# Analyzing LLM Instruction Optimization for Tabular Fact Verification

**Anonymous ACL submission**

## Abstract

Instruction optimization provides a lightweight, model-agnostic approach to enhancing the reasoning performance of large language models (LLMs). This paper presents the first systematic comparison of instruction optimization, based on the DSPy optimization framework, for tabular fact verification. We evaluate four out-of-the-box prompting techniques that cover both text-only prompting and code use: direct prediction, Chain-of-Thought (CoT), ReAct with SQL tools, and CodeAct with Python execution. We study three optimizers from the DSPy framework—COPRO, MiPROv2, and SIMBA— across four benchmarks and two model families.

## 1 Introduction

Verifying natural language claims against structured data is a central capability for trustworthy NLP systems deployed in science, public health, and information quality assurance. While numerous methods have been proposed for tabular fact verification (Yang and Zhu, 2021; Ou and Liu, 2022; Lu et al., 2025; Zhang et al., 2024b, *inter alia*), the resulting systems are often specialized to a particular dataset or fail to outperform simpler prompting approaches.

In this work, we conduct a comparative study of out-of-the-box prompting techniques, paired with instruction optimization, for tabular fact verification. Instruction optimization is a technique that allows for improvements to LLM performance without gradient updates. Since LLMs are known to be sensitive to prompt formulation (Webson and Pavlick, 2022; Leidinger et al., 2023), we analyze the impacts of instruction optimization on practical and generalizable prompting techniques, such as Chain-of-Though (Wei et al., 2022), used with open LLMs.

Recent frameworks for instruction optimization (e.g., DSPy; Khattab et al., 2024) treat multi-step LLM pipelines as programs whose textual parameters can be automatically tuned by search or meta-reasoning, yielding large gains on diverse tasks. Despite this progress, a systematic understanding of how instruction optimization affects tabular fact verification is lacking. The following impacts are particularly underexplored: (1) prompting techniques that differ in their intermediate computation (e.g., direct prediction, CoT, and program-aided reasoning via SQL and Python), (2) optimizer families, and (3) model scale and families. Tool-augmented agents (e.g., ReAct; Yao et al., 2023) promise stronger grounding by interleaving thoughts with executable actions, but their end-to-end effectiveness depends critically on the learned tool interface and execution reliability—factors that instruction optimization may help or hinder.

We present the first comparative study of instruction optimization for tabular fact verification using the DSPy optimization framework. Our study focuses on three optimizers within DSPy: COPRO, MiPROv2, and SIMBA[1]. We analyze these across four benchmarks (TabFact, PubHealthTab, and SciTab, MMSci), four prompting techniques (Direct prediction, CoT, ReAct, and CodeAct), and two base LLMs (Qwen3 and Gemma3). We conduct a comprehensive analysis to address the following research questions:

- What is the impact of optimized instructions on CoT reasoning?
- How does the optimized instructions affect the tool calling behavior of ReAct agents?
- Does program-aided reasoning show consistent advantages over CoT in tabular fact checking?

[1]We restrict MiPROv2 and SIMBA to instruction-only tuning to isolate the effect of instructions from few-shot example selection.

## 2 Related Work

**Table-based Fact Checking** Verifying claims against structured evidence requires compositional reasoning over diverse table schema. TabFact (Chen et al., 2020) established the first large-scale benchmark for binary fact verification on Wikipedia tables. Later datasets incorporated more nuanced labeling schema (e.g., three labels instead of only two) and more complex claims requiring multi-hop reasoning (Wang et al., 2021). Among these, several domain-specific datasets have been created: PubHealthTab (Akhtar et al., 2022), which targets claims about public health, SciTab (Lu et al., 2023), which includes claims from computer science publications, and SciAtomicBench (Zhang et al., 2025), which covers computer science along with other domains such as finance. While fact verification datasets typically present tabular data in textual form, multi-modal datasets have also been created (Yang et al., 2025b). Additionally, some fact-verification datasets mix tabular evidence with text (Aly et al., 2021; Schlichtkrull et al., 2023; Zhao et al., 2024) and figures (Wang et al., 2025; Chan et al., 2024).

Early methods for tabular fact verification used symbolic or programmatic reasoning (Chen et al., 2020; Zhong et al., 2020; Shi et al., 2020; Zhang et al., 2020; Yang et al., 2020; Yang and Zhu, 2021; Ou and Liu, 2022). While some recent work has also made use of neuro-symbolic systems (Glenn et al., 2024; Aly and Vlachos, 2024; Cheng et al., 2023), there has been an increasing focus on adapting and making use of LLMs. To this end, prior works have developed both pre-training (Eisenschlos et al., 2020; Dong and Smith, 2021; Zhang et al., 2024a) and fine-tuning (Wu and Feng, 2024; Jiang et al., 2025) approaches for table-based fact verification, as well as more general table-based reasoning tasks (Herzig et al., 2020; Liu et al., 2022). Additionally, several works propose prompting techniques for improving model reasoning over tables (Wang et al., 2024b; Zhang et al., 2025; Abhyankar et al., 2025; Zhang et al., 2024b). Recently, work has also begun to investigate agentic systems and tool-use for table-based fact verification (Lu et al., 2025; Zhou et al., 2025). However, despite these advances, many systems are computationally intensive or specialized to a particular dataset. In contrast, our work explores computationally light instruction optimization techniques applied to general prompting strategies.

Most closely related to our work are two recent analyses into the challenges of various table understanding tasks, including fact verification. Bhandari et al. (2025) examine how instruction tuning, in-context examples, and model size impact performance on tabular reasoning tasks, while Wu et al. (2025) survey approaches to table understanding tasks more broadly. In contrast to these analyses, our work compares instruction optimization techniques applied to simple prompting strategies (standard baselines such as CoT as well as simple programmatic reasoning models such as ReAct). Additionally, while Bhandari et al. (2025) cover multiple table understanding tasks, our work focuses only on table-based fact verification, opting instead to cover a wider range of datasets tabular fact verification.

## 3 Method

### 3.1 Prompting Techniques

**Chain-of-Thought** Chain-of-thought reasoning (CoT) (Wei et al., 2022) encourages LLMs to generate intermediate reasoning steps before producing the final answer. With CoT, LLMs can decompose a complex query into sub-problems and progressively build the solution in the reasoning traces.

**ReAct** ReAct (Yao et al., 2023) serves as a foundational framework for tool-based agents by interleaving reasoning with task-specific actions. ReAct enables LLMs to interact with external tools, allowing them to collect additional evidence and ground their reasoning in the tool execution output. In our experiments, we evaluate a ReAct agent with access to a standard SQL tool that can execute SQL queries on the table data to retrieve relevant information and perform math operations.

**CodeAct** CodeAct (Wang et al., 2024a) leverages executable Python code as the primary action modality for tool-based agents. Unlike existing paradigms that rely on tool calls in text or JSON formats, CodeAct enables multi-step operations and flexible tool chaining through code execution, allowing the agent to perform sophisticated actions by integrating with Python's control flow and existing libraries. In our experiments, the CodeAgent has no access to pre-defined tools. It generates free-form python codes to process the table data and perform math operations step by step.

### 3.2 Instruction Optimization

In our analysis, we focus on three LLM-based instruction optimization approaches in the DSPy (Khattab et al., 2024) framework: COPRO, MiPROv2 (Opsahl-Ong et al., 2024) and SIMBA.

**DSPy Framework** DSPy is a framework for algorithmically optimizing model prompts and weights, treating LLM pipelines as programmes that can be automatically compiled and optimized.

**COPRO** Cooperative Prompt Optimization (COPRO) systematically explores various candidate instructions in a beam search-like manner and evaluates their performance on the train set. The optimizer iteratively refines the prompt instruction by proposing multiple new candidate instructions based on the N best prompts among previous attempts and their corresponding evaluation scores.

**MiPROv2** Multi-Stage Instruction Prompt Optimization (MiPROv2) is an advanced framework that can refine both the instruction and few-shot demonstrations through a three-stage pipeline. First, the optimizer bootstraps multiple candidate sets of few-shot demonstrations from the training data. Then, it generates diverse prompt instructions and demonstrations based on previously evaluated candidates, the properties of the downstream task, and randomly sampled prompting strategies. Finally, MiPROv2 employs Bayesian optimization method to efficiently search the best combination of candidate instruction and demonstration.

Compared with COPRO, MiPROv2 provides a richer context for the generation of new candidate instructions and performs more efficient evaluation on mini-batches of training data.

**SIMBA** Stochastic Introspective Mini-Batch Ascent (SIMBA) is an introspective prompt optimization algorithm that leverages the language model's capacity for self-reflection to iteratively improve instruction quality. The optimizer identifies challenging training instances where model outputs exhibit high variability, then applies two complementary strategies to refine prompts. One strategy performs contrastive analysis, where the model compares successful and unsuccessful execution traces to generate explicit improvement rules that are appended to the original instruction. Another strategy incorporates successful execution trajectories as few-shot demonstrations.

## 4 Experiments

### 4.1 Datasets

We evaluate the performance of various LLMs on four tabular fact checking datasets: TabFact (Chen et al., 2020), PubHealthTab (Akhtar et al., 2022), SciTab (Lu et al., 2023) and MMSci (Yang et al., 2025b). These datasets cover diverse domains and table types, ranging from general knowledge to specialized data, thereby enabling a more comprehensive evaluation of the generalization ability of different approaches. In SciTab, PubHealthTab, and MMSci, there are three labels: *supports*, *refutes* and *not enough info*; TabFact is a binary classification task with only *supports* and *refutes* labels.

**SciTab** SciTab (Lu et al., 2023) is a benchmark designed for scientific claim verification, leveraging real-world table evidence from scientific publications in machine learning and natural language processing domains. The dataset presents unique challenges in claim ambiguity, compositional reasoning and numerical analysis of scientific data.

**PubHealthTab** PubHealthTab (Akhtar et al., 2022) is a table-based fact checking dataset focusing on public health claims. The evidence tables are extracted from multiple web sources, which exhibit noisy and complex table structure with varying content quality.

**TabFact** TabFact (Chen et al., 2020) is a large-scale table-based fact verification dataset that consists of human-annotated claims with Wikipedia tables as evidence. TabFact provides two test sets that differ in the claim complexity, and we use the complex test set for evaluation.

**MMSci** MMSci (Yang et al., 2025b) is a benchmark for multimodal scientific reasoning across three table-based tasks. We use the table fact verification test set, converting table images to textual format, to evaluate generalization to unseen data.

### 4.2 Optimization

For each considered LLM, we evaluate the performance of different prompting techniques, including direct prompting, CoT, ReAct and CodeAct to study the impact of instruction optimization on both language-based reasoning and program-aided reasoning. We use the same instructions in the system prompt before optimization for different experiments, i.e. `verify the given claim against`

| Dataset | Train | Dev | Test |
|---|---|---|---|
| TabFact | 92,585 | 12,851 | 8,609 |
| SciTab | 210 | 429 | 429 |
| PubHealthTab | 1440 | 177 | 180 |
| MMSci | - | - | 1,038 |

Table 1: Statistics of the fact checking datasets.

`the provided table data`. All the experiments are conducted in zero-shot setting.

### 4.3 Evaluation Setup

**Models and Baselines** We conduct our experiments using Qwen3 (Yang et al., 2025a), Gemma3 (Team et al., 2025) and GPT-4o models, which allows us to systematically investigate the impact of instruction optimization on reasoning and tool-calling behavior across different model families and sizes. The same model is used for proposing candidate instructions and evaluating instruction quality during optimization. To examine the effectiveness of optimized instructions, we compare the model performance in GPT-4o experiments with ReActable (Zhang et al., 2024b), a ReAct framework that uses GPT-4o with human-written instructions and SQL and Python as tools.

**Data processing** Each fact checking dataset is processed into a unified data format. We then split three of the datasets (TabFact, SciTab, and PubHealthTab) into train, development and test sets; our fourth dataset, *MMSci, is used only for evaluation*. We create a hybrid training set for instruction optimization by randomly sampling 100 instances from the training splits of the three datasets. We sample 40 PubHealthTab instances, 40 SciTab instances and 20 TabFact instances to ensure the label distribution of the hybrid dataset is balanced. Statistics of the processed datasets are in Table 1.

**Evaluation metrics** We optimize the instructions using the hybrid train data, and evaluate the performance on the development and test sets of all four datasets with accuracy and macro-F1. During instruction optimization, only accuracy is used to measure the quality of different candidate prompts.

## 5 Results

We report the test performance of different prompting techniques with Qwen3 models on four fact checking datasets in Table 2. For direct prompting and CoT, larger models generally achieve higher accuracy and F1 than their smaller counterparts across most test sets. For program-aided reasoning paradigms (ReAct, CodeAct), increasing model size does not yield significant performance gains. Although larger models have similar baseline performance to smaller versions, they benefit substantially more from instruction optimization and show greater improvement with refined instructions.

The effectiveness of instruction optimization for tabular reasoning is highly dependent on both model scale and the prompting technique. For optimizing CoT reasoning, MiPROv2 brings the most consistent gains, achieving the highest accuracy and F1 on PubHealthTab and TabFact for Qwen3-8B, and showing competitive results across three datasets with Qwen3-32B. For program-based reasoning, SIMBA provides the strongest performance gain on SciTab, particularly for improving ReAct with the Qwen3-32B model. COPRO also offers moderate benefits for Qwen3-32B model but less consistently than SIMBA. This suggests that larger models are better at identifying patterns of successful trajectories through self-reflection and comparative analysis, leading to more effective rules for optimizing tool use in diverse scenarios.

According to Table 3, the general trend observed with the Gemma3 model family is slightly different from Qwen3. The larger Gemma3 model shows consistently higher performance for both CoT reasoning *and* program-aided reasoning. Unlike Qwen3, where the optimizers fail to enhance the performance for ReAct and CodeAct with a smaller model, Gemma3 models respond more positively to instruction optimization across different prompting techniques and show greater improvement with refined instructions at both sizes.

Similar to Qwen3 experiments, MiPROv2 still delivers significant improvements when optimizing CoT. SIMBA performs exceptionally well for improving ReAct and CodeAct, particularly for the larger 27B model. COPRO remains effective for smaller model (12B) but provides smaller incremental gains relative to MiPROv2 and SIMBA. Overall, the Gemma3 model family underperforms Qwen3, even after applying instruction optimization. For both Gemma3 and Qwen3 models, CoT reasoning consistently achieves higher performance than program-aided reasoning paradigms on tabular fact checking.

Table 4 summarizes the test performance of GPT-4o models. Due to budget considerations, GPT-4o models and ReActable are evaluated on a smaller TabFact test set (TabFact-mini) with 400

| Module | Optimizer | 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 |
| Direct | Baseline | **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 | Baseline | 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 | Baseline | **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 | Baseline | **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 |

Table 2: Results of Qwen3-8B and Qwen3-32B on test sets. **Bold** is best performance per method and dataset.

random instances. GPT-4o models demonstrate much stronger baseline performance, and consequently benefit less from instruction optimization than Qwen3 and Gemma3 models. For GPT-4o-mini, MiPROv2 is more effective for improving CoT reasoning, while SIMBA yields greater improvements across the test sets for optimizing ReAct. However, no single optimizer provides consistent performance gains for optimizing CodeAct. For the GPT-4o model, SIMBA performs consistently well and brings improvement to both CoT and ReAct, whereas MiPROv2 is shown to be effective for enhancing CodeAct performance. ReAct with GPT-4o shows slightly worse performance on SciTab and TabFact-mini compared with the ReActable baseline, but it can consistently outperform ReActable across all test sets after SIMBA optimization, which demonstrates the superiority of DSPy-based instruction optimization over manually designed prompts.

According to the test performance on MMSci, we observe that for Qwen3-32B and Gemma3-27B model, the optimized instructions with superior performance on PubHealthTab, SciTab and TabFact often generalize well to MMSci. Specifically, instructions optimized by SIMBA consistently achieves the highest F1 scores on MMSci in both direct prompting and ReAct settings, while CoT instructions learned by MiPROv2 continues to deliver the strongest improvements on MMSci. However, this trend is not observed in GPT-4o models, for which the performance on the other three fact checking datasets is not predictive of test performance on MMSci. Although SIMBA shows strong performance on SciTab and TabFact-mini across direct prompting, CoT and ReAct settings, these performance gains do not consistently transfer to MMSci test data. This may indicate instructions proposed by GPT-4o during SIMBA optimization generalize less effectively on unseen data.

| | | Gemma3-12B | | | | | | | | Gemma3-27B | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Module | Optimizer | PubHealth | | SciTab | | TabFact | | MMSci | | PubHealth | | SciTab | | TabFact | | MMSci | |
| | | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 | Acc | F1 |
| Direct | Baseline | 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 | Baseline | 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 | Baseline | 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 | Baseline | 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** |

Table 3: Results of Gemma3-12B and Gemma3-27B on test sets. **Bold** is best performance per method and dataset.

| | | GPT-4o-mini | | | | | | | | GPT-4o | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Module | Optimizer | 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 | | 52.8 | 52.9 | 46.9 | 46.6 | 64.9 | 41.6 | 50.7 | 38.8 | 82.8 | 82.2 | 67.8 | 67.8 | 91.3 | 91.3 | 84.1 | 61.0 |
| Direct | Baseline | 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 | Baseline | **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 | Baseline | 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 | Baseline | **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** |

Table 4: Results of GPT-4o-mini and GPT-4o on test sets. **Bold** is best performance per method and dataset.

## References

Nikhil Abhyankar, Vivek Gupta, Dan Roth, and Chandan K. Reddy. 2025. H-STAR: LLM-driven hybrid SQL-text adaptive reasoning on tables. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8841–8863, Albuquerque, New Mexico. Association for Computational Linguistics.

Mubashara Akhtar, Oana Cocarascu, and Elena Simperl. 2022. PubHealthTab: A public health table-based dataset for evidence-based fact checking. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1–16, Seattle, United States. Association for Computational Linguistics.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13, Dominican Republic. Association for Computational Linguistics.

Rami Aly and Andreas Vlachos. 2024. TabVer: Tabular fact verification with natural logic. *Transactions of the Association for Computational Linguistics*, 12:1648–1671.

Kushal Raj Bhandari, Sixue Xing, Soham Dan, and Jianxi Gao. 2025. Exploring the robustness of language models for tabular question answering via attention analysis. *Trans. Mach. Learn. Res.*, 2025.

Chu Sern Joel Chan, Aakanksha Naik, Matthew Akamatsu, Hanna Bekele, Erin Bransom, Ian Campbell, and Jenna Sparks. 2024. Overview of the context24 shared task on contextualizing scientific claims. In *Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024)*, pages 12–21, Bangkok, Thailand. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. TabFact: A large-scale dataset for table-based fact verification. In *Proceedings of the Eighth International Conference on Learning Representations*.

Zhoujun Cheng, Tianbao Xie, Peng Shi, Chengzu Li, Rahul Nadkarni, Yushi Hu, Caiming Xiong, Dragomir Radev, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2023. Binding language models in symbolic languages. In *ICLR*. OpenReview.net.

Rui Dong and David Smith. 2021. Structural encoding and pre-training matter: Adapting BERT for table-based fact verification. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2366–2375, Online. Association for Computational Linguistics.

Julian Eisenschlos, Syrine Krichene, and Thomas Müller. 2020. Understanding tables with intermediate pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 281–296, Online. Association for Computational Linguistics.

Parker Glenn, Parag Dakle, Liang Wang, and Preethi Raghavan. 2024. BlendSQL: A scalable dialect for unifying hybrid question answering in relational algebra. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 453–466, Bangkok, Thailand. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Chuang Jiang, Mingyue Cheng, Xiaoyu Tao, Qingyang Mao, Jie Ouyang, and Qi Liu. 2025. Tablemind: An autonomous programmatic agent for tool-augmented table reasoning. *arXiv preprint arXiv:2509.06278*.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. Dspy: Compiling declarative language model calls into state-of-the-art pipelines. In *ICLR*. OpenReview.net.

Alina Leidinger, Robert van Rooij, and Ekaterina Shutova. 2023. The language of prompting: What linguistic properties make a prompt successful? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9210–9232, Singapore. Association for Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022. TAPEX: table pre-training via learning a neural SQL executor. In *ICLR*. OpenReview.net.

Xinyuan Lu, Liangming Pan, Qian Liu, Preslav Nakov, and Min-Yen Kan. 2023. SCITAB: A challenging benchmark for compositional reasoning and claim verification on scientific tables. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7787–7813, Singapore. Association for Computational Linguistics.

Xinyuan Lu, Liangming Pan, Yubo Ma, Preslav Nakov, and Min-Yen Kan. 2025. TART: An open-source tool-augmented framework for explainable table-based reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4323–4339, Albuquerque, New Mexico. Association for Computational Linguistics.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. 2024. Optimizing instructions and demonstrations for multi-stage language model programs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366, Miami, Florida, USA. Association for Computational Linguistics.

Suixin Ou and Yongmei Liu. 2022. Learning to generate programs for table fact verification via structure-aware semantic parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7624–7638, Dublin, Ireland. Association for Computational Linguistics.

Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2023. AVeriTeC: A dataset for real-world claim verification with evidence from the web. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Qi Shi, Yu Zhang, Qingyu Yin, and Ting Liu. 2020. Learn to combine linguistic and symbolic information for table-based fact verification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5335–5346, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Chengye Wang, Yifei Shen, Zexi Kuang, Arman Cohan, and Yilun Zhao. 2025. SciVer: Evaluating foundation models for multimodal scientific claim verification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8562–8579, Vienna, Austria. Association for Computational Linguistics.

Nancy X. R. Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021. SemEval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (SEM-TAB-FACTS). In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 317–326, Online. Association for Computational Linguistics.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024a. Executable code actions elicit better llm agents. In *ICML*.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and 1 others. 2024b. Chain-of-table: Evolving tables in the reasoning chain for table understanding. *arXiv preprint arXiv:2401.04398*.

Albert Webson and Ellie Pavlick. 2022. Do prompt-based models really understand the meaning of their prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2300–2344, Seattle, United States. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Xiaofeng Wu, Alan Ritter, and Wei Xu. 2025. Tabular data understanding with llms: A survey of recent advances and challenges. *arXiv preprint arXiv:2508.00217*.

Zirui Wu and Yansong Feng. 2024. ProTrix: Building models for planning and reasoning over tables with sentence context. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4378–4406, Miami, Florida, USA. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Bohao Yang, Yingji Zhang, Dong Liu, André Freitas, and Chenghua Lin. 2025b. Does table source matter? benchmarking and improving multimodal scientific table understanding and reasoning. *arXiv preprint arXiv:2501.13042*.

Xiaoyu Yang, Feng Nie, Yufei Feng, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Program enhanced fact verification with verbalization and graph attention network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7810–7825, Online. Association for Computational Linguistics.

Xiaoyu Yang and Xiaodan Zhu. 2021. Exploring decomposition for table-based fact verification. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1045–1052, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *Proceedings of the Eleventh International Conference on Learning Representations*. ICLR.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1624–1629, Online. Association for Computational Linguistics.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2024a. TableLlama: Towards open large generalist models for tables. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6024–6044, Mexico City, Mexico. Association for Computational Linguistics.

Yuji Zhang, Qingyun Wang, Cheng Qian, Jiateng Liu, Chenkai Sun, Denghui Zhang, Tarek Abdelzaher, Chengxiang Zhai, Preslav Nakov, and Heng Ji. 2025. Atomic reasoning for scientific table claim verification. *arXiv preprint arXiv:2506.06972*.

Yunjia Zhang, Jordan Henkel, Avrilia Floratou, Joyce Cahoon, Shaleen Deep, and Jignesh M Patel. 2024b. Reactable: enhancing react for table question answering. *Proceedings of the VLDB Endowment*, 17(8):1981–1994.

Yilun Zhao, Yitao Long, Yuru Jiang, Chengye Wang, Weiyuan Chen, Hongjun Liu, Yiming Zhang, Xiangru Tang, Chen Zhao, and Arman Cohan. 2024. FinDVer: Explainable claim verification over long and hybrid-content financial documents.

Wanjun Zhong, Duyu Tang, Zhangyin Feng, Nan Duan, Ming Zhou, Ming Gong, Linjun Shou, Daxin Jiang, Jiahai Wang, and Jian Yin. 2020. LogicalFactChecker: Leveraging logical operations for fact checking with graph module network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6053–6065, Online. Association for Computational Linguistics.

Wei Zhou, Mohsen Mesgar, Annemarie Friedrich, and Heike Adel. 2025. Efficient multi-agent collaboration with tool use for online planning in complex table question answering. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 945–968, Albuquerque, New Mexico. Association for Computational Linguistics.