面向表格事实验证的大语言模型指令优化分析

Xiaotang Du[1] **Giwon Hong**[1] **Wai-Chung Kwan**[1] **Rohit Saxena**[1] **Ivan Titov**[1]
Pasquale Minervini[1,2] **Emily Allaway**[1]1爱丁堡大学，英国 2Miniml.AI，英国
{xiaotang.du, ghong,p.minervini, emily.allaway}@ed.ac.uk

arXiv:2602.17937v1 [cs.CL] 20 Feb 2026

摘要

指令优化提供了一种轻量级、模型无关的方法，可提升大语言模型（LLM）的推理性能。本文首次基于 DSPy 优化框架，对表格事实验证中的指令优化进行了系统比较。我们评估了四种开箱即用的提示技术，涵盖纯文本提示和代码使用：直接预测、思维链（CoT）、结合 SQL 工具的 ReAct，以及使用 Python 执行的 CodeAct。我们在四个基准测试和三个模型系列上，研究了 DSPy 框架中的三种优化器——COPRO、MiPROv2 和 SIMBA。我们发现，指令优化能够持续提升核查准确率；MiPROv2 为 CoT 带来了最稳定的提升，而 SIMBA 对 ReAct 智能体的益处最大，尤其是在模型规模较大时。行为分析表明，SIMBA 通过应用启发式规则，促使推理路径更加直接，从而提升了 CoT 推理中的数值比较能力，并帮助 ReAct 智能体避免不必要的工具调用。在不同提示技术中，CoT 对表格事实核查仍然有效，尤其适用于较小模型。尽管使用较大模型构建的 ReAct 智能体能够取得有竞争力的性能，但仍需要谨慎进行指令优化。

1 引言

根据结构化数据核查自然语言主张，是部署于科学、公共卫生和信息质量保障领域的可信自然语言处理系统的一项核心能力。尽管已有大量针对表格事实验证的方法（Yang 和 Zhu，2021年；Ou 和 Liu，2022年；Lu 等人，2025年；Zhang 等人，2024b，等等），但由此产生的系统往往针对特定数据集进行专门设计，或无法胜过更简单的提示方法。

在本研究中，我们针对表格事实验证，对开箱即用的提示技术与指令优化相结合的方法进行了比较研究。指令优化是一种无需梯度更新即可提升大语言模型性能的技术。鉴于大语言模型已知对提示构造十分敏感（Webson 和 Pavlick，2022 年；Leidinger 等人，2023 年），我们分析了指令优化对实用且可泛化的提示技术（例如思维链（Wei 等人，2022年））的影响。

近期的指令优化框架（例如 DSPy；Khattab 等人，2024年）将多步大语言模型流水线视为程序，可通过搜索或元推理自动调整其文本参数，从而在各种任务上取得显著提升。尽管已有这些进展，我们仍缺乏对指令优化如何影响表格事实验证的系统性认识。以下方面尤其缺乏研究：（1）中间计算方式各异的提示技术（例如直接预测、思维链，以及通过 SQL 和 Python 实现的程序辅助推理）；（2）优化器类别；（3）模型规模和模型系列。工具增强型智能体（例如 ReAct；Yao 等人，2023年）通过交替进行思考与可执行操作，有望增强事实依据，但其端到端有效性在很大程度上取决于学习得到的工具接口和执行可靠性——而指令优化可能会改善或损害这些因素。

我们首次使用 DSPy 优化框架，对表格事实验证中的指令优化开展比较研究。本研究聚焦于 DSPy 中的三种优化器：COPRO 、 MiPROv2 和 SIMBA[1]。我们从四个基准（TabFact、Pub-HealthTab、SciTab 和 MMSci）、四种提示技术（直接预测、思维链、ReAct 和

[1]我们将 MiPROv2 和 SIMBA 限定为仅进行指令调优，以便将指令的影响与少样本示例选择区分开来。

CodeAct）以及三种基础大语言模型（Qwen3、Gemma3、GPT-4o）出发进行分析。我们结合任务指标（准确率、宏平均 F1 值）和行为变化（例如 SQL/Python 调用的频率与成功率、错误分类体系）开展全面分析，以回答以下研究问题：

• 优化后的指令对思维链推理有何影响?

• 优化后的指令如何影响 ReAct 智能体的工具调用行为?

• 在表格事实核查中，程序辅助推理是否始终优于思维链?

我们的实验结果表明，指令优化的影响因提示技术和模型规模而异。与较小模型相比，较大模型从优化后的指令中获益更多，并且在四个测试集上都有更大提升。在不同优化器中，MiPROv2 能为思维链推理带来稳定的增益，而 SIMBA 则能有效改善 ReAct 的工具使用行为。

仔细检查优化后的指令可以发现，MiPROv2 和 SIMBA 优化器会鼓励模型选择直接的推理路径，避免为简单的主张进行不必要的工具调用。使用 SIMBA 优化器时，工具使用频率会降低，工具执行的成功率则会提高。此外，我们发现，SIMBA 优化器在优化后的指令中加入了更多关于数值比较的启发式规则，这有助于提高思维链的整体准确率。我们的代码可在此处获取：
h
t
t
p
s://
github.com/xiaodu0123/tabfact-prompt-optimization。

2 相关工作

基于表格的事实核查 依据结构化证据验证声明，需要对多样的表格模式进行组合推理。TabFact（Chen 等人，2020年）建立了首个面向维基百科表格进行二元事实验证的大规模基准。后续数据集纳入了更细致的标注模式（例如，使用三个标签而非仅两个），以及需要多跳推理的更复杂声明（Wang 等人，2021年）。其中已创建若干领域特定的数据集：PubHealthTab（Akhtar 等人，2022年）针对公共卫生相关声明，SciTab（Lu 等人，2023年）包含来自计算机科学出版物的声明，以及 SciAtomicBench（Zhang 等人，2025年），其涵盖计算机科学以及金融等其他领域。尽管事实核查数据集通常以文本形式呈现表格数据，但也已创建多模态数据集（Yang 等人，2025b）。此外，一些事实核查数据集将表格证据与文本（Aly 等人，2021年；Schlichtkrull 等人，2023年；Zhao 等人，2024年）及图表（Wang 等人，2025年；Chan 等人，2024年）相结合。

早期的表格事实验证方法采用符号推理或程序化推理（Chen 等，2020；Zhong 等，2020；Shi 等，2020；Zhang 等，2020；Yang 等，2020；Yang 和 Zhu，2021；Ou 和 Liu，2022）。尽管近期也有一些研究采用神经符号系统（Glenn 等，2024；Aly 和 Vlachos，2024；Cheng 等，2023），但研究重点正日益转向如何调整并利用大语言模型。为此，先前研究针对基于表格的事实验证提出了预训练（Eisenschlos 等，2020；Dong 和 Smith，2021；Zhang 等，2024a）和微调（Wu 和 Feng，2024；Jiang 等，2025）方法，也探索了更通用的基于表格的推理任务（Herzig 等，2020；Liu 等，2022）。此外，还有多项研究提出提示技术，以提升模型在表格上的推理能力（Wang 等，2024b；Zhang 等，2025；Abhyankar 等，2025；Zhang 等，2024b）。近期，研究也开始关注智能体系统及其在基于表格的事实验证中的工具使用（Lu 等，2025；Zhou 等，2025）。然而，尽管取得了这些进展，许多系统仍然计算开销大，或针对特定数据集进行了专门设计。相比之下，我们探索将计算开销较低的指令优化技术应用于通用提示策略。

与我们的工作最为密切相关的是两项近期分析，它们探讨了各类表格理解任务（包括事实核查）所面临的挑战。Bhandari等人（2025年）考察了指令微调、上下文示例和模型规模如何影响表格推理任务的表现，而Wu等人（2025年）则更广泛地综述了表格理解任务的方法。与这些分析不同，我们的工作比较了应用于简单提示策略的指令优化技术（包括思维链等标准基线，以及ReAct等简单程序化推理模型）。此外，尽管Bhandari等人（2025年）涵盖了多种表格理解任务，我们的工作仅聚焦于基于表格的事实验证，转而覆盖更广泛的表格事实验证数据集。

3 方法

3.1 提示技术

思维链思维链推理（思维链）（Wei等人，2022年）鼓励大语言模型在给出最终答案之前生成中间推理步骤。借助思维链，大语言模型可以将复杂查询分解为子问题，并在推理轨迹中逐步构建解决方案。

ReAct ReAct（Yao 等人，2023年）是基于工具的智能体的一种基础框架，通过交替进行推理与特定任务的行动来运作。ReAct 使大语言模型能够与外部工具交互，从而收集更多证据，并依据工具执行输出进行推理。在我们的实验中，我们评估了一个可使用标准 SQL 工具的 ReAct智能体，该工具能够对表格数据执行 SQL 查询，以检索相关信息并执行数学运算。

CodeAct CodeAct（Wang 等人，2024a）将可执行的 Python 代码作为基于工具的智能体的主要行动模式。与依赖文本或 JSON格式工具调用的现有范式不同，CodeAct 通过代码执行实现多步操作和灵活的工具链调用，使智能体能够结合 Python控制流和现有库执行复杂操作。在我们的实验中，CodeAgent 无法使用预定义工具，而是生成自由形式的 Python代码，逐步处理表格数据并执行数学运算。

3.2 指令优化

在我们的分析中，我们聚焦于 DSPy（Khattab 等人，2024年）框架中的三种基于大语言模型的指令优化方法：COPRO、MiPROv2（Opsahl-Ong 等人，2024年）和 SIMBA。

DSPy 框架 DSPy 是一个用于通过算法优化模型提示和权重的框架，它将大语言模型流水线视为可自动编译和优化的程序。

COPRO 协作式提示优化（CO-PRO）以类似束搜索的方式系统地探索各种候选指令，并评估它们在训练集上的性能。优化器根据此前尝试中排名前 N 的提示及其相应的评估分数，迭代优化提示指令，并提出多个新的候选指令。

MiPROv2 多阶段指令提示优化（MiPROv2）是一种先进框架，可通过三阶段流水线同时优化指令和少样本示例。首先，优化器从训练数据中引导生成多组候选少样本示例。然后，它根据此前评估过的候选项、下游任务的特性以及随机采样的提示策略，生成多样化的提示指令和示例。最后，MiPROv2 使用贝叶斯优化方法，高效地搜索候选指令与示例的最佳组合。

与 COPRO 相比，MiPROv2 为生成新的候选指令提供了更丰富的上下文，并能更高效地评估小批次训练数据。

SIMBA 随机内省式小批次上升（SIMBA）是一种内省式提示优化算法，它利用语言模型的自我反思能力，迭代改进指令质量。优化器会识别模型输出差异较大的棘手训练实例，然后采用两种互补策略来优化提示。一种策略是进行对比分析，让模型比较成功与失败的执行轨迹，从而生成明确的改进规则，并将其附加到原始指令中。另一种策略则将成功的执行轨迹作为少样本示例。

4 实验

4.1 数据集

我们在四个表格事实核查数据集上评估了各种大语言模型的性能：TabFact（Chen 等人，2020年）、PubHealthTab（Akhtar 等人，2022年）、SciTab（Lu 等人，2023年）和 MMSci（Yang 等人，2025b）。这些数据集涵盖了多种领域和表格类型，从通用知识到专业数据，使我们能够更全面地评估不同方法的泛化能力。在 SciTab、PubHealthTab 和 MMSci 中，标签分为三类： 支持、 反驳 和信息不足；TabFact 是一个二元分类任务，仅包含 支持和反驳标签。

SciTab SciTab(Lu 等人，2023年)是一个为科学主张验证而设计的基准，利用来自科学出版物的真实世界表格证据，这些出版物涉及机器学习和自然语言

数据集	训练集	Dev	Test
TabFact	92,585	12,851	8,609
SciTab	210	429	429
PubHealthTab	1440	177	180
MMSci	-	-	1,038

表1：事实核查数据集的统计数据。

处理领域。该数据集在主张歧义、组合推理以及科学数据的数值分析方面提出了独特的挑战。

PubHealthTab PubHealthTab（Akhtar 等人，2022年）是一个基于表格的事实核查数据集，聚焦于公共卫生主张。证据表格提取自多个网络来源，呈现出嘈杂且复杂的表格结构，内容质量参差不齐。

TabFact TabFact（Chen 等人，2020年）是一个大规模基于表格的事实验证数据集，由以维基百科表格作为证据的人工标注的主张组成。TabFact 提供了两个在主张复杂度上有所不同的测试集，我们使用复杂测试集进行评估。

MMSci MMSci（Yang 等人，2025b）是一个面向三项基于表格的任务的多模态科学推理基准。我们使用表格事实验证测试集，将表格图像转换为文本格式，以评估对未见数据的泛化能力。

4.2 优化

对于每个所考虑的大语言模型，我们评估不同提示技术的性能，包括直接提示、思维链、ReAct 和 CodeAct，以研究指令优化对基于语言的推理和程序辅助推理的影响。在优化之前，我们在不同实验的系统提示中使用相同的指令，即根据所提供的表格数据验证给定主张。所有实验均在零样本设置下进行。

4.3 评估设置

模型与基线 我们使用 Qwen3（Yang 等人，2025a）、Gemma3（Team 等人，2025年）和 GPT-4o 模型开展实验，从而能够系统地研究指令优化对不同模型系列和规模的推理及工具调用行为的影响。我们使用同一个模型来提出候选指令并评估指令优化过程中的质量。为考察优化后的指令的有效性，我们将 GPT-4o 实验中的模型性能与 ReActable（Zhang 等人，2024b）进行比较。ReActable 是一种 ReAct 框架，使用 GPT-4o 搭配人工编写的指令，并将 SQL 和 Python 作为工具。

数据处理 每个事实核查数据集都被处理为统一的数据格式。随后，我们将其中三个数据集（TabFact、SciTab 和 PubHealthTab）划分为训练集、开发集和测试集；我们的第四个数据集，*MMSci* 仅用于评估。我们通过从这三个数据集的训练划分中随机抽取 100 个实例，创建用于指令优化的混合训练集。我们抽取 40 个 PubHealthTab 实例、40 个 SciTab 实例和 20 个 TabFact 实例，以确保混合数据集的标签分布均衡。处理后数据集的统计数据见表1。

评估指标 我们使用混合训练数据优化指令，并在全部四个数据集的开发集和测试集上以准确率和宏平均 F1 值评估性能。在指令优化过程中，仅使用准确率来衡量不同候选提示的质量。

5 结果

我们在表 2 中报告了使用 Qwen3 模型的不同提示技术在四个事实核查数据集上的测试性能。对于直接提示和思维链，较大模型在大多数测试集上通常比对应的较小模型取得更高的准确率和 F1 值。对于程序辅助推理范式（ReAct、CodeAct），增大模型规模并未带来显著的性能提升。尽管较大模型与较小版本具有相近的基线性能，但它们从指令优化中获益明显更多，并在使用优化后的指令时表现出更大的改进。

表格推理中指令优化的有效性高度取决于模型规模和提示技术。对于思维链推理的优化，MiPROv2带来的提升最为稳定：在Qwen3-8B上，它在PubHealthTab、TabFact和MM-Sci数据集上均取得最高准确率和F1值；在Qwen3-32B上，它在三个数据集上的表现也颇具竞争力。对于基于程序的推理，SIMBA在SciTab上的性能提升最大，尤其能改善Qwen3-32B模型的ReAct表现。COPRO也为Qwen3-

模块	优化器	Qwen3-8B								Qwen3-32B							
		PubHealth		SciTab		TabFact		MMSci		PubHealth		SciTab		TabFact		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
直接方法	基线	**73.3**	**73.4**	**58.7**	56.5	58.0	52.8	57.2	**41.5**	84.4	82.3	52.9	49.6	64.1	62.8	68.2	46.3
	+COPRO	**73.3**	**73.4**	**58.7**	**56.6**	58.1	52.8	57.0	41.2	84.4	82.7	**53.4**	**51.1**	65.3	63.9	69.5	47.7
	+MiPROv2	72.8	72.9	56.4	54.3	**58.5**	**53.4**	56.6	41.3	84.4	82.3	53.1	50.0	64.1	62.8	68.2	46.3
	+SIMBA	**73.3**	**73.4**	**58.7**	56.5	58.1	52.8	**57.3**	41.4	**85.6**	**84.3**	52.7	50.0	**67.6**	**68.7**	**70.6**	**49.4**
CoT	基线	83.9	82.3	64.3	64.4	77.6	80.5	81.9	58.0	88.3	87.6	66.4	66.4	84.5	86.6	86.5	61.6
	+COPRO	83.9	82.8	**66.2**	**66.2**	76.8	79.9	79.4	56.3	87.2	86.1	67.4	67.3	85.5	87.6	86.7	61.5
	+MiPROv2	**86.1**	**85.7**	66.0	66.0	**80.3**	**83.1**	**82.5**	**59.4**	87.2	86.5	**68.8**	**68.6**	**86.9**	**88.5**	**87.7**	**65.4**
	+SIMBA	82.2	81.4	62.5	62.2	77.6	80.6	81.6	58.7	**90.0**	**89.6**	**68.8**	**68.6**	85.2	87.1	87.0	64.2
ReAct	基线	**86.7**	**86.6**	61.3	61.2	**83.8**	**85.3**	82.5	58.5	87.8	87.4	61.5	60.1	**86.4**	**87.0**	**87.5**	62.6
	+COPRO	84.4	83.9	**62.2**	**62.1**	80.5	81.5	**83.6**	**61.5**	86.1	84.7	62.0	61.5	81.5	84.1	85.0	61.0
	+MiPROv2	81.7	81.1	61.8	61.8	75.5	80.6	82.2	60.0	87.8	87.2	61.5	60.9	84.2	85.2	86.2	63.0
	+SIMBA	86.1	85.2	58.3	58.3	82.9	84.7	80.8	57.3	**90.6**	**90.0**	**66.2**	**65.9**	86.1	**87.0**	85.9	**65.0**
CodeAct	基线	**86.1**	**86.0**	57.1	57.1	82.0	83.5	81.2	59.2	85.6	84.9	58.0	57.5	85.9	87.1	87.5	**66.1**
	+COPRO	82.8	82.2	**59.7**	59.1	80.0	82.2	83.3	**60.7**	**87.2**	**86.6**	62.2	61.8	**86.7**	**87.9**	**88.1**	63.3
	+MiPROv2	**86.1**	85.7	56.9	56.6	80.5	82.0	82.1	59.0	83.9	83.5	59.0	58.2	86.4	87.6	86.5	62.7
	+SIMBA	85.0	84.9	**59.7**	**59.5**	**84.8**	**85.5**	**84.3**	59.8	85.6	85.2	**69.2**	**69.3**	85.4	87.0	86.5	62.6

表 2：Qwen3-8B 和 Qwen3-32B 在测试集上的结果。**粗体** 表示每种方法和数据集中的最佳性能。

32B模型带来了适度收益，但稳定性不如 SIMBA。这表明，较大模型更擅长通过自我反思和比较分析识别成功轨迹中的模式，从而形成更有效的规则，在各种场景中优化工具使用。

根据表3，Gemma3 模型系列呈现出的总体趋势与 Qwen3 略有不同。较大的 Gemma3 模型在思维链推理和程序辅助推理方面均持续表现出更高的性能。与 Qwen3 不同，后者在较小模型上使用优化器无法提升 ReAct 的性能，而 Gemma3 模型对不同提示技术下的指令优化响应更加积极，并且在两种规模下使用改进后的指令都能获得更大的改进。

与 Qwen3 实验类似，MiPROv2 在优化思维链时仍能带来显著改进。SIMBA 在提升 ReAct 和 CodeAct 方面表现出色，尤其是对于较大的 27B 模型。COPRO 对较小的模型（12B）仍然有效，但相较于 MiPROv2 和 SIMBA，其增量收益较小。总体而言，即使应用了指令优化，Gemma3 模型系列的性能仍低于 Qwen3。对于 Gemma3 和 Qwen3 模型，在表格事实核查任务中，与程序辅助推理方法相比，经过指令优化后，思维链推理始终能够取得具有竞争力的性能。

表格4 总结了以下模型在测试集上的性能：GPT-4o 模型。考虑到预算因素，GPT-4o 模型和 ReActable 在较小的 TabFact 测试集（TabFact-mini）上进行评估，该测试集包含 400 个随机实例。GPT-4o 模型的基线性能明显更强，因此从指令优化中获益不如 Qwen3 和 Gemma3 模型。对于 GPT-4o-mini，MiPROv2 在提升思维链推理方面更有效，而 SIMBA 在优化 ReAct 时能在各测试集上带来更大的提升。然而，没有任何单一优化器能够在优化 CodeAct 时持续提升性能。对于 GPT-4o 模型，SIMBA 表现稳定出色，既能提升思维链，也能提升 ReAct；而 MiPROv2 则被证明有助于提升 CodeAct 性能。与 ReActable 基线相比，GPT-4o 的 ReAct 在 SciTab 和 TabFact-mini 上表现略差，但经 SIMBA 优化后，它在所有测试集上的表现都能持续优于 ReActable，这证明了基于 DSPy 的指令优化优于人工设计的提示。

根据 MMSci 上的测试性能，我们观察到，对于 Qwen3-32B 和 Gemma3-27B 模型，在 PubHealthTab、SciTab 和 TabFact 上表现更优的优化后的指令通常能够很好地泛化到 MMSci。具体而言，由 SIMBA 优化的指令在直接提示和 ReAct 设置下均持续在 MMSci 上取得最高的 F1 分数，而由 MiPROv2 学习得到的思维链指令则继续带来

在 MMSci 上最强的改进。然而，在 GPT-4o 模型中并未观察到这一趋势：其在另外三个事实核查数据集上的性能并不能预测其在 MMSci 上的测试性能。尽管 SIMBA 在直接提示、思维链和 ReAct 设置下均在 SciTab 和 TabFact-mini 上表现出强劲性能，但这些性能提升并未一致地迁移到 MMSci 测试数据上。这可能表明，GPT-4o 在 SIMBA 优化过程中提出的指令在未见数据上的泛化效果较差。

为进一步考察指令优化的有效性和泛化能力，我们使用不同的随机种子、质量各异的初始指令以及不同的训练集开展消融实验。我们还通过考虑 TART 框架中引入的一个或多个最常用工具（Lu 等人，2025年）来扩展对 ReAct 智能体的评估。有关实验的更多细节请参见第C节。

5.1 优化后的指令对表格推理的影响

表格 2 和 3提供了证据，表明提示优化能够在直接方法和思维链设置下提升表格推理性能。为对此进行更详细的分析，表格 5和 6 给出了混淆矩阵，用于比较 SIMBA 优化与基线之间的差异。

从表格 5和6 中观察到的最显著模式是，优化提高了反驳预测所占的比例，表明总体而言，这会带来更加保守的预测。为进一步解释这一影响，我们分析了直接模块和思维链模块的优化后的指令。

对于直接提示，我们观察到优化后的指令往往会针对序数信息进行调整。对于使用 SIMBA 优化的 Qwen3-32B，得到的指令（表 7）包含一条额外提示，鼓励模型关注主张中元素的顺序（例如，“如果表格中的顺序与主张矛盾，模块应返回‘反驳’。”）。为了验证这一效果，我们定义了一组序数词（见第A.1节），并分析了优化后的指令在主张是否包含序数词时的有效性（表8）。确实，当真实标签为“反驳”时，与不含序数词的主张相比，优化后的指令使含有序数词的主张取得了显著的 4.5% 性能提升（从 66.1% 提高到 70.6%）。

对于思维链，由表 7中 Qwen3-32B 模型优化的指令 SIMBA 特别擅长数值比较（例如，“该模块应重点比较表格中的数值，以判断主张是得到支持还是遭到反驳”）。按照上述类似流程，我们定义了一组比较术语（第A.2节），并分析了优化后的指令是否会在推理过程中引导出更明确的比较行为，以及这对性能有何影响（表 9）。同样，当真实标签为“反驳”时，优化后的指令使性能提升了 2.4%（从 83.7% 提高到 86.1%）。

表格 8 和 9 验证了优化过程会促使模型关注表格推理所需的特定方面（直接预测和思维链分别关注元素顺序和数值比较），并且这些行为能有效转化为性能提升。

5.2 工具使用行为分析

我们研究了指令优化前后，ReAct 在 SciTab 测试数据上使用 Qwen3 模型时的工具使用行为。对于 32B模型，COPRO 将工具调用频率从 90% 提高到 93%，但错误率也随之上升（成功执行率从 42% 降至 12%）。MiPROv2 和 SIMBA 则都引导模型采用更直接的推理路径：工具使用频率分别从 90% 降至 70.6% 和 35.4%，同时轨迹中的平均推理长度分别增加了 16% 和 33%。这表明，MiPROv2 和 SIMBA 鼓励模型直接解读表格，而不是过度依赖基于工具的验证，并且只在必要时调用工具。优化后，MiPROv2 导致更多工具执行错误，而 SIMBA 的错误率与基线相比基本持平。

对于较小的8B模型，COPRO和MiPROv2将工具使用频率从67%大幅降低至约40%；而SIMBA的工具使用频率与基线相近，同时成功执行率略有提升（从40.8%提升至41.2%）。

5.3 ReAct和CodeAc的错误分析 t

为了评估 ReAct 和 CodeAct 结果的有效性，我们从 Qwen3-32B 在 SciTab测试集上的每种设置中抽取 20 个错误案例，并进行人工错误分析。我们定义了一个由三类构成的高层次错误分类体系：与数据相关、与代码相关以及推理错误，并分-

模块	优化器	Gemma3-12B								Gemma3-27B							
		PubHealth		SciTab		TabFact		MMSci		PubHealth		SciTab		TabFact		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
直接方法	基线	77.8	72.8	48.3	43.4	57.6	54.6	64.7	38.9	82.8	80.4	53.6	50.7	58.6	60.3	66.7	45.0
	+COPRO	80.6	77.2	49.9	46.4	58.8	58.9	65.9	**45.3**	82.8	80.2	51.5	48.2	54.4	59.2	65.7	44.9
	+MiPROv2	80.6	79.4	**55.0**	**54.7**	**63.2**	**64.1**	**66.9**	45.0	82.8	80.3	55.9	54.6	59.2	61.6	**67.4**	46.0
	+SIMBA	**81.7**	**79.5**	54.3	52.8	59.2	60.1	64.5	45.0	**85.6**	**83.7**	**60.6**	**60.6**	**62.9**	**62.9**	67.3	**47.4**
CoT	基线	87.8	86.4	54.3	52.3	75.5	77.7	79.3	54.6	87.8	86.9	62.2	61.9	78.3	80.8	82.9	58.9
	+COPRO	87.8	86.5	57.3	56.4	74.5	76.6	79.8	54.8	**89.4**	**88.7**	61.5	61.3	78.4	81.6	84.6	59.8
	+MiPROv2	87.2	85.6	58.3	57.8	**80.1**	**82.2**	**84.7**	**60.5**	88.9	87.8	**64.8**	**64.4**	**81.4**	**83.4**	**85.8**	**62.5**
	+SIMBA	**89.4**	**88.8**	**60.1**	**59.6**	77.6	79.3	83.2	57.7	88.9	87.6	63.6	63.8	75.8	79.1	81.9	59.0
ReAct	基线	83.9	82.9	49.2	48.7	64.9	72.8	79.9	57.5	87.8	86.8	52.9	52.9	76.3	80.8	82.9	58.7
	+COPRO	**87.2**	**86.5**	**58.3**	**57.1**	77.1	79.4	84.7	**61.0**	85.0	83.6	48.0	47.9	72.9	78.4	69.3	52.5
	+MiPROv2	84.4	83.5	49.0	48.7	64.6	72.5	79.9	57.4	89.4	88.8	**63.6**	**63.2**	82.9	84.4	**86.5**	62.5
	+SIMBA	86.7	85.7	53.4	51.0	**79.8**	**81.1**	**84.8**	59.2	**90.0**	**89.3**	60.4	58.9	**84.0**	**85.0**	85.8	**62.6**
CodeAct	基线	86.7	86.0	51.5	49.8	64.7	72.2	83.2	57.7	87.2	86.2	55.9	56.1	73.6	78.7	85.8	61.3
	+COPRO	**89.4**	**88.9**	54.3	53.4	67.0	74.4	**85.0**	61.1	88.9	87.9	**59.2**	**59.5**	79.0	81.5	84.4	61.1
	+MiPROv2	88.3	87.6	49.9	48.6	**78.9**	**81.6**	84.7	59.0	85.6	84.8	55.5	56.0	81.3	83.6	86.5	62.8
	+SIMBA	85.0	84.0	**55.2**	**54.8**	77.5	79.9	83.4	**61.7**	**89.4**	**88.5**	58.3	56.7	**83.1**	**84.4**	**87.6**	**65.6**

表3：Gemma3-12B 和 Gemma3-27B 在测试集上的结果。**粗体** 表示每种方法和数据集上的最佳性能。

模块	优化器	GPT-4o-mini								GPT-4o							
		PubHealth		SciTab		TabFact-mini		MMSci		PubHealth		SciTab		TabFact-mini		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
ReActable		73.9	73.4	52.7	52.4	85.5	85.5	64.3	48.2	83.3	82.6	67.6	67.6	89.8	89.7	85.4	62.8
直接方法	基线	85.6	85.3	58.3	58.4	65.0	**66.7**	70.0	51.2	**90.6**	**89.8**	65.0	65.0	73.2	74.8	82.1	60.1
	+COPRO	**86.7**	**87.1**	**61.1**	**61.0**	**65.2**	65.7	70.8	51.8	89.4	88.9	64.8	64.7	76.0	77.0	**84.7**	61.2
	+MiPROv2	85.0	85.2	60.1	59.8	63.5	63.9	**71.6**	**52.9**	90.0	89.1	65.0	65.1	74.5	76.0	84.4	**61.7**
	+SIMBA	86.1	85.6	57.1	56.6	60.5	64.0	69.7	51.1	89.4	88.5	**65.3**	**65.2**	**76.5**	**77.3**	82.8	59.8
CoT	基线	**90.6**	**90.1**	62.9	63.0	79.8	82.4	83.0	58.9	87.8	87.2	69.2	69.1	87.8	89.6	87.7	63.7
	+COPRO	90.0	89.6	61.8	61.7	81.0	82.7	83.6	61.3	87.8	87.5	69.7	69.6	88.0	89.8	**88.4**	65.0
	+MiPROv2	89.4	88.9	**64.8**	**64.8**	**81.2**	**83.0**	84.4	60.9	89.4	88.9	**70.6**	**70.5**	88.5	89.9	88.3	**65.8**
	+SIMBA	90.0	89.5	64.3	64.3	78.8	81.1	**84.5**	**62.2**	**90.0**	**89.8**	**70.6**	**70.5**	**90.2**	**91.4**	87.9	64.3
ReAct	基线	87.8	87.3	55.0	53.1	84.8	85.4	84.4	61.2	88.3	87.3	64.1	62.8	90.0	90.3	**89.5**	66.2
	+COPRO	89.4	88.9	59.4	58.4	82.8	83.7	**85.7**	61.8	**89.4**	**88.9**	67.8	67.3	90.2	91.0	88.7	67.0
	+MiPROv2	90.0	89.6	**60.1**	**60.0**	82.5	83.2	84.5	60.3	**89.4**	88.4	66.2	65.6	90.8	91.4	88.5	**67.6**
	+SIMBA	**91.7**	**91.1**	**60.1**	59.9	**84.8**	**86.1**	84.0	**62.2**	88.3	87.6	**68.3**	**68.3**	**91.0**	**92.3**	88.1	64.0
CodeAct	基线	**84.4**	**83.7**	**59.0**	**58.8**	82.5	83.9	84.5	60.4	87.2	86.7	63.4	62.3	90.2	90.8	89.3	**65.4**
	+COPRO	**84.4**	82.8	53.4	52.2	83.5	84.7	**85.4**	**61.3**	89.4	89.0	62.9	60.7	90.5	91.4	**89.7**	64.9
	+MiPROv2	80.6	77.9	52.2	48.9	**85.2**	**86.6**	82.9	57.8	**91.1**	**90.6**	**65.0**	**63.9**	**91.2**	**91.7**	89.2	62.6
	+SIMBA	**84.4**	83.0	55.7	54.9	81.5	83.0	84.1	58.6	88.3	87.6	61.1	60.5	90.0	91.4	89.0	**65.4**

表格 表4：GPT-4o-mini 与 GPT-4o 在测试集上的结果。**粗体**表示各方法与数据的最佳性能 et.

析每个错误实例属于哪一类或哪些类别，并指出同一实例可能属于多个类别。

ReAct 中错误的主要来源是推理问题（45%）和数据相关错误（35 %）。面对复杂主张，ReAct 往往只关注其中部分陈述，在工具规划或推理步骤中忽略其他相关数据。此外，存在执行错误的工具执行轨迹可能导致不合逻辑的推理，使其与所提供的表格数据不一致。数据相关问题包括定义不充分的主张和含噪表格结构，这使 ReAct 难以构建正确的 SQL 查询。

对于 CodeAct，表格中的错误分析11 这表明，错误中占比最高的部分（70%）源于 SciTab 数据集本身的问题，包括主观或定义不充分的主张（例如，未定义的“显著性”概念）、需要表格中未提供的信息才能判断的主张，或错误的标准答案标签。代码相关错误占案例总数的 30% 以上；在这 6 个此类错误中，有 4 个主要由语法问题导致代码执行失败，这表明更稳健的编程智能体可以显著减少此类错误。推理错误约占总数的 15%，且与主张相关，通常涉及对主张的误解，或未能意识到表格缺少所需的信息，而此时本应预测“信息

直接方法		Gold	
(Base→SIMBA)		支持	反驳
Predict	支持	3,870→2,941	2,430→1,270
	反驳	311→1,242	1,648→2,880
	Not 信息充足	125→123	225→153

表5：使用 Qwen3-32B 模型在 TabFact 上进行直接预测时，提示优化前后的混淆矩阵。 → 和 → 分别表示正向变化和负向变化。

CoT		Gold	
（基线→SIMBA）		支持	反驳
Predict	支持	3,643→3,648	429→403
	反驳	483→499	3,628→3,691
	Not 信息充足	180→159	246→209

表 6：Qwen3-32B模型在Tab-Fact上使用思维链时，提示优化前后的混淆矩阵。 → 和 → 分别表示正向和负向变化。

不足”标签。值得注意的是，即使代码执行失败，模型有时也会在推理阶段进行补偿，通过检查生成的代码或表格内容来近似完成预期的计算。

5.4 思维链与ReAct的比较

为了解思维链推理与工具增强型智能体之间的性能差异，我们从 SciTab测试集 中选取了 50 个实例进行详细分析，并评估了使用 MiPROv2 优化指令后 Qwen3-8B 和 Qwen3- 32B 的预测结果。如表 10 所示，在这一子集中，ReAct 在 Qwen3-32B（78 % 对 70%）和 Qwen3-8B（60% 对 54%）上的表现均持续优于思维链。仔细检查这 50 条主张后，我们发现了相互补充的失败模式。

ReAct 的主要优势在于其迭代验证过程，这对于验证复杂或包含多个分句的主张十分有效，尤其是那些涉及多项定量检查的主张，例如“G2S 方法优于 S2S 基线”。即使某些工具调用因执行错误而失败，智能体仍可重新检查表格证据，并在下一步改用基于语言推理的策略。这与 CoT 的推理过程形成鲜明对比，后者往往无法在得出最终判定前汇总所有子声明。此外，CoT 经常误解指标方向性，例如将较低的错误率（如 ADDED/MISS 值）视为性能较差。对于 Qwen3-32B，ReAct 独有的六个成功案例中，有四个涉及 CoT 因误解指标方向性而错误地将主张判定为反驳的情况。其余案例则是 CoT 在面对多分句的改进主张时（例如 “F1 值比基线高出 10.5”）放弃作出判定。8B模型也呈现出类似模式，但更倾向于放弃作出判定。

然而，ReAct 依赖基于工具的验证，这可能会影响它在简单直接的主张上的表现；对于这类主张，只需直接解读表格即可做出判断，例如“BI 和 IS 分别优于预言机”。对于 Qwen3-8B模型，思维链成功而 ReAct 失败的所有实例，都是由工具执行错误造成的。智能体在进行工具调用时容易违反工具模式，并以错误格式表示表格数据。这些执行错误使验证器无法重新审视主张，导致推理过程以信息不足 或反驳标签结束。总的来说，这一比较证实，工具规划有助于更系统地核查证据，但只有在学习得到的工具接口与工具执行器正确对齐时，这些优势才会显现。

5.5 ReAct 与 CodeAct 的比较

我们从 SciTab 测试数据中随机抽取了 100 个实例，以研究基于 Qwen3-32B 预测结果的 ReAct 与 CodeAct 之间的差异。表格展示了 SIMBA 优化前后正确性的分布12。

在基线实验中 ReAct 独有的 17 个正确实例里，我们观察到 CodeAct 在解析表格数据以及处理数值聚合和比较时常常表现不佳。不同于使用 SQL 工具提取相关表格数据，并在后续推理步骤中解决数学问题的 ReAct，CodeAct 在表格解析和数学推理两方面都依赖 Python 代码。生成的 Python 代码通常会引入一种表格表示，或直接将相关表格单元格赋值给变量，因此容易出错。此外，CodeAct 在进行数值比较时往往过于严格，可能无法考虑所有相关条目。经过 SIMBA 优化后，CodeAct 在 17 个错误实例中有 11 个给出了正确预测。我们发现 SIMBA 尤其是——

模块	指令
基线	根据提供的表格数据核验给定的主张。
直接方法（SIMBA）	根据提供的表格数据核验给定的主张。\n\n如果模块收到的主张指定了某种特定处理方法 作为针对给定条件的“第一”或“第二”个备选方案，应仔细核对表格数据，确保主张的 备选方案顺序与表格一致。 [...] 如果表格与主张中的顺序相矛盾，模块应返回“反驳”。[...] 当表格提供了足够的数据来评估主张时，请勿返回“信息不足”。
思维链（SIMBA）	根据提供的表格数据核验给定的主张。\n\n如果主张涉及某种方法在特定阶段的有效性或性能，而表格包含该阶段的评估指标（例如准确率、百分比），那么模块应重点关注 特定阶段，表格包含该阶段的评估指标（例如准确率、百分比），那么模块应聚焦于此 通过比较表格中的数值，判断主张是得到支持还是被反驳。避免假设表格中缺少某些指标 ，除非明确说明。 [...]

表 7：使用 SIMBA 优化器的 Qwen3-32B 模型的基线指令和优化后的指令。表 88 和9 中分析的特征所对应的提示部分已加下划线。为清晰起见，省略了部分指令内容。

指标	包含序数词		不含序数词	
	基线	SIMBA	基线	SIMBA
实例	1,499		7,110	
按真实标签划分的准确率				
全部（总体）	63.7	**69.0**	64.2	**67.3**
反驳	39.0	**70.6**	38.1	**66.1**
支持	**90.2**	67.4	**89.8**	67.4

表8：在包含序数词的实例上，使用 Qwen3-32B 进行直接预测在优化（SIMBA）前后的比较主张中的（第A.1节），来自 TabFact。

指标	基线	SIMBA	重叠
含比较词的实例	1,882	**2,016**	2,382
按金标准标签划分的重叠实例准确率			
全部（总体）	83.0	**84.1**	–
反驳	83.7	**86.1**	–
支持	**82.2**	81.8	–

表9：思维链与Qwen3-32B在优化（SIMBA）前后对包含比较术语的实例的比较 在推理中（第A.2节），来自TabFact。

结果	Qwen3-32B	Qwen3-8B
两者均正确	33	22
仅思维链正确	2	5
仅 ReAct 正确	6	8
两者均错误	9	15

表10：在随机抽取的50条 SciTab 测试集声明上比较思维链与 ReAct。

分类体系	ReAct	CodeAct
数据相关错误	7（35%）	14（70.0%）
代码相关错误	5（25%）	6（30.0%）
推理错误	9（45%）	3（15.0%）

表 11：在 SciTab 上使用 Qwen3-32B 时，ReAct 与 CodeAct 预测中发现的各类错误实例数量。

结果	基线	使用 SIMBA
两者均正确	44	52
仅 ReAct 正确	17	12
仅 CodeAct 正确	7	13
两者均错误	32	23

表 12：ReAct 与 CodeAct 在 100 个随机抽取的 SciTab 测试集实例上的表现对比（使用 Qwen3-32B）

在通过引入启发式规则改进表格解析和简单数值比较方面尤为有效。然而，CodeAct在优化后验证复杂声明时仍表现出有限的推理能力，尤其是当主张包含模糊描述或需要多行比较时，性能较弱。

6 结论

近期关于表格事实核查的研究探讨了工具增强和智能体方法，但外部工具和程序辅助推理是否能为最新的大语言模型带来持续优势，超越基于语言的推理，仍不明确。在这项工作中，我们首次对比了

对表格事实核查的各种提示技术进行比较分析，并考察指令优化对两个模型系列的推理和工具使用行为的影响。我们的分析表明，MiPROv2 优化器为思维链推理带来了显著增益，而 SIMBA 在精炼 ReAct 和 CodeAct 智能体的指令方面被证明是有效的。MiPROv2 和 SIMBA 都鼓励更直接的推理路径，并有助于减少不必要的工具调用。总体而言，思维链仍然是表格事实核查的有力选择，尤其是对于较小模型。与此同时，基于较大模型构建的 ReAct 智能体可以取得具有竞争力的性能，但需要更谨慎的指令优化。

局限性

尽管我们的研究为指令优化对表格事实核查中多种提示技术的影响提供了有价值的见解，但我们承认实验设置存在若干局限性。我们的实验范围仅限于两个特定的模型系列，即 Qwen3 和 Gemma3，每个系列各有两种规模，而我们的发现对其他模型架构或规模大得多的模型的可推广性仍是一个开放性问题。我们的比较分析仅限于三种具有代表性的指令优化方法；对最新技术进行更广泛的综述，例如 GEPA（Agrawal 等人，2025年），可能会揭示不同的模型行为或性能权衡。我们对 ReAct智能体的评估仅使用单一工具来执行 SQL 查询，这无法捕捉多工具场景中工具选择的复杂性。此外，我们没有考虑初始指令质量对优化过程的潜在影响。所有实验均采用相同的、格式简单的种子指令进行。

致谢

我们感谢匿名审稿人富有见地的反馈与意见。Xiaotang Du 部分得到英国研究与创新自然语言处理博士培训中心的支持，该中心由英国研究与创新（资助编号 EP/S022481/1）和爱丁堡大学信息学院资助。Pasquale Minervini 部分获得 ELIAI（爱丁堡集成人工智能实验室）、EPSRC（资助编号 EP/W002876/1）、思科的行业资助以及埃森哲有限责任合伙公司的捐赠资助。Giwon Hong 得到爱丁堡大学信息学院 ILCC 博士项目（信息学院资助方案）的支持。本研究得到爱丁堡国际数据设施（EIDF）以及爱丁堡大学数据驱动创新计划的支持。部分评估使用了由剑桥大学研究计算服务运营的剑桥数据驱动发现服务（CSD3）所提供的资源；这些资源由戴尔 EMC 和英特尔利用工程与自然科学研究委员会的二级资助（资本资助 EP/T022159/1）以及科学技术设施委员会的 DiRAC 资助提供。

参考文献

Nikhil Abhyankar、Vivek Gupta、Dan Roth 与 Chandan K. Reddy。2025年。H-STAR：面向表格的大语言模型驱动的 SQL-文本混合自适应推理。载于《计算语言学协会美洲分会*2025*年会议论文集：人类语言技术（第*1*卷：长论文）》，第8841–8863页，新墨西哥州阿尔伯克基。计算语言学协会。

Lakshya A Agrawal、Shangyin Tan、Dilara Soylu、Noah Ziems、Rishi Khare、Krista Opsahl-Ong、Arnav Singhvi、Herumb Shandilya、Michael J Ryan、Meng Jiang、Christopher Potts、Koushik Sen、Alexandros G. Dimakis、Ion Stoica、Dan Klein、Matei Zaharia 和 Omar Khattab。2025年。 GEPA：反思式提示演化可以优于强化学习。 预印本，arXiv:2507.19457。

Mubashara Akhtar、Oana Cocarascu 和 Elena Simperl。2022年。 PubHealthTab：面向循证事实核查的公共卫生表格数据集。载于《计算语言学协会成果：*NAACL 2022*》，第1–16页，美国西雅图。计算语言学协会。

Rami Aly、Zhijiang Guo、Michael Sejr Schlichtkrull、James Thorne、Andreas Vlachos、Christos Christodoulopoulos、Oana Cocarascu 和 Arpit Mittal。2021年。 非结构化与结构化信息上的事实抽取与验证（FEVEROUS）共享任务。载于 《第四届事实抽取与验证（*FEVER*）研讨会论文集》，第1–13页，多米尼加共和国。计算语言学协会。

Rami Aly 和 Andreas Vlachos。2024年。 TabVer：基于自然逻辑的表格事实核查。计算语言学协会会刊，12：1648–1671。

Kushal Raj Bhandari、Sixue Xing、Soham Dan 和 Jianxi Gao。2025年。通过注意力分析探索语言模型在表格问答中的鲁棒性。机器学习研究汇刊，2025年。

Chu Sern Joel Chan、Aakanksha Naik、Matthew Akamatsu、Hanna Bekele、Erin Bransom、Ian Campbell 和 Jenna Sparks。2024年。关于科学主张语境化的 context24 共享任务概述。载于第四届学术文档处理研讨会论文集（*SDP 2024*），第12–21页，泰国曼谷。计算语言学协会。

Wenhu Chen、Hongmin Wang、Jianshu Chen、Yunkai Zhang、Hong Wang、Shiyang Li、Xiyou Zhou 和 William Yang Wang。2020年。 TabFact：用于基于表格的事实验证的大规模数据集。载于第八届国际学习表征会议论文集。

Zhoujun Cheng、Tianbao Xie、Peng Shi、Chengzu Li、Rahul Nadkarni、YushiHu、Caiming Xiong,

Dragomir Radev、Mari Ostendorf、Luke Zettle moyer、Noah A. Smith 和 Tao Yu。2023年。《在符号语言中绑定语言模型》。发表于*ICLR*。OpenRe- view.net。

Rui Dong与David Smith。2021年。结构编码与预训练至关重要：将BERT适配于基于表格的事实核查。载于计算语言学协会欧洲分会第*16*届会议论文集：主卷，第2366–2375页，在线。计算语言学协会。

Julian Eisenschlos、Syrine Krichene与Thomas Müller。2020年。通过中间预训练理解表格。载于计算语言学协会论文集：*EMNLP 2020*，第281–296页，在线。计算语言学协会。

Parker Glenn、Parag Dakle、Liang Wang与Preethi Raghavan。2024年。BlendSQL：一种用于在关系代数中统一混合式问答的可扩展方言。载于计算语言学协会论文集：*ACL 2024*，第453–466页，曼谷，泰国。计算语言学协会。

Jonathan Herzig、Pawel Krzysztof Nowak、Thomas Müller、Francesco Piccinno 和 Julian Eisenschlos。2020年。TaPas：通过预训练实现弱监督表格解析。载于第*58*届计算语言学协会年会论文集，第4320–4333页，线上举行。计算语言学协会。

Chuang Jiang、Mingyue Cheng、Xiaoyu Tao、Qingyang Mao、Jie Ouyang 和 Qi Liu。2025年。Tablemind：一种用于工具增强的表格推理的自主程序化智能体。*arXiv*预印本 *arXiv:2509.06278*。

Omar Khattab、Arnav Singhvi、Paridhi Maheshwari、Zhiyuan Zhang、Keshav Santhanam、Sri Vard-hamanan、Saiful Haq、Ashutosh Sharma、Thomas T. Joshi、Hanna Moazam、Heather Miller、Matei Za-haria 和 Christopher Potts。2024年。DSPy：将声明式语言模型调用编译为最先进的流水线。载于*ICLR*。OpenReview.net。

Alina Leidinger、Robert van Rooij 和 Ekateri na Shutova。2023年。提示的语言：哪些语言学属性能让提示取得成功？载于计算语言学协会：*EMNLP 2023*年论文集，第 9210–9232 页，新加坡。计算语言学协会。

JasonLiu 和贡献者。2024年。Instructor：一个用于从大语言模型生成结构化输出的程序库。

Qian Liu、Bei Chen、Jiaqi Guo、Morteza Ziyadi、Zeqi Lin、Weizhu Chen 和 Jian-Guang Lou。2022年。TAPEX：通过学习神经网络 SQL 执行器进行表格预训练。载于*ICLR*。OpenReview.net。

Xinyuan Lu、LiangmingPan、Qian Liu、PreslavNakov 和 Min-Yen Kan。2023 年。SCITAB：一个具有挑战性的用于科学表格组合推理和主张核查的基准。载于*2023*年自然语言处理实证方法会议论文集，第 7787–7813 页，新加坡。计算语言学协会。

Xinyuan Lu、Liangming Pan、Yubo Ma、Preslav Nakov 和 Min-Yen Kan。2025年。TART：一个用于可解释表格推理的开源工具增强框架。载于计算语言学协会论文集：*NAACL 2025*，第 4323–4339 页，阿尔伯克基，新墨西哥州。计算语言学协会。

Krista Opsahl-Ong、MichaelJ Ryan、Josh Purtell、David Broman、Christopher Potts、Matei Zaharia 和 Omar Khattab。2024年。优化多阶段语言模型程序的指令和示范。载于*2024*年自然语言处理实证方法会议论文集，第 9340–9366 页，迈阿密，佛罗里达，美国。计算语言学协会。

Suixin Ou 和 Yongmei Liu。2022年。通过结构感知语义解析学习生成用于表格事实验证的程序。收录于第*60*届计算语言学协会年会论文集（第*1*卷：长篇论文），第7624–7638页，爱尔兰都柏林。计算语言学协会。

Michael Schlichtkrull、Zhijiang Guo 和 Andreas Vlachos。2023年。AVeriTeC：一个利用网络证据进行真实世界主张验证的数据集。收录于第三十七届神经信息处理系统大会数据集与基准赛道。

Qi Shi、Yu Zhang、Qingyu Yin 和 Ting Liu。2020年。学习结合语言和符号信息进行基于表格的事实验证。收录于第*28*届国际计算语言学会议论文集，第5335–5346页，西班牙巴塞罗那（线上）。国际计算语言学委员会。

Gemma团队、Aishwarya Kamath、Johan Ferret、Shreya Pathak、Nino Vieillard、RamonaMerhej、Sarah Perrin、Tatiana Matejovicova、Alexandre Ramé、Morgane Rivière、Louis Rouillard、Thomas Mesnard、Geoffrey Cideron、J ean bastien Grill、Sabela Ramos、Edouard Yvinec、Michelle Casbon、Etienne Pot、Ivo Penchev及其他197位作者。2025年。Gemma 3 技术报告。预印本，arXiv:2503.19786。

Chengye Wang、Yifei Shen、Zexi Kuang、Arman Cohan和Yilun Zhao。2025年。SciVer：评估用于多模态科学主张核验的基础模型。载于计算语言学协会第*63*届年会论文集（第*1*卷：长篇论文），第8562–8579页，奥地利维也纳。计算语言学协会。

Nancy X. R. Wang、Diwakar Mahajan、Marina Danilevsky 和 Sara Rosenthal。2021年。SemEval-2021 任务9：事实核查与证据查找

科学文档中的表格数据（SEM-TAB-FACTS）。见《第*15*届语义评估国际研讨会论文集（*SemEval-2021*）》，第317–326页，线上举行。计算语言学协会。

Xingyao Wang、Yangyi Chen、Lifan Yuan、Yizhe Zhang、Yunzhu Li、Hao Peng和Heng Ji。2024a。可执行代码动作能够激发更优的大语言模型智能体。载于 *ICML*。

Zilong Wang、Hao Zhang、Chun-Liang Li、Julian Martin Eisenschlos、Vincent Perot、Zifeng Wang、Lesly Miculicich、Yasuhisa Fujii、Jingbo Shang、Chen-Yu Lee以及其他1人。2024b。Chain-of-Table：在推理链中演化表格以促进表格理解。*arXiv*预印本 *arXiv:2401.04398*。

Albert Webson和Ellie Pavlick。2022年。基于提示的模型真的理解其提示的含义吗？载于《*2022*年计算语言学协会北美分会会议论文集：人类语言技术》，第2300–2344页，美国西雅图。计算语言学协会。

Jason Wei、Xuezhi Wang、Dale Schuurmans、Maarten Bosma、Fei Xia、Ed Chi、Quoc V Le、Denny Zhou 和其他 1 位作者。2022年。思维链提示引发大语言模型的推理。神经信息处理系统进展，35：24824–24837。

Xiaofeng Wu、Alan Ritter 和 Wei Xu。2025年。利用大语言模型理解表格数据：近期进展与挑战综述。*arXiv* 预印本 *arXiv:2508.00217*。

ZiruiWuand Yansong Feng。2024年。ProTrix：构建用于结合句子上下文进行表格规划与推理的模型。载于计算语言学协会论文集：*EMNLP 2024*，第4378–4406 页，美国佛罗里达州迈阿密。计算语言学协会。

An Yang、Anfeng Li、Baosong Yang、Beichen Zhang、Binyuan Hui、Bo Zheng、Bowen Yu、Chang Gao、Chengen Huang、Chenxu Lv、Chujie Zheng、Dayi-heng Liu、Fan Zhou、Fei Huang、Feng Hu、Hao Ge、Haoran Wei、Huan Lin、Jialong Tang 和其他 41 位作者。2025a。Qwen3 技术报告。*arXiv* 预印本 *arXiv:2505.09388*。

Bohao Yang、Yingji Zhang、Dong Liu、André Freitas 和 Chenghua Lin。2025b。表格来源是否重要？多模态科学表格理解与推理的基准测试及改进。*arXiv* 预印本 *arXiv:2501.13042*。

Xiaoyu Yang、Feng Nie、Yufei Feng、Quan Liu、Zhigang Chen 和 Xiaodan Zhu。2020年。结合语言化表达与图注意力网络的程序增强型事实核查。载于*2020*年自然语言处理实证方法会议论文集（*EMNLP*），第 7810–7825 页，线上举行。计算语言学协会。

Xiaoyu Yang 和 Xiaodan Zhu。2021年。探索基于表格的事实验证中的分解方法。载于计算语言学协会论文集：*EMNLP 2021* 年研究成果，第 1045–1052 页，多米尼加共和国蓬塔卡纳。计算语言学协会。

Shunyu Yao、Jeffrey Zhao、Dian Yu、Nan Du、Izhak Shafran、Karthik Narasimhan 和 Yuan Cao。2023年。ReAct：协同语言模型中的推理与行动。载于第十一届国际学习表征会议论文集。ICLR。

Hongzhi Zhang、Yingyao Wang、Sirui Wang、Xuezhi Cao、Fuzheng Zhang 和 Zhongyuan Wang。2020年。基于结构感知 Transformer 的表格事实验证。载于*2020*年自然语言处理实证方法会议论文集（*EMNLP*），第 1624–1629 页，线上举行。计算语言学协会。

Tianshu Zhang、Xiang Yue、Yifei Li 和 Huan Sun。2024a。TableLlama：迈向面向表格的开放式通用大型模型。载于计算语言学协会北美分会*2024*年会议论文集：人类语言技术（第*1*卷：长篇论文），第6024–6044页，墨西哥墨西哥城。计算语言学协会。

Yuji Zhang、Qingyun Wang、Cheng Qian、Jiateng Liu、Chenkai Sun、Denghui Zhang、Tarek Abdelzaher、Chengxiang Zhai、Preslav Nakov 和 Heng Ji。2025年。面向科学表格主张验证的原子推理。*arXiv*预印本 *arXiv:2506.06972*。

Yunjia Zhang、Jordan Henkel、Avrilia Floratou、Joyce Cahoon、Shaleen Deep 和 Jignesh M Patel。2024b。ReacTable：增强 ReAct 以回答表格问题。*VLDB*基金会会刊，17(8)：1981–1994。

Yilun Zhao、Yitao Long、Yuru Jiang、Chengye Wang、Weiyuan Chen、Hongjun Liu、Yiming Zhang、Xi-angru Tang、Chen Zhao 和 Arman Cohan。2024年。FinDVer：针对长篇及混合内容金融文档的可解释主张核查。

Wanjun Zhong、Duyu Tang、Zhangyin Feng、Nan Duan、Ming Zhou、Ming Gong、Linjun Shou、Daxin Jiang、Jiahai Wang 和 Jian Yin。2020年。Logical-FactChecker：利用图模块网络进行逻辑运算以实现事实核查。载于计算语言学协会第*58*届年会论文集，第6053–6065页，线上。计算语言学协会。

Wei Zhou、Mohsen Mesgar、Annemarie Friedrich 和 Heike Adel。2025年。在复杂表格问答中使用工具进行在线规划的高效多智能体协作。载于计算语言学协会北美分会（*NAACL 2025*）论文集，第945–968页，美国新墨西哥州阿尔伯克基。计算语言学协会。

A 实现细节

A.1 序数词列表

表格18 列出了第5.1节实验中使用的序数词。这些词语在实验开始前预先确定，用于识别主张涉及序数信息的情况，例如主张与表格值之间或表格值彼此之间的顺序或排名关系。

A.2 比较术语列表

表格 19 展示了第 5.1 节实验中使用的比较术语列表。这些术语在实验开始前预先确定，并作为代理指标，用于判断模型的推理过程是否涉及主张与表格值之间的比较，或表格值彼此之间的比较。

A.3 超参数设置

我们使用 DSPy 中的指令优化器进行分析。

COPRO 对于 COPRO 优化，我们将广度设为 6，深度设为 3。指令生成的初始温度设为 1.2。

MiPROv2 为确保对各优化器进行一致的比较，我们将 MiPROv2 优化器配置为仅优化提示指令。具体而言，我们将自举示例数量和带标签示例数量均设为 0，并使用中等优化模式。

SIMBA 我们将每次迭代生成的候选项数量设为 6，迭代步数设为 8。少样本示例的最大数量设为 0，以确保仅优化指令。优化过程中，我们使用随机种子 0 对候选程序进行采样。

A.4 数据集细节

我们实验中使用的所有评估数据集（PubHealthTab、SciTab、TabFact、MMSci）均采用 MIT许可证。对于每个数据集，我们均采用其官方 GitHub代码库中的原始版本。

A.5 计算细节

使用 Qwen3-8B、Gemma3-12B 和 Gemma3-27B 进行的实验均在单张配备 80GB GPU 内存的 NVIDIA A100 GPU 上运行。实验使用 Qwen3-32B 进行的实验在两张 NVIDIA A100 GPU 上运行。GPU 小时数取决于模型规模、提示方法、优化器类型和测试数据集的大小。例如，使用 COPRO 和 SIMBA 优化器，在混合训练数据上优化 ReAct 或 CodeAct 中的指令，最多需要 10 小时。使用大型骨干模型（27B/32B），通过 ReAct 或 CodeAct 框架在 TabFact 测试数据上进行评估，通常最多需要 2 到 3 天。

A.6 ReActable 的实现

官方的 ReActable 实现与 GPT-4o 模型不兼容，因为这些模型无法始终遵循上下文示例中定义的输出格式，导致从生成内容中提取代码块时解析失败。为了使用 GPT-4o 模型评估 ReActable，我们通过 Instructor 程序库（Liu 和贡献者，2024年）强制进行结构化输出生成，从而复现该流程。在每次迭代中，模型可以从三种行动中选择：执行 SQL 命令、执行 Python 代码片段，或预测最终判定。代码会在最近一个能够成功处理的中间表上执行。我们将最大迭代次数设为 5，温度设为 0，并将每次迭代生成的词元数限制为 3,500。如果模型在连续迭代中生成重复的代码块，或代码执行失败，则强制模型输出最终标签。

我们通过调整指令以适应任务来扩展提示模板，并设计了包含更详细指令的系统提示，以指导结构化输出生成，如图1、2、3和4所示。在所有三分类事实核查实验中，我们使用相同的上下文演示，这些演示由从 SciTab 训练集中随机抽取的 5 个实例构成。在 TabFact 实验中，我们使用其中 3 个标注了二元标签的上下文示例。

B 优化后的指令

表 13展示了 Qwen3-32B 使用不同优化器得到的优化后的指令。经过 COPRO 优化后，精炼后的指令描述了详细的流程，其中将“设计 SQL 查询以提取相关数据”列为必需的推理步骤。这会迫使智能体在处理简单查询时也使用工具，可能导致大量低质量或不必要的工具调用。使用 MiPROv2 时，优化后的指令

优化器	优化后的指令
基线	请根据提供的表格数据核实给定的主张。
COPRO	你是一名负责判断给定主张是否得到支持或是否不准确的推理智能体，或根据所提供表格中的数据及其上下文标题判定为无法确定。你将分析表格的结构、内容和上下文，使用 execute_sql 工具提取或核实相关证据，并利用检索到的信息进行基于证据的评估。\n\n请遵循以下结构化方法：\n1. 首先仔细阅读主张、表格和标题，以了解上下文以及表格所代表的内容。\n\n2. 设计 SQL 查询，以通过使用带有 table_name 的 execute_sql 工具，直接从表格中提取相关数据针对该主张的 SQL_查询。\n3. 仔细检查查询结果，并结合以下内容进行评估该主张。\n4. 逐步推理你的发现，并明确引用由查询得出的证据以及表格。\n5. 一旦你掌握了足够的信息并得出明确结论，判断该主张是否得到数据支持或与数据相矛盾，请使用 finish工具给出最终答案：支持、不支持或证据不足。\n\n你的答案应以数据为依据并有充分支持，且推理清晰。
MiPROv2	请使用提供的结构化表格数据，仔细核实所提供主张的准确率。你是一名智能体，负责推理并判断该主张是否得到支持、遭到反驳，或是否有足够的信息进行判断。必要时，使用 execute_sql 工具查询表格，以获得更深入的见解；确信核实结果后，使用 finish 工具。推理始终应以数据为依据，避免作出表格未直接支持的假设。
SIMBA	请根据提供的表格数据核实给定的主张。你是一个智能体。在每个回合中，你将获得主张、表格和标题字段作为输入。你还可以查看目前为止的过往轨迹。你的目标是使用一个或多个提供的工具收集必要的信息，以得出答案。为此，你需要在每个轮次中交替给出 next_thought、next_tool_name 和 next_tool_args，并且在完成任务时也要如此。每次工具调用后，你都会收到相应的观察结果，并将其追加到你的轨迹。\n\n在撰写下一条_思考时，你可以推理当前情况并规划后续步骤。选择下一个_工具_名称及其下一项_工具_参数时，工具必须是以下之一：\n\n(1) execute_sql [...]\n(2) finish [...]\n提供下一项_工具_参数时，字段中的值必须采用 JSON 格式\n\n如果表格数据明确包含验证主张所需的信息，模块应立即调用 finish工具，而不执行不必要的 SQL 查询。具体而言，如果主张涉及将 URL 与组织名称匹配，且表格直接列出了这些对应关系，模块应识别这一点，并在不再进行工具调用的情况下结束任务。 [...] 如果主张指的是某个特定指标（例如准确率）和特定设置（例如传导式），模块应核实表格是否明确报告了该指标和设置。如果表格只提供相关指标（例如 F 分数）或未提及该设置，模块就不应假定二者等价，而应得出结论：这些数据并不直接支持该主张。

表 13：用于 Qwen3-32B 模型的 ReAct 智能体的种子指令及由不同优化器（COPRO、MiPROv2、SIMBA）优化后的指令。带下划线的指令突出了各优化器的关键特征。为清晰起见，省略了部分指令内容。

仅包含简短且高层次的指导原则。它明确提到“必要时使用 execute_sql 工具以获得更深入的见解”，允许智能体根据主张复杂度决定何时调用工具。SIMBA 优化会在种子指令后附加具体规则，从而生成冗长的指令，其中包括如何构造工具调用，以及何时应避免构造复杂 SQL 查询等细节。通过提供明确的启发式规则，SIMBA 能有效教会智能体采用更复杂的决策策略，不仅能减少不必要的工具调用，还有助于避免推理陷阱（例如，如果表格中未明确提及指标名称，智能体就不应将不同的评估指标视为相同）。

表格 20 至35 展示了针对不同模型和提示技术，使用不同优化器（COPRO、MiPROv2 和 SIMBA）为 Qwen3-8B、Qwen3-32B、Gemma3-12B 和 Gemma3-27B 模型上的四种提示技术（直接提示、思维链、ReAct 和 CodeAct）生成的优化后的指令。

C 消融研究

C.1 对随机种子的鲁棒性

我们使用不同的随机种子开展消融研究，以验证指令优化方法的敏感性。由于 COPRO 不依赖随机种子，我们仅调整 MiPROv2 和 SIMBA 优化器中的随机种子。在 MiPROv2 和 SIMBA 中，随机种子用于采样提示策略以生成新的候选指令，并创建验证数据的小批量，以评估所提出指令的质量。我们评估了采用思维链提示的 Qwen3-32B 在三个

随机种子。在表格 14中，我们观察到两种优化器在 SciTab、TabFact 和 MMSci 测试集上的 F1 分数均取得了一致的性能提升。在大多数设置下，每种优化器的指标值变化均低于 2%，这表明这些优化器对随机种子的变化并不高度敏感。

C.2 初始指令质量的影响

为评估初始指令质量对指令优化的影响，我们使用三种不同的种子指令，通过思维链提示对 Qwen3- 32B 进行优化。我们考虑了信息量各异的种子指令，包括空指令、对前文分析中使用的默认指令的改写，以及内容更详细的指令。表 15表明，不同优化器对初始指令质量的敏感程度各不相同。COPRO 能从信息更丰富、内容更详细的初始指令中获益；随着种子指令包含的信息增多，其在 TabFact 和 MMSci 上的 F1 分数也稳步提升。这主要是因为 COPRO 仅根据先前的候选指令及其评估分数提出新的候选指令。详细的种子指令可以在早期迭代中为指令生成提供更多指导。

SIMBA 使用简洁的种子指令时性能达到峰值，而在种子指令中加入更多细节则可能导致性能下降。由于 SIMBA 会在初始指令后附加启发式规则，从详细指令开始可能会限制搜索空间，从而对优化后指令的泛化能力产生负面影响。在语义等价的种子指令中，与 COPRO 和 MiPROv2 相比，SIMBA 对表层形式变化最不敏感。使用改写后的初始指令进行 SIMBA 优化后，模型在所有测试集上的性能仍与基线相近。

C.3 使用 TART 工具优化 ReAct

除了 SQL 工具外，我们还研究了配备单个或多个 TART框架中最常用工具的 ReAct 智能体（Lu 等人，2025年）。我们使用 MiPROv2 和 SIMBA 优化以 Qwen3-32B 为骨干模型的 ReAct 智能体的指令，因为根据表 2，这些优化器在提升 ReAct 性能方面更有效。如表 17所示，增加 TART 工具的数量不一定能提升优化后的性能。在大多数实验设置中，随着工具函数数量从 3 增加到 10，优化后测试集上的 F1 分数呈 U 形趋势。与使用前 10 名 TART 工具相比，使用单个工具（例如 get_column_cell_value）的 ReAct 智能体在指令优化前后都能取得更好的性能。get_row_by_name 工具在 PubHealthTab 和 MMSci 上尤其有效，而 get_row_index_by_value 在 SciTab 上表现出色。尽管 ReAct 智能体搭配 TART 工具或其组合在某些评测设置中效果显著，但经过 SIMBA 优化后，其性能仍不如搭配 SQL 工具的对应智能体。

C.4 训练数据配置

我们通过采用三种训练配置进行优化，考察训练数据对指令优化的影响。我们没有使用混合训练数据，而是分别从 SciTab、TabFact 和 PubHealthTab 训练集中随机抽取 100 个实例，构建单一来源的训练集。如表16所示，SIMBA 从使用混合训练集中获益更多，而 MiPROv2 对训练数据分布的敏感度较低。这表明，混合数据可以帮助 SIMBA 学习适用于不同基于表格的事实核查任务的通用规则。相比之下，MiPROv2 遵循人工编写的启发式规则来提出新的候选指令，并主要依靠训练数据理解任务格式。此外，我们发现，在 TabFact 数据上学习得到的优化后的指令也能出人意料地很好地泛化到 SciTab 和 MMSci，这可能归功于 TabFact 中更干净的数据和二元标签分布。

随机种子	优化器	PubHealth		SciTab		TabFact		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
0	基线	88.3	87.6	66.4	66.4	84.5	86.6	86.5	61.6
	+MiPROv2	87.8	87.3	67.4	67.2	85.9	87.8	87.4	63.6
	+SIMBA	87.2	86.2	**68.8**	**68.6**	85.2	87.1	86.4	63.1
9	+MiPROv2	88.3	87.2	68.5	68.5	85.2	87.4	86.6	62.4
	+SIMBA	88.3	87.4	67.8	67.5	86.1	87.4	87.8	**65.1**
42	+MiPROv2	87.8	87.1	67.8	67.8	**86.5**	**88.3**	87.3	**65.1**
	+SIMBA	**88.9**	**88.0**	**68.8**	**68.6**	**86.5**	87.9	**88.5**	64.7

表 14：使用思维链的 Qwen3-32B 随机种子消融研究。**粗体** 表示各数据集上的最佳性能。

种子指令	优化器	PubHealth		SciTab		TabFact		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
根据提供的表格数据核验给定的主张。（默认）	基线	88.3	87.6	66.4	66.4	84.5	86.6	86.5	61.6
	+COPRO	87.2	86.1	67.4	67.3	85.5	87.6	86.7	61.5
	+MiPROv2	87.2	86.5	**68.8**	**68.6**	**86.9**	**88.5**	**87.7**	**65.4**
	+SIMBA	**90.0**	**89.6**	**68.8**	**68.6**	85.2	87.1	87.0	64.2
空字符串	基线	85.6	84.5	68.1	68.2	83.5	85.9	86.4	63.0
	+COPRO	87.2	85.9	67.1	67.0	83.3	85.4	85.5	60.8
	+MiPROv2	85.0	83.8	**69.2**	**69.0**	**86.5**	**88.4**	**87.7**	62.2
	+SIMBA	**87.8**	**86.8**	66.7	66.4	84.1	86.1	86.7	**63.1**
检查表格数据是否支持该主张。	基线	89.4	88.8	67.1	67.2	84.0	86.1	86.3	62.3
	+COPRO	**90.0**	**89.5**	66.4	66.4	84.4	86.6	85.4	62.2
	+MiPROv2	88.3	87.5	66.4	66.2	84.2	86.4	**86.9**	63.2
	+SIMBA	89.4	88.5	**68.5**	**68.2**	**86.2**	**87.6**	86.6	**63.8**
检查下面提供的主张，并系统地比较每个 com- 将陈述的组成部分与表格中相应的值、条目和关系进行比较， 判断该主张是否得到支持， 与表格内容矛盾，或无法根据现有表格信息进行验证。	基线	83.9	83.0	**68.8**	**68.8**	85.9	87.9	**87.0**	62.9
	+COPRO	**87.8**	**86.9**	**68.8**	**68.8**	85.5	87.6	86.9	**63.4**
	+MiPROv2	85.6	84.8	66.2	66.2	**87.0**	**88.7**	**87.0**	63.1
	+SIMBA	**87.8**	86.8	65.7	65.8	85.9	87.5	85.8	62.7

表15：A 关于 Qwen3-32B 思维链种子指令的消融研究。**粗体** 表示各自的最佳性能 ed 指令 和数据集。

训练数据	优化器	PubHealth		SciTab		TabFact		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
混合	基线	88.3	87.6	66.4	66.4	84.5	86.6	86.5	61.6
	+COPRO	87.2	86.1	67.4	67.3	85.5	87.6	86.7	61.5
	+MiPROv2	87.2	86.5	**68.8**	**68.6**	**86.9**	**88.5**	**87.7**	**65.4**
	+SIMBA	**90.0**	**89.6**	**68.8**	**68.6**	85.2	87.1	87.0	64.2
SciTab	+COPRO	87.2	86.5	**69.7**	**69.8**	83.2	85.4	85.5	62.4
	+MiPROv2	**88.9**	**88.0**	68.3	68.2	**86.5**	**88.3**	87.4	62.1
	+SIMBA	87.8	87.3	64.1	63.7	84.8	86.5	**88.3**	**67.4**
TabFact	+COPRO	88.3	88.8	67.4	69.2	85.4	87.0	86.4	87.7
	+MiPROv2	87.2	88.3	**70.2**	**71.3**	86.1	88.1	87.0	88.9
	+SIMBA	**90.0**	**89.3**	64.3	67.5	**87.0**	**88.2**	**88.2**	**89.0**
PubHealth	+COPRO	87.8	87.3	**68.5**	**68.4**	**86.6**	**88.1**	**87.8**	**65.8**
	+MiPROv2	87.2	86.4	66.0	66.0	85.6	87.6	87.4	**65.8**
	+SIMBA	**88.9**	**88.3**	67.4	67.3	85.4	87.1	86.6	62.8

表16：使用思维链的 Qwen3-32B 训练数据消融研究。**粗体** 表示每种训练数据配置和数据集对应的最佳性能。

工具函数	优化器	PubHealth		SciTab		TabFact		MMSci	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
执行_SQL	基线	87.8	87.4	61.5	60.1	82.8	83.3	**87.5**	62.6
	+MiPROv2	87.8	87.2	61.5	60.9	84.2	85.2	86.2	63.0
	+SIMBA	**90.6**	**90.0**	**66.2**	**65.9**	**86.1**	**87.0**	85.9	**65.0**
前 10 名 TART 工具	基线	85.0	84.3	59.4	58.1	83.4	84.8	84.7	61.8
	+MiPROv2	**87.8**	**87.1**	60.1	59.3	**86.0**	**87.4**	**87.8**	**62.4**
	+SIMBA	84.4	83.9	**62.7**	**62.7**	81.6	84.4	81.6	57.8
前 3 名 TART 工具	基线	85.0	84.1	58.0	56.6	83.4	85.0	83.2	59.1
	+MiPROv2	**86.7**	**85.9**	**61.3**	**60.2**	**83.5**	**85.9**	**86.6**	**62.5**
	+SIMBA	84.4	83.9	58.7	58.5	82.1	84.2	82.8	58.6
排名前 5 的 TART 工具	基线	84.4	83.6	**59.7**	58.0	**84.1**	85.6	83.8	59.7
	+MiPROv2	**85.0**	**84.2**	58.3	57.2	83.9	**85.8**	**86.3**	**62.6**
	+SIMBA	82.8	81.6	58.7	**58.1**	82.0	84.0	82.2	58.1
等于_到	基线	**86.7**	**85.9**	59.9	59.6	83.6	84.5	86.1	62.8
	+MiPROv2	86.1	85.0	59.4	58.5	**84.3**	**85.3**	**87.3**	**63.4**
	+SIMBA	85.0	84.7	**66.0**	**65.8**	83.5	84.8	85.8	61.4
按名称获取_列__	基线	87.2	86.6	61.8	60.7	**85.2**	**86.6**	**85.8**	60.8
	+MiPROv2	**87.8**	**87.2**	59.2	58.6	81.6	84.4	84.8	**61.7**
	+SIMBA	83.9	82.5	**63.9**	**64.1**	83.4	85.7	83.7	60.2
获取_列_单元格_值	基线	87.8	86.6	59.9	58.5	83.3	85.1	84.4	**62.1**
	+MiPROv2	**88.3**	**87.8**	60.4	59.7	**84.9**	**86.2**	**86.8**	61.8
	+SIMBA	**88.3**	87.6	**65.3**	**65.1**	81.7	84.0	81.4	59.4
按_名称获取_行_	基线	88.3	87.7	57.8	55.6	85.2	86.6	87.0	63.6
	+MiPROv2	**88.9**	**88.2**	60.1	58.5	**85.4**	86.7	**88.1**	63.0
	+SIMBA	87.2	86.3	**61.1**	**60.7**	85.0	**86.8**	87.5	**66.6**
根据_value获取_row_index_	基线	**87.8**	**87.0**	62.2	61.6	**84.6**	**86.5**	86.5	**63.1**
	+MiPROv2	87.2	86.5	64.1	63.8	82.6	84.4	**86.8**	61.6
	+SIMBA	87.2	86.7	**66.0**	**66.1**	82.4	85.6	86.5	61.2

表17：使用 ReAct 的 Qwen3-32B 在 TART 中所用工具函数的消融研究。**粗体** 表示每种工具函数和数据集上的最佳性能。

序数词
‘第一’，‘第二’，‘第三’，‘第四’，‘第五’，‘第六’，‘第七’，‘第八’，‘第九’，‘第十’，‘下一个’，‘最后一个’， ‘前一个’，‘后续的’，‘最初的’，‘最终的’，‘主要的’，‘次要的’，‘随后的’，‘在前的’，‘开端’， ‘中间’，‘末尾’，‘较早的’，‘较晚的’，‘先前的’，‘后面的’，‘连续的’，‘接连的’，‘按顺序的’

表18：第 5.1 节实验中使用的序数词列表

比较术语
’较低’, ’较高’, ’更大’, ’较小’, ’更大’, ’较少’, ’相等’, ’不相等’, ’更大’, ’更短’, ’更长’, ’更深’, ’更强’, ’更弱’, ’更快’, ’更慢’, ’更早’, ’更晚’, ’更好’, ’更差’, ’优越’, ’较差’, ’最大值’, ’最小值’, ’大于’, ’小于’, ’等于’, ’多于’, ’少于’ , ’至少’ , ’至多’ , ’不大于’ , ’不小于’

表19：第 5.1 节实验中使用的比较术语列表

优化器	指令
基线	根据提供的表格数据验证给定的主张。
COPRO	根据提供的表格数据验证给定的主张。
MiPROv2	分析主张及相关表格数据，判断该主张是受到支持、遭到反驳， 还是根据表格中提供的信息无法确定。仔细检查表格中的细节，以便 评估主张的准确性，并给出清晰且有理有据的结论。
SIMBA	根据提供的表格数据核实给定的主张。

表20：使用不同优化器进行直接提示时，Qwen3-8B模型的基线指令和优化后的指令。

优化器	指令
基线	根据给定的表格数据验证给定的主张。
COPRO	仔细分析表格数据和主张，判断该主张是得到支持、与数据矛盾，还是 无法根据数据确定。请清晰简洁地作出评估，并简要说明你的 推理过程。
MiPROv2	分析主张、表格数据和表格标题，以判断该主张是得到支持、遭到反驳，还是 信息不足，无法核实。请逐步说明你的推理过程，解释你是如何得出结论的 在得出结论时，请明确将最终答案表述为以下选项之一：’支持’、’反驳’或’信息 不足’。
SIMBA	根据提供的表格数据核实给定的主张。\n\n如果模块收到一项比较价格的主张 涉及不同能源来源的价格数值，则应关注表格中给出的实际价格数值，而不是 将该主张解读为比较价格的数量。具体来说，当主张指出某种能源 来源具有“能源价格数值最高”，应理解为比较价格数值，而不是 比较条目数量。模块应直接比较表格中列出的价格数值，并判断 根据实际提供的数值数据，主张是得到支持、遭到反驳，还是信息不足。 \n\n如果模块收到的主张涉及具体差值（例如“每项指标低 1.4”），就应检查表格是否提供了明确的数值或指标，以支持或反驳 这一差值。如果表格中没有这类具体数据，模块就应得出结论：信息“并不 这一差异。如果表格不包含此类特定数据，模块应得出结论，认为存在’不 信息充足’，而不是对数据做出假设。请专注于主张的确切措辞，以及 表格中是否存在具体的数值比较。如果模块收到一条比较两个 模型的主张（例如，’SegMatch 的表现略好于 Audio2vec’），应首先识别表格中所提及模型的所有相关变体。 对于表格中的每个指标，直接比较这些模型的数值。如果 主张指的是某种具体差异（例如，’略好’），请确保表格中的实际数值差异足以支持或反驳该主张。如果表格提供了明确的数值， 该表中的差异是否足以支持或反驳该主张。如果表格提供了明确的数值，来表明】【。 相关模型，据此判断该主张是受到支持、遭到反驳，还是信息不足 。避免对数据作出假设，并着重关注主张的确切措辞以及 表格中是否存在具体的数值比较。

表 21：使用不同优化器的 Qwen3-8B 模型思维链方法的基线指令和优化后的指令。

优化器	指令
基线	请根据提供的表格数据验证给定的主张。
COPRO	你是一名智能体，负责根据给定的表格评估一项主张。你将获得“主张”，将‘表格‘和‘标题‘作为输入。你的目标是分析数据，并判断该主张是否得到支持、与表格矛盾，还是信息不足，无法作出判断。你可以使用提供的工具进行分析并得出结论。你应交替进行思考、编写 SQL 查询和调用工具，以便得出经过充分推理的结论。
MiPROv2	你是一个智能体，负责根据表格核实一项主张。系统会向你提供‘主张‘、‘表格‘和将‘标题‘作为输入，并结合你过去的轨迹。你的目标是确定该主张是否得到支持、遭到反驳，还是信息不足以验证。\n\n为此，你需要：\n1. 仔细分析主张和表格数据。\n2. 运用推理规划下一步，可能包括：\n a) 执行 SQL 查询，使用‘execute_sql‘工具从表格中提取相关数据。此工具接受表格数据、表格名称和 SQL 查询作为参数。\n b) 当你掌握了足够的信息来评估该主张时，使用‘finish‘工具结束任务。\n3. 每次操作后，你都会收到一条观察结果，随后追加到你的轨迹中。\n4. 根据观察结果和你的推理，你将迭代地决定下一步操作，直到你能够确定该主张是得到支持、遭到反驳，还是无法验证。\n\n当编写 ‘next_thought‘ 时，你必须清楚地说明你的推理过程以及下一步计划。选择 ‘next_tool_name‘ 时，请务必在 ‘execute_sql‘ 和 ‘finish‘ 之间进行选择。指定 ‘next_tool_args‘ 时，请确保参数采用 JSON 格式。\n\n你的最终答案应为以下之一：\n- \"supports\"（如果表格数据支持该主张）。\n- \"refutes\"（如果表格数据反驳该主张）。\n- \"信息不足\"：如果表格数据未提供足够的信息来评估该主张。\n\n请保持你的推理过程清晰简洁，并确保根据可用信息说明采取这些行动的理由表格。
SIMBA	根据提供的表格数据验证给定的主张。\n\n你是一个智能体。在每个回合中，你将获得字段‘主张’、‘表格’、‘标题’作为输入。并且你可以查看目前为止自己的历史轨迹。\n你的目标是使用一个或多个所提供的工具，收集生成‘答案’所需的任何必要信息。\n\n要做到这一点，你将在每一轮中交替使用 next_thought、next_tool_name 和 next_tool_args，并且在完成任务时也是如此。\n每次工具调用后，你会收到相应的观察结果，该观察结果会被追加到你的轨迹中。\n\n在撰写 next_thought 时，你可以推理当前情况，并规划未来的步骤。\n在选择 next_tool_name 及其 next_tool_args 时，该工具必须是以下之一：\n\n(1) execute_sql, 其描述为 <desc>对以列表的列表格式提供的表格执行 SQL 查询。参数：table_data：列表的列表，其中第一行包含表头，后续各行包含数据表_name：表格名称用于执行 SQL 查询的数据框 sql_query：要执行的 SQL 查询字符串。返回：格式化后的字符串，包含转换后的表格或错误消息 </desc>。它接受以下参数：{'table_data': {'items': {}, 'type': 'array'}, 'table_name': {'type': 'string'}, 'sql_query': {'type': 'string'}}。\n(2) finish，其描述为 <desc>将任务标记为已完成。也就是说，这表示生成输出（即‘答案’）所需的全部信息现在均已可供提取。</desc>。它接受参数 {}。\n提供‘next_tool_args’时，其中的值该字段必须采用 JSON 格式\n\n如果模块收到一项主张，涉及基于其训练方法（例如，强化学习与学习得到的奖励）对系统进行比较，则应着重识别表格数据中与训练方法直接相关的明确指标。例如，如果该主张提到“学习奖励”或“无强化训练”，模块应查找诸如“学习得到的”或“未使用强化学习”等特定术语，并在“奖励”列中查找。避免仅根据奖励值对简单性做出假设。相反，应聚焦于直接回应主张各组成部分的明确数据。\n\n如果模块收到涉及根据训练方法（例如强化学习与学习得到的奖励）比较系统的主张，应侧重于识别表格数据中与训练方法直接相关的明确指标。例如，如果主张提到“学习奖励”或“未进行强化训练”，该模块应在“奖励”列中查找“学习得到的”或“未使用强化学习”等特定术语。避免仅凭奖励值就对简洁性作出假设。相反，应关注直接回应主张各个组成部分的明确数据。

表22：使用不同优化器的 ReAct 方法下 Qwen3-8B模型的基线指令和优化后的指令。

优化器	指令
基线	请根据所提供的表格数据核验给定的主张。
COPRO	你是一名智能体，负责根据所提供的表格核验一项主张。你将收到 以下输入：“主张”（待验证的陈述）、“表格”（包含行和列的数据集），以及 “标题”（对表格的描述）。你的目标是生成一个 Python 脚本，分析数据以验证 主张。该脚本应将所有相关信息打印到控制台，包括表格的标题和一份 表格数据样本。收集完所有必要信息并验证主张后，将任务标记为 已完成，并使用“finished=True”表示完成，同时提供清晰的最终答案。
MiPROv2	根据提供的表格数据核验给定的主张。你是一名智能体，负责评估 该主张的真实性，依据是表格中的信息。每个回合中，你都会收到以下字段 “主张”、“表格”和“标题”作为输入。\n\n你的目标是生成可执行的 Python 代码来处理 表格数据，以判断该主张是否得到支持、遭到反驳，或信息不足。对于 每次迭代，你都将生成一个代码片段，直接解决任务或逐步推进，最终得出 解决方案。确保将你希望从代码中提取的任何输出打印到控制台，并且代码应 应放在围栏代码块中。当确定该主张真伪所需的全部必要信息 均已可用时，标记‘finished=真‘以表示该过程已完成。\n\n你可以使用 Python 标准库，并可使用所提供的任何函数来分析表格数据。你的代码应精确，并 通过评估表格中提供的数据，直接针对该主张。执行代码后，结果将 用于判定该主张是得到支持、被反驳，还是无法根据给定信息加以验证。
SIMBA	根据所提供的表格数据验证给定的主张。你是一个智能体。对于每个回合，你 将接收字段‘claim‘、‘table‘、‘caption‘作为输入。你的目标是生成可执行的Python代码 以收集生成‘answer‘所需的任何必要信息。对于每次迭代，你将生成一段代码 片段，要么解决该任务，要么朝着解决方案推进。确保你希望提取的任何输出 都会从代码打印到控制台。代码应包含在围栏代码块中。当所有 用于生成输出（‘answer‘）的信息都已可供提取时，请在旁边标记‘finished=真‘ 最终的 Python 代码。\n你可以使用 Python 标准库和以下函数：\n\n如果 模块收到涉及表格数据中未直接提及的特定事件或情形的主张， 就应仔细分析表格内容，判断是否存在任何可以 支持或反驳该主张的间接或相关信息。如果表格未提供任何相关信息，模块应 得出结论：没有足够的信息来支持或反驳该主张。\n\n如果模块收到一项主张， 该主张涉及特定特征或嵌入时，应直接比较这些特征的相关指标 而不是对它们求平均。这种方法有助于发现可能反驳或支持 主张的细微差异。除非明确要求汇总整体性能，否则应避免使用平均值。\n\n如果 模块接收到主张和表格数据，应先将表格数据解析为结构化格式（例如字典列表），并确保所有变量在代码中使用之前都已正确定义。如果表格数据 不是可用的格式，模块应将其转换为结构化格式，并明确列出行和列 。此外，模块应处理任何潜在错误，例如未定义的变量，方法是 和列。此外，该模块应处理任何潜在错误，例如未定义的变量， 检查它们是否存在，并提供有意义的错误消息。这将防止得出错误的结论 这类结论基于错误的数据解析。\n\n如果模块收到的主张涉及特定特征或嵌入，它 应直接比较这些特征的相关指标，而不是对其取平均值。这种方法将 有助于识别可能反驳或支持该主张的细微差异。除非明确要求概括整体性能，否则应避免使用平均值 此外，应确保代码正确识别 并比较主张中提到的具体字段，例如本例中的’选区’和’buta’，以避免 对数据的误解。

表23：使用不同优化器的 CodeAct 方法下，Qwen3-8B模型的基线指令和优化后的指令。

优化器	指令
基线	根据提供的表格数据验证给定的主张。
COPRO	给定一个数据表，通过与其进行全面交叉比对，评估特定主张的真实性 根据表格中提供的信息，清晰地解释得出结论的推理过程。
MiPROv2	根据提供的表格数据核实给定的主张。
SIMBA	根据提供的表格数据核实给定的主张。如果模块收到的主张指定某种 针对给定条件的特定处理方法为“第一种”或“第二种”备选方案，则应仔细核对 表格数据，以确保主张中备选方案的顺序与表格一致。具体而言，如果主张称“X是 第一个备选方案，Y是第二个”，模块应验证表格将X列为“第一备选方案” 并将Y列为相关条件的“第二备选方案”。如果表格与主张的顺序相矛盾 ，模块 应返回“反驳”。\n\n如果模块收到指定特定条件的主张（例如，“英语和 欧洲议会语料库”），则应仅关注表格中与该条件匹配的行。然后应 比较该行中 DocSub 与 DF 的精确率值。如果 DocSub 的精确率低于 DF，且 DF 在该行中具有最高精确率，则该模块应返回“支持”。避免对所有行进行泛化 或误解精确率的相对值。\n\n如果该模块收到一条讨论某系统 训练方法（例如，“未经任何强化训练”）的主张，则应检查表格是否提供了明确的 有关该系统训练方法的信息。如果表格不包含此类信息， 该模块应返回“信息不足”，而不是做出无依据的推断。具体而言，如果 主张涉及系统的简洁性或训练方法，而表格只包含性能指标 （例如 ROUGE 分数），模块应避免对训练方法得出结论，而应 返回“信息不足”。\n\n如果模块收到的主张涉及合计同一俱乐部球员的积分， 则应计算各俱乐部的积分总和并进行比较。具体而言，如果主张称 “科隆鲨鱼的积分排名第一”，模块应计算科隆鲨鱼的积分总和 并将其与其他俱乐部的积分总和进行比较。如果科隆鲨鱼的积分总和不是最高的，模块 应返回“反驳”。当表格提供了足够的数据来评估 主张。

表 24：使用不同优化器进行直接提示时，Qwen3-32B模型的基线指令和优化后的指令

。

优化器	指令
基线	根据所提供的表格数据验证给定的主张。
COPRO	给定一项主张和一张数据表，评估该主张是*完全支持*、*部分支持*、 *矛盾*还是*无法确定*，依据表格中的信息。请提供简洁而全面的 评估说明。考虑所有相关数据点，并确保你的推理直接关联 回到表格的内容。
MiPROv2	通过分析所提供的表格及其标题，彻底审视给定的主张。使用逐步的逻辑 推理来判断该主张是得到支持、被反驳，还是表格中没有足够的信息 来验证它。注意具体细节，并确保结论基于上下文和数据准确无误。
SIMBA	根据给定的表格数据验证主张。如果主张涉及某种方法在特定阶段的有效性或性能 特定阶段中一种方法的，并且该表格包含针对该内容的评估指标（例如，准确率、百分比） 阶段的评估指标（例如准确率、百分比），那么模块应重点比较表格中的数值，以判断该主张是否得到支持 或遭到反驳。除非明确说明，否则不要假定表格缺少某些指标。如果表格显示，在某个阶段纳入某种方法后性能持续 提升，模块就应得出该 方法在该阶段有效的结论，并据此反驳认为它无效的主张。

表 25：使用不同优化器的 Qwen3-32B 模型思维链方法基线指令与优化后的指令。

优化器	指令
基线	根据提供的表格数据验证给定的主张。
COPRO	提出的指令如下：\n\n**指令：**你是一名负责确定.....的推理智能体，判断给定的‘主张’是否得到所提供的‘表格’及其上下文‘标题’中的数据支持、不准确或无法确定。你将分析表格的结构、内容和上下文，并编写一条有效的 SQL 查询，使用‘execute_sql’工具提取或验证相关证据，并使用检索到的信息进行基于证据的评估。\n\n请遵循以下结构化方法：\n1. 首先，仔细阅读‘主张’、‘表格’和‘标题’，以了解上下文及表格所代表的内容。\n2. 设计一条 SQL 查询，通过执行‘execute_sql’工具直接从‘表格’中提取相关数据，并附上针对该主张的‘table_name‘和‘sql_query‘。\n3. 仔细检查查询结果，并结合该主张进行评估。\n4. 逐步推理你的发现，并明确引用从查询和表格中得出的证据。\n5. 一旦你掌握了充分的信息，并对该主张是得到数据支持还是与之矛盾形成了确定性理解，请使用‘finish‘工具给出你的最终答案：‘支持‘、‘不支持‘或‘证据不足‘。\n\n你的答案应基于数据、论证充分，并具有清晰的推理。
MiPROv2	请仔细使用结构化表格数据核实所提供主张的准确率。你是一名智能体，负责推理该主张，并判断它是否得到支持、遭到反驳，或者是否缺少足够的信息来作出判断。必要时，使用‘execute_sql‘工具查询表格，以获取更深入的见解；并在确信核实结果后，使用‘finish‘工具。推理时始终以数据为依据，避免作出表格数据并未直接支持的假设。
SIMBA	根据提供的表格数据验证给定的主张。\n\n你是一个智能体。在每个回合中，你将获得字段‘claim’、‘table’、‘caption’作为输入。你还可以看到截至目前的历史轨迹。\n你的目标是使用一个或多个提供的工具，收集生成‘答案’所需的任何信息。\n\n为此，你需要在每个回合中交替给出下一步_思考、下一个_工具_名称和下一个_工具_参数，完成任务时也要如此。完成任务。\n每次工具调用后，你都会收到相应的观察结果，并将其附加到你的轨迹。\n\n编写下一步_思考时，你可以推理当前情况并规划未来步骤。\n选择下一个_工具_名称及其下一步_工具_参数时，该工具必须是以下之一：\n\n(1) 执行_sql，其描述为 <desc>在以列表的列表格式提供的表格上执行 SQL 查询。参数：table_数据：列表的列表，其中第一行为表头，后续行为数据。表格_名称：表格名称用于执行 SQL 查询的数据框。sql_查询：要执行的 SQL 查询字符串。返回：转换后的表格或错误消息的格式化字符串 </desc>。它接受参数 {'table_data': {'items': {}, 'type': 'array'}, '表格_名称': {'type': '字符串'}, 'SQL_查询': {'type': '字符串'}}.\n(2) finish，其描述为 <desc>标记该任务已完成。也就是说，这表示生成输出所需的所有信息，即“答案”，现在都可以提取。</desc>。它接受参数 {}。\n提供 ‘next_工具_args‘ 时，字段的值必须采用 JSON 格式\n\n如果表格数据明确包含验证主张所需的信息，模块应立即调用 ‘finish‘ 工具，而不执行不必要的 SQL 查询。具体来说，如果主张涉及将 URL 与组织名称进行匹配，且表格直接列出了这些关联，模块应识别这一点，并在无需进一步工具调用的情况下结束任务。避免将验证过程复杂化：当数据已经清晰且易于获取时，不要执行多次查询。\n\n如果表格数据明确包含验证所需的信息，模块应避免执行不必要的 SQL 查询。具体而言，如果主张涉及比较准确率值，而表格直接列出了这些值，模块应直接从表格中提取相关数据，而无需构造复杂的 SQL 查询。此外，模块不应假设准确率的边际提升就足以支持关于纳入新概念（例如词义层级信息）的主张，除非数据明确支持这一结论。\n\n如果主张涉及特定指标（例如准确率）和特定设置（例如，传导式），模块应核实表格是否明确报告了该指标和设置。如果表格仅提供相关指标（例如 F 分数）或未提及该设置，模块不应假定二者等价，而应得出结论：数据并不直接支持该主张。

表 26：使用不同优化器时，采用 ReAct 方法的 Qwen3-32B模型的基线和优化后的指令。

优化器	指令
基线	请根据提供的表格数据验证给定的主张。
COPRO	你将担任数据验证助手。你的任务是评估提供的“主张”是否得到支持 根据‘表格’（一种具有键值结构的字典）中的数据，该表格附有‘标题’（对表格内容的简要 说明）。生成简洁实用的 Python 代码，根据表格验证主张。 代码应打印相关结果，以便判断该主张是否属实。当评估主张所需的数据已全部提取或任务已完成时， 请以‘finished=真’作为结束标记，并 将你的发现输出到答案字段中。
MiPROv2	通过生成用于解读所提供表格数据的Python代码，验证给定主张的准确率，包括 检查数值趋势、比较以及表格标题中的上下文细节。作为一名 智能体，你将收到一项主张、一个表格和一个标题作为输入。你的任务是生成Python代码，该代码要么 直接回答该主张，要么帮助收集信息以支持最终答案。代码应当写成 迭代式的，并应输出相关信息以供评估。当该主张能够被明确 回答时，请在代码旁用‘finished=真‘标明。你可以使用Python 标准库 以及支持分析所需的任何相关函数。请确保输出清晰且以结构化 格式打印。
SIMBA	根据提供的表格数据验证给定的主张。\n\n你是一个智能体。每个回合中，你将 收到作为输入的字段‘主张’、‘表格’和‘标题’。\n你的目标是生成可执行的 Python 代码，以 收集生成‘答案’所需的任何信息。每次迭代中，你将生成一个代码片段 要么解决该任务，要么朝着解决方案推进。确保你希望提取的任何输出都会从 代码中打印到控制台。代码应放在围栏代码块中。当所有用于 生成输出（“答案”）的信息都可提取时，在最终 Python 旁标记“finished=真” 代码。你可以使用 Python 标准库以及以下函数：如果该模块收到 关于分类器方法的主张（例如，“我们使用基于规则的分类器”），它应生成代码 以打印表格数据和标题供检查，且不对这种关系做出假设 在数据与主张之间。代码应经过结构化处理，确保数据清晰呈现，以便 提取器进行解读。\n\n如果模块收到关于总体趋势的主张（例如，“阿片类药物处方 在 2010—2015 年间有所下降”），且表格包含各项指标变化所涉及县的百分比， 则应生成代码，打印原始表格数据和标题以供检查。避免尝试直接处理 或分析数据，因为表格可能未提供验证主张所需的信息。确保 代码在语法上正确，方法是正确转义换行符并使用三引号字符串 如有需要。如果该模块收到关于两个事件之间时间顺序关系的主张，并且 表格包含这些事件发生的年份，则应生成能够正确解析年份并检查 时间顺序的代码。通过正确缩进所有代码块，确保代码在语法上正确， 尤其是在 ’if’ 和 ’else’ 等控制流语句之后。如果该模块收到关于总体 趋势的主张，并且表格包含各县在不同指标上发生变化的县的百分比，则应生成代码 以打印原始表格数据和标题供检查。避免尝试处理或分析数据 直接方法，因为表格可能不包含验证主张所需的信息。请确保代码 在语法上正确，方法是正确转义换行符，并在需要时使用三引号字符串。

表27: 使用不同优化器的 CodeAct 方法下，Qwen3-32B 模型的基线指令和优化后的指令 us
优化器。

优化器	指令
基线	请根据提供的表格数据核查给定的主张。
COPRO	你是一名严谨可靠的事实核查助手。系统会向你提供一条主张和一份数据表。 你的任务是根据数据严谨地评估该主张，并明确将其分类为“支持”、 “矛盾”或“无法确定”。请简明扼要地解释分类依据，并引用具体的数据 点作为无可辩驳的证据。如果该主张被判定为“无法确定”，请清楚说明 得出确切评估所需的具体信息。请以清晰、 准确和简洁为优先组织回答，确保读者能够顺畅理解你的推理。
MiPROv2	你是一名事实核查助手。给定一项主张、一张表格以及描述该表格的标题，确定该 主张是否被表格反驳。如果该主张与表格中的信息矛盾，则返回 \"refutes\"；如果该主张得到表格支持，则返回 \"supports\" 否则返回其他结果。重点比较主张中的断言与表格中呈现的数据， 并结合表格标题理解上下文。
SIMBA	根据提供的表格数据核实给定的主张。\n\n在评估涉及数值的主张时 与表格数据进行比较时，请仔细提取表格中的相关数值。具体而言， 当主张涉及比较两个数值时，请确保比较的是正确的数值（例如，比较 「氢可酮」和「氢吗啡酮」的「换算因子」）。如果一个数值大于另一个，而 主张却断言相反，则将该主张归类为「反驳」。优先进行直接数值比较，而不要受任何 可能对表格上下文产生的误读影响。在评估涉及与表格数据进行数值比较的主张时， 应优先进行直接数值比较。具体而言，提取相关数值 （本例中为死亡人数和住院人数），并对其进行比较。如果主张指出两者存在差异，而这些数字 支持这一差异，则归类为“支持”。如果主张指出的数值差异与实际情况相反， 则归类为“反驳”。请仔细留意主张的措辞（例如“低得多”“较高”“相等”）， 以确保比较准确。评估有关出席人数变化的声明时，直接比较 指定日期的出席人数。如果第二个日期的出席人数明显低于 第一个日期，则将该主张归类为“支持”。请留意主张的措辞（例如“显著下降”） 并确保数值差异与该表述一致。数值比较优先于任何其他 上下文因素。\n\n评估有关修复的影像资料的主张时，请仔细查看相关故事和剧集对应的“有修复影像资料的缺失剧集” 与相关故事和回合对应的“修复的影像资料”列。如果表格显示该故事的第4集码中特。 编号为澳大利亚032的第4集有修复的影像资料，则将该主张归类为“支持”。优先考虑主张与 表格数据之间的直接匹配，尤其是在主张明确提及具体剧集和故事编号时。

表28：使用多种优化器进行直接提示时，Gemma3-12B模型的基线指令和优化后的指令。

优化器	指令
基线	请根据提供的表格数据核查给定的主张。
COPRO	你是一名专门从事数据驱动验证的资深事实核查员。你的任务是严谨地评估一项 根据给定表格中呈现的信息，对给定主张进行判断。确定该主张是否明确 支持、明确矛盾，还是表格中没有与该主张相关的信息（未 提及）。请提供简洁而全面的判断理由，并始终引用表格中的具体数据 点来支撑你的推理。解释时应优先保证清晰和精确。
MiPROv2	你是一名主张验证专家。请分析所提供的主张及其附带的表格，判断该主张是否 主张是支持、反驳，还是信息不足以验证。请提供详细的分步 推理过程，解释你如何得出结论。请明确说明最终答案为\"支持，\" \"反驳，\"或\"信息不足。
SIMBA	根据所提供的表格数据验证给定的主张。\n\n验证主张与表格是否相符时，请仔细 检查表格内容和单位。不要尝试进行未明确 在表格中提供。如果表格缺少直接支持或反驳该主张所需的必要信息，请回答 以“信息不足”。避免做出假设或进行并非直接得到支持的计算 由数据。\n\n在基于表格数据分析有关性能提升的声明时，尤其是当 表格包含 ROUGE 分数时，请仔细考虑任何附带的标题或描述。如果标题 表明所评估的系统优化的是与被比较指标不同的指标（例如， 针对学习奖励而非 ROUGE 进行优化），不要假定更高的 ROUGE 分数会自动 意味着性能提升。相反，应将该系统的得分与其他系统的得分进行比较，并考虑 优化策略。如果表格中没有明确证据表明其性能优于其他系统， 则回答“信息不足”。\n\n分析基于表格数据的准确率主张时，首先要找出所有 与该主张相关的模型（例如，使用 TVMAX 的模型）。然后，系统地比较这些模型的“测试集标准 总体”得分（或相关的准确率指标），以确定哪一个模型得分最高。 在完成这一比较之前，不要过早断定没有模型符合该主张的标准。 比较。分析有关能源价格的主张时，请重点关注表格中提供的数值。 该主张可能指的是价格本身，而不是价格的数量。请仔细比较主张中提到的相关价格 与表格中的数值，忽略时间或百分比变化等无关细节。如果 主张要求比较两个特定项目（例如，西德克萨斯中质原油和天然气），则在作出判定时只考虑这些 项目。

表29：使用各种优化器和思维链方法的 Gemma3-12B 模型的基线指令和优化后的指令。

优化器	指令
基线	根据提供的表格数据核查给定的主张。
COPRO	你是一个声明验证代理。你会收到一个主张、一张表格和一个标题。你的任务是判断是否 主张在表格中得到支持、存在矛盾，或未被提及。分析表格数据，并考虑标题 作为上下文。请提供简洁的答案，表明该主张是得到支持、存在矛盾，还是未被提及。 如果该主张得到支持或存在矛盾，请简要说明表格数据如何支持该主张或与该主张相矛盾。
MiPROv2	对照所提供的表格数据验证给定主张。\n\n你是一个智能体。在每个回合中，你将 获得字段‘claim‘、‘table‘、‘caption‘作为输入。并且你可以看到迄今为止的过往轨迹。\n你的目标是 使用一个或多个所提供的工具来收集生成“答案”所需的任何信息。要 执行此操作时，你将在每一轮中交替输出 next_思考、next_工具_名称和 next_工具_参数，并且在 完成任务时也是如此。每次工具调用后，你都会收到一个结果观察结果，该观察结果会追加到你的 轨迹中。编写 next_思考时，你可以分析当前情况并规划接下来的 步骤。选择 next_工具_名称及其 next_工具_参数时，工具必须是以下工具之一：(1) execute_sql, 其描述为 <desc>在以列表的列表格式提供的表格上执行 SQL 查询。参数：table_数据： 列表的列表，其中第一行包含表头，后续行包含表格数据_name：表格名称 用于执行 SQL 查询的数据框 sql_query：要执行的 SQL 查询字符串。返回：格式化后的字符串，内容为 转换后的表格或错误消息 </desc>。它接受以下参数：{’table_data’: {’items’: {}, ’type’: ’array’}, ’table_name’: {’type’: ’string’}, ’sql_query’: {’type’: ’string’}}。\n(2) finish，其描述为 <desc>将任务标记为 已完成。也就是说，这表示生成输出所需的所有信息，即‘答案’，现已 可供提取。</desc>。它接受的参数为 {}。\n提供‘next_tool_args’时，其中的值 字段必须采用 JSON 格式
SIMBA	根据给定的表格数据验证所提供的主张。\n\n你是一个智能体。在每个回合中，你将获得 字段‘claim’、‘table’、‘caption’作为输入。并且你可以查看截至目前的过往轨迹。\n你的目标是 使用一个或多个提供的工具收集生成‘答案’所需的任何信息。\n\n为 此，你需要在每个回合中交替输出 next_thought、next_tool_name 和 next_tool_args，并且在 完成任务。\n每次工具调用后，你都会收到相应的观察结果，并将其追加到你的 轨迹中。\n\n撰写下一条_思考时，你可以分析当前情况并规划后续 步骤。\n选择下一个_工具_名称及其下一组_工具_参数时，工具必须是以下之一：\n\n(1) execute_sql, 其描述为 <desc>在以列表的列表格式提供的表格上执行 SQL 查询。参数：table_data： 列表的列表，首行为表头，后续行为表格数据。table_name：表格名称 用于执行 SQL 查询的数据框。sql_query：要执行的 SQL 查询字符串。返回：格式化后的字符串，内容为 转换后的表格或错误消息 </desc>。它接受以下参数：{’table_data’: {’items’: {}, ’type’: ’array’}, ’table_name’: {’type’: ’string’}, ’sql_query’: {’type’: ’string’}}。\n(2) finish，其描述为 <desc>将任务标记为 已完成。也就是说，这表示生成输出（即‘答案’）所需的所有信息现在都已 可供提取。</desc>。它接受参数 {}。\n提供‘next_tool_args’时，字段中的值必须 采用 JSON 格式\n\n尝试使用‘execute_sql’工具时，请仔细查看该工具的 文档，确保‘table_data’的格式为列表的列表，其中第一个内部列表表示 表头行及其后的列表表示数据行。调用工具之前，请仔细检查 ‘table_data‘的结构，确认其符合此格式。如果多次尝试执行 SQL 查询均失败，请放弃 使用该工具，转而依靠对初始输入中所提供表格数据的直接分析。

表 30：使用不同优化器的 ReAct 方法下，Gemma3-12B 模型的基线指令和优化后的指令。

优化器	指令
基线	根据所提供的表格数据验证给定的主张。
COPRO	你是一名主张验证器，擅长对照表格数据专业分析声明。给定一项主张、一个表格和一个标题， 你必须判定该主张是“真”、“假”还是“信息不足”。请注重精确率和准确率。 请考虑所有相关的列和行。你的回答*必须*包含推理过程和最终 判定。请先简要概述你的策略，然后分析数据，最后给出结论。 只有经过全面分析后，才能给出判定。
MiPROv2	你是一名专业的数据分析师，负责根据表格数据核实主张。你将收到一项主张、一份 表格及其标题。你的目标是分析表格，并判断该主张是得到支持、遭到反驳， 还是信息不足，无法验证。\n\n为此，你将首先生成 Python代码， 以提取并分析表格中的相关数据。生成的代码应打印到控制台 中的围栏代码块内。这段代码可能涉及解析表格、计算统计数据或进行 比较。\n\n生成代码后，执行代码并查看输出。根据输出以及你对主张和表格的 理解，判断该主张是得到支持、遭到反驳，还是信息不足，无法验证。 足够的信息。\n\n最后，连同最终的 Python 代码一起标记为‘finished=True‘。确保所有必要的 信息均已提取后，再标记为‘finished=True‘。\n\n你可以使用 Python 标准库。
SIMBA	根据提供的表格数据核验给定的主张。\n\n你是一个智能体。每个回合中，你将 收到‘主张’、‘表格’、‘标题’字段作为输入。\n你的目标是生成可执行的 Python 代码，以 收集生成‘答案’所需的任何信息。\n每次迭代中，你将生成一个代码片段 能够解决任务或推动解决方案取得进展。\n确保将你想要提取的任何输出 代码打印到控制台。代码应放在围栏代码块中。\n当所有用于生成输出（‘答案‘）的信息 生成输出（‘答案’）可供提取时，在最终 Python 旁标记‘finished=真’ 代码旁标记 ‘finished{v1}True‘。\n你可以使用 Python 标准库和以下函数：\n\n收到 ‘表格‘输入时，该输入是包含表格数据的字符串，请使用 ‘pandas‘ 程序库直接从字符串中解析表格数据 。具体来说，使用 ‘StringIO‘ 和 ‘pd.read_csv‘ 将表格字符串读入 pandas 数据框。 这样，您就可以使用 DataFrame 索引访问表格数据，并避免出现「名称错误」 这种错误会在尝试访问未定义变量时发生。

表 31：使用不同优化器的 CodeAct 方法下，Gemma3-12B 模型的基线指令和优化后的指令。

优化器	指令
基线	根据提供的表格数据核实给定的主张。
COPRO	你是一名事实核查助手。你的任务是判定给定的主张是否得到支持、遭到反驳或未找到。 在所提供的数据表格中。\n\n请按以下步骤操作：\n\n1. **仔细阅读主张。** 理解 它的确切含义。\n2. **分析数据表格。** 找出相关的行和列，以评估该主张。\n3. **确定关系：**\n * **支持：** 如果表格中的数据直接证实该主张。\n * **反驳：** 如果表格中的数据直接否定该主张。\n * **未找到：** 如果表格没有 包含与该主张相关的信息，或者信息不足以核实该主张。\n4. **简洁地作答 ，且*仅*使用以下标签之一：**“支持”、“反驳”或“未找到”。不要附上解释 或其他文本。
MiPROv2	你是事实核查专家。给定一个主张、一张表格以及描述该表格的标题，请判断表格是否支持该主张， 表格是否反驳该主张，或者表格提供的信息是否不足以作出 判定。\n\n请仔细分析主张和表格内容。注意单位、具体数据点， 以及整体背景。你的回答应为以下其中一项：’支持’、’反驳’或’信息不足 信息”。示例：主张：澳大利亚的故事编号032中，第4集是缺失剧集 且有修复的影像资料表格：‖博士	季	故事编号	连续剧	集数	缺失剧集剩余影像总时长 （分：秒）| 有修复影像资料的缺失剧集	国家/地区	来源	格式 | 影像总时长（分：秒）‖...标题：神秘博士缺失剧集答案：支持现在，请运用你的 专业知识来处理给定的主张、表格和标题。
SIMBA	根据提供的表格数据验证给定的主张。\n\n看到一项主张和一张表格时，重点关注 直接将表格中的数值与主张中的陈述进行比较。在本例中，该主张称“添加 逻辑值”并非表现最佳。表格显示，“添加逻辑值”的 F1 值为 80.85%，“添加逻辑值+专家”的 F1 值为 80.90%，而其他方法的 F1 分数较低或相近。因此，该主张被反驳， 因为添加逻辑值确实表现良好，并非最差。如果遇到类似主张，请明确 检查表格数据是否与主张中的断言相矛盾。\n\n评估有关有效性的主张时 对于基于表格数据的方法，应优先识别与该方法相关的性能指标。 如果这些指标表明性能良好（例如，成功率高、错误率低），那么关于该方法 「*无效*」的主张很可能会被反驳。相反，如果这些指标显示性能不佳，那么该主张就 会得到支持。请密切关注表格中的比较值，以确定该方法是否优于 其他方法，即使绝对值并不特别高。\n\n当你收到一项主张和一张包含 数值数据的表格时，请明确比较主张中的数字与表格中的数字。如果主张中的数字 表格明确与主张相矛盾（例如，主张称数值处于某个范围内，但表格显示 数值超出该范围），则输出“反驳”。如果可以进行数值比较，不要默认输出“信息不足”。 \n\n当你收到一项主张和一张包含数值数据的表格时，*务必*仔细阅读标题， 以理解这些数字代表什么，以及它们与主张有何关联。如果标题表明这些 数字无法与主张中的断言直接比较（例如，优化的是不同的指标），或者 主张涉及表格中未呈现的因素，则输出“信息不足”。不要试图作出 如果上下文表明仅依据数值比较进行判断并不合适。

表32：使用不同优化器进行直接提示时，Gemma3-27B模型的基线指令和优化后的指令。

优化器	指令
基线	根据提供的表格数据核实给定的主张。
COPRO	你是一位一丝不苟的事实核查员。你的任务是确定给定的主张是否得到以下数据的支持 所提供的表格。\n\n首先，仔细分析主张，并找出验证该主张所需的关键信息 。然后，查看表格以查找相关数据。\n\n根据这一比较，从以下选项中选择一个作答： \n\n* **支持：**如果表格数据直接支持该主张。\n* **矛盾：**如果 表格数据直接与该主张矛盾。\n* **信息不足：**如果表格不包含 足够的信息来验证该主张。\n\n请在给出最终答案*之前*陈述推理过程。说明你如何 通过引用表格中的具体数据点得出结论。
MiPROv2	你是一名技术娴熟的事实核查员，负责确保以下内容中关键信息的准确性 公共卫生公告。一旦分类错误，可能造成严重后果。给定一项主张、一张数据表， 以及一段描述该表格的标题，你的任务是仔细对照数据核实该主张。请提供 逐步推理过程，清晰说明你是如何得出结论的。然后明确指出 根据表格，该主张是“支持”、“反驳”还是“信息不足”。准确性至关重要。
SIMBA	根据提供的表格数据验证给定的主张。\n\n在分析有关统计显著性的主张时， 应优先确认表格是否明确标示了相关模型和条件的统计显著性（例如，使用星号或其他 标记）。如果表格显示，在预言机设置下列为「我们的模型」的模型取得了具有统计显著性的改进 即使没有将其与同一模型在非预言机设置下的结果直接比较， 「我们的模型未能取得统计显著性更高的 BLEU分数」这一主张 很可能被反驳。在分析声明时，*严格*关注表格是否包含直接 与主张相关的信息。避免基于相关数据（如准确率 本例中的分数）进行推断或得出结论，如果主张涉及不同方面（如词义层级信息的使用）。如果 表格并未明确针对该主张，则默认采用“信息不足”，即使你能够就相关 概念进行推理。在分析声明时，严格遵循表格中呈现的信息。如果表格 定义了范围或边界（如“低于17%”），避免仅仅将主张解释为“不准确” 因为某个数值落在该范围内。相反，应关注表格是否*直接*支持、反驳主张，或缺少 有关该主张的信息。如果表格仅提供定义或分类，却没有说明 主张是否属实，则默认判定为“信息不足”。

表33：使用不同优化器的思维链方法下，Gemma3-27B 模型的基线指令和优化后的指令

。

优化器	指令
基线	请根据提供的表格数据核查给定的主张。
COPRO	你是一名准确率极高的声明验证代理。你的任务是判断给定的主张是否得到支持根据所提供表格中的数据。你将获得一条「主张」、一个「表格」（采用包含表头行的列表的列表格式），以及描述该表格的「标题」。\n\n请按照以下步骤操作：\n\n1. **理解主张：** 仔细阅读「主张」，找出其中涉及的关键实体以及所断言的关系。\n2. **分析表格：** 检查「表格」的结构、表头和数据，以了解其中包含的信息。参考「标题」获取有用的背景信息。\n3. **构造 SQL 查询：** 构造一条针对该「表格」执行后即可直接回答「主张」所提出问题的 SQL 查询。SQL 查询应尽可能简单直接尽可能。\n4. **执行 SQL 查询：** 使用 ‘execute_sql‘ 工具，并提供 ‘table_data‘、所选的 ‘table_name‘（例如，\"my_table\"）以及编写的 ‘sql_query‘。\n5. **解读结果：** 分析 ‘execute_sql‘ 工具返回的结果。\n6. **验证主张：** 根据查询结果，判断该主张是得到支持（真）、相互矛盾（假），还是表格中的信息不足以作出判断（未知）。\n7. **完成：** 使用 ‘finish‘ 工具表示任务已完成。\n\n你的回答应当准确、简洁，并且仅依据 ‘表格‘ 中提供的信息。优先使用 SQL 直接针对主张核心断言的查询。
MiPROv2	你是一名事实核查代理，负责根据表格数据验证声明。你将获得一项主张，一张表格和一个表格标题。你的目标是判断该表格是支持、反驳该主张，还是提供的信息不足以验证该主张。\n\n**操作指南：**\n\n1. **推理：**仔细阅读主张、表格标题和表格数据。制定一个清晰的计划来确定答案。考虑可能的诸如单位、数据类型和模糊语言等问题。\n2. **工具使用（execute_sql）：** 如果表格很复杂或需要进行特定的数据提取，请使用‘execute_sql‘工具查询表格。编写能够直接回应主张的 SQL 查询。请记得处理可能出现的错误（例如，无效的列名、数据类型问题）。如果 SQL 查询失败，请改用手动检查。\n3. **手动检查：** 如果‘execute_sql‘失败或没有必要使用，请手动检查表格以查找相关信息。\n4. **最终判断：** 根据你的推理和提取的证据，判断表格是*支持*主张、*反驳*主张，还是提供*信息不足*，无法进行判定。5. **输出：**请逐步说明你的推理，然后给出最终答案，答案只能是“支持”、“反驳”或“信息不足”。**重要注意事项：**优先考虑准确率。仔细核查你的推理和证据。* 请留意表格标题；它通常会提供关键背景信息。* 如果主张涉及比较，请确保比较的是正确的数据点。* 如果表格不包含回答该主张所需的信息，请选择“信息不足”。
SIMBA	根据提供的表格数据核验给定的主张。\n\n你是一个智能体。在每个回合中，你将会获得“主张”“表格”“标题”字段作为输入，并且可以查看截至目前的过往轨迹。\n你的目标是使用一个或多个提供的工具收集必要的信息，以生成“答案”。\n\n为此，你将在每一轮中交替输出 next_thought、next_tool_name 和 next_tool_args，完成任务时也一样。任务结束时也是如此。\n每次工具调用后，你都会收到相应的观察结果，并将其追加到你的轨迹中。\n\n编写 next_thought 时，你可以分析当前情况并规划后续步骤。\n选择下一个_工具_名称及其下一个_工具_参数时，工具必须是以下之一：\n\n(1) execute_sql，其描述为 <desc>以列表的列表格式对给定表格执行 SQL 查询。参数：table_data：列表的列表，其中第一行包含表头，后续行包含表格数据 table_name：表格名称用于执行 SQL 查询的数据框 sql_query：要执行的 SQL 查询字符串。返回：转换后的表格的格式化字符串或错误消息 </desc>。它接受以下参数：{’table_data’: {’items’: {}, ’type’: ’array’}, ’table_name’: {’type’: ’string’}, ’sql_query’: {’type’: ’string’}}。\n(2) finish，其描述为 <desc>标记任务已完成。也就是说，这表示生成输出（即“答案”）所需的所有信息现在均可提取。</desc>。它接受 {} 个参数。\n提供“next_tool_args”时，字段中的值必须为 JSON 格式\n\n当直接以列表的列表形式收到表格数据时，请勿尝试使用“execute_sql”工具。请直接分析所提供的数据来回答问题。优先直接分析所提供的数据，然后再考虑使用 SQL 查询，尤其是在初次 SQL 尝试失败时。\n\n当表格数据以列表的列表形式直接提供时，应优先直接分析数据，而不是尝试使用 SQL。SQL 适用于大型数据集上的复杂查询，但对于直接提供的小型表格，直接方法分析效率更高，也能避免潜在错误。查看输入中是否包含“列表的列表”或“表格”等关键词，以判断是否适合直接分析。\n\n遇到优势比（OR）为 1.00 时，尤其是在涉及基线百分比主张的情境中，应考虑明确说明基线率等于声称的百分比这一假设。不要仅仅因为表格没有直接说明百分比，就立即断定“信息不足”；相反，应结合 OR 值探索合理的解读与主张相关。分析表格时，应优先关注与主张最直接相关的值。在本例中，主张询问的是急性期的灵敏度*百分比*。虽然表格中出现了 98%，但那是特异度值，而不是灵敏度值。请查看“灵敏度 (%)”列和“急性期”行，以找到正确的值（17%）。不要被表格中与问题没有直接关系的其他数字分散注意力。如果输入的“表格”以列表的列表形式提供（而不是数据库连接或类似形式），应优先直接分析数据，而不是尝试使用 SQL。SQL 更适用于大型数据集或复杂查询，但对于直接提供的小型表格，直接分析更加高效，且可以避免潜在错误。查看输入中是否包含“列表的列表”或“表格”等关键词，以判断直接分析是否合适。如果 SQL 查询反复失败，应立即切换到直接数据分析。

表34：使用不同优化器的 ReAct 方法下 Gemma3-27B 模型的基线指令和优化后的指令。

优化器	指令
基线	根据提供的表格数据验证给定的主张。
COPRO	你是一名技艺高超的智能体，专门根据表格中呈现的数据验证声明。你将获得一‘主张‘、一个‘表格‘（以字符串表示）以及描述该表格的‘标题‘。你的任务是判断该主张是否得到表格中信息的支持。\n\n你将按以下步骤进行：\n\n1. **理解表格：**解析表格字符串，以理解其结构（行、列、表头）。\n2. **提取相关数据：**识别表格中与‘主张‘相关的数据。\n3. **验证主张：**比较‘主张‘中的信息与所提取的数据，以判断该主张为真、假或未知（如果表格未包含足够的信息来验证该主张）。\n4. **生成Python代码：**编写Python用于完成步骤 1-3 的代码。代码必须打印最终答案（‘TRUE’、‘FALSE’ 或 ‘UNKNOWN’）到标准输出。重点提取并比较回答问题所需的关键信息。\n5. **标记完成：**在最终代码块之后包含 ‘finished=True’，以便提取答案。\n\n输出中不要包含任何对话内容或解释；只需Python代码和 ‘finished=True’ 标记。
MiPROv2	你是一位技艺精湛的数据分析师，负责根据表格数据核验主张。你将收到一个“主张”（需要核验的陈述）、一个“表格”（其中的数据以分隔字符串格式呈现）以及一个“标题”（用于描述表格）。你的目标是判断该“主张”是否得到支持、遭到反驳，或是该“表格”提供的信息不足（“信息不足”）。\n\n你必须生成 Python 代码，使用“pandas”程序库解析“表格”数据。表格数据以“|”为分隔符。从表格中提取相关信息以评估主张。确保代码能够妥善处理潜在错误，例如数据缺失或用于评估主张的表格。确保你的代码能够妥善处理潜在错误，例如数据缺失或数据类型不正确。打印最终的布尔结果（真或假），表示该主张是否得到支持表格数据，并将结果输出到控制台。\n\n你的回答必须遵循以下步骤：\n\n1. **解析表格：**使用‘pandas’读取所提供字符串中的表格数据。使用‘sep='|'‘正确解析数据。清理数据，移除列名前后以及单元格值前后的空白。\n2. **分析主张：**仔细阅读并理解主张。确定需要从表格中获取哪些具体数据点来验证该主张。\n3. **提取并比较：**从‘pandas’数据框中提取相关数据点。执行根据表格数据进行必要的比较或计算，以确定主张为真还是假。\n4. **打印结果：**将‘真’或‘假’打印到控制台，以表示分析结果。\n5. **标记完成：**打印结果后，添加‘Finished: True’以表示任务已完成。\n\n你只能将布尔值（真或假）和‘Finished: True’打印到控制台；不得输出其他文字或解释。
SIMBA	根据提供的表格数据验证给定的主张。\n\n你是一个智能体。对于每个回合，你将收到字段“claim”、“table”、“caption”作为输入。\n你的目标是生成可执行的 Python 代码，以收集生成“answer”所需的任何信息。\n每次迭代中，你将生成一个代码片段，用于解决任务或推进解决进程。\n请确保将你想要提取的任何输出通过代码打印到控制台。代码应放在围栏代码块中。\n当生成输出（“answer”）所需的所有信息都已收集完毕，可以提取时，请在最终 Python 代码旁标记“finished=True”代码。\n你可以使用 Python 标准库和以下函数：\n\n生成代码时要解析表格，请特别注意语法，尤其是缩进。确保所有“else”语句都有对应的缩进代码块。在完成代码之前，可以考虑添加一个简单的测试用例，以验证解析逻辑能否在少量表格数据上正常运行。此外，还可以考虑添加错误处理，以捕获潜在的解析问题，并提供有用的错误消息。\n\n解析表格时，请格外注意分隔符。该表格使用 '|' 作为分隔符，但你最初使用的是 '\\|'。请确保分隔符与表格的格式完全一致。解析后还要仔细检查你尝试访问的列名该表格，以避免出现键错误。可以考虑添加一个小型测试用例，使用部分表格数据验证解析逻辑，然后再进行完整分析。\n\n解析表格时，请仔细检查分隔符在‘pd.read_csv‘中使用的分隔符。该表格使用 '|' 作为分隔符。请确保已正确指定。此外，创建数据框后，请验证列数和分配的列名，确保它们与表格结构一致。在继续进行之前，添加一个小型测试用例，使用部分表格数据验证解析逻辑完整分析。如果解析失败，则返回一条简单的错误消息，而不是崩溃。生成 Python 代码时，请极其注意缩进。不正确的缩进是错误的常见来源。在最终确定代码之前，使用代码检查工具或在本地运行代码以捕获这些错误。具体而言，确保所有代码块紧跟控制流语句（例如‘if‘、‘else‘、‘try‘、‘except‘）之后的部分都已正确缩进。

表 35：使用各种优化器的 Gemma3-27B 模型在 CodeAct 方法下的基线与优化后的指令。

系统提示（两类）

你需要使用 SQL、Python 或直接给出答案，逐步根据表格核查声明。

规则：1. 每一步：选择操作_类型 = SQL | Python | 答案。2. SQL/Python → 将可执行代码填入“code”。绝不要在“code”中放入标签。3. 答案 → 将“label”准确填写为“supported”或“refuted”。4. 表格名称为 DF。列名中包含空格时，使用反引号括起来：‘Col Name‘。5. 执行器会运行你的代码并返回一个中间表。6. 代码应简短（<300 字符）。在最新的中间表上进行操作。

重要："代码"= 仅限可执行的 SQL/Python。"标签" =仅限最终裁决。切勿将二者混合。

图 1：用于两类表格事实核查任务的系统提示。

系统提示（三类）

你需要逐步对照表格核验声明，使用 SQL、Python 或直接方法给出答案。

规则：1. 每一步：选择操作_类型 = SQL | Python |答案。2. SQL/Python → 将可执行代码填入“code”。绝不要在“code”中放入标签。3. 答案 → 将“label”准确填写为“supported”、“refuted”或“not verifiable”。4. 表格名称为 DF。列名中包含空格时，使用反引号括起来：‘Col Name‘。5. 执行器会运行你的代码并返回一个中间表。6. 代码应简短（<300字符）。在最新的中间表上进行操作。

标签：
- “支持”：表格确认该主张。- “反驳”：表格与该主张相矛盾。- “无法验证”：表格缺少相关信息（仅在检查数据之后）。

重要：“代码”= 只能包含可执行的 SQL/Python。“标签” = 只能包含最终判断。切勿混用。

图 2：用于三类基于表格的事实核查任务的系统提示。

任务提示（两类）

数据库表格 DF 如下所示：表格标题：{table caption} {table}

请根据上述数据回答以下问题：“{claim}” 给定表格是否支持或反驳上述主张？请根据问题和表格逐步生成 SQL 或 Python 代码，以正确回答问题。每一步都生成 SQL 代码来处理查询，或生成 Python 代码来重新整理数据。请将代码放在“```”中，外部执行器将运行生成的代码，并将中间表格反馈给你。如果有把握，请直接回答问题。

图 3：用于两类表格事实核查任务的任务提示模板。

任务提示（三类）

数据库表 DF 如下所示：表格标题：{table caption} {table}

请根据上述数据回答以下问题：“{claim}” 根据给定的表格，上述主张是得到支持、遭到反驳，还是无法验证？请根据问题和表格逐步生成 SQL 或 Python 代码，以正确回答问题。每一步都可以选择生成 SQL 代码来处理查询、生成 Python 代码来重新整理数据，或直接回答问题并给出解释。¨请将代码放在‘“中，外部执行器将运行生成的代码，并将中间表格反馈给你。如果有把握，请直接回答问题。答案必须且只能是以下标签之一：supports、refutes 或 not enough info。

图 4：用于三类表格事实核查任务的任务提示模板。