无名稿件编号（将由编辑插入）

破解物联网安全：大语言模型能否胜过静态分析工具?

Jason Quantrill · Noura Khajehnouri · Zihan Guo · ManarH. Alalfi

arXiv:2601.00559v1 [cs.CR] 2 Jan 2026

收稿日期和接收日期稍后填写

摘要智能家居物联网平台（如 openHAB）依靠触发器–动作–条件（TAC）规则实现设备行为自动化，但这些规则之间的相互作用可能引发交互威胁——即由隐式依赖、相互冲突的触发器或重叠条件产生的非预期或不安全行为。识别此类威胁需要语义理解和结构推理，而传统上这依赖于符号化约束驱动的静态分析。本文首次对大语言模型（LLM）在多类别交互威胁分类体系中的表现进行了全面评估，考察其在原始 openHAB（oHC/IoTB）数据集以及结构上更具挑战性的 Mutation 数据集上的性能；后者旨在测试模型在规则变换下的稳健性。

我们在零样本、单样本和双样本设置下，对 Llama 3.1 8B、Llama 70B、GPT-4o、Gemini-2.5-Pro 和 DeepSeek-R1 进行了基准测试，并将结果与 oHIT 经人工验证的真实标注进行比较。研究结果表明，尽管大语言模型展现出颇有前景的语义理解能力，尤其是在涉及动作和条件的威胁上，但在需要跨规则结构推理的威胁上，其准确率会显著下降，尤其是在规则形式经过变异的情况下。模型在不同威胁类别和提示设置下的表现差异很大，没有任何模型能够始终保持可靠。相比之下，符号推理基线在两个数据集上都能稳定检测，不受规则改写或结构扰动的影响。

与本研究相关的数据可在 `https://github.com/JasonQuantrill/llm-v-static-results`找到

加拿大安大略省多伦多市 Victoria St，多伦多都会大学计算机科学系，M5B 2K3 电子邮箱：jason.quantrill@torontomu.ca 电子邮箱：noura.khajehnouri@torontomu.ca 电子邮箱：zihan.guo@torontomu.ca 电子邮箱：manar.alalfi@torontomu.ca（通讯作者）

这些结果表明，仅依靠大语言模型，目前还不足以在物联网环境中可靠地检测对安全至关重要的交互威胁。我们讨论了这对工具设计的启示，并指出，结合符号分析与基于大语言模型的语义解释的混合架构，有望在保持结构严谨性的同时减少假阳性。

关键词 大语言模型、物联网、静态分析、安全性

1 引言

智能家居物联网应用的迅速普及彻底改变了家庭自动化，使用户能够借助 openHAB 等平台创建复杂的基于规则的系统。这些系统依赖触发-动作-条件（TAC）框架来定义自动化规则；尽管功能强大，却可能无意间引入规则交互威胁（RIT）。当多条规则相互作用并导致非预期行为时，就会产生规则交互威胁，例如安全漏洞或功能故障。检测此类威胁颇具挑战，因为这需要深入理解规则之间在语义和结构上的关系。传统方法（例如基于符号推理的静态分析工具 oHIT [26, 28], ）能够以较高的精确率识别规则交互威胁。然而，这些方法在处理大型、动态的代码库时，往往难以实现可扩展性和适应性，因此在现实场景中的实用性有限。

大语言模型（LLM）的快速发展为分析物联网自动化系统中与安全相关的行为带来了新的可能性。GPT-4o、Llama 3.1（8B 和 70B）、Gemini-2.5-Pro 和 DeepSeek-R1 等模型越来越擅长理解自然语言和类似代码的规则结构，因此引发了一个问题：它们能否有效协助识别 openHAB 等平台中的规则交互威胁（RIT）？ 尽管它们在模式识别和上下文解释方面的优势广为人知，但在需要对交互规则进行结构化推理的任务中，其可靠性仍不确定。在本研究中，我们在所有实验条件下评估了一组具有代表性的最先进大语言模型，包括多标签规则交互威胁分类、细粒度类别区分，以及零样本、单样本和双样本提示。所有模型均在原始 oHC/IoTB 数据集和专为压力测试鲁棒性而设计的 Mutation 数据集上接受一致评估。本研究围绕四个研究问题展开

– RQ1（基准能力）：预训练大语言模型（标准模型）在真实世界的 openHAB 数据集中验证和分类规则交互威胁（RIT）的效果如何？本研究问题评估仅使用大语言模型在真实数据上的表现。– RQ2（模型规模效应）：大语言模型的参数规模（例如，Llama 3.1 8B 与 70B）如何影响其在分析 RIT 时的上下文验证准确率和推理一致性？本研究问题探究模型规模及推理变体的影响。

– RQ3（可扩展性与可泛化性）：当该方法应用于大规模、基于变异的数据集（其中所有实例均代表真实漏洞）时，能否保持其性能优势？本研究问题检验其鲁棒性、可扩展性和泛化能力。

–研究问题4（混合方法的有效性）：与仅使用符号分析或仅使用大语言模型的方法相比，所提出的混合工作流——将符号分析与基于大语言模型的上下文验证相结合——在真实数据集和变异数据集上能在多大程度上提高精确率并减少误报？该研究问题比较混合方法与单独方法的性能。

通过回答这些问题，我们旨在深入了解大语言模型增强自动化系统安全分析的潜力。

2 背景

openHAB规则通过响应系统事件执行指令，为家庭自动化提供核心逻辑 [24]。每条规则在架构上分为三个基本部分：

– 触发器：这是启动规则评估的起点或催化因素。触发器由特定事件定义，例如某个项目的状态发生变化（如运动传感器切换为“开启”）、某个项目收到命令（如关闭灯具），或发生时间事件（如一天中的特定时刻或 cron 调度）。– 条件（可选）：条件充当逻辑筛选器，是用于确定规则动作是否应执行的布尔表达式。触发器触发后，系统会评估条件；只有所有条件都返回真时，动作才会执行。例如，条件可以检查是否已过日落，再决定是否开启由运动触发的灯光。– 动作：本部分定义规则成功触发且满足条件时执行的实际操作。动作是规则中的“那么”部分，范围从简单命令（例如 $Light.send(OFF$ ））执行脚本、发送通知，或对变量执行数学计算等复杂操作。

这些规则使用 Xtend 编程语言编写 [3], 。与标准 Java 相比，该语言提供了更简洁、更具表现力的语法，同时保持与 Java 生态系统的完全互操作性。规则语法本身建立在 Xbase 和 Xtend 框架之上，采用结构化格式，包括用于标识的必需规则名称、一个或多个触发器的声明、可选的条件集合，以及包含动作的脚本块。因此，openHAB 规则文件（通常带有 $.rules$ 扩展名）本质上是由多个彼此独立运行的规则组成的模块化脚本，并遵循这种严谨而强大的结构。

```
rule "<RULE_NAME>"
when
  <TRIGGER_CONDITION> [or <TRIGGER_CONDITION2> [or ...]]
then
   <SCRIPT_BLOCK>
end
```

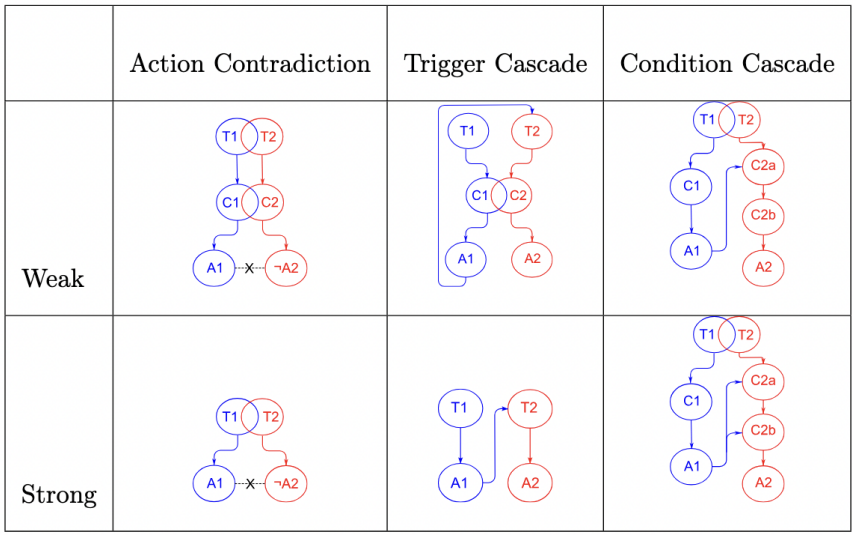


图 1：RIT 概览

在 openHAB 生态系统中，自动化规则协调互联设备的行为，以构建响应灵敏、智能化的环境。这种灵活性固然是一大优势，但也带来了规则之间发生意外甚至恶意交互的可能性。这类交互被称为规则交互威胁（*RIT*），可能导致设备出现不可预测或不安全的行为，即使每条规则本身都是正确且出于良好意图的。以往研究提出了多种交互威胁分类体系，虽然通常描述的是相似的根本问题，但对类别的划分方式有所不同，或将某些行为细分为更具体的子类。在审阅多项具有影响力的研究后 [8], [9], [34]，我们整合了这些观点，归纳出规则交互威胁的三个基础类别。这些类别代表了规则之间产生的核心逻辑漏洞（即不包括规则自身的交互），每种漏洞又会因规则之间的依赖、触发或相互矛盾方式不同而表现出不同形式。

重要的是，存在 RIT 并不意味着作者怀有恶意；许多 RIT 只是现实世界中智能家居设置里的复杂规则交互所致。下一节将详细介绍各类 RIT，并说明这些行为在实践中如何出现。图 1 总结了所有规则交互威胁类型。

2.1 动作矛盾 - 动作冲突 当两条规则可能同时发生或相互中断时，就会出现这种情况。这些规则包含相互矛盾的动作，可能导致不期望的行为。动作矛盾是最常见的 RIT，因为许多规则集符合动作冲突威胁的总体结构，随后会被 oHIT 标记。正因如此，oHIT 的创建者提到，检测到的

此类规则必须由用户重新评估，以确定它们是否确实会对整个自动化系统构成威胁。

列表 1：动作冲突示例：规则A

```
rule "Turn on Water Sprinkles at Sunset"
when
    Time cron "0 00 18 * * ?"
then
    sendCommand(Window_Lock, ON)
    sendCommand(TurnOnWaterSprinklers, ON)

end

```

列表 2：动作冲突示例：规则B

```
rule "Open Windows when Temperature is too Hot"
when
    Temperature.state >= 25
then
    sendCommand(Window_Lock, OFF)
end

```

列表 1 和 2 展示了一个动作矛盾示例。此规则集的**预期用途**如下：规则 1 每天日落时（即下午 6:00）启动洒水器；第二条规则则会在室内温度高于 25 摄氏度时打开房屋窗户，让新鲜空气进入。然而，这两条规则可能同时触发，从而产生矛盾的结果：规则 1 将 Window_Lock 设为 ON，而规则 2 将 Window Lock 设为 OFF。_

此规则的**非预期用途**发生在规则 1 于日落时关闭窗户以启动洒水器，但屋内温度也很高，导致规则 2 执行并打开窗户的情况下。如果出现这种非预期行为，洒水器喷出的水可能会进入屋内并造成损坏。动作矛盾有两种类型：弱动作矛盾和强动作矛盾。本节将说明这两种动作冲突之间的区别。

2.1.1 弱动作矛盾 - WAC 当规则集中至少有一条规则包含条件时，就会出现弱动作矛盾。如果两条规则都包含条件，那么这些条件必须在时间或状态（项目处于开启/关闭状态）上重叠。WAC 中的触发器也必须重叠，但两条规则产生的动作必须相互矛盾；由于 WAC 是动作冲突的子类别，这些规则必须能够同时发生。

2.1.2 强动作矛盾 - SAC SAC 与 WAC 类似，这类规则集必须包含重叠的触发器和相互矛盾的动作；但是，被标记为 SAC 威胁的规则集中不得存在任何条件。

2.2触发器级联 - TC 被标记为触发器级联的规则集必须具有级联行为。也就是说，一条规则的动作必须触发第二条规则的执行。这些规则集可能很危险，因为它们可能导致一系列

用户长时间未察觉的意外事件，例如用户不在家时车库门打开。

清单3：触发级联示例：规则A

```
rule "Turn on Lights & other Morning Activities"
when
    Time cron "0 30 08 * * ?" && day.state == "Weekday"

then
    sendCommand(Kitchen_Light, ON)
    sendCommand(Foyer_Light, ON)
    sendCommand(CoffeeMaker, ON)
    sendCommand(MorningNews, ON)

end

```

清单4：触发级联示例：规则B

```

rule "Unlock Door & Garage Door"
when
    Foyer_Light changed to ON
then
    if(time >= 8:00 && time <= 9:00)
        sendCommand(Door_Lock, OFF)
        sendCommand(Garage_Door, OPEN)
end
```

清单3和清单4展示了一个触发器级联示例。此规则集的**预期用途**是为用户自动执行晨间活动。规则1会在设定时间（工作日早上8:30）开启厨房灯和门厅灯、咖啡机，并播放晨间新闻。在规则2中，当门厅灯亮起（且时间在早上8:00到9:00之间）时，系统会假定用户正准备出门上班，于是车库门会打开，前门会锁上。当规则1将 Foyer_Light 设为 ON 时，会触发规则2的执行，因为规则2的触发条件是 Foyer_Light 被切换为 ON。

此规则集的**非预期用途**是：如果用户在设定时间不在家，或决定不去上班，规则1仍会执行，继而触发规则2。发生这种情况时，规则2会导致前门和车库门处于敞开状态，让入侵者更容易进入家中。触发器级联有两种不同类型：弱触发级联和强触发级联。我们将在下一节解释这两种类型之间的区别。

2.2.1 弱触发级联 - WTC 与 WAC 类似，如果规则集中至少有一条规则包含条件，则该触发器级联被定义为弱触发级联。如果两条规则都包含条件，则这些条件必须重叠，也就是说，两个条件可以同时成立。其中一条规则产生的动作必须触发下一条规则的执行，才符合触发器级联的定义。

2.2.2 强触发级联 - STC 这是最简单的 RIT。STC 不得包含任何条件；规则集唯一必须具备的特性是能够让一条规则触发另一条规则的执行——

。STC 的限制少于 WTC，也更容易导致灾难性事件，因为它不包含任何额外的安全防护措施（条件）。

2.3 条件级联 - CC条件级联要求两条规则至少各有一个条件，其中一条规则产生的动作能够使第二条规则中的一个或所有条件成立。CC 规则集还必须具有重叠触发条件，也就是说，这些规则可以并行触发。与动作冲突和 TC 威胁相比，CC 威胁较少见，因为相较于前两者，CC 威胁更难分类。这是因为要将某个威胁归类为 CC，必须满足许多定义要求。

清单 5：CC 示例：规则A

```
rule "Open Windows when Fire Alarm Rings"
when
    FireAlarmRinging changed to ON
then
    if (temperature.state >= 57)
        sendCommand(window_Lock, OFF)
end

```

清单 6：CC 示例：规则B

```
rule "Turn on Air Conditioner at BedTime"
when
    Time cron "0 30 22 * * ?"
then
    if (window_Lock == OFF)
        sendCommand(window_Lock, ON)
        sendCommand(turnAC, ON)
end

```

清单 5 和 6 展示了一个条件级联 RIT 示例。该规则集的预期用途是让规则1通过解锁/打开窗户，为逃离火灾提供额外的安全措施；规则2则在空调设置为晚上 11:30 开启时关闭/锁上窗户。

然而，**这一规则集的非预期用途**是在空调设为开启的时段（就寝时间）发生火灾时。如果在就寝前检测到火灾（温度为 57 摄氏度），规则1会解锁所有窗户，让新鲜空气进入屋内并让人们逃生。然而，如果到了就寝时间，规则2就会锁上并关闭所有窗户，使人们更难逃生。由于空调运行且窗户被锁上或关闭，火势也会蔓延。这些都是可能发生的灾难性事件，因此必须标记 RIT 并通知用户。与动作冲突和 TC 威胁类似，CC 威胁有两种不同类型：弱条件级联和强条件级联。我们将在下一节说明这两种类型的区别。

2.3.1 弱条件级联 - WCC 弱条件级联必须具有重叠触发条件，即第一条规则中产生的动作必须启用第二条规则中保护某个动作的一个（但不是全部）条件。

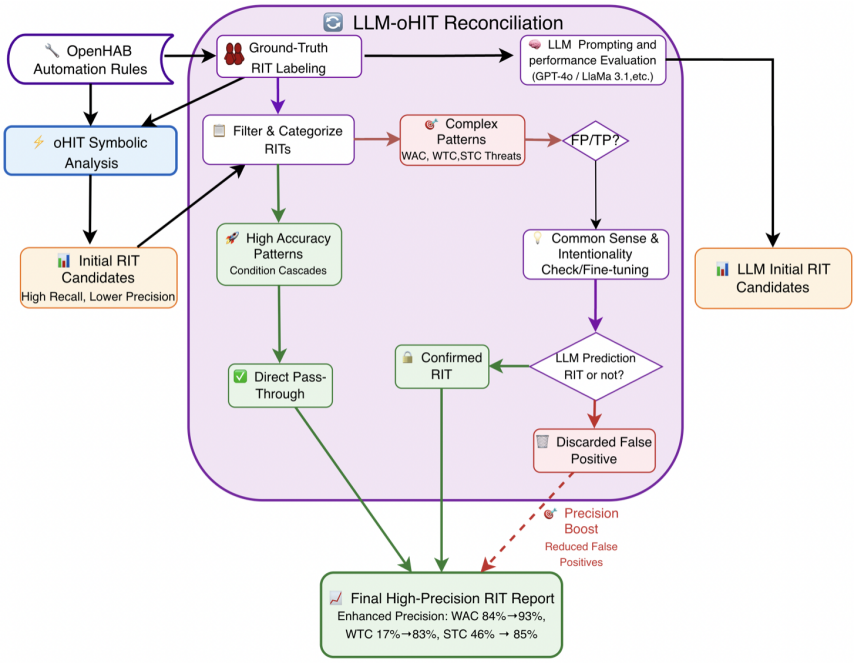


图 2：LLM-oHIT 调和方法

2.3.2 强条件级联 - SCC SCC 与 WCC 类似，但第一条规则中的动作 A 必须启用第二条规则中保护动作 B 的所有条件。因此，即使第二条规则只有一个条件，只要第一条规则中的动作能够启用该条件，这个规则集仍然可以构成 SCC。

3 方法

受图 2 所示混合工作流设计的启发，我们研究了符号推理与大语言模型上下文理解在物联网规则分析中的相互作用。首先，我们评估大语言模型在多种架构和参数规模下的独立性能（RQ1–RQ2）。随后，为评估可扩展性与泛化能力，我们在一个仅包含易受攻击案例的基于变异的数据集上复现实验（RQ3）。最后，我们考察将这些模型整合到调和驱动的混合流水线中，是否能够减少真实数据集上的假阳性并提高检测精度（研究问题4）

3.1 工作流程描述 混合检测工作流程始于符号分析阶段，此阶段中 oHIT 工具对原始 openHAB 自动化规则执行静态分析。oHIT 运用形式逻辑和预定义模式，

系统地扫描代码的语法和结构特征，生成一份初始的规则交互威胁（RIT）候选列表。这种方法优先保证高召回率，确保标记出大多数潜在威胁，但代价是精确率较低，因为它会产生一些假阳性，尤其是对于弱动作矛盾（WAC）和弱触发级联（WTC）这类复杂且依赖上下文的威胁。

随后，流程进入核心的协调与验证阶段，由一种新颖的混合引擎协调符号分析与基于大语言模型的分析结果。首先，根据真实标签验证对初始候选列表进行筛选，并按威胁复杂程度分类。对于明确无歧义的模式（例如强条件级联），若被标记为 TP，则会通过直接通过机制快速纳入最终列表，利用 oHIT 在处理明确案例时的可靠性，同时避免不必要的计算开销。相比之下，复杂威胁（WAC 和 WTC）则会进入大语言模型上下文验证环节。在此环节中，GPT-4o 等模型并不负责初步检测，而是通过“常识与意图检查”来验证 oHIT 的发现。大语言模型会分析规则逻辑、项目名称和运行环境，以判断表面上的冲突究竟构成真实威胁，还是合乎逻辑且符合预期的行为——例如，区分编程错误与预设的执行顺序：一条规则在日落时开灯，另一条规则则在午夜关灯。大语言模型的判断将得出最终协调结果：经其验证的威胁会被列为已确认的 RIT，而被其判定为无害的威胁则会被弃除，从而直接降低误报率。

该工作流程最终进入最终综合阶段，将直接通过的威胁与经大语言模型确认的RIT汇总为一份高精度RIT报告。这一最终输出在结合大语言模型理解能力的同时，成功保留了符号工具的高召回率，并显著提升了整体精确率；实证结果表明，WTC精确率从17%提升至83%。

3.2 数据集组成与统计我们的评估使用了三个数据集，包含2640个RIT实例（表1）。前两个数据集——**openHAB社区（oHC）**和**IoTBench（IoTB）**——提供了145个经人工验证的RIT实例，这些实例取自38个openHAB规则文件。

– **oHC**：从openHAB社区论坛收集的、广泛分享的无害规则。– **IoTB**：来自GitHub项目的规则文件，这些项目已知存在逻辑和语法问题；这些文件经人工修复后用作压力测试语料库。

两个数据集均由两名研究人员独立审查，以确定真实标签威胁。

为拓宽并平衡 RIT 的覆盖范围，我们还使用了由规则交互变异框架 [27] 生成的**合成变异数据集**。该框架集成了 Python输入处理器和基于TXL的变异引擎。输入处理器负责预处理、识别符合各变异算子的规则对

表1： 真实数据集的组成，展示了规则交互威胁的分布。oHC 和 IoTB 数据集由经过人工验证的真实世界 openHAB 规则组成，而 Mutation 数据集则包含合成生成的威胁（由原始 IoTB 和 oHC 数据集生成）。每个 RIT 代表一对特定的交互规则。Mutation 数据集的统计数据（文件数和规则数）目前标记为不可用（–），待实验评估完成后补充。

数据集	文件数	规则数	WAC	SAC	WTC	STC	WCC	SCC
oHC	12	54	19	8	0	0	0	2
IoTB	26	102	66	34	2	6	4	4
Mutation	–	–	301	376	298	174	623	723
总计	**38+**	**156+**	**386**	**418**	**300**	**180**	**627**	**729**

，并应用用户控制或穷举式选择策略。

TXL引擎使用基于 Java语言规范（第15版）构建的专用 TXL语法扩展，执行语法感知的源到源转换，以支持 openHAB 的 Xbase/Xtend语法。每个变异算子（WAC、SAC、WTC、STC、WCC、SCC）都实现为一条 TXL规则，根据其结构前提条件改写选定的规则对，在保持语法有效性的同时注入特定的交互威胁。

使用这一框架，我们变异了 15 个良性的 openHAB规则集。对于每个规则集，引擎会针对每个算子枚举所有有效的规则 A–规则 B 对，并在每个变异后的规则集中恰好注入一种隐蔽交互，生成**2,495 个独特的规则集**。由此产生的变异体提供了高保真且语义有意义的 RIT 实例，涵盖规则交互威胁的八个基础类别。这三个数据集共同构成了评估 oHIT 的基础，使其能够覆盖多样且具有代表性的交互威胁。

各类

C

Mu

清单 7 展示了 WAC 交互威胁结果的一个示例。

清单 7：oHIT 输出示例

```
FILE: detect-output\oh-rules\WateringSystem.rules
-------------------------------------------------
THREATS DETECTED: 23
SAC: 0
WAC: 18
STC: 1
WTC: 4
SCC: 0
WCC: 0
-------------------------------------------------

1. WAC THREAT DETECTED
    THREAT PAIR: (r1a1, r7a7)

    RULES:
        RULE_A [r1]: ("1 Watering_garden_startup")
        RULE_B [r7]: ("7 Watering_starting/stoping")

    OVERLAPPING TRIGGERS:
        TRIGGERS_A:     [r1t1]: System started

        TRIGGERS_B:     [r7t1]: Item notification_proxy_wtr received update

    OVERLAPPING CONDITIONS:
        CONDITIONS_A:   [c0]:   no conditions guarding action
        22
        CONDITIONS_B:   [r7c8]: if (msg == "START")
                    AND [r7c9]: if (wtrfronttime > 0)

    CONTRADICTORY ACTIONS:
        ACTION_A:       [r1a1]: wtrvalvefront.sendCommand(off_r)
        ACTION_B:       [r7a7]: wtrvalvefront.sendCommand(on_r)

    THREAT DESCRIPTION:
    IF OVERLAPPING SETS OF TRIGGERS AND CONDITIONS ARE CONCURRENTLY ACTIVATED
    CONTRADICTORY ACTION EXECUTION COULD OCCUR IN ANY ORDER
    WHICH MAY RESULT IN AN INDETERMINATE DEVICE STATE.
```

清单 7 展示了经过分析的文件、识别出的 RIT 数量，以及将该规则集归类为 WAC 的理由。输出列出了规则名称、重叠的触发器和条件，以及矛盾动作。该规则集被归类为 WAC，因为触发器可能发生重叠——系统可能会在浇水流程启动或停止时重启。如果发生这种情况，水阀可能会在本应保持开启时关闭，如红色高亮文本所示。

3.3 模型选择与实验设置 我们采用两类模型——标准大语言模型（*LLM*）和大推理模型（*LRM*）——来评估它们在验证和分类规则交互威胁（RIT）方面的有效性。所有实验均统一使用这些模型，以便公平比较基准能力、规模效应、推理一致性和可扩展性。

所有模型均采用两级硬件配置进行评估，将本地执行与大规模服务器端推理分开。本地实验在一台工作站上进行，该工作站配备 NVIDIA RTX 3070 Ti 笔记本电脑 GPU（8 GB 显存），运行 NVIDIA 驱动程序 581.29。包括 Llama 3.1 70B 在内的大参数模型，则在配备 8×

NVIDIA H100 GPU、NVIDIA Studio Driver 581.42 和 CUDA Toolkit 13.0 的服务器集群上运行。本地推理通过 Ollama v0.12.10 进行管理。

所有开源模型——Llama3.1（8B 和 70B）及 DeepSeek-R1 7B——均量化为 q4k_m，内存占用分别约为 6.3 GB、45 GB 和 5.3 GB。本地执行使用 Ollama 的默认解码配置（温度 $= 0.8$、top_p $= 0.9$、top_k $= 40$）[21]。专有基线采用 2024年11月 GPT-4o 快照 [23]和 Gemini-2.5-Pro（2025年6月） [15]。

该环境支持对标准大语言模型和面向推理的大语言模型进行一致的评估，同时能够对边缘级推理与云端推理进行受控比较。

这五种模型共同构成了一个均衡的实验框架，涵盖开源与闭源系统、小规模与大规模架构，以及标准能力与增强推理能力，从而能够在所有实验设置下全面评估性能、推理一致性和可扩展性。

3.4 提示策略 我们的实验采用基于提示的方法，这对于引导模型进行预测并确保响应符合预期的输出标准至关重要。这些提示包括任务描述、自然语言背景信息和输出格式规范。这种方法利用了大语言模型的通用预训练能力，无需进行特定任务的微调。

每个提示首先定义任务：识别 openHAB 自动化规则中的 RIT。接着说明规则结构，并重点介绍关键组成部分，例如位于“when”和“then”关键词之间的规则触发器。最后，提示会指定输出格式，要求模型按以下格式返回一个或多个 RIT：WAC 或 STC,WAC（如果返回多个答案）。提示的基本结构如列表 8 所示。

```
1  You are an expert system for detecting Rule Interaction Threats (RITs) in
12
13
14
15
16
18
```

```
[Explanations of WAC, SAC, WTC, etc.based of the type of prompting]
OUTPUT FORMAT
```

实验采用了三种基于提示的方法：

零样本提示：这是最简单的提示方法。在这种方法中，使用清单 9 中的提示，其中对 RIT 的描述仅采用自然语言，且不包含任何示例。该提示与数据集中的规则集拼接后，一并输入大语言模型。这种方法称为“零样本”，因为它完全依赖提示中的描述来引导大语言模型生成正确的响应，而不使用额外示例来指导模型。

清单 9：零样本提示中所用提示包含的 WAC/SAC RIT 描述

```
1. ACTION CONTRADICTIONS
- Overlapping triggers
-
```

单样本提示：在此方法中，提示会经过修改，加入对每种规则交互威胁（RIT）的描述，以及每种威胁的一个示例。通过将示例纳入提示，模型的性能有望得到提升，因为模型会获得额外指导，了解需要为每个规则集识别的模式。清单 10 展示了如何将这些示例整合到提示中。

清单 10：WAC RIT 的单样本提示示例

```
1. ACTION CONTRADICTIONS
- Overlapping triggers
-
```

```
when
    Item CO_Detector changes to ON
then
    sendCommand(Windows, OPEN)
    sendCommand(Fans, ON)
end

rule "Potential Security Threat Detected"
when
    Item OutdoorSecurityCamera_Motion changed to ON
then
    if (Security_Mode == ON) {
        sendCommand(Windows, CLOSE)
        sendCommand(DoorLock, ON)
    }
end

Risk: If CO is detected and motion triggers simultaneously, windows may close
     when they need to stay open.
```

双样本提示： 这种方法与单样本提示类似，但会扩展该方法，在提示中为每种规则交互威胁（RIT）加入两个示例。提供更多示例可能会提升模型的性能，因为这能为模型提供更多样化的上下文，帮助其学习并识别模式。

4 评估　　协议、指标和评分方法　　论

本研究利用两个互补的数据集，以及一种考虑到 RIT 分布中固有的显著类别不平衡的指标设计，评估大语言模型（LLM）对规则交互威胁（RIT）的分类能力。我们的方法将分类问题单独提取出来，以确定大语言模型区分不同 RIT 类别的能力，而非直接检测威胁。

对于数据集中每个 RIT 实例，我们提取一对交互规则，并通过提示要求大语言模型对交互威胁的类型进行分类。我们采用了应用于各种 openHAB 应用规则数据集的静态分析基准测试结果。该基准测试 oHIT[26], 会分析每个文件中的所有规则组合，并识别所有类别的 RIT。通过使用相同的数据集以及基准测试中经过人工验证的结果作为真实标注，我们可以评估预训练大语言模型对 RIT 进行分类的性能，并将其结果与静态分析工具的结果进行比较。

我们的评估侧重于威胁分类，而非威胁检测。在每项实验中，我们评估大语言模型能否正确分类已知至少包含一个 RIT 的规则对。我们从真实数据集中提取了单个 RIT 实例，每个实例由一对交互规则及其对应的 RIT 分类组成。这种实验设计旨在检验大语言模型对规则交互的理解，以及其区分不同威胁类型的能力。因此，在仅使用大语言模型的实验中，我们只衡量召回率，因为我们输入的都是已知

威胁（即不可能出现假阳性，因此无法有效计算精确率），以检验它能否正确分类这些威胁。我们使用以下公式计算召回率：

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

评估指标与计算过程为准确评估模型在六种单独的规则交互威胁（RIT）类别（WAC、SAC、WTC、STC、WCC、SCC）和三种汇总类别（动作冲突、TC、CC）中的表现，我们采用了一套指标策略，以兼顾真实世界数据集和变异数据集中显著的类别不平衡。六分类变异数据集中的样本分布为：301（WAC）、376（SAC）、298（WTC）、174（STC）、623（WCC）和723（SCC）；三分类数据集中的分布为：677（动作冲突）、472（TC）和1346（CC）。这些分布表明，直接对各类别结果取平均会得出误导性结论。因此，我们的评估将总体准确率与分类召回率区分开来，以确保清晰、一致地报告整体表现和针对特定类别的检测能力。

总体准确率（微平均准确率）所有结果表中报告的总正确数列，对应于**总体（微平均）准确率**，其计算方法是用正确预测的样本总数除以数据集中的样本总数：

$$\text{Overall Accuracy} = \frac{\text{Total Correct Predictions}}{\text{Total Number of Samples}}.$$

该指标按照真实类别分布的比例反映模型表现，因此容易受到类别不平衡的影响。例如，在实验 C（零样本）中，表8显示 Gemini 2.5 Pro 在 2,495 个样本中正确预测了 2,188 个，总体准确率为 87.70%。需要特别指出的是，该分数并非对各类别百分比取平均所得，而是反映模型在整个数据集上的预测正确情况。

分类召回率各个 RIT 类别列（例如，动作冲突、TC、CC）表示**分类召回率**，其定义如下：

$$\text{Recall}_i = \frac{\text{Correct Predictions for Class } i}{\text{Total True Samples of Class } i}.$$

这些数值量化了模型在每个类别内识别实例的有效程度，与该类别出现的频率无关。例如，在同一项实验中，Gemini 2.5 Pro 的分类召回率——AC 为 59.53%、TC 为 94.28%、CC 为 99.55%——反映了模型针对特定类别的检测能力。所有百分比均采用统一的舍入方法，并计算至两位有效数字。

选择指标的依据 总体准确率与**分类召回率**相结合，可以均衡且易于理解地呈现模型性能。总体准确率符合现实部署场景，因为正确分类率取决于自然形成的偏斜分布。与此同时，分类召回率可避免多数类掩盖少数类别中的性能不足。综合来看，这些指标全面呈现了模型的总体行为以及细粒度的类别层面效能。

评估条件 我们在两种不同的条件下进行了实验，以探索不同的部署场景： **允许多个回答（实验 A、C、E）：** 在此条件下，大语

5 结果与讨论

为回答**RQ1（基准能力）**——即考察预训练大语言模型在真实*openHAB* 数据集中验证并分类规则交互威胁（RIT）的能力——结果显示，不同矛盾类别的基准推理表现存在明显差异。在受评模型中，*Llama-8b* 展现出最强的零样本能力，在不提供任何语境示例的情况下，总体准确率达到64.83%。它在弱动作矛盾（*WAC*）（80.49%）和弱条件矛盾（*WCC*）（75%）上的高准确率表明，较小的预训练大语言模型能够可靠地检测规则逻辑中明确或表层的不一致。然而，它在强动作矛盾（*SAC*）（51.11%）、强触发矛盾（*STC*）（16.67%）和强条件矛盾（*SCC*）（0%）上的表现不佳，显露出其在处理需要更深入语义推理的复杂或多层矛盾时存在明显局限。此外，在单样本（36.55%）和双样本（39.31%）设置下，准确率有所下降，这表明语境示例未必能提升较小模型的基准推理能力，甚至可能造成负面影响；这可能是因为模型在少样本设置下对语境的编码较为僵化，适应能力有限。相比之下，*Llama-70b*尽管参数规模显著更大，但表现并不稳定，其零样本设置下的准确率为50.34%，而单样本和双样本设置下均降至31.72%。

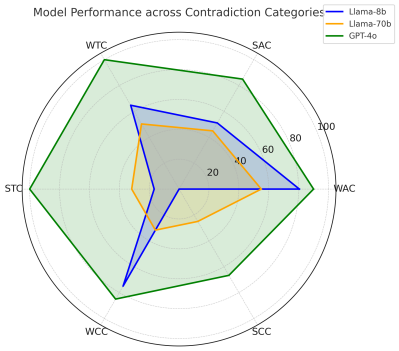
表 2：实验 A：使用 RIT 的全部 6 个类别并允许多个回答

模型	WAC	SAC	WTC	STC	WCC	SCC	总正确数
Llama-8b 零样本	80.49%	51.11%	50.00%	16.67%	75.00%	0.00%	64.83%
Llama-8b 单样本	53.66%	15.56%	0.00%	0.00%	50.00%	0.00%	36.55%
Llama-8b 双样本	54.88%	15.56%	100.00%	33.33%	25.00%	0.00%	39.31%
Llama-70b 零样本	67.07%	31.11%	0.00%	33.33%	50.00%	0.00%	50.34%
Llama-70b 单样本提示	52.44%	2.22%	0.00%	16.67%	25.00%	0.00%	31.72%
Llama-70b 双样本提示	51.22%	2.22%	50.00%	16.67%	25.00%	0.00%	31.72%
GPT-4o 零样本	32.93%	60.00%	100.00%	100.00%	0.00%	66.67%	45.52%
GPT-4o 单次提示	25.61%	33.33%	50.00%	100.00%	50.00%	66.67%	54.27%
GPT-4o 两次提示	29.27%	40.00%	50.00%	100.00%	75.00%	66.67%	60.16%
Gemini-2.5-Pro 零样本	75.61%	48.89%	50.00%	100.00%	25.00%	50.00%	65.52%
Gemini-2.5-Pro 单样本	43.90%	53.33%	50.00%	100.00%	25.00%	66.67%	49.66%
Gemini-2.5-Pro 双样本	47.56%	35.56%	0.00%	100.00%	25.00%	50.00%	44.83%
Deepseek-r1-7b 零样本	56.10%	53.33%	50.00%	33.33%	25.00%	16.67%	51.72%
Deepseek-r1-7b 单样本	75.61%	26.67%	0.00%	16.67%	25.00%	16.67%	53.10%
Deepseek-r1-7b 两次示例	64.63%	6.67%	50.00%	0.00%	0.00%	0.00%	39.31%

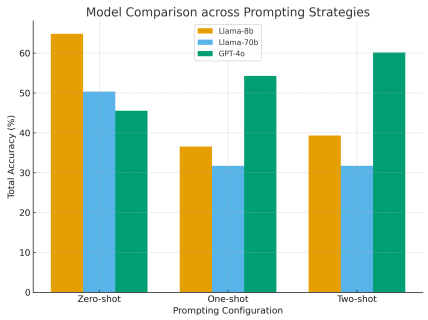
这种不稳定性表明，仅靠扩大模型规模并不能保证提升语境推理能力。结果说明，若缺乏充分的对齐或推理优化，较大的模型可能会过拟合局部语境线索；当它们需要验证涉及细微跨规则依赖关系的RIT时，性能便会下降。

针对**RQ2（模型规模效应）**——该问题探究模型规模如何影响推理一致性和上下文验证准确率——研究结果显示，*GPT-4o* 在所有设置下均优于两种 Llama 模型，并展现出显著的上下文学习提升趋势。其总准确率从 45.52%（零样本）稳步提升至 54.27%（单样本）和 60.16%（双样本），表明在提供少量示例后，它能够有效适应。与 Llama 模型不同，*GPT-4o* 在所有配置下的弱触发矛盾（*WTC*）和强触发矛盾（*STC*）任务中均保持 100% 的准确率，并在强条件矛盾（*SCC*）任务中持续取得较高分数（66.67%），表明其具备先进的上下文整合能力和多步推理能力。

雷达图（图3a）进一步印证了这一解读：GPT-4o 在所有矛盾类型中都保持了全面且均衡的性能表现，而两个 Llama 变体则呈现出明显的不均衡，仅在弱矛盾中表现出色，在强矛盾中却表现骤降。柱状图对比（图3b）进一步展现了这种差异——随着上下文信息的增加，GPT-4o 的性能稳步提升，而 Llama 模型的表现则有所下降。综合来看，这些发现证实，规模扩展与推理对齐共同决定了模型鲁棒性：仅靠参数规模并不足够，但若与有效的推理优化相结合——如 GPT-4o 所示——便能实现稳定且具备上下文感知能力的 RIT 验证。



(a) 模型在各矛盾类别（WAC、SAC、WTC、STC、WCC、SCC）上的表现。



(b) 不同提示配置下总体准确率趋势的比较。

图 3：大语言模型在不同 RIT矛盾类型和提示配置下的推理表现及扩展行为比较。

表 3：实验 B：使用 RIT 的全部 6 个类别，且不允许多个回答

模型	WAC	SAC	WTC	STC	WCC	SCC	总正确数
Llama-8b 零样本	18.29%	0.00%	0.00%	16.67%	75.00%	0.00%	13.10%
Llama-8b 单样本	4.88%	2.22%	0.00%	16.67%	25.00%	0.00%	4.83%
Llama-8b 双样本	10.98%	0.00%	50.00%	0.00%	25.00%	0.00%	7.59%
Llama-70b 零样本	51.22%	0.00%	0.00%	50.00%	50.00%	0.00%	32.41%
Llama-70b 单样本提示	50.00%	0.00%	50.00%	33.33%	0.00%	0.00%	30.34%
Llama-70b 双样本提示	28.05%	13.33%	100.00%	50.00%	0.00%	0.00%	23.45%
GPT-4o 零样本	29.27%	53.33%	100.00%	100.00%	25.00%	50.00%	59.60%
GPT-4o 单次提示	25.61%	31.11%	50.00%	100.00%	100.00%	66.67%	62.23%
GPT-4o 两次提示	13.41%	28.89%	0.00%	83.33%	75.00%	100.00%	50.11%
Gemini-2.5-Pro 零样本	51.22%	51.11%	50.00%	100.00%	25.00%	50.00%	52.41%
Gemini-2.5-Pro 单样本	47.56%	51.11%	100.00%	100.00%	25.00%	50.00%	51.03%
Gemini-2.5-Pro 双样本	46.34%	53.33%	0.00%	100.00%	25.00%	50.00%	49.66%
Deepseek-r1-7b 零样本	3.66%	6.67%	0.00%	66.67%	0.00%	0.00%	6.90%
Deepseek-r1-7b 单样本	4.88%	0.00%	0.00%	66.67%	0.00%	0.00%	5.52%
Deepseek-r1-7b 两次示例	4.88%	2.22%	0.00%	33.33%	0.00%	0.00%	4.83%

单响应约束的影响：当模型被限制为只能输出一种 RIT 分类（实验 B）时，与多响应设置相比，所有架构的总体准确率都显著下降。这一降幅凸显了基于大语言模型的威胁验证中固有的精确率–召回率权衡。在更严格的单响应条件下，*Llama-8b* 和 *Deepseek-r1-7b* 等较小模型的总体准确率大幅降低，分别跌至 15% 和 7% 以下，表明在被迫只选择一种解读时，它们的信心和泛化能力有限。*Llama-70b* 保持了尚算稳定的表现（约 30%），但仍未能

以有效发挥其规模优势。相比之下，*GPT-4o* 和*Gemini-2.5-Pro* 等推理优化模型表现出相对较强的韧性，即使在单标签约束下，准确率仍超过 50%。尤其是*GPT-4o*，保持了稳健的性能（59.6–62.2%），表明其置信度校准和内部推理一致性更优。

将这些发现与多响应结果进行比较可以看出，允许多个预测虽提高了召回率，却掩盖了不确定性；而单响应条件则揭示了每个模型真正的精确率和推理严谨性。这一对比证实，尽管更大且经过对齐优化的模型在更严格的约束下仍能维持平衡的推理能力，但规模较小或对齐程度较低的模型往往会过拟合情境线索，并难以进行明确分类。因此，从多响应评估转向单响应评估，能够更清晰地反映模型在自主RIT验证系统部署中的可靠性。

总而言之，在 RQ1 下，预训练大语言模型展现出识别 RIT 的能力有限但并非微不足道，尤其是在弱矛盾情境中。然而，当受到单响应约束时，较小模型的这些能力会显著下降，暴露出在消除歧义后进行置信度校准和果断推理方面的挑战。在 RQ2 下，只有与推理对齐相结合时，规模效应才仍然显著——这表明仅增大参数规模并不能确保稳健性。诸如*GPT-4o* 和*Gemini-2.5-Pro* 之类的模型，在两种评估条件下都表现出更强的情境适应性和语义连贯性，并在分析*openHAB* 数据集中的基于规则的矛盾时保持了均衡的精确率和召回率。总体而言，这些发现强调，决定 RIT 在不同部署约束下能否得到一致且具备情境意识的验证的，是推理对齐，而非单纯的规模。

缩减类别评估：简化威胁分类体系的影响实验 C 考察了在分类空间缩减为三个主要 RIT 类别——动作矛盾（AC）、触发矛盾（TC）和条件矛盾（CC）——并允许给出多个回答时，模型的表现。如表 4 所示，简化分类体系后，几乎所有模型的准确率都大幅提升，这表明降低语义复杂度有助于大语言模型更有效地泛化到不同威胁类型。

在零样本条件下，*Llama-8b* 和 *Deepseek-r1-7b* 等较小模型的总体准确率最高（最高达 88–91%），较其在完整六类别设置下的表现显著提升。这表明，当决策空间受到限制时，这些模型能够更可靠地将上下文线索映射到较宽泛的矛盾模式。同样，*Llama-70b* 和 *Gemini-2.5-Pro* 等较大模型的准确率稳定在中等范围（60–69%），反映出分类边界减少后，类别对齐有所改善。

相比之下，*GPT-4o* 在三类别和六类别实验中的结果均相对稳定（58–64%），这表明无论标签粒度如何，其泛化能力都较为稳健，

表 4：实验 C：仅使用 3 类 RIT，并允许多个回答

模型	AC	TC	CC	总正确数
Llama-8b 零样本	92.91%	100.00%	20.00%	88.28%
Llama-8b 单样本	88.19%	100.00%	30.00%	84.83%
Llama-8b 双样本	56.69%	87.50%	30.00%	56.55%
Llama-70b 零样本	73.23%	50.00%	30.00%	68.97%
Llama-70b 单样本提示	63.78%	50.00%	20.00%	60.00%
Llama-70b 双样本提示	66.14%	62.50%	50.00%	64.83%
GPT-4o 零样本	25.20%	75.00%	90.00%	63.40%
GPT-4o 单次提示	25.20%	87.50%	80.00%	64.23%
GPT-4o 两次提示	29.13%	87.50%	60.00%	58.88%
Gemini-2.5-Pro 零样本	57.48%	87.50%	100.00%	62.07%
Gemini-2.5-Pro 单样本	55.12%	100.00%	100.00%	60.69%
Gemini-2.5-Pro 双样本	55.91%	100.00%	100.00%	61.38%
Deepseek-r1-7b 零样本	96.06%	87.50%	30.00%	91.03%
Deepseek-r1-7b 单样本	96.06%	75.00%	30.00%	90.34%
Deepseek-r1-7b 两次示例	55.91%	75.00%	60.00%	57.24%

这种一致性凸显了其均衡的推理能力和语义抽象能力，能够避免对类别结构产生过拟合。

相比之下，这些发现表明，类别简化缩小了小型模型与大型模型之间的推理差距，提高了基线准确率，但也降低了用于区分更深层推理对齐能力的挑战性。归根结底，简化后的分类体系凸显出，尽管结构简化能够提升表层准确率，但复杂的多类别推理仍然是区分 GPT-4o 和 Gemini-2.5-Pro 等先进推理优化架构的真正要素。

简化分类体系下的单响应评估 实验 D 研究了在简化的三类别 RIT分类体系（动作矛盾（*AC*）、触发器矛盾（*TC*）和条件矛盾（*CC*））中施加单响应约束的影响。如表5所示，与允许多个响应的实验 C 相比，整体性能有所下降。这一结果进一步印证了实验 B 中先前观察到的精确率–召回率权衡，表明虽然减少类别数量能够简化推理空间，但将输出限制为单个标签会暴露出模型在置信度方面的潜在局限。

较小规模的模型，如*Llama-8b*和*Deepseek-r1-7b*，表现出明显的性能下降（总准确率低于25–56%），这证实了尽管单标签约束受益于分类体系的简化，但这些模型仍对该约束较为敏感。相比之下，较大规模的模型，如*Llama-70b* 以及面向推理的

表5：实验 D：仅使用 3 类 RIT，且不允许多个回答

模型	AC	TC	CC	总正确数
Llama-8b 零样本	52.76%	25.00%	10.00%	48.28%
Llama-8b 单样本	58.27%	50.00%	30.00%	55.86%
Llama-8b 双样本	50.39%	0.00%	0.00%	44.14%
Llama-70b 零样本	62.99%	12.50%	40.00%	58.62%
Llama-70b 单样本提示	64.57%	25.00%	30.00%	60.00%
Llama-70b 双样本提示	67.72%	0.00%	40.00%	62.07%
GPT-4o 零样本	30.71%	87.50%	60.00%	59.40%
GPT-4o 单次提示	28.35%	75.00%	60.00%	54.45%
GPT-4o 两次提示	28.35%	75.00%	70.00%	57.78%
Gemini-2.5-Pro 零样本	48.03%	87.50%	70.00%	51.72%
Gemini-2.5-Pro 单样本	44.88%	100.00%	60.00%	48.97%
Gemini-2.5-Pro 双样本	44.88%	100.00%	60.00%	48.97%
Deepseek-r1-7b 零样本	19.69%	0.00%	40.00%	20.00%
Deepseek-r1-7b 单样本	21.26%	0.00%	70.00%	23.45%
Deepseek-r1-7b 两次示例	22.05%	0.00%	40.00%	22.07%

v

架构（如*GPT-4o* ）保持了中等程度的稳定性（58–62%），表明规模扩展与推理对齐能够共同减轻过度承诺错误。有趣的是，*Gemini-2.5-Pro* 在不同类别上的准确率表现不均衡——在TC上达到100%，但总体准确率不稳定——这表明其倾向于根据显著的上下文触发器进行激进的决策。

相比之下，实验 C（多响应）虽然提高了召回率并掩盖了不确定性，但实验 D 更清晰地呈现了各模型在受限决策压力下的推理规范性。结果证实，即使在类别减少的情境下，对齐优化模型（*GPT-4o*、*Llama-70b*）的推理仍比规模较小或对齐程度较低的模型更加连贯，且置信度校准得更好。总之，类别简化与单响应约束之间的相互作用，划定了表层泛化与稳健语义推理之间的界限，并拓展了 RQ1 和 RQ2 的研究发现。

总体而言，当各模型被限制为只能输出单一 RIT 标签时，其准确率均持续下降，这再次印证了先前分析中发现的精确率–召回率权衡。对于*Deepseek-r1-7b* 等较小型架构，这种下降尤为显著：其总准确率从 90% 以上降至约 23%，表明模型在被迫做出单一分类时，置信度会大幅降低。相反，*GPT-4o* 和*Llama-70b* 等较大型或更注重推理的模型则保持了相对稳定的表现（约为 55–60%），展现出召回率与精确率之间更均衡的校准结果。这些发现表明，尽管多响应的灵活性有助于

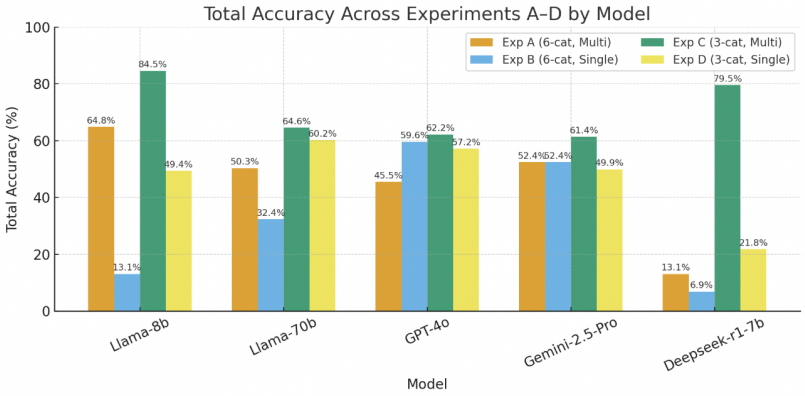


图 4：实验 A–D 中所有模型的总准确率对比，展示了六类别和三类别 RIT 分类体系下多响应与单响应条件中的性能趋势。

表 6：变异数据集上的实验 A

模型	WAC	SAC	WTC	STC	WCC	SCC	总正确数
Llama-8b 零样本	74.42%	36.44%	75.17%	22.41%	57.78%	12.17%	42.97%
Llama-8b 单样本	60.13%	24.47%	74.50%	12.64%	63.72%	8.85%	39.20%
Llama-8b 双样本	82.06%	19.95%	83.56%	11.49%	41.25%	4.15%	35.19%
Llama-70b 零样本	72.43%	9.84%	35.91%	13.79%	53.13%	2.07%	29.34%
Llama-70b 单样本提示	17.61%	4.79%	31.21%	12.64%	29.37%	2.77%	15.59%
Llama-70b 双样本提示	37.87%	19.68%	30.54%	20.11%	26.48%	2.35%	19.88%
Gemini-2.5-Pro 零样本	42.86%	39.36%	87.25%	81.03%	41.89%	51.73%	52.63%
Gemini-2.5-Pro 单样本	36.21%	55.32%	76.85%	80.46%	49.76%	63.35%	58.28%
Gemini-2.5-Pro 双样本	35.88%	60.11%	19.13%	89.08%	48.31%	57.95%	50.74%
Deepseek-r1-7b 零样本	76.08%	36.17%	22.15%	31.61%	30.02%	24.07%	33.95%
Deepseek-r1-7b 单样本	80.40%	21.81%	8.72%	45.40%	24.40%	24.62%	30.42%
Deepseek-r1-7b 两次示例	76.08%	13.83%	11.74%	35.06%	19.10%	25.73%	27.33%

提高召回率，但单响应评估更能准确反映模型推理在真实 RIT 验证情境中的可靠性。图 4 展示了所有受评估模型在四项实验（实验 A-D）中的表现对比。

5.1 可扩展性与泛化能力（RQ3）在此前使用 openHAB 和 IoTB 数据集开展实验的基础上，RQ3 将关注重点从受控条件下的准确率转向大规模场景下的鲁棒性。变异数据集的规模大出几个数量级，且完全由真实矛盾案例构成，因此对泛化能力提出了更严峻的考验。这种设置消除了歧义，增加了结构多样性，并让模型面对数百个合成生成但语义一致的漏洞

表 7：在变异数据集上的实验 B

模型	WAC	SAC	WTC	STC	WCC	SCC	总正确数
Llama-8b 零样本	15.95%	0.80%	27.18%	37.36%	23.27%	1.38%	14.11%
Llama-8b 单样本	11.96%	1.06%	23.49%	13.22%	40.61%	6.36%	17.31%
Llama-8b 双样本	14.95%	0.00%	24.83%	29.31%	28.41%	0.69%	14.11%
Llama-70b 零样本	35.55%	6.38%	25.50%	15.52%	38.68%	0.00%	19.04%
Llama-70b 单样本提示	12.62%	7.98%	46.98%	13.79%	37.24%	0.28%	18.68%
Llama-70b 双样本提示	16.94%	37.77%	48.66%	23.56%	26.48%	0.00%	21.80%
Gemini-2.5-Pro 零样本	34.55%	39.89%	80.54%	74.14%	37.72%	48.41%	48.42%
Gemini-2.5-Pro 单样本	31.89%	54.79%	72.15%	76.44%	47.51%	60.86%	55.55%
Gemini-2.5-Pro 双样本	30.90%	58.24%	18.79%	87.36%	46.39%	55.46%	48.50%
Deepseek-r1-7b 零样本	10.30%	4.52%	0.67%	44.25%	6.10%	2.35%	7.29%
Deepseek-r1-7b 单样本	5.65%	1.33%	0.67%	65.52%	2.09%	4.84%	7.45%
Deepseek-r1-7b 两次示例	6.31%	2.66%	0.67%	58.05%	2.57%	10.10%	8.86%

实验 A（六类别分类体系，允许多个回答）

在 oHC/IoTB 数据集上，较小的 Llama 变体和推理优化模型展现出各有长短的表现。Llama-8b 的零样本总分高达 64.83%，在 WAC 和 WCC 上表现良好，但在 SCC 上表现不佳。Gemini-2.5-Pro 的零样本成绩为 65.52%，而 GPT-4o 使用双样本输入后进一步提升至 60.16%。将相同的评估应用于变异数据集后发现，大多数模型的表现显著下降：Llama-8b 在 0/1/2 样本设置下的总分分别降至 42.97%/39.20%/35.19%，而 Llama-70b 则从约 29.3% 降至 15.6%，随后回升至 19.9%。Deepseek 的表现降至 30% 多。相比之下，Gemini-2.5-Pro 在变异数据集上的总分仍显著较高（52.6–58.3%），表明其对变异驱动的真实漏洞实例具有更强的泛化能力。在多响应评估中，推理对齐架构仍保持优势；而面对规模更大、语义更多样的数据集时，较小的 Llama 模型表现则大幅下滑。

实验 B（六类别分类体系，仅限单响应）

将输出限制为单个响应后，模型间的差异更加明显。在 oHC/IoTB 数据集上，许多模型的单响应总得分大幅下降（例如，Llama-8b 在零样本设置下仅为 13.10%），而 GPT-4o 依然表现稳健（不同样本设置下为 59.6–62.2%），Gemini 则稳定在 50%出头。在变异数据集上，单响应表现进一步下滑：Llama-8b 仍处于较低水平（14–17%），Llama-70b 略有提升但依旧较弱（19–21.8%），Deepseek 则仍然表现不佳（7–9%）。Gemini-2.5-Pro 在所有变异样本中始终保持中等水平的单标签准确率，这表明针对推理和对齐进行优化的模型更能应对严格的决策约束。这些结果表明，单响应评估暴露了模型在置信度校准方面的局限，而推理/对齐能力有助于提高其选出正确明确标签的准确率。

表8：Mutation 数据集上的实验 C

模型	AC	TC	CC	总正确数
Llama-8b 零样本	84.93%	96.82%	27.19%	56.03%
Llama-8b 单样本	77.55%	94.70%	20.51%	50.02%
Llama-8b 双样本	55.39%	70.97%	48.51%	54.63%
Llama-70b 零样本	79.76%	40.04%	31.72%	46.33%
Llama-70b 单样本提示	75.63%	23.73%	23.40%	37.64%
Llama-70b 双样本提示	65.58%	34.53%	41.75%	46.85%
Gemini-2.5-Pro 零样本	59.53%	94.28%	99.55%	87.70%
Gemini-2.5-Pro 单样本	57.02%	93.43%	98.14%	86.09%
Gemini-2.5-Pro 双样本	49.93%	93.86%	98.22%	84.29%
Deepseek-r1-7b 零样本	94.68%	83.26%	30.39%	57.84%
Deepseek-r1-7b 单样本	97.64%	83.69%	28.53%	57.72%
Deepseek-r1-7b 两次示例	66.47%	67.16%	68.57%	67.74%

实验 C（三分类法，允许多个回答）

降低标签粒度后，所有模型的绝对总得分均有所提高，这一趋势在变异数据集上也同样存在。在变异实验 C 中，Gemini-2.5-Pro 的总得分非常高（不同样本设置下为 ~84–88%），Deepseek 也有显著提升（~57–67%）。Llama-8b 和 Llama-70b 虽有小幅进步（分别为 ~50–56% 和 ~37–46%），但仍落后于推理优化模型。尽管降低分类体系的粒度缩小了多响应模式下小型模型之间的差距，推理对齐架构仍取得了最大幅度的提升。降低分类体系粒度对所有模型都有益，但对齐和推理优化带来的性能最为稳健，也最具泛化能力。

实验 D（三分类法，仅限单响应）

当三分类空间与单响应约束结合时，整体表现低于多响应评估，但对于某些模型，高于六分类单响应评估。在变异数据集上，Gemini-2.5-Pro 的单响应总成绩保持强劲（69.5–70.5%），表明其在更严格的决策条件下具有较高的精确率和稳定性。Llama-70b 的得分处于 30 多到 40 出头（35–41%），Deepseek徘徊在 30 出头，而 Llama-8b 相较六分类单响应表现略有提升（29–33%）。尽管变异数据集削弱了在 oHC/IoTB 上观察到的部分优势，但推理对齐模型保留单标签能力的表现远胜于未对齐的 Llama 变体，展现出其在受限评估下的稳定性和韧性。

表9：变异数据集上的实验 D

模型	AC	TC	CC	总正确数
Llama-8b 零样本	41.95%	30.30%	23.70%	29.90%
Llama-8b 单样本	44.61%	23.31%	27.12%	31.14%
Llama-8b 双样本	53.47%	31.99%	22.88%	32.91%
Llama-70b 零样本	63.81%	23.94%	35.51%	41.00%
Llama-70b 单样本提示	53.32%	23.94%	33.58%	37.11%
Llama-70b 双样本提示	62.48%	20.13%	26.97%	35.31%
Gemini-2.5-Pro 零样本	46.53%	92.37%	73.11%	69.54%
Gemini-2.5-Pro 单样本	45.49%	91.95%	75.63%	70.54%
Gemini-2.5-Pro 双样本	44.46%	79.24%	80.24%	70.34%
Deepseek-r1-7b 零样本	34.42%	6.78%	35.29%	29.66%
Deepseek-r1-7b 单样本	34.12%	8.26%	44.73%	34.95%
Deepseek-r1-7b 两次示例	26.14%	9.96%	47.18%	34.43%

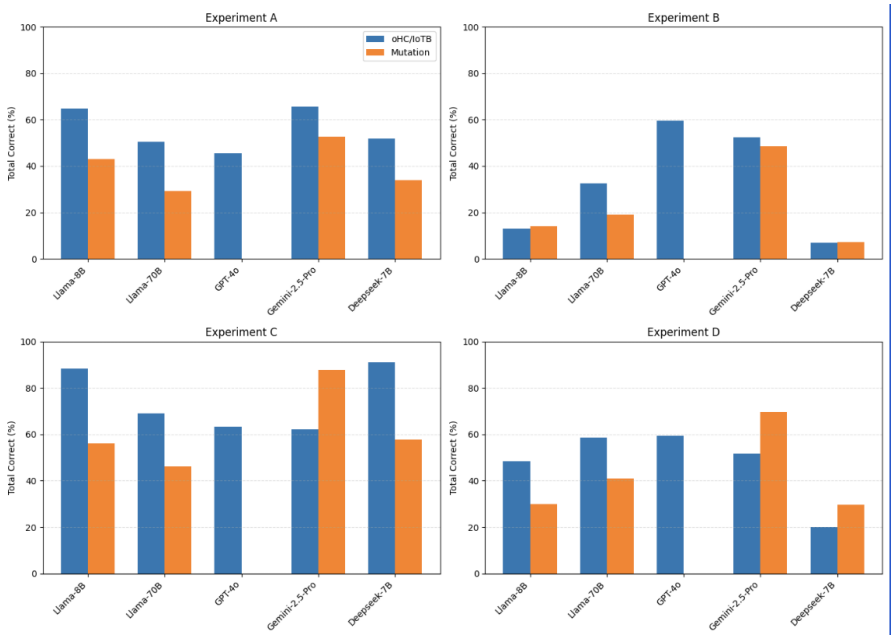


图5：比较实验 A–D（Mutation 与 oHC/IoTB 数据集）中所有模型的总准确率，展示了在六类别和三类别 RIT 分类体系下，多响应和单响应条件中的性能趋势。

在全部四项实验中，都呈现出一个清晰的规律。具备明确推理能力并经过对齐的模型，例如 Gemini-2.5-Pro，以及在较小程度上 GPT-

4o 在大型变异数据集和更严格的单响应约束下，仍能保持性能优势。这些模型的泛化能力更强，即使每个实例都代表真实漏洞也是如此。相比之下，Llama 系列模型，尤其是未经对齐调优的较大变体，以及较小的架构，在变异数据上表现大幅下降，在单响应或多类别压力下尤其明显。将分类体系缩减为三类可以提升所有模型的准确率，但会掩盖泛化能力方面更深层的差异。对齐优化模型保持着更高的绝对性能和稳健的泛化能力，而未对齐模型仅有小幅提升，且仍然脆弱。总的来说，这些发现表明，真正的可扩展性和泛化能力主要依赖推理与对齐能力，而非单靠参数数量。

5.2 与静态分析工具的比较 作为后续实验，

尽管大语言模型在检测 RIT 方面展现出潜力，但其性能仍不及 oHIT 等静态分析工具在召回率和一致性方面的表现。模型无法可靠地区分相似的 RIT 模式，且依赖提示设计，这凸显了采用混合方法的必要性：将符号推理与大语言模型结合起来，以减少假阳性并提高准确性。

5.3 混合方案：将大语言模型与静态分析相结合 本小节回答**研究问题4**。在静态分析中，我们注意到，假阳性总是由有限且可明确识别的原因造成，通常与静态分析工具缺乏上下文理解能力有关。例如，oHIT 中假阳性的一个主要来源，是在判断触发器重叠时出现的问题。比如，由于变量命名约定，oHIT 可能会错误地判定触发器重叠，例如，触发器 A 被设置为在*“cron ’0 8 * * *””*（上午 8 点）触发，而触发器 B 则在*“Sun_Is_Setting_Event changes to ON”*时触发。人类显然能看出上午 8 点与日落永远不会重叠，但静态分析缺乏常识，无法做出这种区分。

表10：实验 G：通过检测 oHIT 中的假阳性来补充静态分析工具

Tool	方法	WAC	SAC	WTC	STC	WCC	SCC	总计
oHIT	静态	83.68%	88.24%	16.67%	46.15%	100.00%	100.00%	72.53%
oHIT + Llama-8b	零样本	62.24%	41.18%	50.00%	76.92%	100.00%	100.00%	71.72%
oHIT + Llama-8b	单样本	29.59%	58.82%	66.67%	53.85%	100.00%	100.00%	68.15%
oHIT + Llama-8b	双样本	33.67%	45.10%	66.67%	61.54%	100.00%	100.00%	67.83%
oHIT + GPT-4o	零样本	91.84%	94.12%	58.33%	69.23%	100.00%	100.00%	85.59%
oHIT + GPT-4o	单样本	90.82%	96.08%	41.67%	53.85%	100.00%	100.00%	80.40%
oHIT + GPT-4o	双样本	92.86%	98.04%	66.67%	76.92%	100.00%	100.00%	89.08%
oHIT + Gemini 2.5	零样本	89.80%	96.08%	83.33%	53.85%	100.00%	100.00%	87.18%
oHIT + Gemini 2.5	单样本	89.80%	98.04%	83.33%	84.62%	100.00%	100.00%	92.63%
oHIT + Gemini 2.5	双样本	89.80%	98.04%	83.33%	84.62%	100.00%	100.00%	92.63%

oHIT 中另一个重要的假阳性来源，是它对触发器级联威胁的检测。触发器级联可能是危险的，也可能是有意为之，而静态分析无法区分两者。

为了评估大语言模型如何增强符号规则交互流水线，我们使用三种混合配置——oHIT + Llama-8B、oHIT + GPT- 4o 和 oHIT + Gemini 2.5（推理）——在 oHC + IoTB数据集上进行了评估。这些实验将 oHIT 的静态检测结果作为大语言模型“裁定”阶段的输入，由模型判断每个候选交互是否构成真实或虚假的 RIT。表10中报告的“总计”数值代表经过大语言模型裁定步骤后达到的最终精确率。为控制运行成本，同时涵盖不同的推理风格，我们选择了三种互补的模型：

– Llama-8B——轻量、高性价比的非推理基线；– GPT-4o——大型、高保真的非推理模型；– Gemini 2.5 Reasoning——一种面向推理的模型，在试点测试显示其一致性更高后，取代了 DeepSeek-R1。

每种混合变体均在零样本、单样本和双样本提示配置下运行，使用以下生成参数：温度 $= 0.2$、top_p$= 0.95$、max_output_tokens $= 2048$。

判定流水线将 oHIT 的 RIT 候选项拆解为细粒度推理子任务，每项均由大语言模型独立作答：

– 触发器重叠分析 – 判断两个规则触发器是否可能同时触发（例如，日出_事件与日落_事件）。– 触发器级联安全性 – 判断触发器级联是否可能是用户有意设置的，或是否可能产生不良行为。– 动作冲突检查 – 比较两个动作，判断它们是否为同一设备属性赋予不兼容的值。

混合架构通过利用大语言模型关于触发器意图和设备关系的语义先验，持续减少了假阳性。Gemini 2.5 模型取得了最佳结果。所有 RIT 类别的精确率均有所提升或保持稳定，使总体精确率提高

表 11：oHIT 在变异数据集上的性能

类别	规则集总数	正确检测	未检测	检测率
SAC	376	376	0	100.00%
WAC	301	299	2	99.34%
STC	174	158	16	90.80%
WTC	298	263	35	88.26%
SCC	723	690	33	95.44%
WCC	623	595	28	95.51%
总计	**2,495**	**2,381**	**114**	**95.43%**

表 12：模型在 oHIT 于变异数据集上产生的假阴性中的最佳表现

类别	规则集总数	正确检测	未检测	检测率
SAC	0	-	-	-
WAC	2	2	0	100.00%
STC	16	2	14	12.50%
WTC	35	18	17	51.43%
SCC	33	33	0	100.00%
WCC	28	6	22	21.43%
总计	**114**	**61**	**53**	**53.51%**

从 0.73（仅使用 oHIT 的基线）提升至最佳混合设置（两次采样）下的 0.93。这些结果证实，模块化的符号与大语言模型流水线能够显著提高物联网安全验证的准确性，而无需重新训练或改动原有的静态分析引擎。

除了在静态分析流水线中加入大语言模型来判定假阳性和假阴性之外，我们还进行了镜像实验，以确定其在判定假阴性方面的价值。由于 oHIT 在合并后的 IoTB 和 oHC 数据集上达到了 100% 的召回率，因此假阴性是根据 oHIT 在变异数据集上的表现得出的。这一恢复阶段的提示策略与验证阶段有着根本差异。在判断一项阳性识别是否有效时，会向大语言模型提供来自静态分析工具的上下文线索。具体而言，包括被标记的 RIT、相关条件以及涉及的具体规则。然而，在向大语言模型输入阴性样本时，这些上下文线索并不存在；大语言模型必须进行盲目搜索，以识别静态分析遗漏的威胁。

表 11 详细说明了 oHIT 在变异数据集上的表现。该工具表现出色，检测率达到 95.43%，但仍有漏检，主要源于两个方面：类型处理差异和流水线僵化。具体而言，条件级联变异（SCC/WCC）有时会产生复杂类型（例如 PointType），导致静态解析器无法解析；而依赖 openHAB 内部状态传播（postUpdate 触发 sendCommand）的触发器级联变异（STC/WTC），则会被 oHIT 更严格的事件匹配逻辑拒绝。

大语言模型尝试找回这些遗漏威胁的结果见表 12。对于 STC 和 WTC，结果表明，导致静态分析失败的特定细节——主要涉及发布更新与发送命令之间不匹配的差异——

表 13：按实验和提示策略划分的每条提示令牌数（使用 Ollama 默认分词器，不包括模型规则集）

RIT 分类	三字母 RIT		两字母 RIT	
实验	实验 A	实验 B	实验 C	实验 D
零样本	492	527	441	480
单样本	961	997	703	753
双样本	1928	1992	935	972

对大语言模型而言同样难以理解。相反，模型在 SCC 上表现极为出色，成功找回了 100% 的假阴性。这一成功凸显了二者的关键互补性：oHIT 未能发现这些威胁，是因为它无法适应变异工具生成的句法异常；而大语言模型则能够通过语义近似来适应这些不规则情况。WCC 的结果与模型在实验 A 中的整体表现高度一致，证实了涉及部分条件的“弱”威胁，无论实验设置如何，对模型而言始终是一项挑战。

5.4 令牌长度、上下文使用与性能特征

在所有实验中，系统提示、指令和规则集的组合使提示长度上限达到 2,200 个令牌。最大的基础提示——实验 B（双样本）——包含 1,992 个令牌，该数值使用每个模型对应 .gguf 文件中的默认分词器计算得出。所有本地模型均配置了 8,192 个令牌的上下文窗口，因此未发生输入截断，见表 13。

性能分析显示，不同部署模式之间存在明显差异。Llama 3.1 8B 模型实现了快速的本地推理（2.57 秒），且其内存需求仍在现代边缘加速器的承载能力范围内。尽管 DeepSeek-R1 7B 的内存占用仅为 5.3 GB，但由于其推理流程，延迟显著更高（20.42 秒）。云端推理消除了硬件限制，但带来了显著延迟；Gemini-2.5-Pro 的平均响应时间为 24.55 秒，见表14。

可靠性方面的差异十分显著。Gemini 曾产生一次空白输出，并累计出现 6,340 次 HTTP 错误，包括 HTTP 429 （请求过多）和 HTTP 503 （服务不可用），有时需要多次调用 API 才能获得有效响应。这些故障凸显了云卸载控制架构存在的结构性脆弱问题：API 配额和服务可用性会损害实时性保证。

总体而言，尽管云端大型推理模型具备更强的推理能力，但对于确定性要求高、对延迟敏感的物联网任务，本地执行仍不可或缺；要实现稳健的自主运行，可能需要采用混合方案。

表14：不同模型和计算资源的平均生成时间

计算资源	移动端 3070ti 8GB GDDR6			H100 服务器
模型	Llama 3.1 8B	Deepseek R1 7b	Gemini-2.5-Pro	Llama 3.1 70B
平均生成时间	2.57秒	20.42秒	24.55秒	13.58秒

6 相关工作

6.1 物联网漏洞检测工具目前已有许多用于分析物联网应用、确保其安全的方法。值得关注的工具包括：Soteria [5], IoTSan [20], iRuler [34], 和 IoTCom [2]，它们都采用静态分析技术，从源代码中提取某种中间表示，并将其输入模型检查器，以分析设备交互。

物联网分析领域的主要工具包括 Soteria [5], 、IoTSan [20], 、iRuler [34], 和 IoTCom [2]。这些工具通过提取中间代码表示并应用模型检查，对 SmartThings 和 IFTTT 应用进行检查，从而采用静态分析。其他工具，如 IoTGuard [6], 、IoTBox [17], 和 HAWatcher [14],，则依赖动态方法，在程序执行期间进行分析 [1]。混合方法结合了这两种方法，例如 SmartFuzz [31]和 IOTSAFE [11] [1]。SafeTAP [18], 、AutoTAP [36], 和 MenShen[4] 等工具专注于物联网系统设计，帮助用户从一开始就创建安全的设备规则。这些进展使人们能够更深入地分析 SmartThings 应用中的设备交互、事件流和权限，从而提升安全性和可靠性。

在开源家庭自动化领域，openHAB 是一个广泛使用的平台，但注重安全的开发仍然有限。有两种工具旨在支持多个平台。VISCR [19] 会生成基于树的代码抽象，并为 SmartThings、openHAB、IFTTT 和 MUD 应用采用基于图的策略。PatrIoT [35] 则定义安全属性，并在系统运行期间防止策略违规，同时支持 SmartThings、openHAB 和 EVA ICS。

6.2 大语言模型软件漏洞检测方法对近期文献的回顾揭示了一个显著趋势：研究者正采用多种方法、针对不同目标，利用大语言模型进行源代码安全分析。多项研究探讨了将大语言模型（如 GPT-4 和 Claude）直接用于特定场景下的漏洞检测，例如智能合约。比如，Chen 等人通过提示工程评估 ChatGPT 检测智能合约漏洞的效果，并将其与其他检测工具进行比较 [7]。David 等人则采用思维链方法，使用 GPT-4 和 Claude 审计智能合约，并考察代码上下文的长度如何影响分析结果 [10]。这些研究表明，尽管大语言模型能够有效识别漏洞，但也可能产生假阳性，并且在处理较长代码上下文时存在局限性。

其他研究探索了更专业的大语言模型的能力，例如 CodeBERT 和 GPT 系列中的模型在不同类型分析中的表现

例如代码补全和错误修复。Omar 等人采用基于 CodeBERT 的 Transformer 语言模型，并对其进行微调，以检测软件漏洞；随后将其与 LSTM 模型和静态分析工具进行比较 [22]。Pearce 等人使用 Codex、Jurassic-1 和 polycoder，并采用不同的提示技术，研究零样本漏洞修复 [25]。Sandoval 等人评估了由 Codex 驱动的代码补全助手对开发人员所编写代码的安全影响 [29]。

许多研究致力于提升大语言模型的性能，通常采用专门训练或特定技术。例如，在 [13] Ferrag 等人推出了 SecureFalcon，这是一种经过微调的轻量级大语言模型，采用迁移学习和对比学习；He 等人则探索使用 Codex、PaLM、AlphaCode 和 CodeGen 进行受控代码生成 [16]。Setak 等人研究了利用教师大语言模型提升大语言模型执行代码变异的能力，并在 Llama3 [30], 上进行了实验；Tihanyi 等人提出一种方法，将大语言模型与 ESBMC 模型检查器集成，在修复代码时将模型检查器的输出作为提示 [32]。Venkatesh 等人对包括 GPT 3.5 Turbo、GPT 4 和 llama2 在内的 26 个大语言模型进行了全面评估，用于 Python 程序的调用图分析和类型推断 [33]。最后，Zhou 等人将 12 个开源大语言模型和 ChatGPT 与 15 种 SAST 工具进行了比较，并提出了一种新颖的代码仓库级漏洞检测方法 [38]。他们采用了多种基于提示的方法，例如零样本提示、思维链（CoT）提示和微调。这些论文展示了微调、专门训练及其他方法在优化大语言模型、使其适用于源代码分析任务方面的潜力。

为将我们对规则交互威胁（RIT）的研究置于背景之中，我们考察了近期有关软件安全领域大语言模型应用，以及物联网漏洞检测具体挑战的综合性调查。这些研究为我们的混合方案提供了理论基础，既凸显了生成式模型的潜力，也强调了在安全关键环境中静态分析依然不可或缺。

Zhu 等人 [39] 对软件安全领域的大语言模型进行了系统综述，并将其应用归纳为模糊测试、单元测试、程序修复和错误分类等关键领域。他们将这些技术分解为预处理、提示生成和后处理等阶段，并指出，尽管大语言模型显著减少了人工工作量，但仍存在明显局限。他们的一项重要发现与我们在 RQ1 中的实验结果相呼应：在零样本设置下直接使用的大语言模型往往难以精确检测漏洞，准确率较低，因此需要进行微调或采用更具引导性的使用方式。这一局限也印证了我们采用“LLM-oHIT 协调”工作流的决定：让大语言模型充当后处理验证器，而非独立的检测引擎。

相反，Zhou 等人 [37] 考察了模型本身固有的安全影响，重点关注幻觉、偏见和可靠性等风险。他们的综述指出，尽管 ChatGPT 等模型在自然语言处理（NLP）方面取得了显著进步，但它们

容易生成看似合理却不真实的内容——这种现象称为幻觉——并且对对抗性操纵高度敏感。这项分析为我们在变异数据集（RQ3）中观察到的性能下降提供了理论依据。面对强触发级联（STC）所要求的严格结构逻辑时，模型往往优先考虑语义流畅性，而非逻辑正确性；Zhou 等人指出，这一可靠性差距是大语言模型应用于高风险决策场景的主要障碍。物联网固件中的漏洞检测将范围缩小到物联网（IoT）领域，

Feng 等人 [12] 将漏洞检测方法分为四个不同类别：基于仿真的测试、自动代码分析、网络模糊测试和手动逆向工程。他们的综述突出了物联网领域所面临的独特挑战，尤其是架构的多样性，以及为动态分析创建准确仿真环境的困难。最重要的是，Feng 等人指出，尽管自动代码分析（静态分析）——即我们的基线 oHIT 工具所属的类别——无需设备实体即可运行，但此类代码匹配解决方案适用于大规模测试。他们还强调，这类方法不可避免地会面临假阳性和假阴性问题，因此很难在效率与准确性之间取得平衡。在我们的工作中，我们将这些错误解释为静态分析缺乏运行时上下文的表现。针对 Feng 等人所发现的这一具体空白，我们提出了一种混合架构，利用大语言模型的语义理解能力，过滤传统静态分析工具生成的缺乏上下文的假阳性结果。

7 效度威胁

内部效度。 本研究结合了自然产生的 openHAB 规则和系统生成的变异规则，涵盖了广泛的交互模式。尽管原始 oHC/IoTB 数据集存在类别不平衡——弱动作矛盾（WAC）在真实标注中占比更高——但这种分布反映了现实世界中的部署情况，因此增强了生态效度。为确保没有任何类别主导评估，我们进行了归一化处理，并开展了四项受控实验（A–D），以区分类别范围和响应约束的影响。虽然大语言模型可能会生成多个合理的 RIT，但评估统一以 oHIT 已验证标签为准，以保持方法上的一致性。这种保守的评分方式为大语言模型的性能提供了严格的下限。

构念效度。 威胁类别（WAC、SAC、WTC、STC、WCC、SCC）源自成熟的基于 TAC 的分类体系，并在符号分析和基于大语言模型的分析中始终如一地应用。通过在四种受控实验配置下评估模型，我们明确考察了模型行为是否取决于类别粒度或输出约束。尽管大语言模型有时会提出 oHIT 分类之外的其他威胁

，我们的评分框架优先考虑可复现性和精确率。未来的工作可以扩展评估范围，纳入对大语言模型发现的其他威胁的验证；但当前方法确保了所有模型和数据集中的构念稳定性与可比性。

外部效度。 使用来自两个独立社区代码库的 openHAB 规则，并结合大型基于变异的语料库，扩大了语法和语义变化的覆盖范围。该变异数据集引入了在自然代码库中并不常见、但对评估大语言模型泛化能力很有价值的规则结构。尽管研究结果未必能直接推广至所有物联网平台，但 openHAB 所采用的触发器-动作-条件模型能够代表当代智能家居生态系统。此外，纳入多个模型系列（Llama 3.1、GPT-4o、Gemini 2.5 Pro、DeepSeek-R1）有助于支持研究结果在不同架构范式和参数规模上的可泛化性。

结论效度。 模型性能通过各类别准确率以及四种受控设置下汇总的总正确数指标进行衡量。这些互补的评估视角有助于减轻数据集不平衡或模型对主要类别过拟合所导致的偏见。在自然数据集和基于变异的数据集上进行双重评估，增强了结果的可靠性，尤其是在评估模型面对结构变化时的鲁棒性方面。

总而言之，自然数据集与合成数据集的结合、多种提示策略以及受控的评估配置，为所报告的研究发现提供了方法论严谨的基础。尽管在进一步平衡数据集和针对性模型微调方面仍有改进空间，但当前实验设计对大语言模型检测规则交互威胁的能力进行了有力且审慎的评估。

8 结论

本研究对基于大语言模型的交互威胁检测进行了全面评估，涵盖多种规则表示、威胁类别和提示配置。结果呈现出一个明确的规律：现代大语言模型虽然展现出对触发器–动作–条件规则有意义的语义理解能力，尤其擅长识别涉及明显动作与条件关系的威胁，但在需要更深层结构推理时，其可靠性会急剧下降。这一局限在 Mutation 数据集中表现得最为突出。即使是性能最强的模型也出现了显著的性能下滑，这凸显出当前大语言模型对表层形式高度敏感，缺乏实现可靠自动化安全分析所需的鲁棒性。在各项实验中，没有任何单一模型或提示策略能够保持稳定表现，而且某一威胁类别上的提升往往伴随着其他类别上的退步。相比之下，符号推理基线在两个数据集中始终保持稳定，不受结构改写影响，也更符合安全关键型物联网环境的需求。这些发现表明，以当前形态而言，大语言模型无法取代用于交互威胁检测的形式化分析或符号分析，尤其是在规则存在复杂依赖关系

或涉及非平凡结构变换时。然而，模型在语义模式识别方面的优势为未来发展指明了充满希望的方向。将符号推理用于提供结构保证，并结合大语言模型进行语义解释和歧义消解的混合方案，可为下一代分析工具奠定平衡的基础。这类架构能够利用大语言模型降低噪声、丰富上下文，同时依靠符号方法确保正确性、鲁棒性和可复现性。总体而言，本研究展示了将大语言模型应用于物联网交互威胁检测的机遇与当前局限，并为未来研究指明了清晰方向：构建有原则的混合系统，结合符号验证与大规模语言理解各自的优势，为智能家居生态系统提供可靠的安全分析。

声明

– 经费：不适用– 伦理审批：不适用– 知情同意：是– 作者贡献：实验和写作由第一作者和第二作者 Noura Khajehnouri 与 Jason Quantrill 完成，实验设计、支持、指导和编辑由第三作者 Manar Alalfi 博士负责– 数据可用性声明：与本研究相关的数据可在 `https://github.com/JasonQuantrill/llm-v-static-results` 获取– 利益冲突：不适用– 稿件中的临床试验编号：不适用

参考文献

1. L. Abuserrieh 和 M. H. Alalfi。物联网系统安全与安保验证综述。*IEEE Access*，12:138627–138645，2024年。 2. Mohannad Alhanahnah、Clay Stevens 和 Hamid Bagheri。物联网系统中交互威胁的可扩展分析。载于第*29*届 *ACM SIGSOFT* 软件测试与分析国际研讨会论文集，第272–285页，2020年。 3. Lorenzo Bettini。使用 *Xtext* 和 *Xtend* 实现领域特定语言——第二版。Packt Publishing，第2版，2016年。 4. Lei Bu、Wen Xiong、Chieh-Jan Mike Liang、Shi Han、Dongmei Zhang、Shan Lin 和 Xuandong Li。系统化确保实时家庭自动化物联网系统的可信度。*ACM* 网络物理系统汇刊，2(3):1–23，2018年。 5. Z Berkay Celik、Patrick McDaniel 和 Gang Tan。Soteria：自动化物联网安全与安保分析。载于*2018*年 *USENIX* 年度技术会议，第147–158页，2018年。

6. Z Berkay Celik、Gang Tan 和 Patrick D McDaniel。IoTGuard：在商用物联网中动态实施安全与安保策略。载于*NDSS*，2019年。 7. Chong Chen、Jianzhong Su、Jiachi Chen、Yanlin Wang、Tingting Bi、Jianxing Yu、Yanli Wang、Xingwei Lin、Ting Chen 和 Zibin Zheng。当 ChatGPT 遇上智能合约漏洞检测：我们究竟走了多远？，2024 年。 8. L. Chen、C. Wang、C. Chen、C. Huang、M. Chen、X. 和 Zhang。TapChecker：一种用于触发器-动作编程的轻量级基于SMT的冲突分析工具。载于*IEEE*物联网期刊，第*11*卷，第*12*期。IEEE，2024年。 9. Xuyang Chen、Xiaolu Zhang、Michael Elliot、Xiaoyin Wang 和 Feng Wang。修补漏水的水龙头：触发器-动作编程（TAP）的安全问题、检测技术与解决方案综述。计算机与安全，120:102812，2022年。 10. Isaac David、Liyi Zhou、Kaihua Qin、Dawn Song、Lorenzo Cavallaro 和 Arthur Gervais。你还需要手动审计智能合约吗？，2023年。 11. Wenbo Ding、Hongxin Hu 和 Long Cheng。IOTSAFE：通过真实物联网物理交互发现来实施安全与安保策略。载于第*28*届网络与分布式系统安全研讨会（*NDSS 2021*），2021年。 12. Xiaotao Feng、Xiaogang Zhu、Qing-Long Han、Wei Zhou、Sheng Wen 和 Yang Xiang。物联网设备固件漏洞检测综述。*IEEE/CAA*《自动化学报》，10(1):25–41，2023年。 13. Mohamed Amine Ferrag、Ammar Battah、Norbert Tihanyi、Ridhi Jain、Diana Maimut、Fatima Alwahedi、Thierry Lestable、Narinderjit Singh Thandi、Abdechakour Mechri、Merouane Debbah 和 Lucas C. Cordeiro。Securefalcon：使用大语言模型进行自动化软件漏洞检测，我们究竟走到哪一步了？，2024年。 14. Chenglong Fu、Qiang Zeng 和 Xiaojiang Du。 {HAWatcher}：{面向语义的} 应用化智能家居异常检测。载于第*30*届 *USENIX* 安全研讨会（*USENIX Security 21*），第4223–4240页，2021年。 15. Google。Gemini API模型，2025年。模型版本：Gemini 2.5 Pro（2025年6月）。 16. Jingxuan He 和 Martin Vechev。面向代码的大语言模型：安全强化与对抗性测试。载于*2023*年 *ACM SIGSAC*计算机与通信安全会议论文集，CCS '23，第1865–1879页。ACM，2023年11月。 17. Hong Jin Kang、Sheng Qin Sim 和 David Lo。IoTBox：通过沙箱挖掘防范物联网系统中的交互威胁。载于*2021*年第*14*届 *IEEE* 软件测试、验证与确认会议（*ICST*），第182–193页。IEEE，2021年。 18. McKenna McCall、Faysal Hossain Shezan、Abhishek Bichhawat、Camille Cobb、Limin Jia、Yuan Tian、Cooper Grace 和 Mitchell Yang。SAFE-TAP：一种高效的触发器-动作程序增量分析器。卡内基梅隆大学，2021年。

19. Vasudevan Nagendra、Arani Bhattacharya、Vinod Yegneswaran、Amir Rahmati 和 Samir R Das。VISCR：用于保护动态消费者物联网基础设施的直观、无冲突自动化。*arXiv* 预印本 *arXiv:1907.13288*，2019年。20. Dang Tu Nguyen、Chengyu Song、Zhiyun Qian、Srikanth V Krishnamurthy、Edward JM Colbert 和 Patrick McDaniel。IoTSan：强化物联网系统的安全性。载于第*14*届新兴网络实验与技术国际会议论文集，第191–203页，2018年。21. Ollama。Ollama 模型文件文档，2024年。访问日期：2025年11月5日。22. Marwan Omar 和 Stavros Shiaeles。Vuldetect：一种使用语言模型检测软件漏洞的新技术。载于*2023*年*IEEE*网络安全与韧性国际会议（*CSR*），第105–110页，2023年。23. OpenAI。模型 - OpenAI API（gpt-4o），2024年。快照：gpt-4o-2024-11-20。24. OpenHAB。赋能智能家居。https://www.openhab.org/，2021年。25. Hammond Pearce、Benjamin Tan、Baleegh Ahmad、Ramesh Karri 和 Brendan Dolan-Gavitt。探究使用大语言模型进行零样本漏洞修复，2022年。26. J Quantrill 和 H. M Alalfi。oHIT——用于识别物联网系统中 openHAB 交互威胁的框架。https://github.com/oHIT-Detection/oHIT。27. J Quantrill 和 H. M Alalfi。RIT-Mutator：系统化生成 openHAB 中的规则交互威胁。载于已提交，共29页，2025年。28. J Quantrill、N Khajehnouri 和 H. M Alalfi。oHIT——用于识别物联网系统中 openHAB 交互威胁的框架。载于计算安全、隐私与信任（*SEPT*），*IEEE*计算机、软件与应用会议（*COMPSAC 2025*）。，共10页，2025年。29. Gustavo Sandoval、Hammond Pearce、Teo Nys、Ramesh Karri、Siddharth Garg 和 Brendan Dolan-Gavitt。Lost at C：关于大语言模型代码助手安全影响的用户研究，2023年。30. Mohammad Setak 和 Pooria Madani。针对代码变异微调大语言模型：网络威胁的新时代，2024年。31. Lwin Khin Shar、Ta Nguyen Binh Duong、Lingxiao Jiang、David Lo、Wei Minn、Glenn Kiah Yong Yeo 和 Eugene Kim。SmartFuzz：用于测试 SmartThings 应用的自动化智能模糊测试方法。载于*2020*年第*27*届亚太软件工程会议（*APSEC*），第365–374页。IEEE，2020年。32. Norbert Tihanyi、Ridhi Jain、Yiannis Charalambous、Mohamed Amine Ferrag、Youcheng Sun 和 Lucas C. Cordeiro。软件安全新时代：借助大语言模型和形式化验证实现自愈软件，2024年。33. Ashwin Prasad Shivarpatna Venkatesh、Samkutty Sabu、Amir M. Mir、Sofia Reis 和 Eric Bodden。大语言模型在静态分析中的兴起：通过微基准测试进行初步考察，2024年。

34. Qi Wang、Pubali Datta、Wei Yang、Si Liu、Adam Bates 和 Carl A Gunter。绘制触发-动作物联网平台的攻击面。载于*2019*年*ACM SIGSAC*计算机与通信安全会议论文集，第1439–1453页，2019年。35. Moosa Yahyazadeh、Syed Rafiul Hussain、Endadul Hoque 和 Omar Chowdhury。Patriot：策略辅助的弹性可编程物联网系统。载于运行时验证：第*20*届国际会议（*RV 2020*），美国加利福尼亚州洛杉矶，*2020*年*10*月*6*日至*9*日，论文集，第151–171页，柏林、海德堡，2020年。Springer-Verlag。36. Lefan Zhang、Weijia He、Jesse Martinez、Noah Brackenbury、Shan Lu 和 Blase Ur。AutoTap：利用线性时序逻辑性质合成和修复触发器-动作程序。载于*2019*年*IEEE/ACM*第*41*届国际软件工程会议（*ICSE*），第281–291页。IEEE，2019年。37. Wei Zhou、Xiaogang Zhu、Qing-Long Han、Lin Li、Xiao Chen、Sheng Wen 和 Yang Xiang。使用大语言模型的安全性：侧重于ChatGPT的综述。*IEEE/CAA*《自动化学报》，12(1)：1–26，2025年。38. Xin Zhou、Duc-Manh Tran、Thanh Le-Cong、Ting Zhang、Ivana Clairine Irsan、Joshua Sumarlin、Bach Le 和 David Lo。静态应用程序安全测试工具与大语言模型在代码仓库级漏洞检测方面的比较，2024年。39. Xiaogang Zhu、Wei Zhou、Qing-Long Han、Wanlun Ma、Sheng Wen 和 Yang Xiang。当软件安全遇上大语言模型：综述。*IEEE/CAA*《自动化学报》，12(2)：317–334，2025年。