TRACE: Tree-Structured Reasoning for Multi-Table Multi-Hop Fact-Checking

Anonymous ACL submission

Abstract

Fact-checking aims to assess whether natural-language statements are supported by reliable evidence and is crucial for improving the integrity of automated information systems. Multi-table multi-hop fact-checking is challenging, because evidence is scattered across multiple tables, and claim verification requires composing multiple tables as evidence through structured reasoning. We present a comprehensive study of this setting by introducing two new multi-table fact verification testbeds—**OTT-FV** and **Hybrid-FV**—and by proposing **TRACE**, a **tree-structured** framework that explicitly organizes both evidence retrieval and multi-table reasoning. Specifically, TRACE constructs a semantic tree that highlights meaning-bearing subclaims and their relations, performs node-level multi-table retrieval to gather a compact set of relevant evidence for each node, and composes evidence-aware representations bottom-up to produce a final decision grounded in retrieved tabular evidence. Experiments on TabFact, OTT-FV, and Hybrid-FV datasets show consistent improvements over strong baselines across diverse LLM backbones under both gold and open evidence settings. Extensive analyses and case studies further indicate that stronger tree construction improves retrieval and verification, and that task-specific node-level retrieval is crucial compared to off-the-shelf retrievers.[1]

1 Introduction

Fact-checking is an important tool for improving the trustworthiness of information systems. While many existing works verify claims against free-form text evidence (Zhou et al., 2019; Liu et al., 2020), tables are also a critical evidence source because they store precise facts in a compact and structured form. However, table fact-checking remains relatively underexplored (Chen et al., 2019; Lu et al., 2023). Existing settings often assume that all evidence is contained in a single table or that gold evidence tables are given in advance, whereas in realistic scenarios evidence may be distributed across multiple tables and must be retrieved before verification (Yao et al., 2023).

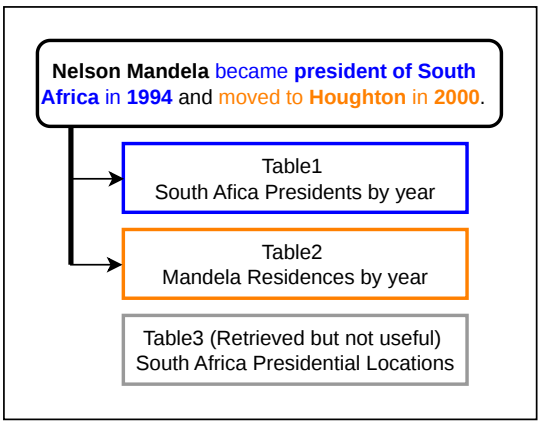

Figure 1: **A motivating example.** Verifying one claim may require combining evidence from multiple tables.

This makes *multi-table multi-hop fact-checking* particularly challenging: a system must both retrieve relevant tables and compose evidence across them through multi-hop reasoning. These two challenges are tightly coupled—retrieval errors propagate to downstream reasoning, while weak reasoning signals make retrieval harder (Zheng et al., 2024). As a result, flat retrieval followed by direct inference often misses critical evidence and provides limited control over intermediate reasoning, especially in open-evidence settings with unavoidable retrieval noise (Chen et al., 2019).

A natural solution is to impose explicit structure on both retrieval and reasoning. However, a key question remains: **what structure should be used, and how can it be made actionable for retrieval and cross-table composition?** We argue that multi-table verification is inherently compositional—complex claims requiring multi-table verification can often be decomposed into meaning-

[1] Code and datasets are included in the supplementary material and will be released upon publication.

bearing subclaims connected by logical relations, e.g., conjunction, chaining, comparison, and aggregation (Yang et al., 2018; Chen et al., 2020b). An effective system should expose this compositionality to guide retrieval and provide an explicit computation graph for multi-hop reasoning. We illustrate such a case in Figure 1, where verifying a single claim requires combining evidence from multiple tables.

Motivated by this view, we propose **TRACE** (**T**ree-Structured **R**e**A**soning for Multi-Table Multi-Hop Fact-**Ch**E**cking), a tree-structured framework that makes both retrieval and reasoning explicit. Given a claim, TRACE constructs a semantic tree whose nodes correspond to subclaims, retrieves evidence for each node, and composes node representations bottom-up to produce the final verdict. This design turns global retrieval into focused subclaim retrieval and enables controlled cross-table reasoning with an explicit intermediate structure.

To support systematic study in this setting, we also introduce two new multi-table fact verification testbeds: **OTT-FV** and **Hybrid-FV**, derived from OTT-QA (Chen et al., 2020a) and HybridQA (Chen et al., 2020b) and annotated for claim verification. They stress multi-hop cross-table reasoning with 2/3/4-hop reasoning and 7 types of cross-table compositions, and we provide both *gold* and *open* evidence variants. Together with TabFact (Chen et al., 2019), they allow us to evaluate both end-to-end verification and the behavior of retrieval and reasoning.

Experiments on TabFact, OTT-FV, and Hybrid-FV show that TRACE consistently improves verification performance and evidence coverage across multiple backbones under both gold and open settings. Further analyses show that tree structure, node-level retrieval, and explicit composition are all important for robust multi-table fact-checking.

This paper makes three main contributions:

1. **Framework:** We propose **TRACE**, a tree-structured framework for multi-table multi-hop fact-checking with explicit retrieval and compositional reasoning.
2. **Testbeds:** We construct two new testbeds, **OTT-FV** and **Hybrid-FV**, with both gold and open evidence variants.
3. **Analyses:** We provide systematic analyses of tree construction, retrieval design, and composition for multi-table fact-checking.

2 Related Work

Multi-hop fact-checking. Fact-checking studies whether natural-language claims are supported by evidence, often requiring both retrieval and reasoning (Thorne et al., 2018). However, many works rely on free-form texts as evidence and neglect the importance of tabular evidence (Zhou et al., 2019; Liu et al., 2020; Zhang et al., 2024). We focus on tables as an important evidence source. Recent works explore multi-hop fact-checking (Zhang and Lee, 2025; Zheng et al., 2025; Zhao et al., 2020; Subramanian and Lee, 2020) but still ignore tables.

Table fact-checking. Prior work has mainly focused on *single-table* verification. TabFact (Chen et al., 2019) establishes this benchmark and inspires two main directions: linearizing tables for joint encoding with the claim (Dong and Smith, 2021; Herzig et al., 2020) and modeling verification through latent operations or structured execution (Chen et al., 2019; Wang et al., 2021). More recent work improves performance with stronger table-aware pretraining and backbones (Gu et al., 2022). However, they do not explicitly address the cross-table retrieval and compositional reasoning challenges in multi-table multi-hop verification.

Reasoning over structured and unstructured data. For completeness, we briefly review existing work for other related tasks. Multi-hop question-answering highlights the difficulty of retrieving and composing evidence across structured and unstructured sources, e.g., HybridQA (Chen et al., 2020b) and OTT-QA (Chen et al., 2020a). However, they target *answer extraction* rather than *claim verification*. Another line of work injects graph structure into verification and multi-hop reasoning (Zhong et al., 2020; Zhou et al., 2019; Chen et al., 2025; Zou et al., 2023). Others use semantic parsing, e.g., Abstract Meaning Representation (AMR), to support decomposition and reasoning (Banarescu et al., 2013; Deng et al., 2022).

Positioning of our work. Different from prior single-table verification methods and multi-hop QA formulations, we provide a comprehensive study of *multi-table multi-hop fact-checking* by (i) introducing two claim verification testbeds, OTT-FV and Hybrid-FV; (ii) proposing TRACE, a tree-structured framework that explicitly combines node-level retrieval with bottom-up compositional reasoning, supported by systematic analyses and case studies;

(iii) studying not only one tree-construction strategy, but also how different tree structures affect retrieval and verification.

3 Methodology

3.1 Task Definition and Notations

Following TabFact (Chen et al., 2019), we study multi-table multi-hop fact-checking as a binary classification problem, though our model is generalizable to more fine-grained classification when other labels, e.g., not enough information, are available. Given a natural-language claim x and a corpus of tables $\mathcal{T}$, the goal is to predict a verdict $y \in \mathcal{Y} = \{\text{SUPPORTED}, \text{REFUTED}\}$. We consider two evaluation settings: **Gold** evidence, where relevant evidence is provided; and **Open** evidence, where a system must retrieve evidence from $\mathcal{T}$.

Our framework constructs a semantic tree structure for each claim and performs node-level retrieval, node encoding, and bottom-up compositional reasoning. Let the semantic tree be $G = (V, E)$, where V is the set of nodes and E is the set of tree edges. Each node $v \in V$ is associated with a node query q_v produced during tree construction. For each node, we first retrieve a small set of relevant tables E_v and then rank rows within them to obtain row-level evidence R_v. Based on the retrieved evidence, we compute an evidence-aware node representation. Finally, we compose node representations from leaves to the root to predict y. Figure 2 provides an overview of the model architecture.

3.2 Semantic Tree Construction

Since a claim that needs multiple tables for verification usually contains complex information, we decompose it for fine-grained modeling. Given a claim x, we first obtain a semantic parse and normalize it into a *tree-structured* semantic representation $G = (V, E)$. In this work, we mainly use Abstract Meaning Representation (AMR)-style semantic parses (Banarescu et al., 2013) for their robustness and efficiency. Alternative tree constructors, including dependency trees, simple rule-based trees, and LLM-generated trees, are investigated in Section 4.3.

Each node $v \in V$ corresponds to a semantically coherent subclaim. We then derive the node query q_v directly from the normalized tree node by lightly canonicalizing its surface form: we remove AMR-specific markers and relation artifacts, and retain only the core lexical content (typically a word or short phrase) that best represents the node semantics for retrieval.

3.3 Node-level Evidence Retrieval

In the **Open** setting, the system must retrieve evidence from a large table corpus $\mathcal{T}$, where relevant evidence may be scattered across multiple tables. After constructing the semantic tree, we perform retrieval *independently for each node* using its derived query q_v. Since node queries are typically short and focused, while evidence is represented as structured tables, we employ a task-specific SentenceBERT-based retriever (Reimers and Gurevych, 2019), which is fine-tuned to better align such semantic queries with serialized tabular evidence.

Concretely, we linearize each candidate table into a text sequence that preserves basic structural cues, including the title, headers, and row content. Formally, we represent each table as:

```
Title: <table title> || Header: <h1 | h2
| ...> || Rows: <row1> || <row2> || ...
```

Each row is linearized by concatenating its cell values with a vertical bar (|). We truncate long titles, headers, and row content when necessary so that the serialized table fits within the retriever's maximum input length. For example, a table with title *NBA Finals*, headers *Year, Champion, Runner-up*, and two rows is serialized as:

```
Title: NBA Finals || Header: Year |
Champion | Runner-up || Rows: 2020 |
Lakers | Heat || 2021 | Bucks | Suns.
```

This linearization preserves key table structure while remaining simple and efficient for dense retrieval.

To train the retriever, we use node queries derived from the semantic tree as supervision on the query side, paired with annotated evidence tables as positives. To improve discrimination among closely related tables, we further sample hard negatives from the same table family as the positive tables. The retriever is then optimized to assign higher similarity to relevant query–table pairs than to irrelevant but confusable ones.

At inference time, we first retrieve tables by scoring the node query against serialized tables:

$$s(q_v, t) = \text{sim}(\phi(q_v), \phi(\text{lin}(t))), \qquad (1)$$

where $\phi(\cdot)$ denotes the fine-tuned SentenceBERT encoder, $\text{lin}(\cdot)$ is the table linearization function,

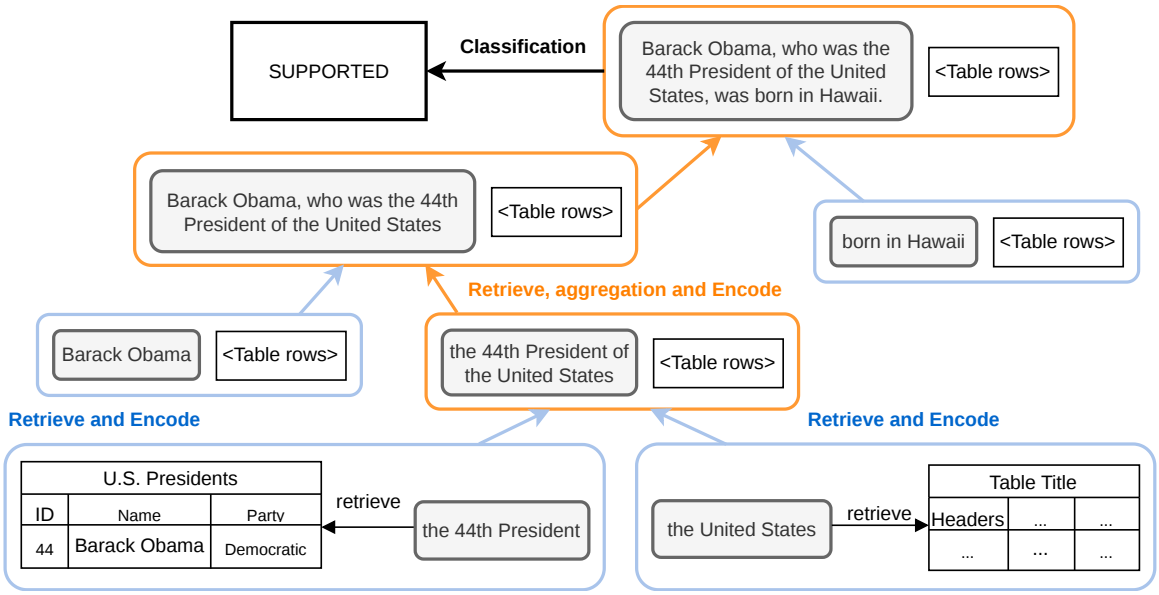

Figure 2: Overview of TRACE. We construct a semantic tree with node queries, retrieve evidence per node, encode node queries and evidence, and compose representations bottom-up to make the final decision at the root.

and $\mathrm{sim}(\cdot,\cdot)$ is a similarity measure (e.g., cosine). For each node $v \in V$, we select the top-k scoring tables:

$$E_v = f_{\text{top-}k}\{s(q_v, t) \mid t \in \mathcal{T}\}. \tag{2}$$

k is a hyperparameter; empirically, we set $k = 10$.

We then perform a row-level reranking stage to refine evidence at a finer granularity. For each node v, we collect rows from the retrieved tables in E_v, linearize each row together with its table context, and score it against the same node query using the same retriever. We retain the top-n rows as the final node-level evidence:

$$R_v = f_{\text{top-}n}\{s(q_v, r) \mid r \in \mathcal{R}(E_v)\}, \tag{3}$$

where $\mathcal{R}(E_v)$ denotes the set of candidate rows from the retrieved tables. These top-ranked rows are passed to the downstream node encoder. Empirically, we set $n = 10$.

3.4 Evidence-aware Node Encoding

We encode each node together with its row-level evidence using a pretrained LLM backbone encoder $f_\theta(\cdot)$. For node v, we first build an evidence context $\mathrm{ctx}(R_v)$ by linearizing the reranked rows in R_v and concatenating them in rank order. We then form the encoder input with a fixed template that combines the node query and its evidence context:

$$\mathbf{z}_v = \mathrm{Format}(q_v, \mathrm{ctx}(R_v)), \tag{4}$$

where $\mathrm{ctx}(R_v)$ denotes the serialized evidence context constructed from the row-level evidence R_v, and $\mathrm{Format}(\cdot,\cdot)$ denotes a simple input templating function that places the query and evidence into a single sequence. We provide the exact serialization and formatting template in Appendix C.2.

3.5 Bottom-up Compositional Reasoning

To perform multi-hop reasoning across nodes, we aggregate node representations following a post-order traversal from leaves to root. For a leaf node v, we directly set its composed representation to its evidence-aware representation, i.e., $\tilde{\mathbf{h}}_v = \mathbf{h}_v$. For a non-leaf node v with children $\mathcal{C}(v)$, we compose its own evidence-aware representation $\mathbf{h}_v$ with the set of composed child representations $\{\tilde{\mathbf{h}}_c\}_{c\in\mathcal{C}(v)}$. We instantiate composition using **PMA + FiLM** (Lee et al., 2019; Perez et al., 2018; de Vries et al., 2017), where PMA provides a permutation-invariant pooling operator for a variable-size set of child nodes, and FiLM conditionally modulates the parent representation based on the aggregated child summary.

PMA over child nodes. We apply Pooling by Multihead Attention (PMA) (Lee et al., 2019) to summarize the child representations into a fixed-size vector. Since different nodes may have different numbers of children, PMA provides a permutation-invariant way to aggregate a variable-size set of subclaims. Concretely, given the child set $\{\tilde{\mathbf{h}}_c\}_{c\in\mathcal{C}(v)}$, we use a learnable seed vector $\mathbf{s}$ to attend over the child representations:

$$\mathbf{u}_v = \mathrm{MHA}\Big(\mathbf{s}, \{\tilde{\mathbf{h}}_c\}_{c\in\mathcal{C}(v)}, \{\tilde{\mathbf{h}}_c\}_{c\in\mathcal{C}(v)}\Big), \tag{5}$$

where $\mathrm{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ denotes multi-head attention with query $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$. We use the resulting vector $\mathbf{u}_v \in \mathbb{R}^d$ as the aggregated summary of the child nodes.

FiLM modulation. We use Feature-wise Linear Modulation (FiLM) (Perez et al., 2018; de Vries et al., 2017) to condition the parent node on aggregated child information. Specifically, given the child summary $\mathbf{u}_v$, we predict a feature-wise scale and shift:

$$\Delta\boldsymbol{\gamma}_v,\ \boldsymbol{\beta}_v = g(\mathbf{u}_v), \qquad \boldsymbol{\gamma}_v = 1 + \tanh(\Delta\boldsymbol{\gamma}_v), \tag{6}$$

where $g(\cdot)$ is an MLP. We then modulate the parent representation as

$$\tilde{\mathbf{h}}_v = \boldsymbol{\gamma}_v \odot \mathbf{h}_v + \boldsymbol{\beta}_v, \tag{7}$$

where $\odot$ denotes element-wise product. The reparameterization $\boldsymbol{\gamma}_v = 1 + \tanh(\Delta\boldsymbol{\gamma}_v)$ keeps the scaling centered around 1.

3.6 Root-level Prediction and Training

Let r be the root node. After bottom-up composition, we obtain $\tilde{\mathbf{h}}_r$ and predict the veracity by

$$\hat{\mathbf{p}} = \mathrm{softmax}(\mathbf{W}\tilde{\mathbf{h}}_r + \mathbf{b}), \qquad \hat{y} = \arg\max_{y \in \mathcal{Y}} \hat{\mathbf{p}}_y, \tag{8}$$

where $\mathbf{W}$ and $\mathbf{b}$ are trainable parameters. Given retrieved evidence, we jointly train the encoder, composition modules, and classifier using cross-entropy loss:

$$\mathcal{L} = -\sum_{y \in \mathcal{Y}} p_y \log \hat{p}_y. \tag{9}$$

4 Experiments

4.1 Experimental Setup

Task settings. We evaluate under two evidence settings. In the **Gold** setting, the tabular evidence required for verification is provided, and the system verifies the claim within the constrained evidence space. In the **Open** setting, the system must first retrieve relevant tables (and rows) from a large table corpus before verifying the claims.

Datasets. We evaluate on three benchmarks: two multi-table multi-hop fact-checking testbeds, **OTT-FV** and **Hybrid-FV**, and a widely used single-table fact-checking benchmark, **TabFact**. Though TabFact is a single-table benchmark, the open evidence setting still evaluates multi-table reasoning ability (Yao et al., 2023). Thus, for completeness purpose, we include it for experiments. Detailed dataset construction and statistics are provided in Appendix A.

Baselines. We compare TRACE against several groups of representative baselines. *First*, we include two **general LLM backbones**, **DeBERTa-v3** (He et al., 2021) and **Qwen3-8B** (Yang and Team, 2025). *Second*, we compare against **table verification and reasoning methods**, including **PASTA** (Gu et al., 2022) and **Table-llama** (Zhang et al., 2023). *Third*, we include **LLM-based fact-checking methods**, namely **Factcheck-GPT** (Wang et al., 2023) and **DCUF** (Hu et al., 2022). *Fourth*, we evaluate **industrial LLMs**, e.g., **GPT-3.5-turbo** (OpenAI, 2023) and **Gemini-2.5-pro** (Comanici and Team, 2025). Finally, to show the adaptability of our framework, we backbone **TRACE** with different LLMs, including **DeBERTa**, **Qwen3-8B**, **PASTA**, and **Table-llama**.

Metrics. Following prior work on table-based and evidence-grounded fact verification (Chen et al., 2019), we report **Accuracy** and **Macro-F1**. For evidence retrieval, we additionally report evidence coverage metrics (Zheng et al., 2024).

4.2 Main Results and Analyses

Overall results. Table 1 presents the claim verification results. In the realistic **Open** setting that retrieves multiple tables for fact-checking, TRACE achieves the strongest performance across **all three benchmarks**, demonstrating the benefit of explicitly structuring both retrieval and reasoning for multi-table verification. The gains are particularly consistent on the two multi-table benchmarks, where evidence is distributed across tables, and multi-hop composition is required.

In the **Gold** setting, our approach remains competitive and achieves the best results on TabFact and OTT-FV with suitable backbones, while strong LLM-based verifiers can be advantageous on Hybrid-FV. This pattern suggests that when evidence is provided, model capacity and table-specific pretraining dominate; when evidence must be retrieved, our structured retrieval-and-composition pipeline becomes a key differentiator.

TRACE is backbone-agnostic. A key takeaway is that TRACE improves diverse LLM backbone families. Compared to the corresponding backbone-only models, adding our framework yields consistent gains in Open evidence. For instance, on

Model	TabFact		OTT-FV		Hybrid-FV	
	Gold	Open	Gold	Open	Gold	Open
DeBERTa-v3	86.1/85.4	78.8/77.9	67.4/66.2	62.5/61.7	56.2/56.3	53.7/53.1
Qwen3-8B	86.7/86.0	80.5/79.6	68.2/67.1	64.6/63.8	60.1/60.1	53.3/54.2
PASTA	89.4/87.7	78.5/77.6	70.1/69.0	62.8/61.9	58.2/56.2	54.1/56.8
Table-llama	87.5/86.6	80.8/79.9	69.0/67.9	65.0/64.1	51.2/50.3	53.5/52.6
DCUF	86.1/85.2	82.7/81.9	64.7/63.6	63.1/62.2	56.1/55.0	50.8/49.9
Factcheck-GPT	82.0/81.3	80.6/79.8	74.6/73.8	68.3/67.5	**63.3/64.2**	56.4/55.5
GPT-3.5-turbo	70.1/69.4	72.4/71.0	68.2/67.2	65.3/66.1	62.4/56.7	54.2/55.4
Gemini-2.5-pro	76.3/75.4	74.2/73.4	70.9/69.8	66.9/66.0	62.8/61.9	55.2/54.3
Ours (DeBERTa)	88.8/88.1	83.1/82.3	72.8/71.9	72.1/71.2	61.8/60.9	59.2/**59.1**
Ours (Qwen3-8B)	**89.4/88.6**	**83.5/82.7**	74.2/73.3	**72.4/71.6**	61.0/60.1	**59.7**/58.9
Ours (PASTA)	87.1/86.5	81.8/81.3	**76.3/74.2**	66.7/67.2	60.8/59.3	55.7/56.4
Ours (Table-llama)	87.8/86.3	81.2/80.3	71.0/68.4	69.5/65.7	52.4/50.6	55.5/54.4

Table 1: Fact-checking results (Accuracy (%) / Macro-F1 (%)) on three benchmarks under Gold and Open settings.

TabFact (Open), DeBERTa-v3 is improved from 78.8%/77.9% to 83.1%/82.3%, Qwen3-8B from 80.5%/79.6% to 83.5%/82.7%, and PASTA from 78.5%/77.6% to 81.8%/81.3%. Similar improvements hold on **OTT-FV (Open)** and on **Hybrid-FV (Open)**. Results indicate that the gains primarily come from semantic tree structure decomposition, node-level retrieval, and bottom-up composition, rather than from backbone-specific methods.

Why Open evidence benefits more. The larger improvements in Open evidence align with our design goal: tree structure produces focused node queries, turning a global retrieval problem into a set of semantically grounded sub-retrieval tasks, and the bottom-up composer explicitly aggregates node-level verification signals to support multi-hop reasoning. When retrieval noise and missing evidence are prevalent, this structured pipeline provides a stronger inductive bias than flat retrieval followed by one-shot inference.

Discussion on large table backbones. Finally, we observe that the incremental gains on very strong table-centric backbones (e.g., Table-llama) can be smaller on some settings, suggesting that model capacity and table pretraining may already capture part of the compositional behavior implicitly. This motivates our additional analyses and ablations on tree construction, retrievers, and composition modules in the following sections.

4.3 Tree Construction Analyses

Goal. We examine how different semantic tree construction strategies affect both end-to-end verification and retrieval quality. We keep all other components fixed (the number of retained evidence rows and the composition module) and only vary how a claim is converted into a semantic tree. We follow Zheng et al. (2024) and report **Macro-F1** and **Evidence Recall@U**, where Recall@U is computed over the *union* of retrieved rows across all nodes (after deduplication).

Overall trend: tree structure matters. Table 2 shows that removing the tree and verifying with a single root query (*No tree, root-only*) performs the worst on both benchmarks (OTT-FV: 61.3% F1 / 40.7% recall; Hybrid-FV: 53.7% F1 / 18.7% recall). This indicates that in multi-table multi-hop settings, a one-shot formulation struggles to cover critical evidence and yields less reliable reasoning. Introducing a tree consistently improves both F1 and evidence recall, supporting the value of explicit decomposition and node-level retrieval and reasoning when evidence is scattered across tables.

More semantic structure yields larger gains. Among non-LLM tree builders, we observe a clear performance gain from weaker to stronger structure. A simple rule-based segmentation tree improves over the no-tree baseline but remains limited. Dependency-tree construction (Honnibal and Montani, 2017) further increases both F1 and evidence recall. AMR-based tree achieves substantially better results on both metrics. These results suggest that tree construction quality directly affects the quality of node queries and subsequent composition, and that more semantically aligned structure

Tree Construction	OTT-FV		Hybrid-FV	
	Macro-F1 (%)	Evid. Recall@U (%)	Macro-F1 (%)	Evid. Recall@U (%)
AMR-tools tree	71.2	80.4	59.1	46.8
Dependency tree	68.9	70.6	57.2	40.5
Simple rule-based segmentation tree	65.0	52.5	55.9	34.4
LLM-generated tree	75.6	84.6	63.5	56.8
No tree (root-only)	61.3	40.7	53.7	18.7

Table 2: **Tree construction analyses.** F1 score and evidence recall across five tree construction strategies.

provides a stronger starting point for both retrieval and reasoning.

LLM-generated trees and headroom. The LLM-generated tree achieves the strongest performance and evidence coverage on both datasets (OTT-FV: 75.6%/84.6%; Hybrid-FV: 63.5%/56.8%), indicating additional headroom when tree construction can better align subclaims and produce more effective node queries. Meanwhile, this highlights tree construction as a potential future work on improving tree quality while maintaining stability and cost efficiency.

Takeaway. Overall, tree construction has a substantial impact on both verification and evidence recall. Even simple trees provide clear benefits over a no-tree formulation, while more semantically informed trees further amplify these gains.

4.4 Retrieval Method Study

We evaluate how the node-level retriever affects overall performance by keeping the rest of TRACE fixed (AMR-tools tree construction, the number of retained evidence rows, backbone, and composition module) and only swapping the retriever. We report end-to-end **Macro-F1** and **Evid. Recall@U**.

Overall finding: node-level retrieval is critical and benefits from task-specific design. As shown in Table 3, our task-specific retriever achieves the best performance on both datasets. Replacing it with off-the-shelf BM25 (Robertson and Zaragoza, 2009) or SentenceBERT leads to substantial drops in both Macro-F1 and evidence recall, indicating that TRACE depends not only on structured composition but also on high-quality evidence selection aligned with node queries.

Retrieval quality directly propagates to end-to-end verification. On OTT-FV, lower evidence coverage from BM25 and SentenceBERT is accompanied by correspondingly lower Macro-F1, showing that retrieval failures directly limit downstream verification. The same pattern is even stronger on Hybrid-FV, where evidence is more dispersed and precise subclaim alignment is more important; generic retrievers are therefore less effective at matching node queries to the correct rows, leaving insufficient evidence for downstream composition.

Takeaway. Retriever choice has a decisive impact on TRACE. Compared with BM25 and SentenceBERT, our task-specific node-level retriever substantially improves both evidence coverage and end-to-end verification, supporting the need for retrieval tailored to tree-structured decomposition.

4.5 Component Ablations

We quantify the contribution of two key composition components in TRACE: **PMA** (attention-based pooling over child node representations) and **FiLM** (feature-wise modulation conditioned on aggregated child information). We conduct ablations under the **Open** setting and report **F1**, keeping all other components fixed.

Overall finding: both PMA and FiLM are important, with PMA often more critical. As shown in Table 4, removing either component consistently degrades performance across datasets and backbones, indicating that TRACE's gains rely on explicit cross-node composition rather than incidental factors. In most cases, **w/o PMA** leads to larger drops than **w/o FiLM**. For example, with DeBERTa on TabFact, performance drops from 82.3% to 79.6% without FiLM and to 77.8% without PMA. The same pattern largely holds across other datasets and backbones, suggesting that effective child aggregation is central to bottom-up reasoning, while FiLM provides additional gains by conditioning the parent representation on the composed child state.

Multi-table benchmarks are more sensitive to composition. Compared to the single-table benchmark TabFact, the multi-table benchmarks (OTT-FV and Hybrid-FV) generally show larger degradation when composition is weakened, especially under **w/o PMA**. This is consistent with

Retriever	OTT-FV		Hybrid-FV	
	Macro-F1 (%)	Evid. Recall@U (%)	Macro-F1 (%)	Evid. Recall@U (%)
Ours	71.2	80.4	59.1	46.8
BM25	64.5	52.3	52.5	21.9
SentenceBERT	66.1	63.7	53.9	30.5

Table 3: **Retrieval method study.** Comparison of different node-level retrievers on OTT-FV and Hybrid-FV.

Model	TabFact (Open)			OTT-FV (Open)			Hybrid-FV (Open)		
	Full	w/o FiLM	w/o PMA	Full	w/o FiLM	w/o PMA	Full	w/o FiLM	w/o PMA
Ours (DeBERTa)	**82.3**	79.6	77.8	**71.2**	66.9	64.1	**59.1**	55.4	52.6
Ours (Qwen3-8B)	**82.7**	80.1	78.3	**71.6**	67.8	65.0	**58.9**	55.9	53.1
Ours (PASTA)	**81.3**	78.7	76.9	**67.2**	63.1	60.4	**56.4**	53.0	51.7
Ours (Table-llama)	**80.3**	78.2	74.7	**65.7**	60.9	60.4	**54.4**	53.8	54.1

Table 4: Component ablation study (Open setting, F1 score (%)).

Model	OTT-FV Accuracy (%)			Hybrid-FV Accuracy (%)		
	2-hop	3-hop	4-hop	2-hop	3-hop	4-hop
Ours (DeBERTa)	72.8	63.9	56.1	61.8	57.3	54.8
Ours (Qwen3-8B)	74.2	65.1	57.3	61.0	57.9	55.6
Ours (PASTA)	76.3	66.4	58.2	60.8	56.8	54.3
Ours (Table-llama)	71.6	67.0	59.5	57.2	55.6	52.8

Table 5: **Hop breakdown on OTT-FV and Hybrid-FV.** Accuracy grouped by hop count. As hop count increases, performance converges toward the random baseline (50%). Our models remain consistently above chance, indicating stronger robustness under high-hop multi-table reasoning.

the intuition that multi-table multi-hop verification requires combining multiple local verification signals across subclaims. We also observe smaller drops for Table-llama on Hybrid-FV, suggesting that strong table-centric backbones may implicitly capture part of the compositional behavior or that performance in this regime is more constrained by retrieval bottlenecks.

Takeaway. Both PMA and FiLM are important components of TRACE: PMA supports stable aggregation over variable numbers of child nodes, while FiLM enables conditional fusion between parent evidence and composed child information. The consistent drops under ablation confirm the importance of explicit cross-node compositional reasoning.

4.6 Hop Breakdown

We analyze performance under different reasoning depths by grouping examples by hop count (2/3/4-hop) and reporting **Accuracy**. Since the task is binary, random guessing corresponds to **50%** accuracy.

Overall trend: higher hops are harder, yet models remain above chance. As shown in Table 5, all backbones show a consistent drop from 2-hop to 3-hop to 4-hop on both datasets, confirming that verification becomes harder as reasoning depth increases. The degradation is larger on OTT-FV, while Hybrid-FV stays closer to the 50% boundary in the hardest bucket, suggesting a more difficult regime with stronger retrieval or composition bottlenecks. Across backbones, differences are clearer in 2-hop cases but become smaller at 4-hop, consistent with increasing reasoning depth amplifying shared limitations such as missing evidence and error accumulation.

Takeaway. Hop breakdown confirms the core challenge of multi-table multi-hop fact-checking: performance approaches the random baseline as hop count increases. Nevertheless, our models remain consistently above chance even in 4-hop settings, indicating non-trivial robustness under high-hop reasoning.

Case study. We present a case study to visually illustrate our framework. See Appendix E.1.

5 Conclusion

We study multi-table multi-hop fact-checking and present **TRACE**, a semantic tree-based framework that explicitly structures both retrieval and fact-checking reasoning. TRACE constructs a semantic tree with node queries, performs node-level evidence retrieval, and composes evidence-aware node representations bottom-up to make a final verification decision. We introduce two multi-table fact verification testbeds, **OTT-FV** and **Hybrid-FV**. TRACE achieves consistent improvements, with particularly strong benefits under open evidence. Our analyses further show that tree construction quality and task-specific node-level retrieval are critical, and that the composition modules contribute substantially to performance.

Limitations

Our study has several limitations. First, although TRACE improves multi-table multi-hop fact-checking by introducing tree-structured decomposition and bottom-up reasoning, its performance still depends on the quality of the constructed semantic tree. Errors in parsing or subclaim decomposition may propagate to downstream retrieval and composition, especially in high-hop settings where reasoning chains are longer and intermediate mistakes accumulate more easily.

Second, our retrieval framework is still constrained by evidence coverage. Even with node-level retrieval, the correct evidence may be missed when relevant rows are lexically sparse, semantically implicit, or distributed across tables in ways that are difficult to capture with the current retrieval setup. As a result, some failures may stem from retrieval bottlenecks rather than reasoning errors alone.

Finally, stronger tree construction methods, such as LLM-based decomposition, may improve performance but also introduce additional computational cost and variability. This creates a practical trade-off between accuracy, stability, and efficiency that is not fully explored in the current work.

Potential Risks

Our work studies fact-checking over tabular evidence and is intended to improve the reliability of automated information systems. However, such systems may still make incorrect verification decisions when the retrieved evidence is incomplete, misleading, or only partially relevant. If deployed in real-world settings without human oversight, these errors could lead users to place undue trust in incorrect judgments. In addition, although our datasets are constructed from public table corpora and are designed for research use, automated claim generation and verification pipelines may inherit biases or factual inaccuracies from the underlying source tables. We therefore view TRACE as a decision-support tool for research rather than a fully reliable replacement for human fact-checking in high-stakes settings.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, and William Yang Wang. 2020a. Open question answering over tables and text. *CoRR*, abs/2010.10439.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2019. Tabfact: A large-scale dataset for table-based fact verification. *CoRR*, abs/1909.02164.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020b. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Yingjian Chen, Haoran Liu, Yinhong Liu, Jinxiang Xie, Rui Yang, Han Yuan, Yanran Fu, Peng Yuan Zhou, Qingyu Chen, James Caverlee, and 1 others. 2025. Graphcheck: Breaking long-term text barriers with extracted knowledge graph-powered fact-checking. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14976–14995.

Gabriela Comanici and Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Harm de Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron Courville. 2017. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhenyun Deng, Yonghua Zhu, Yang Chen, Michael Witbrock, and Patricia Riddle. 2022. Interpretable amr-based question decomposition for multi-hop question answering. *CoRR*, abs/2206.08486.

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*.

Zihui Gu, Ju Fan, Nan Tang, Preslav Nakov, Xiaoman Zhao, and Xiaoyong Du. 2022. PASTA: Table-operations aware fact verification via sentence-table cloze pre-training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

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

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. https://spacy.io/. To appear.

Nan Hu, Zirui Wu, Yuxuan Lai, Xiao Liu, and Yansong Feng. 2022. Dual-channel evidence fusion for fact verification over texts and tables. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 7342–7351.

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.

OpenAI. 2023. Gpt-3.5 turbo model. https://developers.openai.com/api/docs/models/gpt-3.5-turbo. Accessed 2026-03-12.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. 2018. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc.

Shyam Subramanian and Kyumin Lee. 2020. Hierarchical evidence set modeling for automated fact extraction and verification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7798–7809.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 809–819.

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). *CoRR*, abs/2105.13995.

Yuhan Wang, Siva Reddy, and 1 others. 2023. Factcheck-GPT: End-to-end fine-grained document-level fact-checking and correction of LLM responses. *CoRR*, abs/2311.09000.

An Yang and Qwen Team. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Barry Menglong Yao, Aditya Shah, Lichao Sun, Jin-Hee Cho, and Lifu Huang. 2023. End-to-end multimodal fact-checking and explanation generation: A challenging dataset and models. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2733–2743.

Congzhi Zhang, Linhai Zhang, and Deyu Zhou. 2024. Causal walk: Debiasing multi-hop fact verification with front-door adjustment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 19533–19541.

Delvin Ce Zhang and Dongwon Lee. 2025. Correct: Context-and reference-augmented reasoning and prompting for fact-checking. 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 3007–3019.

Tianyi Zhang and 1 others. 2023. Towards open large generalist models for tables. *CoRR*, abs/2311.09206.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *International conference on learning representations*.

Liwen Zheng, Chaozhuo Li, Litian Zhang, Haoran Jia, Senzhang Wang, Zheng Liu, and Xi Zhang. 2025. Mrr-fv: Unlocking complex fact verification with multi-hop retrieval and reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26066–26074.

Liwen Zheng, Chaozhuo Li, Xi Zhang, Yu-Ming Shang, Feiran Huang, and Haoran Jia. 2024. Evidence retrieval is almost all you need for fact verification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9274–9281.

Wanjun Zhong, Wenhan Xiong, Nan Tang, Xuguang Duan, and William Yang Wang. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. Gear: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901. Association for Computational Linguistics.

Anni Zou, Zhuosheng Zhang, and Hai Zhao. 2023. Decker: Double check with heterogeneous knowledge for commonsense fact verification. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11891–11904.

A Dataset Construction and Statistics

A.1 Source Table Corpora

To construct multi-table multi-hop fact verification benchmarks, we build on the source table corpora underlying two existing QA resources. For **OTT-FV**, we use the official table corpus released with OTT-QA. For **Hybrid-FV**, we use the Wikipedia table pool underlying HybridQA, i.e., the original source corpus rather than only the subset of tables explicitly appearing in HybridQA examples. This setup allows our Open evidence evaluation to more faithfully reflect large-scale retrieval over a realistic table pool.

A.2 Claim Construction

We construct claims by first sampling groups of related tables from the source corpus and then prompting `gpt-4o` to identify table combinations that support a fact requiring information from multiple tables. The goal is to create claims that cannot be verified from a single table alone, and instead require cross-table composition. This design encourages multi-table and multi-hop reasoning patterns rather than single-table lookup.

More specifically, given a set of candidate tables, we ask `gpt-4o` to determine whether there exist supporting tables with overlapping or related information that can be combined into a true factual claim. For each valid instance, the model is instructed to generate (1) a concise claim, (2) the identifiers of the supporting tables, and (3) a relation type describing how the evidence is combined. The allowed relation types are CONJUNCTIVE, CHAINED, COMPARATIVE, CONDITIONAL, CAUSAL, AGGREGATION, and TAXONOMIC. If no suitable table combination can be found, the model outputs `no`. We provide the full prompt in Appendix A.3 for reproducibility.

For **supported** claims, the generated statement must be fully grounded in the selected tables. For **refuted** claims, we minimally perturb a supported claim so that the incorrect part directly conflicts with the table evidence, rather than introducing content that is merely unsupported or unverifiable. This construction ensures that refuted examples remain evidence-based and require genuine verification against the tables.

All generated claim–label pairs are then manually re-annotated by 20 graduate students in natural language processing. Annotators verify whether the claim is supported or refuted by the designated tables and correct noisy labels produced during automatic generation. The agreement between the `gpt-4o`-proposed label and the final human judgment is **88.10%**, indicating that manual verification is necessary to ensure dataset quality.

A.3 Prompt for Claim Construction

We use the following prompt to guide `gpt-4o` in constructing multi-table claims:

> You are given several tables from a series. Your task is to check whether there are TWO tables that contain overlapping or related information that can be combined to construct a TRUE fact-based claim. A valid claim must be verifiable using information from both tables (neither table alone is sufficient).
>
> Each table has a unique ID (uid). Your output must have exactly four lines:
>
> 1. A single concise factual claim (similar in style to FEVER/HoVer). 2. The uid of the first supporting table. 3. The uid of the second supporting table. 4. One logic-relation label, chosen from the list: CONJUNCTIVE, CHAINED, COMPARATIVE, CONDITIONAL, CAUSAL, AGGREGATION, TAXONOMIC.
>
> Do not quote or paraphrase the table contents inside the claim. If no suitable pair of tables can be found, output exactly one line: `no`. Output nothing else (no explanations, no blank lines).

For 3-hop and 4-hop examples, we use the same prompt template with only minimal modifications: the required number of supporting tables and the number of returned table IDs are adjusted accordingly.

year	country	president	term_note
1994	South Africa	Nelson Mandela	Took office on 1994-05-10
1995	South Africa	Nelson Mandela	In office
1996	South Africa	Nelson Mandela	In office
1997	South Africa	Nelson Mandela	In office
1998	South Africa	Nelson Mandela	In office
1999	South Africa	Nelson Mandela / Thabo Mbeki	Mandela served until 1999-06-14; Mbeki succeeded him in June 1999
2000	South Africa	Thabo Mbeki	In office

Table 6: South Africa presidents by year.

year	person	main_role
1993	Nelson Mandela	ANC leader / anti-apartheid statesman
1994	Nelson Mandela	President of South Africa
1995	Nelson Mandela	President of South Africa
1996	Nelson Mandela	President of South Africa
1997	Nelson Mandela	President of South Africa
1998	Nelson Mandela	President of South Africa
1999	Nelson Mandela	President of South Africa / retired statesman
2000	Nelson Mandela	Former president; public humanitarian work

Table 7: Nelson Mandela roles by year.

A.4 Dataset Statistics

The final dataset statistics are as follows.

OTT-FV. OTT-FV contains **951** training examples and **435** test examples, corresponding to **68.61% / 31.39%** of the full dataset, respectively. The underlying table pool contains approximately **400K** tables. The label distribution is **55.47%** supported and **44.53%** refuted. Its hop-count distribution is: **2-hop: 1087**, **3-hop: 152**, and **4-hop: 147**. Its relation-type distribution is: **Conjunctive: 649**, **Chained: 333**, **Aggregation: 183**, **Comparative: 132**, **Taxonomic: 69**, **Conditional: 18**, and **Causal: 2**.

Hybrid-FV. Hybrid-FV contains **2400** training examples and **800** test examples, corresponding to **75.00% / 25.00%** of the full dataset, respectively. The underlying table pool contains approximately **1M** tables. The label distribution is **53.15%** supported and **46.85%** refuted. Its hop-count distribution is: **2-hop: 2260**, **3-hop: 680**, and **4-hop: 260**. Its relation-type distribution is: **Conjunctive: 1624**, **Chained: 845**, **Aggregation: 257**, **Comparative: 239**, **Conditional: 109**, **Taxonomic: 107**, and **Causal: 19**.

A.5 Data Privacy, Human Annotation and Sensitive Content

Our datasets are constructed from public table corpora derived from Wikipedia and prior QA resources. As a result, some tables may contain names of real people or other publicly available entity information. We do not collect private personal data, and we do not add new personally identifying information beyond what already appears in the public source tables. During dataset construction, we manually verify generated claims and labels, and we do not intentionally include offensive, hateful, or sexual content. Because the source corpora are large and open-domain, we cannot completely rule out the presence of isolated sensitive or offensive entries inherited from the original tables; however, the benchmark is intended for research use on public factual data rather than for profiling individuals.

To verify the quality of generated claim–label pairs, we recruited 20 graduate students from our university as volunteer annotators. Annotators were shown a generated claim together with the corresponding gold table(s), and were asked to judge whether the claim was factually supported or refuted by the provided evidence. They were also asked to correct annotation mistakes when necessary. Participation was voluntary and unpaid. Annotators were informed that their annotations would be used for research and paper writing, and all participation was based on consent.

Because the annotation task involved only verification over public tabular data and did not involve private personal data, intervention, or high-risk human-subject protocols, we did not seek separate ethics review board approval.

period	person	location	residence_type
late 1992–1999	Nelson Mandela	Houghton, Johannesburg	
1994–1999	Nelson Mandela	Pretoria	
1994–1999	Nelson Mandela	Cape Town	
from retirement in 1999 (incl. 2000)	Nelson Mandela	Houghton, Johannesburg	Post-presidential home/office base

Table 8: Mandela residence-related evidence by period.

capital	type	country
Pretoria	administrative capital	South Africa
Cape Town	legislative capital	South Africa
Bloemfontein	judicial capital	South Africa

Table 9: South Africa capitals.

site	city
Union Buildings	Pretoria
Mahlamba Ndlopfu	Pretoria
Tuynhuys	Cape Town
Genadendal	Cape Town

Table 10: South African official government sites.

A.6 Use of AI Assistants

We used GPT-4o during dataset construction to generate candidate fact-checking claims from related tables. The model was used only as a data generation aid: all generated claim–label pairs were subsequently manually verified and corrected by human annotators before inclusion in the dataset. We did not rely on AI-generated outputs without human verification.

B AMR-guided Tree Construction

B.1 AMR Parser

We use `transition-amr-parser` to obtain an AMR-style semantic graph for each claim. The parser output serves as the starting point for our semantic tree construction.

B.2 Tree Normalization

Our semantic tree should be understood as a structured decomposition of the sentence rather than a direct one-to-one copy of the AMR graph. In practice, we extract entity nodes and attribute/relation structures from the AMR graph and normalize them into a tree suitable for bottom-up composition.

At a high level, the normalization process groups words into the same subtree based on their distances in the AMR graph and in the generated subtrees associated with entities. Intuitively, words that are semantically closer in the AMR structure are more likely to be assigned to the same branch of the tree. This produces a hierarchical decomposition in which lower-level nodes correspond to smaller semantic units and higher-level nodes compose them into larger meaning-bearing subclaims.

year	person	event
1993	Nelson Mandela	Won Nobel Peace Prize with F. W. de Klerk
1994	Nelson Mandela	Inaugurated as President of South Africa
1998	Nelson Mandela	Married Graça Machel
1999	Nelson Mandela	Stepped down after one term as President

Table 11: Nelson Mandela notable milestones.

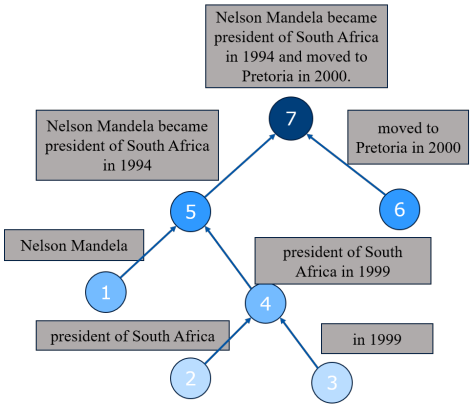

Figure 3: Illustrative example of AMR-guided semantic tree construction.

Figure 3 shows an illustrative example. For the claim *"Nelson Mandela became president of South Africa in 1994 and moved to Pretoria in 2000."*, the tree first composes *Nelson Mandela* with *president of South Africa*, and then with *in 1994* to form the left branch. In parallel, it combines *moved to Pretoria* with *in 2000* to form the right branch. The root node finally merges these two branches into the full claim.

C Retrieval and Evidence Formatting

C.1 Retrieval Unit

In our implementation, the retrieval unit is a **table**, not an individual row. Each candidate table is serialized into a compact textual representation, and the retriever scores the relevance between a node query and the serialized table. This design keeps retrieval simple and scalable over large table corpora.

C.2 Evidence Serialization for Node Encoding

For node encoding, we serialize the node query together with the retrieved evidence rows into a single text sequence. Concretely, given a node query q_v and a ranked list of retrieved rows $\mathcal{R}_v = \{r_1, r_2, \ldots, r_n\}$, we construct the input as:

`[Query]`: <*node query*>
`[Evidence]`
`[Row1]` <*header1*>: <*cell1*> <*header2*>: <*cell2*> ...
`[Row2]` <*header1*>: <*cell1*> <*header2*>: <*cell2*> ...
⋮
`[Row`n`]` <*header1*>: <*cell1*> <*header2*>: <*cell2*> ...

That is, each evidence row is linearized as a sequence of `header: cell` pairs, preserving the original column names so that the encoder can access both cell values and their semantic roles. Rows are concatenated in retrieval order, with special markers (`[Query]`, `[Evidence]`, `[Row`i`]`) used to distinguish the query from different evidence rows.

For example, if the node query is *"Nelson Mandela presidency in 1994"*, one retrieved row may be serialized as:

`[Row1] Year: 1994 President: Nelson Mandela Country: South Africa`

This serialized sequence is then fed into the node encoder to obtain the evidence-aware representation of the node.

C.3 Task-specific Retriever

Our task-specific retriever is based on Sentence-BERT, which we fine-tune to better match short node queries with serialized tables. Positive examples are constructed from a node query paired with its corresponding gold table(s), while negative examples are constructed by pairing the query with other tables from the same table series as the gold table. At inference time, we use the simplest dense retrieval setup: cosine similarity between the query embedding and serialized-table embedding over the full table pool.

The retriever is trained with the following settings:

- backbone: SentenceBERT
- batch size: 64
- learning rate: 1×10^{-5}
- number of epochs: 3
- warmup ratio: 0.05
- mixed precision: fp16
- max input length: 2048

D Implementation Details

D.1 Training Hyperparameters

For **DeBERTa** and **PASTA**, we use:

- optimizer: AdamW
- learning rate: 2×10^{-5}
- number of epochs: 3
- batch size: 16
- seed: 42
- max input length: 512

For **Qwen** and **Table-llama**, we use:

- optimizer: AdamW
- learning rate: 2×10^{-5}
- number of epochs: 3
- batch size: 12
- seed: 42
- max input length: 2048

D.2 Complexity

For a tree with $|N|$ nodes, TRACE makes one retrieval call and one node encoding per node, giving an overall complexity of $O(|N|)$ retrieval and encoding steps. The bottom-up composition is also linear in the tree size, i.e., $O(|N|)$ for a tree. In practice, the semantic trees are small and the evidence per node is capped at Top-k, which keeps the overall computation manageable in open-evidence settings.

D.3 Hardware

All experiments are conducted on the university's NVIDIA A100 GPU cluster.

E Baseline Details

Local backbones. DeBERTa-v3 and Qwen3-8B are deployed locally.

Official reproductions. PASTA, Table-llama, DCUF, and Factcheck-GPT are reproduced from their official repositories.

API-based LLMs. GPT-3.5-turbo and Gemini-2.5-pro are evaluated via their official APIs. We use zero-shot prompting and otherwise keep the default API settings.

E.1 Case Study: Multi-table Retrieval and Compositional Reasoning

We present a real example to illustrate how tree-structured retrieval helps TRACE recover evidence distributed across multiple tables, while a flat retriever is more easily distracted by only part of the claim. Tables 6–11 show a set of potentially retrievable related tables.

Claim. *"Nelson Mandela became president of South Africa in 1994 and moved to Pretoria in 2000."*

Gold label. REFUTED

Why this example is challenging. This claim contains two subclaims: (i) Nelson Mandela became president of South Africa in 1994, and (ii) he moved to Pretoria in 2000. The first subclaim is correct, while the second is false. As shown in Figure 3, the semantic tree decomposes the claim into these two parts, enabling the system to retrieve and verify them separately before composing a final decision at the root.

Relevant evidence. The first subclaim is supported by multiple tables. In `south_africa_presidents_by_year`, the entry for 1994 lists Nelson Mandela as president, with a note indicating that he took office on 1994-05-10. In `mandela_roles_by_year`, the entry for 1994 states that Nelson Mandela's main role was *President of South Africa*, again confirming the first half of the claim.

The second subclaim requires more careful reasoning. A flat retriever is easily attracted by tables mentioning Pretoria, such as `south_africa_capitals` and `south_africa_government_sites`, because Pretoria is strongly associated with South African government and presidential offices. However, these tables do not establish that Mandela *moved to Pretoria in 2000*. Instead, the relevant evidence comes from `mandela_residence_by_period`, which shows that Mandela had an official presidential residence in Pretoria during 1994–1999, but from retirement in 1999 (including 2000), his post-presidential home/office base was in *Houghton, Johannesburg*. Therefore, the claim that he *moved to Pretoria in 2000* is contradicted by the evidence.

Why tree-structured retrieval helps. The semantic tree separates the claim into a "presidency" branch and a "moved to Pretoria in 2000" branch. This decomposition allows node-level retrieval to focus on different evidence needs: one branch retrieves tables about Mandela's role in 1994, while the other retrieves tables about residence/location around 2000. In contrast, a flat root-level retriever tends to prioritize the more salient first half of the claim—*Mandela*, *president*, and *1994*—and may only retrieve partially relevant Pretoria-related tables for the second half. As a result, plain retrieval often captures only the supported portion of the claim, while missing the crucial contradictory evidence needed for the full verification decision.

Bottom-up reasoning. After node-level retrieval, TRACE verifies the two branches separately. The left branch establishes that Mandela indeed became president in 1994. The right branch, however, finds that although Pretoria is associated with presidential residence during Mandela's term, the evidence for the period after his retirement places him in Houghton, Johannesburg rather than Pretoria in 2000. When these two branch-level results are composed at the root, the final verdict is REFUTED.

Takeaway. This example highlights the main advantage of TRACE: by decomposing a claim into semantically coherent subclaims, the framework can retrieve evidence for each part separately and avoid being dominated by the most salient supported fragment of the claim. In multi-table fact-checking, this tree-structured retrieval-and-composition process is especially important when one part of the claim is correct but another part is contradicted by less obvious evidence.