消融分析表明，~~VIPER中逐步加入的~~每个组件都能在全部六个VRC平台上带来稳定的准确率提升。以GPT-4o作为后端时，仅使用原始视觉输入的基线配置（R1）准确率较低，从数美的7.1%到VTT的31.4%不等。通过检测在R2中引入结构化感知后，性能显著提升，所有平台的准确率均超过60%，数美达到77.1%。完整模型（R3）进一步实现大幅跃升，在各平台上的准确率达到86.5–98.5%。

表3： **六个 VRC 平台上各项基准测试的准确率（%）**（ =总体最佳； =总体第二； =总体最差； –=无数据）

基准测试集	后端	VTT	极验	网易	DingX	数美	小豆
Holistic [8,9]	–	88.12	85.74	86.37	88.52	95.84	79.28
GraphNet [18]	–	89.47	76.92	72.18	85.34	99.87	89.65
Oedipus [26]	GPT-4o	70.23	61.47	66.82	73.16	58.44	64.08
	DeepSeek	63.72	55.93	60.37	68.14	57.26	59.53
	kimi	72.48	65.27	69.15	74.63	62.83	70.38
	Grok-2	66.42	59.87	62.74	70.94	60.18	63.27
人类	–	87.60	90.70	95.12	95.40	93.20	91.80
Ablation-R1	GPT-4o	31.36	26.80	16.41	24.87	7.07	10.99
	DeepSeek	28.42	24.36	18.47	23.12	8.63	11.82
	kimi	32.74	27.92	19.88	24.97	9.12	12.96
	Grok-2	29.18	27.44	17.32	26.38	6.77	10.46
消融实验-R2	GPT-4o	75.79	61.75	66.86	73.98	77.08	62.31
	DeepSeek	69.75	58.24	63.12	70.46	77.84	60.24
	Kimi	67.85	55.36	61.42	68.24	70.42	58.33
	Grok-2	72.21	63.15	64.53	74.82	72.24	59.67
R3（VIPER）	GPT-4o	97.33	86.45	92.61	91.48	98.45	92.59
	DeepSeek	92.96	83.25	93.00	87.88	97.50	87.76
	kimi	88.00	85.20	91.92	89.20	95.50	87.86
	Grok-2	92.50	81.91	89.00	86.00	96.50	92.00

这表明，将检测、空间推理和层次化提示相结合是必不可少的，尤其有助于解决杂乱视觉场景中的关系查询。

消融方案的有效性可推广至其他后端。DeepSeek 呈现出相同的上升趋势：准确率从 8.6–28.4%（R1）提升至 58.2–77.8%（R2），最终达到 83.3–97.5%（R3）。Kimi 的初始基线略高（在 VTT 上最高达 32.7%），但在 R3 中收敛至较低的上限 85.2–95.5%，表明其与结构化提示的兼容性较弱。Grok-2 同样展现出稳步提升，从 6.8–29.2%（R1）提升至 59.7–74.8%（R2），最终在 R3 中达到 81.9–96.5%。尽管不同后端之间存在特定差异，例如 Kimi 的平台期表现较弱，而 Grok 在极验和丁香上表现突出，但所有后端从 R1 到 R3 均持续单调提升，这证实了我们的分阶段集成策略既有效又稳健。

◦ **分析**。除总体准确率外，消融结果还揭示了各阶段明显的失败模式。从 R1 到 R2 的准确率大幅提升表明，R1 中的错误主要源于感知失败：直接处理原始图像的大语言模型经常会错认较小、重复出现或外观相似的物体，进而导致指代选择错误。虽然 R2 通过将推理建立在明确的候选对象之上，缓解了大多数此类错误，但当多个物体都部分符合查询属性时，仍会出现残余错误，尤其是在仅位置关系的情况下。从 R2 到 R3 的进一步提升表明，这些残余错误主要与推理有关

而非感知。层次化提示和空间推理通过明确编码关系约束，帮助区分那些似是而非的候选对象。

2 与基准测试集比较（图7）。VIPER 在全部六个 VRC 基准测试中都创下新的 SOTA 表现，始终优于以往方法，准确率也接近人类水平。

在推理要求最高的 VTT 数据集上，VIPER（GPT-4o 后端）的准确率达到 97.33%，比人工基线（87.60%）高出 10 个百分点，并优于 Oedipus（70.23%）、GraphNet（89.47%）和 Holistic（88.12%）。与表现最强的既有大语言模型求解器相比，绝对准确率提升超过 27 个百分点。

在极验上，VIPER 的准确率达到 86.45%，接近人工基线（90.70%），远高于 Oedipus（61.47%）和 GraphNet（76.92%）。在侧重方向和颜色匹配的网易上，其准确率为 92.61%——略低于人类（95.12%），但远超 Oedipus（66.82%）和 GraphNet（72.18%）。在空间关系较简单的丁香上，VIPER 的准确率达到 91.48%，缩小了与人类（95.40%）之间的差距，并以超过 18 个百分点的优势胜过 Oedipus（73.16%）。

在数美平台上，由于物体多样性较低且重复提示较多，VIPER 的准确率达到 98.45%，略高于 Holistic 方法（95.84%），并远超 Oedipus（58.44%）。最后，在小盾平台上，VIPER 的准确率达到 92.59%，优于 Oedipus（64.08%），并与人类表现（91.80%）十分接近。

◦ **分析。** 基准测试集结果表明，VIPER 的提升具有系统性，而非仅针对特定数据集。最大的提升出现在推理密集型基准测试集（例如 VTT 和网易）上，这些数据集主要涉及多个物体之间的关系和组合约束，说明仅有结构化感知并不足够，还需要推理。在某些平台（例如极验和丁香）上，与人类表现之间仍存在差距，这在很大程度上是由含糊或描述不充分的查询造成的，因为可能有多个物体满足约束条件。在这种情况下，人类往往会依赖语境偏差，而 VIPER 则严格遵循形式化的查询语义。因此，这些错误反映的是挑战设计本身存在的固有歧义，而非感知或推理能力的不足。

3大语言模型骨干网络比较 (图8)。采用 GPT-4o 的 VIPER 表现最为稳定且成绩最佳。在 VTT（97.33%）、极验（86.45%）、丁香（91.48%）、数美（98.45%）和小盾（92.59%）上均取得领先的准确率，既展现了应对高复杂度推理任务的稳健性，也体现了处理较简单空间挑战的可靠性。采用 DeepSeek 的 VIPER 表现也颇具竞争力，在网易（93.00%）上超越其他方案，并在 VTT（92.96%）和数美（97.50%）上保持较高分数。不过，它在不同基准测试集上的准确率波动更大，在极验（83.25%）和丁香（87.88%）上的表现较低在极验（83.25%）和丁香（87.88%）上的数值。

VIPER 搭配 Kimi 的表现较为均衡，但优势不够突出：它在各数据集上的表现都较为可靠，在网易和极验上分别达到 91.92% 和 85.20%，但在任何单一平台上都未能领先。它在 VTT（88.00%）和小盾（87.86%）上的相对劣势，凸显了其在推理密集型或对象密集型场景中的能力有限。最后，VIPER 搭配 Grok-2 在小盾（92.00%）和数美（96.50%）上表现出色，在这些场景中可与表现最佳的模型相媲美。然而，其表现并不均衡，在丁香（86.00%）和极验（81.91%）上的成绩明显下降，成为该组中最低的分数。

◦**分析。** 骨干模型比较表明，VIPER 能从更强的大语言模型中受益，但并非完全依赖它们。GPT-4o 的整体表现最佳，尤其是在 VTT 和极验等推理密集型基准测试中，这表明其对组合查询具有更强的鲁棒性。DeepSeek 在属性聚焦型数据集（如网易）上表现良好，但在不同平台间的方差更高，说明其容易受到空间歧义的影响。Kimi 和 Grok-2 的表现则更不均衡，在复杂布局上的性能上限较低或出现大幅下滑，表明它们与结构化感知输出的兼容性较弱。这些结果表明，问题源于后端特定失效模式，而非单一的主导瓶颈。

4响应时间比较（图9）。我们进一步比较了 VIPER 与先前视觉推理验证码求解器的平均响应时间，以评估计算开销。现有方法（如 Holistic 和 GraphNet）表现出极低的延迟，平均完成时间低于 0.2 秒。然而，这些方法是以准确率为代价实现这一速度的，因为它们依赖浅层启发式方法或基于图的匹配，而非完整的推理流程。相比之下，Oedipus 展现出更长的响应时间（平均 3 秒），同时整合了更先进的感知模块，但在解决高复杂度挑战方面仍显不足。

由于采用由大语言模型驱动的多阶段推理流程，VIPER 的延迟较高。根据骨干模型的不同，响应时间为 5–11 秒（表5），其中 GPT-4o 提供最快的响应（平均 5–6 秒），而 Kimi 和 DeepSeek 的执行速度较慢（9–11 秒）。Grok-2 处于中间水平，平均需要 7–9 秒。值得注意的是，VIPER 的总体延迟（跨平台平均为 8.3 秒）仍与人工基线（7.0 秒）相当，这表明其额外计算成本与人类问题求解的计算成本处于同一数量级。

◦ **分析。** 响应时间比较凸显了延迟与推理能力之间固有的权衡。Holistic 和 Graph-Net 等轻量级求解器依赖浅层

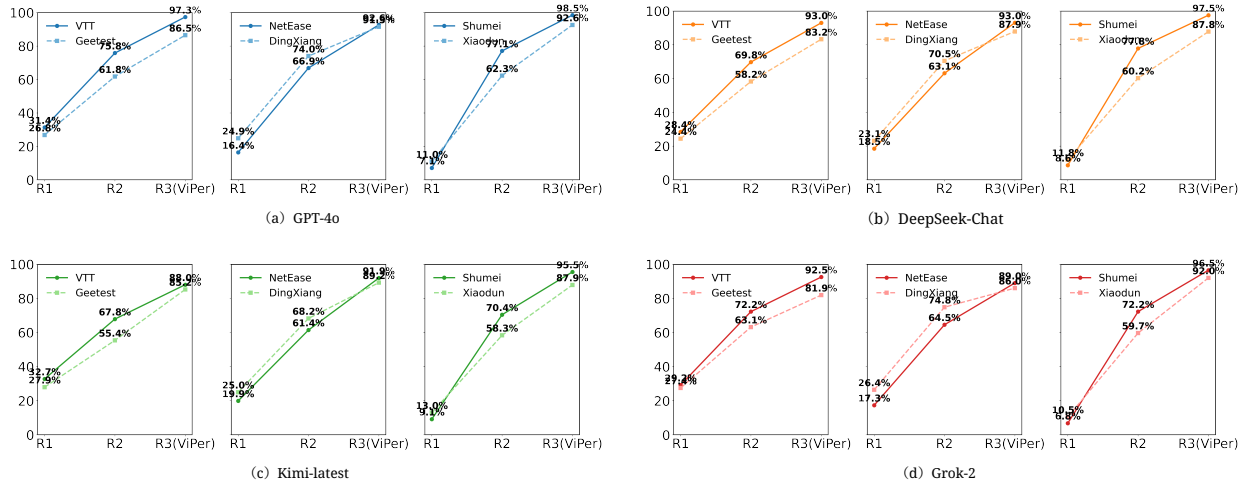


图 6：**VIPER框架在四个大语言模型后端上的组件级消融实验。** 每个面板报告 R1（仅大语言模型推理）、R2（结合结构化感知）和 R3（完整 VIPER）的准确率。

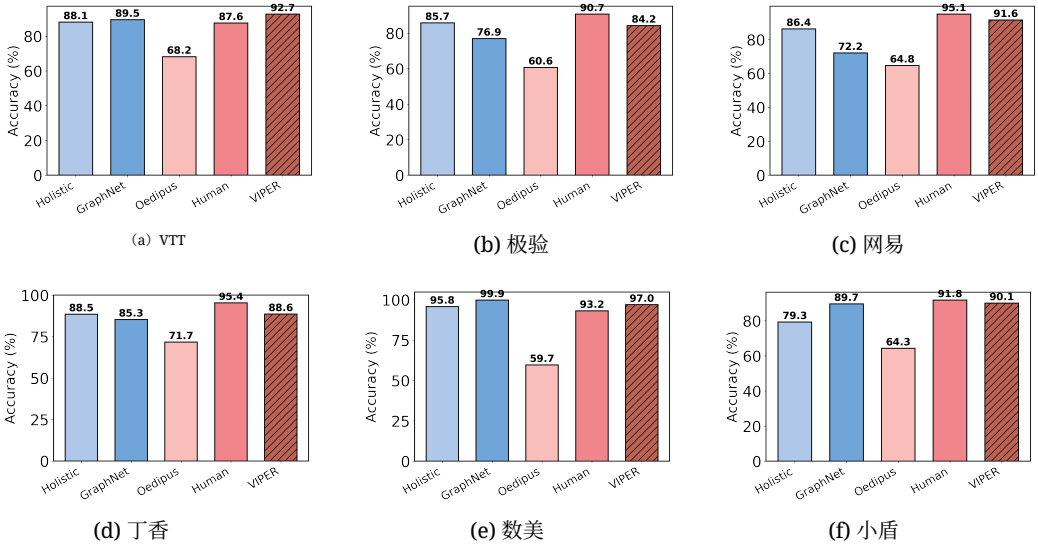


图 7：**与六个 VRC 平台基准测试集的比较。**每个子图报告不同方法的平均准确率，并用斜线柱突出显示 VIPER。

启发式方法，因此在推理密集型视觉推理验证码上的表现不佳。相比之下，VIPER 由于需要进行感知和大语言模型推理，延迟较高，但这些开销带来了显著更高的准确率。值得注意的是，VIPER 的端到端延迟仍与人工解题时间相当，这表明其计算成本反映的是任务本身的内在复杂性，而非效率低下。

5.4 多对象检测器评估

训练与架构（表1）。我们在一个精心整理的 1,200 张已标注视觉推理挑战图像数据集上训练了统一的多目标检测器，这些图像从六家提供商处均匀抽取：VTT、丁香、极验、网易、数美和小盾（每家 200 张）。数据集划分为 800 个训练样本、200 个验证样本和 200 个留出测试样本，以确保评估均衡。训练前，所有图像均调整为 640×640 像素。

检测器采用现代单阶段无锚框 YOLO 风格架构（YOLOv11 [31]），由用于多尺度特征提取的卷积骨干网络、特征聚合颈部，以及用于分类和边界框回归的解耦检测头组成。每个检测到的对象都会被赋予一个复合标签，用于编码其形状、颜色和方向。训练使用单块 NVIDIA RTX 4090 GPU 和 AdamW 优化器，批量大小为 32，训练轮次为 3000。

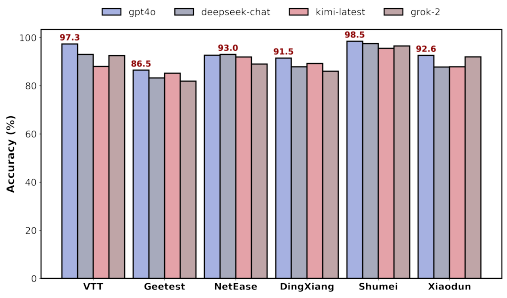


图 8：**四种大语言模型后端的准确率比较**。所有求解器共享相同的视觉感知与推理流程，仅大语言模型后端有所不同。

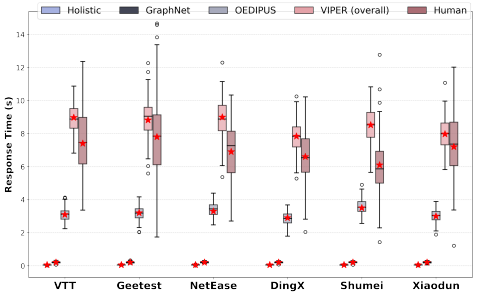


图 9：**响应时间比较**五种以上求解器在六个 VRC 平台上的平均响应时间。

评估 （表 4）。为评估我们的多目标检测器的性能，我们报告整体 COCO 风格指标，以及按目标尺度和属性分层统计的结果。检测器的精确率为 0.998，召回率为 0.956，mAP@50 为 0.984，mAP@50:95 为 0.919。分层分析表明，检测器在不同尺度下表现稳定：小型物体的 $AP_s = 0.912$，中等物体的 $AP_m = 0.944$，大型物体的 $AP_l = 0.963$，表明其对目标尺寸变化具有很强的鲁棒性。AR@100 和每张图像的真阳性/假阳性等属性级指标，进一步凸显了检测器以极少假阳性捕捉细粒度特征的能力。

此外，各平台评估（C 部分）展现出较强的跨域泛化能力：例如，VTT 和 DingX 上的 mAP@50:95 分别达到 0.965 和 0.951，而在极验和小盾上仍分别达到 0.907 和 0.857。该检测器在所有平台上的 mAP@50 也接近完美（≥ 0.95），六个验证平台上的均值± 标准差为 0.932 ± 0.009。运行时性能分析（D 部分）进一步证实了该模型的高效性：输入尺寸为 640 时，模型包含 25.6M 个参数，计算量为 72.4 十亿次浮点运算，在图形处理器上的单张图像延迟为 7.8 ms（128 张图像/秒），非极大值抑制的开销仅为 0.9 ms。此外，附录B.3中的图12和图13展示了具有代表性的定性表现。

本体范围与泛化讨论。VIPER 中的多对象检测器运行于固定的本体之上，该本体包含 91 种复合类别，用于编码形状、颜色和方向。该本体源自对六家主要 VRC 提供方的实证分析，涵盖图像中出现的视觉基元以及其查询所引用的语义属性，因此反映的是主流 VRC 部署的共同结构，而非任意的设计选择。

该检测器展现出强大的跨平台泛化能力（表 4），表明该本体捕捉的是提供商不变特征，而不是对特定模板过拟合。当未来的视觉推理验证码引入新的视觉关系（例如未见过的形状、颜色和关系）时，其影响仅限于感知层：本体之外的对象会被保守地过滤掉或误分类，从而产生更小的候选集，而不是生成虚假的目标。攻击成功率会因感知错误而下降，但下游的感知—推理流程和推理逻辑保持不变。

表 4：**检测器验证摘要。**

（A）总体 COCO 风格指标

精确率	召回率	mAP@50	mAP@50:95	$F1_{opt}$	最佳轮次
0.998	0.956	0.984	0.919	0.968	178

(B) 分层平均精度（按尺寸/属性划分）

AP_s	AP_m	AP_l	AR@100	每张图像的真阳性数	每张图像的假阳性数
0.912	0.944	0.963	0.931	3.52	0.11

（C）各平台平均精度均值

VTT	极验	网易	DingX	数美	小盾
IoU 阈值为 *0.5* 时的平均精度均值					
0.999	0.998	0.998	1.000	0.950	0.958
IoU 阈值为 *0.5* 至 *0.95* 时的平均精度均值					
0.965	0.907	0.910	0.951	0.922	0.957

（D）运行时间统计

# 参数量	浮点运算次数	Lat.	吞吐量	NMS	均值±标准差
25.6M	72.4G	7.8ms	128/s	0.9ms	0.932 ± 0.009

AP_s，*AP_m*，*AP_l* 分别表示小型、中等和大型物体的平均精度；浮点运算次数 表示在 640×640 分辨率下的浮点运算次数；延迟 和吞吐量 分别指延迟和吞吐量；非极大值抑制 是非极大值抑制的缩写。

6 针对 VRC 攻击的防御措施

防御方法。 视觉推理验证码求解器的关键弱点在于依赖狭窄的语言模板。一旦模型见过足够多以固定风格表述的指令，就能将这些提示词映射到可预测的视觉定位模式，从而绕过预期的推理过程。为应对这一问题，我们提出模板空间随机化（*TSR*），这是一种轻量级防御方法，能够扩展挑战的语言表述范围，同时不改变其语义。TSR 从三个方面丰富提示词：（i）对物体属性进行同义词替换（例如，将“红色”替换为“猩红色”）；（ii）进行关系改写和极性翻转（例如，将“在……左边” ↔ “不在……

表 5：各验证码平台的**平均响应时间**（秒）。结果以均值 ±标准差表示。

求解器	VTT	极验	网易	丁香	数美	小盾
Holistic	0.05 (±0.01)	0.05 (±0.01)	0.05 (±0.01)	0.05 (±0.01)	0.05 (±0.01)	0.05（±0.01）
GraphNet	0.20 (±0.05)	0.20 (±0.05)	0.20 (±0.05)	0.20 (±0.05)	0.20 (±0.05)	0.20 (±0.05)
Oedipus	3.10 (±0.40)	3.20（±0.45）	3.30 (±0.42)	2.90 (±0.38)	3。50 (±0.44)	3。00 (±0.41)
VIPER (gpt-4o)	5.31 (±1.39)	5.10 (±1.44)	6.17 (±1.53)	6.06（±0.95）	5.54 (±1.12)	4。42 (±0.89)
VIPER（DeepSeek）	10.20 (±1.12)	9.80 (±1.35)	9.90 (±1.25)	9.10 (±1.40)	8.50 (±1.10)	8.70 (±1.30)
VIPER（Kimi）	9.80（±1.43）	11.40（±1.77）	9.58（±3.02）	7.20（±0.20）	11.37（±2.45）	9。27（±2.00）
VIPER（Grok-2）	8。57 (±1.11)	8。97 (±1.10)	8.33（±0.35）	7.00 (±2.07)	6.67 (±1.20)	6.50 (±1.85)
VIPER（总体）	**8.97**	**8.82**	**8.99**	**7.84**	**8.52**	**7.97**
人工基线	7。40 (±2.10)	7.80 (±2.30)	6.90 (±1.90)	6.60 (±1.80)	6.10（±1.70）	7.20（±2.00）

右边”）；以及（iii）采用可选间接指引，要求用户解析一个额外的参照物（例如，“点击与 T 左边圆锥体颜色相同的对象”）。

评估。 我们使用基准测试集中的 100 张原始 VRC 图像，对 TSR 进行了初步评估。对于每张图像，我们随机生成了六种问题提示变体，分别对应三个转换维度：物体属性的同义词替换（S）、关系改写与极性翻转（R），以及可选间接指引（I）；同时还包括这些维度的两两组合（S+R、S+I）和完整组合（S+R+I）。由此构建了一个包含 600 条测试查询的受控基准测试集。值得注意的是，视觉内容和标准答案均保持不变；我们仅通过 TSR 改变问题的语言表层形式。随后，我们使用四种有代表性的求解器——VIPER（GPT-4o）、Oedipus、GraphNet 和 Holistic——应对这些挑战，以初步评估它们在 TSR 下的鲁棒性。尽管测试规模有限，这些结果仍初步表明，仅语言表层形式的变化就能显著降低求解器性能，而不改变任务语义。

表6：**TSR有效性（成功率，%）**。六个平台的平均值。S = 同义词；R = 关系改写/极性；I = 迂回表达。

求解器	原始	S	R	I	S+R	S+I	S+R+I
VIPER（GPT-4o）	93.60	91.20	90.10	89.50	86.40	86.20	85.20
Oedipus	65.80	61.00	56.90	58.70	49.10	50.80	45.30
GraphNet	83.20	81.00	79.80	78.40	71.60	74.10	68.40
Holistic	89.40	87.70	88.30	87.90	86.90	86.50	84.90

研究结果。 TSR 会在不同程度上降低求解器性能（表 6）。VIPER（GPT-4o）仍相对具有韧性，在完整的 S+R+I 设置下，性能仅从 93.60% 小幅下降至 85.20%。Holistic 的稳定性也相近，仅从 89.40% 略降至 84.90%。相比之下，Oedipus 更为脆弱，性能从 65.80% 降至 45.30%；GraphNet 则从 83.20% 降至 68.40%。结果表明，即使底层图像和正确答案保持不变，问题提示中的简单语言变异性也会显著影响求解器性能。换言之，对表层语言变化的鲁棒性仍是一项尚待解决的挑战，这凸显出求解器往往依赖脆弱的语言—视觉对齐，而非更深层的语义理解。

7 局限性与未来工作

首先，我们使用先进的大语言模型来评估所提出框架的能力上限。我们没有系统地研究较小或轻量级的大语言模型骨干网络，因此无法充分区分VIPER的模块化设计与先进模型本身的推理能力各自的贡献。了解在资源受限场景下，模型规模缩小时性能如何下降，仍是未来研究的重要方向。

其次，本研究的另一项局限是缺乏系统的计算成本分析。VIPER采用多阶段流程，将多目标检测器与基于大语言模型的推理相结合；使用大语言模型时，这不可避免地会带来不小的推理延迟和API成本。我们的评估侧重于准确率和鲁棒性，并未量化成本—性能权衡。因此，成本敏感型分析留待未来工作开展。

第三，VIPER关注的是一种特定类别的基于挑战的验证码，即视觉推理验证码。虽然视觉推理验证码是实践中最具挑战性且部署最广泛的验证码类别之一，但我们的研究结果无法直接推广到行为型或交互式系统等其他范式。将感知—推理框架扩展到更广泛的验证码类别，或探究此类扩展的局限性，仍是一个有待解决的研究问题。

最后，我们的实验仅评估了所提出的 TSR 防御对自动化求解器的有效性，并未开展人类受试者研究，以验证经过混淆或重定向的语言对于合法用户而言是否仍然直观且可解。因此，

TSR 的实际可用性尚无法得到充分证实，而评估此类防御下人类的成功率和认知负荷仍是未来的研究方向。

8 相关工作

验证码演变。验证码的设计长期以来都体现了防御方案与自动化求解器之间的军备竞赛。早期的文本验证码利用光学字符识别的弱点，引入扭曲、杂乱元素和背景噪声 [32, 33]。随着视觉模型不断改进，图像型验证码将挑战转向物体识别和语义理解，但模型能力的提升削弱了其鲁棒性，同时加剧了可用性问题 [4, 34, 35]。为平衡安全性与用户体验，研究人员引入了交互式验证码，依赖被认为超出自动化能力的更高层次推理任务 [36–39]。近年来，reCAPTCHA v3 [14] 放弃了谜题，转而采用基于风险的隐形检查，通过分析行为生物特征（例如光标动态、击键节奏）来透明地分配信任评分。此外，相关研究还探索了对抗性验证码 [40–42],其中嵌入了人类无法感知的扰动，以误导自动化求解器。

针对验证码的攻击。大量研究证明，文本验证码方案容易受到自动化攻击。Gao 等人 [43] 表明，微软的双层验证码可在 9.05 秒内以 44.6% 的准确率破解；Tang 等人 [2]则开发了针对文本验证码的快速深度学习攻击方法。后续研究利用验证码合成器生成合成挑战样本，用于训练基线求解器 [44]。更先进的方法则采用循环一致生成对抗网络 [45],、掩码自动编码器 [46], 、卷积神经网络架构 [47, 48], 和无监督学习 [49],，为各种验证码格式构建端到端求解器。研究还表明，即使是抗分割设计也仍然容易受到攻击 [5]。

除了文本型方案，图像型验证码也容易受到攻击。研究人员结合视觉对齐技术（例如模板匹配、边缘检测）与对类人鼠标动态的模仿，绕过了滑块验证码 [50, 51]。研究人员还利用在增强数据集上训练的深度学习模型，或通过对离散角度区间进行分类，攻破了基于旋转的验证码 [52, 53]。

基于大语言模型的视觉推理。 近期的进展通过将结构化感知与大语言模型的组合能力相结合，拓展了视觉推理 [54–56]。视觉问答（VQA） [57, 58] 和视觉蕴含 [59] 等早期任务凸显了将语言查询与视觉语义对齐的必要性。知识增强基准测试集（例如 OK-VQA、A-OKVQA）进一步揭示了外部依据的重要性 [60, 61]。近期的视觉–语言模型，如 Flamingo [62],、PaLM-E [63],和 BLIP-2 [64] ，表明大语言模型推理可以与预训练感知模块结合，但实现稳健集成的系统性方法仍有待探索 [23, 24]。

9 结论

我们提出了 VIPER，这是一种将多目标检测与自适应大语言模型推理相结合的感知-推理攻击框架。通过将视觉基础与符号推理解耦，VIPER 无需重新训练即可泛化至异构视觉推理验证码。我们通过四种协议对其进行了评估：(i) 组件消融，结果显示准确率单调提升，最高达到 93.2%；(ii) 基准测试集对比，性能优于 GraphNet（ 83.2%）、OEDI-PUS（65.8%）和 Holistic 基线（89.5 %）；(iii) 大语言模型骨干网络替换，在 GPT-4o、DeepSeek、Kimi 和 Grok 上均保持稳健性能（>90%）；(iv) 响应时间分析，延迟与人类解题相当。总体而言，VIPER 在大多数数据集上的准确率与人类相当，并为视觉推理验证码求解器确立了新的上限，凸显了构建既便于人类求解、又能抵御机器破解的视觉推理验证码防御机制的迫切性。

伦理考量

本研究探讨 VRC 的安全性。VRC 是一种广泛部署的人机交互证明。我们认识到，自动化验证码求解研究具有双重用途：它有助于设计更强健、可由人类求解的防御机制，但也可能被滥用于绕过防护措施。我们的目标是推动负责任的风险缓解，而非助长滥用；我们不会提供可直接部署的攻击工具，也不会自动化大规模滥用行为。

利益相关方。我们识别出若干可能受到影响的利益相关方群体。(1) 服务提供商（例如 VRC 供应商以及依赖其服务的网站）在漏洞遭到利用时可能面临更高的安全风险。(2) 终端用户可能受到间接影响，因为验证码防御能力减弱可能导致滥用行为增多（例如垃圾信息或欺诈）。(3) 研究人员和系统设计者能够更清楚地了解当前视觉推理验证码防御的局限性。(4) 更广泛的互联网生态系统（包括在安全研究中经常被忽视的社群）可能受到防御策略变化的影响。(5)作者自身也必须在科学披露与助长滥用的风险之间取得平衡。

潜在影响和双重用途。 这项工作的主要风险在于，通用型 VRC 求解器可能被滥用于规避安全机制，从而导致垃圾信息、欺诈、

或拒绝服务放大。同时，了解这些漏洞也具有积极影响：这能帮助供应商评估视觉推理验证码是否仍能提供有意义的安全保证，并为开发人类可解但机器难以解决的挑战提供指导。我们认为，只要采取适当的防护措施，透明分析所带来的益处就超过其风险。

缓解措施与防护措施。首先，所有实验均在离线环境下进行，未与生产系统或实时用户流量交互。基准测试集通过公开演示门户或自行注册的 API 密钥收集，避免干扰真实服务。其次，我们践行负责任披露：在发表前向所有 VRC 提供方通报了研究发现。第三，我们支持可复现性，但发布的材料仅限于研究代码和脱敏数据集；敏感标识符均已移除。访问权限仅限研究用途。

人类受试者。我们的人类基线研究风险极低，且未使用欺骗手段。参与者获知了任务目的，提供了知情同意，并可随时退出。我们未收集任何个人身份信息，所有结果均以汇总形式呈现。

数据收集。我们的研究仅使用在正常使用限制下收集的公开 VRC 挑战，未绕过访问控制，也未与后端系统交互。我们不会收集个人数据或用户标识符。所有数据集均按照统一规程整理，以确保实验完整性和可复现性。

开放科学

为提高透明度和可复现性，我们在以下地址发布了源代码和基准测试资源：
https://zenodo.org/records/17971722。该软件包包含完整的 VIPER 流程（模型配置、预处理脚本和消融模块），以及六个视觉推理验证码数据集（VTT、极验、网易、丁香、数美、小盾）的清理和标准化版本。我们还提供了基于 COCO 风格指标的统一评估协议。本次发布符合更广泛的开放科学政策。

参考文献

[1] E. Uzun 等，“rtcaptcha：一种基于实时验证码的活体检测系统。”*NDSS*，2018年。[2] M. Tang、H. Gao 等，“关于利用深度学习技术破解文本验证码和设计图像验证码的研究”，*TIFS*，2018年。[3] C. Shi、S. Ji 等，“文本验证码已死？大规模部署与实证研究”，*CCS*，2020年。

[4] M. Guerar等，“全民验证码：关于人类与计算机二十年困境的综述”，《*ACM*计算调查》，2021。

[5] P. Wang、H. Gao等，“文本验证码攻击及其鲁棒性的实验研究”，《ACM计算调查》，2023年。

[6] B. Ousat、E. Schafir等，“验证码问题：对现代网络中脆弱安全特性的分析”，载于*WWW*，2024年。

[7] H. Wang、F. Zheng等，“一种基于视觉推理的验证码设计”，*ICASSP*，2018年。

[8] Y. Gao 等，“视觉推理验证码安全性研究”，*USENIX* 安全研讨会，2021。

[9] P. Wang 等，“视觉推理验证码安全性的扩展研究”，TDSC，2023年。[10] Z. Ding 等，“IllusionCAPTCHA：一种基于视觉错觉的验证码”，载于 WWW，2025。[11] Yiyepianzhou 等，“VTT 验证码数据集（ViRC）”，https://cloud.tencent.com/product/ captcha，2024年。[12] 极验，“极验验证码平台”，https://www.geetest.com/en，2025。[13] BuiltWith，“整个互联网的验证码使用统计”，https://trends.builtwith.com/widgets/captcha/traffic/Entire-Internet，2024年。[14]谷歌，“reCAPTCHA v3 文档”，网页，2018年。 [在线]。可用地址：https://developers.google.com/recaptcha/docs/v3[15] D. George、W. Lehrach 等，“一种以高数据效率进行训练并破解文本验证码的生成式视觉模型”，《科学》，2017。[16] A. Plesner、T. Vontobel 和 R. Wattenhofer，“破解 reCAPTCHA v2”，载于 COMPSAC，2024年。[17] X. Teoh 等，“验证码仍然对机器人难解吗？使用智能体视觉语言模型进行通用视觉验证码求解”，USENIX 安全研讨会，2025。[18] B. Xu 和 H. Wang，“VRC-GraphNet：一种基于图神经网络、用于攻击视觉推理验证码的推理框架”，SecureComm，2023年。[19] J. Achiam 等，“GPT-4 技术报告”，arXiv 预印本 arXiv:2303.08774，2023年。

[20] xAI，《Grok by xAI》https://x.ai，2024年。

[21] P. Xu、X. Zhu 和 D. A. Clifton，《Transformer 多模态学习：一项综述》，*TPAMI*，2023年。

[22] H. Zhou 等，“面向电信的大语言模型（LLM）：原理、关键技术与机遇综述”，*IEEE Communications Surveys & Tutorials*，2024年。

[23] M. Yu 等，“可信大语言模型智能体综述：威胁与对策”，*KDD*，2025。

[24] Y. Chang、X. Wang 等，“大语言模型评测综述”，*TIST*，2024年。

[25] J. Yang 等，“在实践中释放大语言模型的力量：ChatGPT 综述”，*TKDD*，2024年。

[26] J. Deng 等人，《OEDIPUS：增强型大语言模型验证码求解器》，USENIX 安全研讨会，2024年。[27] Xiaodun AI，《Xiaodun 验证码平台》，https: //xiaodun.com/，2025。[28] 网易，《网易验证码服务》，https://dun.163.com/trial/space-inference，2025。[29] 顶象科技，《顶象验证码平台》，https://www.dun.163.com，2025。[30] 数美科技，《数美验证码服务》，https: //www.ishumei.com/trial/captcha.html，2025。[31] U. Team，《Yolov11（社区原型）》，https: //github.com/ultralytics/ultralytics，2024年。[32] H. Gao、J. Yan 等人，《一种针对文本验证码的简单通用攻击》，载于 NDSS，2016年。[33] E. Bursztein、J. Aigrain、A. Moscicki 和 J. C. Mitchell，《末日将至：文本验证码的通用求解》{验证码}，载于 USENIX WOOT，2014年。[34] S. Zhang、X. Yi 等人，《Privcaptcha：通过交互式验证码促进对应用隐私政策的有效理解》，载于 CHI，2025。[35] F. Alqahtani 等人，《基于图像的验证码能否抵御基于机器学习的攻击？一项实验研究》，《Computers & Security》，2020年。[36] X. Li、Y. Chen 等人，《vrcaptcha：探索虚拟现实中的验证码设计》，载于 CHI，2021。[37] X. Teoh 等人，《通过混合视觉交互模型检测验证码伪装的钓鱼网站》，USENIX 安全研讨会，2024年。[38] A. Searles 等人，《现代验证码的实证研究与评估》{验证码}，USENIX 安全研讨会，2023年。

[39] A. Gangwal、P. S. Reddy 和 C. y. Sagar，《瑞士奶酪验证码：一种用于机器人检测的新型多重屏障机制》，*SAC*，2025。

[40] C. Shi、X. Xu、S. Ji 等，“对抗性验证码”，《*IEEE*控制论汇刊》，2021。

[41] P. Wang 等，“利用对抗样本提升音频验证码的安全性”，*TDSC*，2023年。

[42] N. Zhang 等，“利用生成对抗学习抵御暗网文本验证码以实现主动网络威胁情报”，*TMIS*，2022年。

[43] H. Gao、M. Tang 等，“微软双层验证码安全性研究”，*TIFS*，2017。

[44] G. Ye、Z. Tang等，“另一种文本验证码求解器：一种基于生成对抗网络的方法”，载于*CCS*，2018年。[45] C. Li、X. Chen等，“基于循环一致生成对抗网络的文本验证码端到端攻击”，《神经计算》，2021年。[46] R. Zhao等，“Geesolver：一种采用自监督学习、用于破解文本验证码的通用、高效且简易的求解器”，*S&P*，2023年。[47] Y. Zi、H. Gao、Z. Cheng 和 Y. Liu，“一种针对文本验证码的端到端攻击”，*TIFS*，2019年。[48] B. B. Sengül 和 M. B. Alver，“多种 CAPTCHA类型的识别：使用卷积神经网络和组合数据集”，*S˙IU*，2025。[49] S. Tian等，“结合无监督学习与表征学习、用于破解文本验证码的通用求解器”，载于*WWW*，2020年。[50] H. Dai Nguyen等，“C-frame：真实环境中的验证码攻击的特征描述与测量”，*S&P*，2024年。[51] G. Chang等，“基于行为验证的滑块验证码的鲁棒性”，2024年。[52] R. Gossweiler、M. Kamvar 和 S. Baluja，“CAPTCHA近况如何？一种基于图像方向的验证码”，载于*WWW*，2009年。[53] M. Tang、H. Gao等，“破解文本验证码及设计图像验证码的深度学习技术研究”，*TIFS*，2018年。[54] T. Brown、B. Mann等，“语言模型是少样本学习器”，*NeurIPS*，2020年。[55] A. Chowdhery、S. Narang等，“PaLM：利用Pathways扩展语言建模”，*JMLR*，2023年。

[56] H. Touvron、T. Lavril等，“Llama：开放且高效的基础语言模型”，*arXiv*预印本 *arXiv:2302.13971*，2023年。

[57] S. Antol、A. Agrawal 等，“VQA：视觉问答”，国际计算机视觉大会，2015。

[58] Y. Goyal、T. Khot 等，“让 VQA 中的 V 发挥作用：提升图像理解在视觉问答中的作用”，计算机视觉与模式识别会议，2017。

[59] N. Xie 等，“视觉蕴含：一项用于细粒度图像理解的新任务（SNLI-VE）”，*arXiv* 预印本 *arXiv:1901.06706*，2019。

[60] K. Marino、M. Rastegari、A. Farhadi 和 R. Mottaghi，“OK-VQA：一项需要外部知识的视觉问答基准测试集”，计算机视觉与模式识别会议，2019。

[61] D. Schwenk、A. Khandelwal、C. Clark 等，“A-OKVQA：一项利用世界知识进行视觉问答的基准测试集”，欧洲计算机视觉会议，2022年。

[62] J.-B. Alayrac 等人，《Flamingo：用于少样本学习的视觉语言模型》，NeurIPS，2022年。[63] D. Driess、F. Xia 等人，《PaLM-E：具身多模态语言模型》，载于国际机器学习大会，2023年。[64] J. Li、D. Li 等人，《BLIP-2：利用冻结图像编码器和大语言模型引导语言-图像预训练》，载于国际机器学习大会，2023年。[65] L. von Ahn、M. Blum 等人，《自动区分人类与计算机》，美国计算机协会通讯，2004年。[66] G. Mori 和 J. Malik，《在对抗性杂乱场景中识别对象：破解视觉验证码》，计算机视觉与模式识别会议，2003年。[67] R. Gossweiler、M. Kamvar 和 S. Baluja，《验证码怎么了？一种基于图像方向的验证码》，载于 WWW，2009年。[68] K. Chellapilla、K. Larson、P. Simard 和 M. Czerwinski，《设计人性化的人机交互证明（HIPs）》，载于 CHI，2005年。[69] L. von Ahn、B. Maurer、C. McMillen 等人，《reCAPTCHA：通过网络安全措施实现基于人类的字符识别》，《科学》，2008年。[70] 谷歌，《reCAPTCHA v2》，
h
tt
ps://cloud.google.com/security/products/recaptcha，2014年。[71] M. Hossen 等人，《一种用于谷歌图像 reCAPTCHA v2 的基于目标检测的求解器》，国际信息安全与密码学研究会议，2020年。

[72] J. Li、L. Fu 等，“Mi-captcha：使用莫尼图像增强验证码的安全性”，*AAAI*，2025。

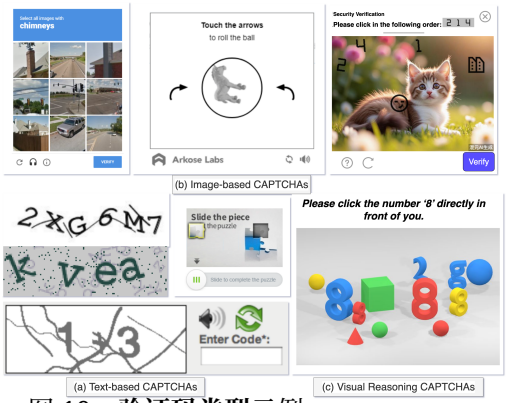


图 10：**验证码类型**示例。

技术补充

A.1 验证码

文本验证码（图10（a））。这些是最早且应用最广泛的人机交互证明。其设计原则是扭曲或模糊字符，使其对人类仍然清晰可辨，但难以通过光学字符识别（OCR）识别。Input CAPTCHA [65],Gimpy [66], 和 PessimalPrint [67] 等方案通过引入背景杂乱、字符重叠或降低字体渲染质量来提高鲁棒性 [6, 68, 69]。

图像型验证码（图10（b））。这类验证码利用人类视觉感知来区分人类和机器人。与基于文本的方案相比，它们提供了更丰富的交互模式，并且对光学字符识别攻击的抵抗力更强 [70–72]。我们将设计归纳为四种交互模式：

- 单击：用户需要点击单张图像中的特定区域（例如，“点击山”），或在多张子图中识别特定物体（“选择所有包含猫的图像”）。这种类型应用广泛（例如 Google reCAPTCHA v2）。然而，深度学习可以训练出能够破解基于点击的验证码的模型，并识别交通信号灯、公交车或人行横道等常见对象类别。

- 顺序点击：用户必须按照特定顺序点击指定的符号、字符或物体（例如，“按顺序点击汉字”）。这提高了机器人破解验证码的难度，因为它将物体识别与记忆和序列跟踪结合起来。
- 滑块：滑块验证码要求用户拖动拼图块或滑块，使其在图像中与目标位置吻合。此类验证码还会结合行为分析（例如“追踪鼠标移动模式”），以进一步区分人类输入和脚本。
- 基于旋转的验证码：这类验证码要求用户旋转图像，使其方向正确（例如，“将倒置的建筑转正”）。这类任务对人类来说很简单，但对机器人而言并不容易，尤其是在无法使用可靠的方向分类器时。

表 7：**GPT-4o 在七种广泛使用的验证码上的零样本表现**。

Type	测试用例	准确率（%）	平均耗时（秒）
视觉推理 CAPTCHA	100	**31**	3.55
文本验证码	100	46	2.41
旋转验证码	100	45	4.38
FunCAPTCHA	100	39	2.31
滑块验证码	100	40	2.17
3D验证码	100	54	2.03
reCAPTCHA	100	50	4.11

视觉推理验证码（图10（c））。VRC 要求用户进行多步视觉推理，即对复杂场景进行空间、比较和逻辑推理 [10]。每个挑战都会呈现一个包含多个对象（例如几何形状、字母、数字）的场景，并附上一条编码查询逻辑的自然语言指令。解答 VRC 需要理解指令、判断对象之间的关系（例如“在……左侧”、“与……颜色相同”或“最大的”），并准确定位正确的目标区域。

关于 CAPTCHA 类型的实证研究。 表 7 报告了 GPT-4o 在主要 CAPTCHA 类型上的零样本准确率。尽管基于文本的验证码、滑块验证码和 reCAPTCHA 已经暴露出明显弱点，但视觉推理验证码却是最难的类型：GPT-4o 的准确率仅为**31.2%** 。失败源于杂乱且充满噪声的场景，以及需要组合推理的多约束查询。这些结果凸显了视觉推理验证码作为验证码设计中最难的前沿 的地位，即便是最先进的大语言模型也难以应对。

B 实验补充

B.1 数据集详情

为了评估模型在各种真实世界验证码部署场景中的泛化能力，我们从六个广泛使用的 VRC 数据集中整理出一个基准测试集。这些数据集在视觉构成、推理复杂度和任务类型方面各不相同，涵盖了广泛的挑战特征（表2）。

VTT（腾讯）。每个 VTT 挑战都包含一条自然语言指令和一幅图像，其中有 10 到 20 个合成三维对象，例如字母、形状和数字。VTT 问题涵盖三个类别：属性识别、比较推理和空间推理，涉及复杂布局和多样的对象变体。

极验。 极验验证码与 VTT 相似，但采用更简单的构图，包含 7–10 个规则几何物体。提示词针对形状、颜色、尺寸和相对位置。与 VTT 相比，这类验证码通常会引入遮挡或含糊不清的空间关系，使目标部分被遮挡或位置不明确。不同的物体和提示词格式进一步加剧了跨域差异。

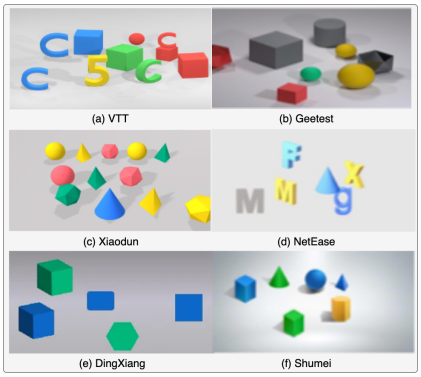


图11：六个广泛部署的 VRC 数据集

小盾。 小盾的空间推理验证码受到 VTT 启发。每道题会呈现 12–14 个 3D 物体（字母、数字、形状），要求根据共有属性（例如颜色相同）和空间关系（在……上方）进行推理。与 VTT 不同，小盾将物体密集地堆叠排列，因此经常出现遮挡和基于接触的关系，增加了检测难度。

网易。网易验证码数据集的每幅图像包含 5 到 7 个规则几何对象或字母数字物体。提示词主要涉及基于方向的筛选条件（例如“朝向侧面”）、属性一致性（例如“颜色相同”）以及基本的方向推理。与 VTT 和小盾相比，该数据集的场景布局更加简洁，但问题仍要求具备物体级理解能力。

丁香。 丁香验证码图像通常显示 5 个平面或几何物体，主要考查基于位置的推理，例如“左侧的物体”、“上方”或“距离最近”。其中不涉及抽象逻辑推理或多阶段依赖关系。

数美。 数美采用了更简约的风格。每张图像仅包含 6 个规则的三维几何形状。提示词格式固定：始终要求用户点击具有特定颜色和形状的最小对象。



图12：**定性检测结果**。检测器的预测示例。每个边界框都编码了预测的复合标签及其置信度分数。

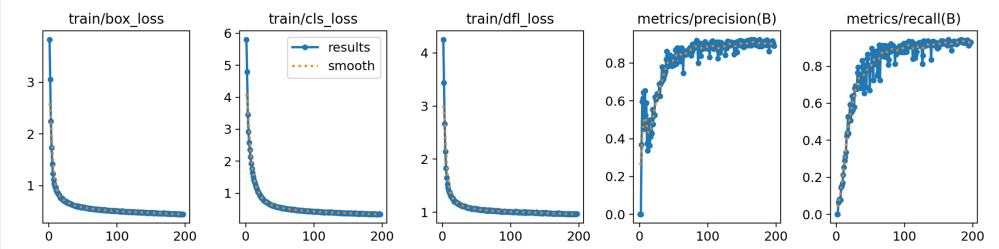


图13：**多目标检测模型的训练与验证指标**。图中展示了模型在 200 个训练轮次中的收敛情况。指标包括：训练损失（边界框_损失、分类_损失、分布焦点损失_损失）和评估指标（精确率、召回率）。

B.2 人工基线

我们在与所有自动化方法相同的基准测试集设置下建立了人工基线。我们招募了 10 名参与者来完成基准测试集中的一个代表性子集。参与者为年龄在 20 至 35 岁之间的大学生和教职员工志愿者。我们指出，这可能会限制人工基线的外部效度；更广泛的人口统计覆盖范围将留待未来研究。

对于每个 VRC平台，向参与者展示了从模型实验所用的相同评估语料库中均匀随机抽取的 100 个验证码挑战（每个平台共计 1，000 个）。我们记录了每次试验的正确性和响应延迟。表8 汇总了相关统计数据。

表8：**人工基线（10 名参与者，每个平台 × 100 完成 100 个挑战）**。对于每个 VRC平台，我们报告评估项目数（N）、正确答案总数、平均准确率及参与者间标准差、平均响应时间及参与者间标准差，以及全部 1,000 次试验的累计时间。

平台	N	正确答案	准确率（%）	平均时间（秒）	总时间
VTT	1000	876	87。60 (±3.10)	7.40 (±2.10)	02:03:20
极验	1000	907	90.70 (±2.70)	7。80 (±2.30)	02:10:00
网易	1000	951	95.12 (±2.20)	6.90 (±1.90)	01:55:00
丁香	1000	954	95.40 (±2.00)	6.60 (±1.80)	01:50:00
数美	1000	932	93。20 (±2.50)	6.10 (±1.70)	01:41:40
小盾	1000	918	91.80 (±2.80)	7.20 (±2.00)	02:00:00

B.3 多目标检测器性能

定性结果。除了定量指标外，我们还在图中展示了定性检测结果。12该图展示了我们的多目标检测器在多个 VRC 平台上生成的代表性预测结果。每个边界框都标注了预测的复合标签（涵盖形状、颜色和方向属性）以及置信度分数。结果表明，即使在杂乱背景和不同尺度下，检测器仍能可靠地定位和分类各类物体，这证实了感知模块在支持下游推理方面的鲁棒性。

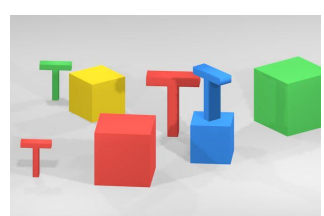
训练动态。 图 13显示，所有训练和验证损失（边界框、分类、分布焦点损失）在前 20 个训练轮次中急剧下降，并稳定在接近零的水平；验证曲线紧随其后，仅表现出轻微方差。精确率和召回率迅速上升，并在 100 个训练轮次后超过 0.90，证实了定位准确，且泛化稳定。平滑曲线（橙色）与原始结果高度吻合，表明各训练轮次之间的方差较低，优化过程稳健。

C 案例研究

我们通过一个典型示例，展示 VIPER 如何处理 VRC（即端到端感知推理）。本示例逐一呈现各模块的中间输出，展示原始图像和查询如何逐步转换为最终答案。

步骤1：输入。给出图像和查询：

“请点击正下方的对象
字母 T。”



步骤 2：感知输出。检测器处理输入图像，并生成一组结构化的对象候选项，其中包含其类别、坐标和边界框。完整输出如下：

```
[
  {"对象":"字母 T","位置":
                                                    "边界框
":.
                                                    {"对象":
"字
```

```
[442 2, 267 8], {"
对象":"立方体","位置":...."边界性":....54 ]},
{"
对象":"侧面","位置":...."边界性":....80 ]},
{"
对象":"红色","位置":...."边界性":....93 ]},
{"
对象":"前方","位置":...."边界性":....]}
]
```


步骤3：QIE 结果。QIE 处理自然语言指令 q，并为其中提及的物体生成结构化符号记录。对于此查询，输出为：


```
[
  {"物体数据 ": {" 形状 ": "T", " 颜色":  "", "
      方向":  ""}}
]
```


步骤 4：Integrator 输出。Integrator 将视觉检测器提供的结构化检测结果与 QIE 生成的符号记录进行对齐。生成的表示可作为一组经过精炼的候选对象，用于后续关系推理。输出为：


```
[{2 [ ], "对象":"
字母
490
前方
边界
母 '
355
T",
139
```


步骤 5：RPIE 输出。给定检测到的参照物（“字母 *T*”），RPIE 会沿指定方向（“下方”）将每个中心点移动固定偏移量 Δ，并选择边界框包含探测点的检测结果，将其标记为“!maybe result”。


```
[2 {"Object": ["字母 T", "蓝色", "侧面"],"位置":[442.5, 168.2]}, 3
{"(
26
```



```
4 {"Object": ["字母 T", "前方", "红色"],"位置":[358.8, 192.3]}, 5 {"
Ob                                                              ": [
"字
```


步骤 6：动态提示。动态提示明确指示大语言模型转换查询、推理相对空间关系，并从标记为 ! maybe result 的候选项中识别正确的目标对象。本例构造的提示词如下：

> **动态提示词：**你是一名 3D 推理验证码助手。你将首先收到验证码问题，然后收到验证码图像，以及视觉模型生成的输出（其中可能包含错误）。重要提示：这是一个涉及相对位置关系的问题。请仔细思考参照物与目标对象之间的关系。最后，请提供目标对象的坐标，而不是参照物的坐标。示例：- “ 点击圆柱体上方的字母 ” 意味着点击字母，而不是圆柱体。- “点击 ‘j’ 下方的对象”意味着点击对象，而不是字符 ‘j’。请从标记为“!maybe result”的候选项中选择答案。问题：“点击字母 ‘T’ 正下方的对象”

步骤 7：大语言模型分析器输出。大语言模型接收 RPIE 模块生成的动态提示词和候选对象。它会推理它们的位置关系，并选出最终答案。本例中的输出为：

> 根据对齐后的检测结果，字母“T”正下方的对象是**蓝色立方体**。蓝色立方体的坐标为**(442.2,267.8)**。

步骤 8：验证。系统会检查大语言模型分析器预测的坐标是否落在目标区域的真实标注边界框内。

> 标注的目标边界框定义如下：左侧 =
> 0 5670, = 0 7352, = 0 4469, = 0 7288 。右侧。顶部。底部。。归一化点 (0.6503, 0.5822) 位于该区域内。**结果：验证成功**。
>
> dataset/VTT_dataset/images/captcha_1003.png
>
> 