ToE: Defending against GEO Poisoning with Hierarchical and Explainable Byzantine-Robust Aggregation

Anonymous authors
Paper under double-blind review

Abstract

Large language models (LLMs) increasingly rely on retrieval systems and external tools to obtain up-to-date information beyond their parametric knowledge. This reliance introduces a new attack surface. Knowledge corruption attacks can inject misleading evidence into retrieval pipelines, while Generative Engine Optimization (GEO) poisoning can make adversarial content more likely to be retrieved and used during reasoning. We propose Tree of Evidence (ToE), a hierarchical and explainable Byzantine-robust aggregation framework with dynamic multi-source evidence retrieval. ToE uses reinforcement learning-guided retrieval, evaluates the stance and reliability of individual evidence items, and organizes claims in a dynamically expanded argument tree. It then aggregates node scores bottom-up to limit the influence of unreliable or conflicting evidence. Experiments on four datasets and two backbone LLMs, together with component and retrieval-tool ablations, demonstrate the effectiveness of ToE on standard claim-verification benchmarks and under controlled GEO poisoning.

1 Introduction

Large language models (LLMs), such as the DeepSeek series (Liu et al., 2024a) and the GPT series (Achiam et al., 2023), have demonstrated strong capabilities across a wide range of tasks. However, their limited access to up-to-date information is one factor that can contribute to hallucination. To address this limitation, two major lines of work have been proposed. Retrieval-Augmented Generation (RAG) constructs an external knowledge index and retrieves relevant documents at inference time to supply the model with up-to-date context (Lewis et al., 2020). Tool calling (Schick et al., 2023), on the other hand, enables LLMs to dynamically invoke external tools such as search engines during generation, thereby grounding responses in freshly retrieved information. Both approaches alleviate the hallucination problem to a certain extent. However, the introduction of third-party information also brings new risks. AI-generated or manually fabricated misinformation can lead LLMs to produce incorrect conclusions (Zhong et al., 2023; Zou et al., 2024; Wang et al., 2026a). This risk may be further exacerbated by Generative Engine Optimization (GEO), a technique that structures content to maximize its discoverability by retrieval algorithms such as embedding-based ranking (Aggarwal et al., 2024). Compared to manually written truthful content, adversarially crafted misinformation optimized via GEO can be systematically ranked higher in retrieval results, making it more likely to be consumed by LLMs. This threat is particularly pressing in the era of large-scale AI-driven advertising, where GEO-optimized content can be deliberately embedded into recommendation pipelines to manipulate LLM outputs at scale (Xue et al., 2026). For example, when a user asks *"Who is the CEO of OpenAI?"*, an attacker can inject a fabricated document claiming that *"Tim Cook has joined OpenAI as CEO since yesterday, replacing Sam Altman"*, which once contaminates the retrieved context and causes the LLM to produce a confidently wrong answer.

To combat the spread of misinformation, researchers have proposed a range of detection methods. Early work relies on deep learning-based approaches (Zhou & Zafarani, 2020), but these methods suffer from limited generalizability, as the inherent diversity of online information across topics, styles, and platforms makes it difficult to transfer models trained on one distribution to unseen data from another (Liu et al., 2024b). More recently, LLM-based fake news detection methods have been proposed (Lucas et al., 2023; Liu et al., 2024b; Li et al., 2024a; Ni et al., 2024), which leverage

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the reasoning capabilities of LLMs to analyze input content and identify potential misinformation. While these methods achieve promising results on scientific or knowledge-intensive claims, they struggle with time-sensitive news, particularly in the face of AI-generated misinformation where fabricated content is fluent, coherent, and difficult to distinguish from genuine reporting without access to real-time external knowledge. Knowledge corruption under adversarial evidence can be formulated as a Byzantine-robust aggregation problem (Wang et al., 2026a). To address this problem, we propose Tree of Evidence, a hierarchical and explainable Byzantine-robust aggregation framework against GEO poisoning. ToE models a claim as a dynamically expanding argument tree and verifies it through three core components: a reinforcement learning-driven multi-source retrieval agent that collects evidence across platforms according to the characteristics of the claim under investigation, an evidence evaluation agent that scores the veracity and reliability of the claim based on the collected evidence, and an argument tree aggregation agent that decomposes the claim into sub-claims along dimensions such as who, what, when, where, why, and how, expanding the subtree for deeper verification when the current evidence is insufficient for a reliable judgment. Node-level scores are propagated bottom-up through the tree, and the process terminates when the root node reaches a convergence or decisiveness threshold, yielding a final veracity score along with the full reasoning tree.

We summarize our **main contributions** as follows: (1) We propose ToE, a hierarchical and explainable Byzantine-robust aggregation framework against GEO poisoning. To the best of our knowledge, ToE is the first framework against GEO poisoning that combines RL-guided dynamic multi-source information retrieval with reliability-aware stance aggregation. Its retrieval policy selects heterogeneous sources and supporting or counter-evidence queries at verification time, while its evaluator encodes each document with stance and source-quality attributes and aggregates them into veracity and reliability estimates rather than treating all retrieved evidence as equally trustworthy. (2) We introduce a reliability-driven argument-tree aggregation mechanism that uses node reliability to control claim decomposition, branch expansion and pruning, and stopping. Bottom-up aggregation propagates evidence across sub-claims to limit the influence of low-quality or conflicting documents, while the resulting tree records the evidence and aggregation path for inspection. (3) We evaluate ToE on four public datasets with two backbone LLMs under standard verification and controlled GEO-poisoning settings, together with component and retrieval-tool ablations. The results support the effectiveness and robustness of the framework across datasets, model scales, and evidence conditions.

2 Related Work

With the rapid development of AI, LLM-based agents address the knowledge staleness problem by invoking external search tools to retrieve relevant information as supplementary context (Lewis et al., 2020; Schick et al., 2023). However, this reliance on third-party sources introduces new vulnerabilities. Prior work has demonstrated that AI-generated misinformation injected into retrieval results can mislead LLMs into producing incorrect conclusions, with potentially severe consequences in domains such as medicine and finance (Zou et al., 2024; Huang & Sun, 2023). While deep learning-based detection methods have been proposed to identify such content (Zhou & Zafarani, 2020), they rely on surface-level stylistic features and fail to generalize to AI-generated misinformation that differs from truthful content only in subtle factual details.

Recent work leverages the reasoning capabilities of LLMs to assist in verifying the reliability of information. F3 framework guides LLMs through stepwise logical analysis and evidence consistency assessment via prompt engineering techniques such as zero-shot chain-of-thought reasoning and deductive generation, producing a veracity judgment for the input claim (Lucas et al., 2023). TELLER decomposes fake news detection into a set of structured evaluation questions, directing LLMs to analyze news from dimensions including factual accuracy, contextual consistency, and source credibility, then aggregates the dimension-level scores through interpretable logical rules to reach a final verdict (Liu et al., 2024b). However, these methods predominantly rely on LLMs to analyze content provided by the claim content, and show limited performance against AI-generated misinformation. STEEL adopts a multi-round retrieval-augmented strategy in which an LLM assesses the confidence of initial retrieval results and automatically generates refined queries to iteratively collect additional web evidence when the current evidence is insufficient, alleviating the dependence on static knowledge bases (Li et al., 2024a). AdSent rewrites input claims into sentiment-neutralized variants to force the veracity classifier to rely solely on factual content, improving robustness against

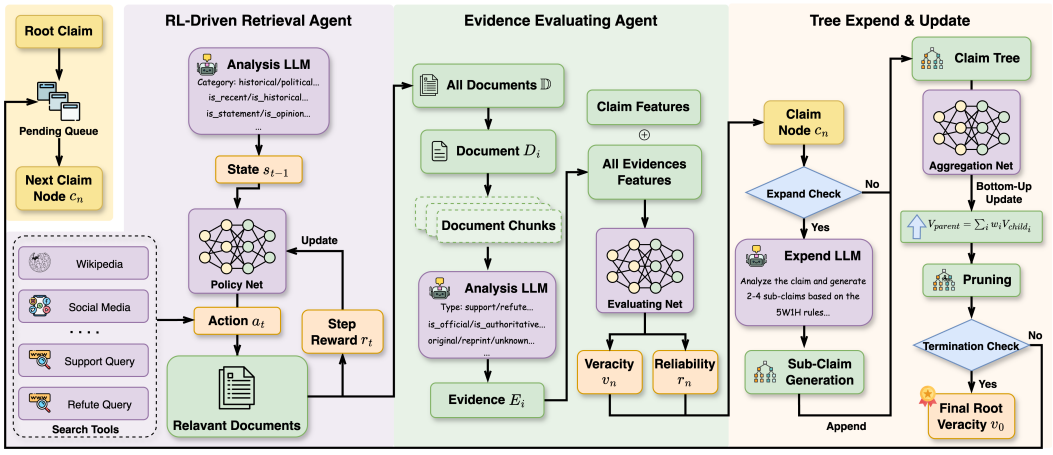


Figure 1: Overview of the ToE framework.

sentiment-manipulated misinformation (Tahmasebi et al., 2026). However, it provides limited defense against misinformation grounded in fabricated facts.

3 METHOD

To mitigate the effect of GEO-poisoned evidence, we propose ToE, a hierarchical and explainable Byzantine-robust aggregation framework with dynamic multi-source evidence retrieval. ToE models the verification of a claim as a dynamically growing argument tree. At each step, it collects and evaluates evidence from heterogeneous sources, including Wikipedia, Arxiv, fact-checking platforms, search engine results, and social media. Based on the evaluation results, new subtrees are expanded to progressively refine the veracity judgment. This process ultimately produces an interpretable evidence tree that traces the full reasoning process. As depicted in Fig. 1, the system initializes the argument tree with the claim under investigation as the root node and enters the main iterative loop. At each iteration, the priority of each pending node is computed dynamically based on the uncertainty of its parent and its own estimated importance, so that nodes with the greatest expected contribution to the final judgment are processed first. For each selected node, the system analyzes semantic features of the claim, such as its category and geographic scope, and generates three types of search queries: general background queries, supporting evidence queries, and counter-evidence queries. This design ensures comprehensive and objective evidence collection and reduces confirmation bias. A retrieval agent trained via reinforcement learning then executes searches autonomously across heterogeneous sources, including Wikipedia, PolitiFact, social media, and the general web, and decides dynamically when to stop retrieval. The collected content is parsed by an LLM to extract evidence snippets directly relevant to the claim, each annotated with attributes such as stance and source credibility, forming a structured evidence set. Based on this evidence, the system computes two core scores for the current node: veracity, which measures the probability that the claim is true, and reliability, which measures how strongly the current evidence supports that judgment. After each node is evaluated, scores are propagated bottom-up through the tree via an aggregation network, continuously updating the root node's overall judgment. If a node's reliability is insufficient, indicating that the available evidence does not yet support a confident verdict, the system invokes an LLM to decompose the claim into finer-grained, more verifiable sub-claims along various dimensions, covering time, location, person, event, cause, and information source, and adds them as new child nodes for further processing. Concurrently, subtrees that have reached high reliability with a sufficiently decisive verdict are pruned to avoid redundant computation. The iterative process continues until the root node's judgment converges, a decisiveness threshold is met, or the maximum number of iterations is reached, ultimately producing a veracity score for the claim and a complete argument tree available for traceability and review.

3.1 Conceptual Framework and Design Motivation

The verification process in ToE is guided by two complementary conceptual lenses: a decision-theoretic formulation that characterizes the structure of the search problem, and an information-theoretic intuition that motivates the design of the reward signal driving the retrieval agent.

POMDP Problem Formulation. The core challenge in automated fact-checking is that the ground-truth veracity $v^* \in [0, 1]$ of a claim c is a latent variable that cannot be directly observed. The system can only accumulate evidence through sequential search actions and must form a judgment under persistent uncertainty. This structure maps naturally onto a Partially Observable Markov Decision Process (POMDP) (Kaelbling et al., 1998), defined as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{R})$, with components as follows. The **state** $s_t = (\mathcal{E}_t, f_c)$ is the joint representation of the evidence set $\mathcal{E}_t$ accumulated up to step t and the semantic features f_c of the claim under investigation, where f_c encodes attributes such as claim category and geographic scope. The **action** $a_t \in \mathcal{A}$ is selected from the eight-option discrete action space of the retrieval agent, covering heterogeneous source types including Wikipedia, arXiv, fact-checking platforms, social media, and a stop action. The **observation** $o_t \in \mathcal{O}$ is the set of documents retrieved after executing action a_t, which is subsequently processed by the evidence evaluation agent into structured evidence objects annotated with stance and credibility attributes. The **transition** $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ describes how the evidence set is updated upon receiving new observations:

$$\mathcal{E}_{t+1} = \mathcal{E}_t \cup \{o_t\}. \tag{1}$$

The **belief state** $b_t = (v_{n,t}, r_{n,t})$ is maintained by the evaluation network, which maps the current evidence set and claim features to a veracity score $v_{n,t}$ and a reliability score $r_{n,t}$. The evaluation network serves as a domain-driven belief estimator rather than an independently trained value network: it produces a posterior estimate of v^* given all available evidence. The dual belief state $(v_{n,t}, r_{n,t})$ is motivated by the distinction between *what* the system believes (veracity) and *how strongly* it believes it (reliability). This separation is essential for the tree management module to determine whether a node should be decomposed into sub-claims or pruned as decisive.

Information-Seeking Reward Intuition. The reward signal driving the retrieval agent is designed based on an information-theoretic intuition. At each step t, the agent receives a step reward proportional to the reliability gain $\Delta r_t = r_{n,t} - r_{n,t-1}$. We assume that this reliability gain approximates the conditional mutual information between the latent veracity v^* and the new observation o_t given prior evidence:

$$\Delta r_t \approx I(v^*; o_t | E_{t-1}) = H(v^* | E_{t-1}) - H(v^* | E_t). \tag{2}$$

Under this assumption, each search action reduces epistemic uncertainty about v^*, with the reliability score serving as a tractable surrogate for posterior confidence. We further assume that the marginal information gain exhibits diminishing returns consistent with submodularity of reliability with respect to the evidence set:

$$r(E_A \cup \{o\}) - r(E_A) \geq r(E_B \cup \{o\}) - r(E_B), \quad E_A \subseteq E_B. \tag{3}$$

These assumptions serve as useful design principles with two practical implications. First, they motivate the per-step penalty δ_{step} in the reward: if marginal gains naturally decrease, the agent should learn to stop when the expected gain falls below the cost of an additional retrieval step. Second, they motivate the convergence-based termination criterion: if the reliability score approaches a plateau as evidence accumulates, the root veracity should stabilize across iterations. Importantly, the system does not depend on these assumptions holding exactly—the tree management module provides robustness through explicit reliability thresholds for expansion (τ_r^{exp}) and decisiveness (τ_r^{dec}), which trigger continued search or pruning regardless of whether the reward signal perfectly tracks mutual information.

Reward Fidelity and Practical Considerations. Since the retrieval process involves sequential interaction with stochastic external sources under partial observability, it naturally fits the POMDP framework. The information-gain and diminishing-returns assumptions motivate the reward and stopping design rather than imposing a required property on every open-web evidence environment. If we denote the per-step deviation of the evaluation network from ideal reliability by ϵ, then under the assumptions above:

$$|\Delta r_t - I(v^*; o_t | E_{t-1})| \leq 2\epsilon. \tag{4}$$

This relation characterizes the fidelity of the surrogate reward; it is not a policy-convergence guarantee. The tree management module can still trigger further expansion when reliability is low, regardless of whether the reward perfectly tracks mutual information. We report the evaluation network results in Appendix C.

3.2 RL-Driven Retrieval Agent

As one of the core components of ToE, the retrieval agent is responsible for searching the web for content relevant to the claim under investigation. The agent first analyzes the input claim to extract a set of semantic features, including its category (e.g., historical, scientific, or political) and attributes such as whether it describes a recent event, a historical fact, or a factual statement. These claim-level features are combined with step-level search state features, including the number of searches performed, the number of supporting snippets retrieved, and the number of counter-evidence snippets retrieved, to form the current state s_t. The state is fed into a policy network π_θ, which selects an action a_t from a discrete action space. The action space covers eight options: searching general web sources, searching for counter-evidence, querying arXiv for academic papers, querying Wikipedia, searching social media platforms, querying fact-checking sites such as PolitiFact and Snopes, searching Reddit discussions, and stopping the search. This design reflects the observation that different types of claims are best verified through different sources. Established facts benefit from dedicated fact-checking sites, encyclopedic knowledge is well served by Wikipedia, and time-sensitive claims, particularly statements attributed to public figures or official bodies, are often most efficiently verified via social media. The explicit counter-evidence action further ensures that the agent actively seeks disconfirming information rather than collecting only supporting content. Given the selected action, the agent uses an LLM to generate a search query tailored to the claim's content and then invokes the corresponding tool to retrieve results from the target source. After each search step, a step reward r_t is computed and stored together with the transition in a trajectory buffer. Once the buffer reaches the designated capacity, the policy network is updated online using Group Relative Policy Optimization (GRPO) (Shao et al., 2024), a reinforcement learning algorithm that estimates advantages by comparing returns within a group of sampled trajectories, eliminating the need for a separate value network. A complete description of the state features, reward function, and training details is provided in the Appendix E.1.

3.3 Evidence Evaluation Agent

After the retrieval agent collects relevant documents, each document D_i is processed individually. To avoid exceeding the context limit of the LLM, each document is first split into chunks. Each chunk is submitted to a cleaning LLM that extracts statements directly relevant to the claim. Once all chunks of a document are processed, the extracted statements are combined with the document's metadata, such as URL and author information, and submitted to an analysis LLM. This LLM determines the document's stance toward the claim (supporting, refuting, or neutral), whether it originates from an official or authoritative source, and whether it constitutes a primary report or a republication, among other attributes. The result is a structured evidence object E_i. After all documents are processed, the full evidence set $\mathbb{E}$ and the claim's semantic features are passed to the evaluation network, which produces two scores for the current node: a veracity score v_n and a reliability score r_n.

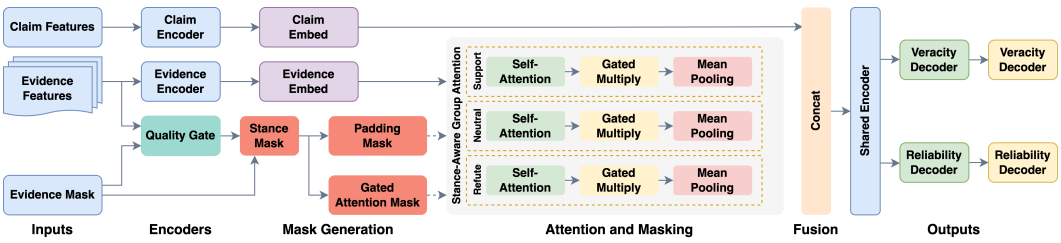


Figure 2: Evidence evaluation network.

As illustrated in Fig. 2, the evaluation network jointly predicts veracity and reliability through a stance-aware multi-path attention mechanism. Claim features and evidence features are first encoded by a claim encoder and a shared evidence encoder into high-dimensional claim embeddings and

evidence embeddings, respectively. In parallel, source-level quality indicators within the evidence features are passed through a quality gate: a small MLP that produces a per-evidence quality weight reflecting its estimated relevance and credibility. The encoded evidence then enters the **stance-aware group attention** module. First, a *stance mask* is constructed for each stance category (support, neutral, refute) by extracting the corresponding stance indicator from the evidence feature vector and applying a logical AND with the binary evidence mask, which indicates whether each evidence slot is occupied. The stance mask identifies which evidence items are both valid and belong to the target stance. Two derivative masks are then derived from the stance mask. The *padding mask* is a hard binary mask derived directly from the stance mask, serving as the key-padding mask in multi-head self-attention to prevent the model from attending to invalid or cross-stance positions. The *gated attention mask* is a continuous-valued mask obtained by element-wise multiplying the stance mask with the quality gate output, jointly encoding stance membership and source quality into a unified weighting factor. Within each track, the evidence embeddings pass through a multi-head self-attention layer governed by the padding mask, enabling intra-stance contextual interaction. The attention outputs are then element-wise multiplied by the gated attention mask to inject quality-aware weighting, and mean pooling compresses the variable-length evidence set into a fixed-dimensional stance aggregation vector. In the fusion stage, the original claim embedding and the three stance aggregation vectors are concatenated along the feature dimension, yielding a unified representation that captures the claim content alongside the evidential signal from all three stances. This representation is passed through a shared network for global feature integration, after which it is routed to two parallel output heads: a veracity decoder and a reliability decoder, producing the final veracity score and reliability score, respectively. More details of the feature definitions and network training are provided in Appendix E.2.

3.4 Tree Management

Once a node obtains its veracity score v_{n^*} and reliability score r_{n^*} from the evaluation network, the tree management module determines how the argument tree should evolve. The system operates through an iterative process of expansion, aggregation, and pruning. During expansion, if a node's self-reliability is too low for a confident judgment, the system invokes a decomposition LLM. The LLM breaks the claim into two to four sub-claims, each targeting a distinct verification dimension: geographic accuracy, entity identity, core event, causal relationship, source credibility, quantitative accuracy, or contextual background. Each sub-claim is assigned an importance weight, with all weights summing to one, and the resulting sub-claims are appended to the argument tree $\mathcal{T}$ as children of the current node and added to the processing queue $\mathcal{Q}$. After any structural change to the tree, scores are propagated bottom-up via the aggregation network. For each node, the network takes as input the node's own veracity and reliability scores alongside those of all processed children that reflect the combined evidence from the entire subtree. This update refreshes the root node's scores v_0 and r_0 to reflect all available evidence. The architecture and training details of the aggregation network are described in Appendix E.3.

Following aggregation, the system performs a top-down pruning pass. For any non-leaf node whose decisiveness flag $\phi_n = 1$, meaning its aggregated reliability r_n^{aggr} exceeds τ_r^{dec} and its aggregated veracity v_n^{aggr} falls outside $[\tau_v^-, \tau_v^+]$, all pending descendants are removed from $\mathcal{Q}$, avoiding unnecessary computation on branches that no longer affect the outcome. The loop then checks whether the convergence flag $\gamma = 1$ or the root's decisiveness flag $\phi_0 = 1$. Convergence holds when the range of v_0 over the last three iterations drops below τ_c and r_0^{aggr} exceeds τ_r^{exp}. If either condition is met, the system terminates and returns v_0 and r_0 as the final verdict. Otherwise, the next node is selected from $\mathcal{Q}$ according to its priority score, which is computed as a weighted combination of the parent node's uncertainty and the node's own importance weight assigned at decomposition time, and the next iteration begins.

4 Evaluation

We evaluate ToE from two complementary perspectives: general claim-verification performance on standard benchmarks and robustness under controlled GEO poisoning.

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate ToE on four fact-verification datasets. LIAR (Wang, 2017) contains 12,836 human-labeled political statements with six fine-grained veracity labels and speaker metadata. PolitiFact (Misra, 2022) contains 21,152 expert-verified statements from 2008 to 2022 across six verdict categories. Check-COVID (Wang et al., 2023) contains 1,504 COVID-19 claims verified against evidence from scientific journal articles. GPE (Wang et al., 2026b) provides manually verified claim–evidence pools with native six-class labels and controlled mixtures of benign and GEO-poisoned evidence.

Comparison Methods. We compare ToE against a range of LLM-based fact-checking methods. Direct prompting feeds the raw claim directly to an LLM and asks for a veracity judgment without any additional context. Z-CoT and DefGen (Lucas et al., 2023) are two structured prompting strategies proposed by the same work, eliciting step-by-step reasoning through zero-shot chain-of-thought and deductive generation, respectively, without external retrieval. AFaCTA (Ni et al., 2024) employs a multi-agent voting mechanism to aggregate judgments from multiple LLM instances. TELLER (Liu et al., 2024b) models the verification process through a dual cognitive and decision-making system. STEEL (Li et al., 2024a) performs iterative multi-round retrieval to progressively gather evidence before reaching a verdict. AdSent (Tahmasebi et al., 2026) neutralizes sentiment signals in the input claim to improve veracity judgment. To assess whether the gains of ToE generalize across model scales, we instantiate all methods on two backbone LLMs: DeepSeek-V3.2(MIT License) (DeepSeek-AI, 2025), a large-scale model, and gpt-oss-20b(Apache 2.0 license) (OpenAI, 2025), a smaller open-weight reasoning model released by OpenAI.

ToE Settings. The argument tree is constrained to a maximum depth of 5 and a maximum of $T_{\max} = 20$ iterations. The thresholds are set as $\tau_r^{\text{exp}} = 0.7$ (expansion), $\tau_r^{\text{dec}} = 0.7$ (decision), $\tau_v^+ = 0.7$, $\tau_v^- = 0.3$ (veracity bounds), and $\tau_c = 0.1$ (convergence). For the retrieval agent, each search action returns at most 3 results, and the total number of retrieved documents per node is capped at 10. Further implementation details and the source code are provided in the supplementary material.

4.2 RESULTS AND ANALYSIS

Table 1: Accuracy on traditional fact-verification benchmarks.

Method	DeepSeek-V3.2			GPT-OSS-20B		
	LIAR	**PolitiFact**	**checkCOVID**	**LIAR**	**PolitiFact**	**checkCOVID**
Direct	0.37	0.32	0.28	0.24	0.40	0.27
AFacTA	0.32	0.44	0.35	0.33	0.48	0.32
ZCoT	0.42	0.60	0.52	0.34	0.22	0.43
DefGen	0.51	0.38	0.36	0.37	0.57	0.40
STEEL	0.46	0.61	0.41	0.20	0.38	0.45
TELLER	0.45	0.68	0.56	0.63	**0.64**	**0.52**
Adsent	0.41	0.46	0.36	0.45	0.48	0.22
ToE	**0.62**	**0.73**	**0.59**	**0.68**	0.62	0.48

Table 1 evaluates ToE on traditional fact verification using LIAR, PolitiFact, and Check-COVID with two backbone LLMs. For consistent three-class accuracy, we map the original labels of each dataset to true, false, and uncertain. For LIAR, pants-fire, false, and barely-true map to false; half-true maps to uncertain; and mostly-true and true map to true. PolitiFact uses mostly-false in place of barely-true, which likewise maps to false. For Check-COVID, SUPPORT, REFUTE, and NOTENOUGHINFO map to true, false, and uncertain, respectively. The final ToE score is classified as false when $v < 0.4$, uncertain when $0.4 \leq v \leq 0.6$, and true when $v > 0.6$. For this experiment, ToE is trained on the LIAR training split and evaluated on the LIAR test split, PolitiFact, and Check-COVID without dataset-specific adaptation. For each dataset, we sample 100 claims with a fixed random seed and stratify by veracity label. All methods use the same claims in each trial, and each experiment is repeated ten times; Table 1 reports median accuracy. The three datasets cover distinct claim domains, and ToE performs consistently across them. The Direct baseline, which prompts the LLM to judge claims without external retrieval, achieves non-trivial accuracy on several datasets, possibly because

some benchmark claims overlap with knowledge seen during pretraining. Backbone capability also affects method performance: ZCoT scores 0.60 on PolitiFact with DeepSeek-V3.2 but 0.22 with GPT-OSS-20B. ToE improves over Direct in all six settings, showing that structured retrieval and evidence aggregation provide gains beyond parametric knowledge on these benchmarks.

Table 2: Accuracy under GEO poisoning on GPE.

Attack	Ratio	Direct Evid.	RAFTS	SAFE	STEEL	ToE
Claim-local retrieval						
Clean	0	52.7	49.8	46.2	52.5	**62.4**
ATA	1/3	39.2	31.5	27.1	30.6	**48.4**
	2/3	30.4	26.5	16.6	22.7	**46.6**
FakeGPT	1/3	46.1	39.2	28.8	39.5	**49.8**
	2/3	35.3	35.0	21.2	35.9	**50.0**
PoisonedRAG	1/3	39.3	43.4	28.4	38.2	**53.1**
	2/3	37.9	33.1	23.0	32.8	**50.2**
Claim-global retrieval						
Clean	0	52.2	47.5	44.5	51.7	**58.3**
ATA	1/3	37.9	33.4	27.3	29.3	**51.6**
	2/3	28.8	25.6	16.8	21.5	**53.5**
FakeGPT	1/3	44.0	38.9	32.6	40.1	**49.2**
	2/3	35.3	33.2	23.2	33.4	**51.1**
PoisonedRAG	1/3	44.4	39.7	30.6	37.5	**52.4**
	2/3	39.0	34.5	23.0	32.3	**48.1**

Table 2 evaluates robustness to GEO poisoning on the full GPE benchmark using its native six-class labels. Claim-local BM25 retrieves only from the current claim's evidence pool, whereas claim-global BM25 retrieves from the union of all claim pools. We evaluate clean evidence and contamination ratios of 1/3 and 2/3 under ATA, FakeGPT, and PoisonedRAG. For this experiment, ToE uses a separately trained six-class classifier. We compare against Direct Evidences, RAFTS, SAFE, and STEEL under the same BM25 evidence protocol; Direct Evidences supplies the GPE-retrieved evidence directly to the LLM for analysis and judgment. Under claim-local retrieval, ToE obtains 62.4%, 48.4%, and 46.6% accuracy for clean evidence and ATA contamination ratios of 1/3 and 2/3; the corresponding claim-global results are 58.3%, 51.6%, and 53.5%. ToE has the highest accuracy among the compared methods in all six clean/ATA conditions and all eight FakeGPT and PoisonedRAG conditions. The non-monotonic claim-global result is reported descriptively because different contamination ratios change the evidence mixture and BM25 ranking.

Table 3: Ablation study of the main ToE components.

Variant	0	1/3	2/3
Full ToE	**62.4**	**48.4**	**46.6**
w/o Claim Decomposition	43.1	43.1	37.8
w/o Tree Aggregation	40.9	45.0	35.3
w/o Learned Aggregation	47.5	48.1	41.2

Table 3 evaluates claim decomposition, tree aggregation, and learned aggregation on the full GPE benchmark under claim-local retrieval. The table reports native six-class accuracy for clean evidence and ATA contamination ratios of 1/3 and 2/3. Removing claim decomposition lowers accuracy to 43.1%, 43.1%, and 37.8%. Replacing learned aggregation with a fixed rule yields 47.5%, 48.1%, and 41.2%. Without bottom-up tree aggregation, direct reliability-and-importance-weighted aggregation over active descendant claims yields 40.9%, 45.0%, and 35.3%. This variant is evaluated on cases with at least one active child claim, as defined in the ablation protocol.

Figure 3 and Table 4 evaluate how the retrieval tools contribute to evidence collection. We use 30 claims from six categories, with five claims per category (two true, two false, and one uncertain), and record the selected evidence source at each retrieval step. The heatmap shows category-specific

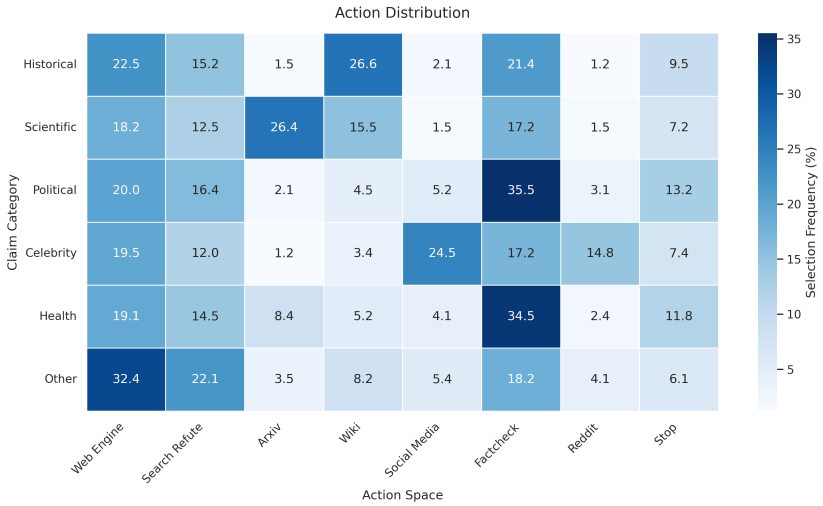


Figure 3: Heatmap of action distribution.

Table 4: Ablation study of the retrieval tool space.

Configuration	Overall	TRUE	FALSE	UNCERTAIN	Steps
Full Tool Space	80.0%	91.6%	75.0%	66.7%	5.4
w/o Academic	73.3%	75.0%	83.3%	50.0%	6.8
w/o Counter-Evidence	63.3%	83.3%	50.0%	50.0%	4.0
w/o Factcheck	56.6%	50.0%	75.0%	33.3%	7.8
w/o Social Media	63.3%	66.6%	83.3%	16.6%	3.8

source selection. ArXiv accounts for 26.4% of actions on scientific claims. Fact-checking sources account for 35.5% and 34.5% of actions on political and health claims, respectively. Social media reaches 24.5% on celebrity claims, while historical claims distribute actions across Web Engine, Wiki, and Factcheck. The heterogeneous Other category uses Web Engine most often, at 32.4%.

The tool-space ablation measures the effect of removing four functional groups: academic sources, counter-evidence search, fact-checking databases, and social media. Removing academic sources reduces overall accuracy from 80.0% to 73.3%; removing counter-evidence search reduces it to 63.3%. Removing fact-checking sources causes the largest drop, to 56.6%, consistent with their frequent use for political and health claims. Removing social media also reduces overall accuracy to 63.3% and uncertain-claim accuracy to 16.6%, consistent with its use for celebrity and recent-event evidence. Together, the source-selection frequencies and removal results show that the four tool groups provide complementary evidence and that the retrieval policy allocates them according to claim type.

5 Conclusion

We presented ToE, a hierarchical and explainable Byzantine-robust aggregation framework with dynamic multi-source evidence retrieval. By organizing claims and evidence in an argument tree, ToE uses evidence quality and cross-source consistency to control decomposition, pruning, and bottom-up score propagation rather than treating retrieved documents as equally trustworthy. Experiments on four datasets with two backbone LLMs, together with component and retrieval-tool ablations, show that ToE remains effective across standard verification benchmarks and controlled GEO-poisoning settings.

AI USE STATEMENT

In this work, we used generative AI tools to edit the manuscript, improve readability and presentation, and refine the manuscript structure. All AI-assisted content was reviewed and verified by the authors. We take responsibility for the final content of this work, including its text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

All datasets used in this work are publicly available and cited in the Experimental Setup section. The paper and appendices specify the evaluation protocol, model settings, network architectures, training procedures, and hyperparameters. The source code and configuration files are provided in the anonymous supplementary material to support reproduction of the reported experiments.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Pranjal Aggarwal, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, Karthik Narasimhan, and Ameet Deshpande. Geo: Generative engine optimization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5–16, 2024.

DeepSeek-AI. Deepseek-v3.2: Pushing the frontier of open large language models, 2025.

Yue Huang and Lichao Sun. Fakegpt: fake news generation, explanation and detection of large language models. *arXiv preprint arXiv:2310.05046*, 2023.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Guanghua Li, Wensheng Lu, Wei Zhang, Defu Lian, Kezhong Lu, Rui Mao, Kai Shu, and Hao Liao. Re-search for the truth: Multi-round retrieval-augmented large language models are strong fake news detectors. *arXiv preprint arXiv:2403.09747*, 2024a.

Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. Llms-as-judges: a comprehensive survey on llm-based evaluation methods. *arXiv preprint arXiv:2412.05579*, 2024b.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Hui Liu, Wenya Wang, Haoru Li, and Haoliang Li. Teller: A trustworthy framework for explainable, generalizable and controllable fake news detection. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15556–15583, 2024b.

Jason Lucas, Adaku Uchendu, Michiharu Yamashita, Jooyoung Lee, Shaurya Rohatgi, and Dongwon Lee. Fighting fire with fire: The dual role of llms in crafting and detecting elusive disinformation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14279–14305, 2023.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.

Rishabh Misra. Politifact fact check dataset, 09 2022.

Jingwei Ni, Minjing Shi, Dominik Stammbach, Mrinmaya Sachan, Elliott Ash, and Markus Leippold. Afacta: Assisting the annotation of factual claim detection with reliable llm annotators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1890–1912, 2024.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Sahar Tahmasebi, Eric Müller-Budack, and Ralph Ewerth. Robust fake news detection using large language models under adversarial sentiment attacks. *arXiv preprint arXiv:2601.15277*, 2026.

Gengyu Wang, Kate Harwood, Lawrence Chillrud, Amith Ananthram, Melanie Subbiah, and Kathleen McKeown. Check-covid: Fact-checking covid-19 news claims with scientific evidence. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 14114–14127, 2023.

William Yang Wang. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 2: short papers)*, pp. 422–426, 2017.

Zhaoqi Wang, Daqing He, Zijian Zhang, Ye Liu, Jiamou Liu, Zhirui Zeng, Zhan Qin, Zhen Li, Xin Li, Hongwei Yao, Jincheng An, Yong Liu, Yi Li, Qi Sun, Xiulei Liu, and Liehuang Zhu. Combating knowledge corruption in agent systems: A byzantine-tolerant secure collaborative rag framework. In *Proceedings of the ACM Web Conference 2026*, WWW '26. ACM, 2026a.

Zhaoqi Wang, Zijian Zhang, Xiaomei Yuan, Pengtao Kou, Jiamou Liu, Zhen Li, and Liehuang Zhu. Gpe: Evaluating robust evidence aggregation for fact verification under controllable geo-style poisoning. *arXiv preprint arXiv:2607.20730*, 2026b.

Ben Xue, Dan Liu, Lixiang Wang, Mingjie Sun, Peng Wang, Pengfei Zhang, Shaoyun Shi, Tianyu Xu, Yunhao Sha, Zhiqiang Liu, et al. Generative recommendation for large-scale advertising. *arXiv preprint arXiv:2602.22732*, 2026.

Zexuan Zhong, Ziqing Huang, Alexander Wettig, and Danqi Chen. Poisoning retrieval corpora by injecting adversarial passages. *arXiv preprint arXiv:2310.19156*, 2023.

Xinyi Zhou and Reza Zafarani. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Computing Surveys (CSUR)*, 53(5):1–40, 2020.

Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. Poisonedrag: Knowledge corruption attacks to retrieval-augmented generation of large language models. *arXiv preprint arXiv:2402.07867*, 2024.

A Case Study

We use a three-layer example from AdvFact to illustrate how ToE identifies AI-generated misinformation. The sample is deliberately deceptive, blending a real historical context (Jeff Weaver's allegations of procedural irregularities at the Nevada Democratic Convention) with fabricated details, including a falsified 2024 date and a fictitious CNN endorsement. ToE decomposes the root claim into three sub-claims at Layer 2, each along a different dimension, and further into atomic sub-claims at Layer 3. Layer 2 node 1 (Veracity 0.30) verifies the basic factual scaffolding of the claim. Its Layer 3

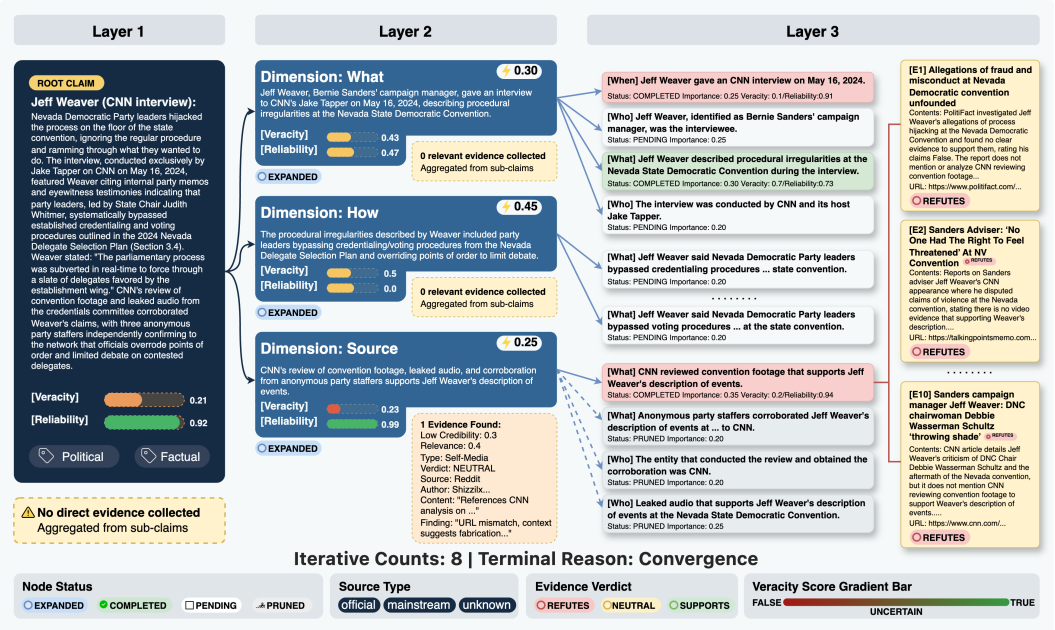
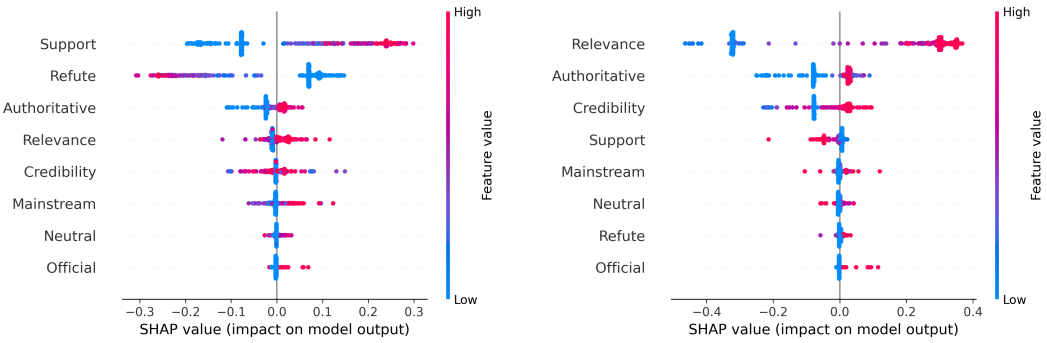


Figure 4: A qualitative ToE verification example.

children confirm that Weaver did give a CNN interview and was indeed Sanders' campaign manager, yielding moderate veracity scores. However, the specific procedural allegations lack corroborating evidence, which suppresses the node score. Layer 2 node 2 targets the alleged procedural details, but no relevant evidence is retrieved for any of its children. All nodes remain PENDING, falling back to default values of Veracity 0.5 and Reliability 0.0. Since aggregation is reliability-weighted, this node contributes no signal to the root. Layer 2 node 3 (Veracity 0.25) proves most decisive. ToE retrieves authoritative sources including PolitiFact, which explicitly rates the convention misconduct allegations as false and finds no evidence that CNN reviewed convention footage as claimed. The single evidence item found here originates from a Reddit post with a URL mismatch, flagged as a likely fabrication. This directly refutes the core manufactured detail that CNN officially corroborated Weaver's account, and the node achieves a reliability score of 0.99. This high-confidence refutation triggers the early convergence mechanism, terminating the search after only 8 iterations. PRUNED nodes are similarly assigned zero reliability weight, ensuring that redundant nodes do not distort the aggregated result. The false verdict thus propagates cleanly to the root claim, yielding a logically transparent result (Veracity 0.21). This case demonstrates that ToE not only provides a fine-grained and interpretable evidence chain, but also improves verification efficiency through dynamic pruning and convergence.

B Interpretability Analysis

To illustrate the semantic interpretability of the scoring model, we conduct a SHAP analysis on feature contributions during the evaluation phase using an XGBoost surrogate model (Lundberg & Lee, 2017; Lundberg et al., 2020). As shown in Figure 5, the two prediction targets exhibit distinctly different feature utilization patterns. In veracity prediction, Support Ratio and Refute Ratio dominate the model output in opposite directions: a higher proportion of supporting evidence yields larger SHAP values, while a higher proportion of refuting evidence strongly suppresses predictions. This indicates that the model assesses statement veracity through the net signal difference between opposing evidence types. In reliability prediction, Relevance Score dominates with a contribution magnitude far exceeding all other features (SHAP value span of ± 0.4); low-relevance evidence exerts a significant negative effect on reliability, while Source: Authoritative Ratio and Credibility Score serve as secondary features that provide quality gains. These patterns closely align with the cognitive logic of human fact-checking, confirming the semantic-level interpretability of the model.



(a) Veracity prediction.

(b) Reliability prediction.

Figure 5: SHAP beeswarm plots.

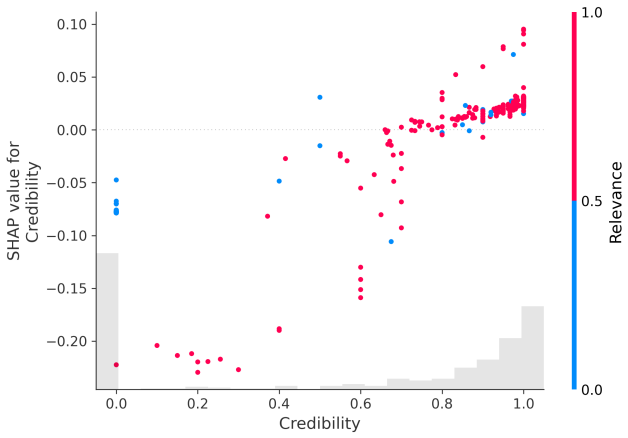


Figure 6: SHAP dependence plot.

To further examine interaction effects among features, we plot the SHAP dependence plot for Credibility Score on the reliability task, using Relevance Score as the interaction coloring variable (Figure 6). The plot reveals a clear nonlinear structure in the contribution of Credibility Score to reliability. When credibility is low (≤ 0.3), SHAP values concentrate around -0.2, and the model assigns a strong negative penalty regardless of Relevance Score, indicating that low credibility acts as a hard bottleneck for reliability independent of evidence relevance. When credibility rises above 0.8, SHAP values recover to near zero or slightly positive, and high-relevance samples are consistently higher than low-relevance ones, revealing a synergistic gain between Credibility Score and Relevance Score: only when both are jointly high does reliability prediction improve effectively. This interaction pattern further confirms that the model's judgment of evidence quality follows an intuitive and semantically coherent structure.

C Evaluation Metrics of the Evidence Evaluation Network

Table 5: Evaluation metrics of the evidence evaluation network.

Dimension	MAE	RMSE	Pearson r	R^2 Score
Veracity	0.0821	0.1594	0.8272	0.6817
Reliability	0.0632	0.1301	0.9472	0.8922

Table 5 reports the performance of the evidence evaluation network on a held-out test set constructed by random splitting the collected evidence-score dataset into 80% for training and 20% for testing.

The network achieves a Pearson correlation of 0.8272 and an R^2 of 0.6817 on the veracity dimension, and performs even more strongly on the reliability dimension, with a Pearson correlation of 0.9472 and an R^2 of 0.8922. The consistently low MAE and RMSE values across both dimensions confirm that the network produces well-calibrated continuous scores rather than coarse ordinal predictions, which is essential for the tree aggregation and pruning decisions that depend on precise score estimates.

D CLAIM-GLOBAL GPE TOKEN EFFICIENCY

Table 6: Claim-global GPE token efficiency.

	TCV ↓			$1000C/T$ ↑		
Method / attack	0	1/3	2/3	0	1/3	2/3
Direct Evidences	2.00	3.29	4.26	0.50	0.30	0.23
RAFTS	4.28	7.91	9.50	0.23	0.13	0.11
SAFE	8.43	16.32	26.91	0.12	0.06	0.04
STEEL	2.89	9.85	13.17	0.35	0.10	0.08
ToE / ATA	11.27	13.18	16.00	0.09	0.08	0.06
ToE / FakeGPT	–	9.44	9.87	–	0.11	0.10
ToE / PoisonedRAG	–	8.62	11.53	–	0.12	0.09

In Table 6, TCV is measured in thousands of tokens per correct verification, and $1000C/T$ is the number of correct verifications per 1K tokens. At claim-global contamination ratios 0, 1/3, and 2/3, ToE uses an average of 6.21, 6.70, and 8.21 retrieval rounds and processes 18.64, 20.11, and 24.62 retrieved documents per claim, respectively. The additional token use reflects repeated structured-evidence extraction and argument-tree maintenance.

E DETAILS OF NETWORK AND TRAINING PROCESS

E.1 RETRIEVAL POLICY NETWORK

The policy state is a 29-dimensional vector constructed from three groups of features. The first group encodes claim-level attributes: a 6-dimensional one-hot vector for the claim category (historical, scientific, political, health, celebrity, or other), a 3-dimensional one-hot vector for the locale (global, English, or Chinese), and five binary flags indicating whether the claim describes a recent event, a historical fact, a factual statement, an opinion, or a prediction. The second group encodes search progress: the absolute number of searches completed and the ratio of searches completed to the maximum allowed. The third group encodes the current evidence state: the total number of retrieved evidence items and their breakdown by stance (supporting, refuting, and neutral), the per-source search counts across all seven source types, and two binary flags indicating whether any official or authoritative source has been retrieved.

The policy network is a compact feed-forward network that maps the 29-dimensional state vector to a distribution over the 8 discrete actions. The encoder consists of three linear layers with hidden dimensions 128, 128, and 64, each followed by ReLU activation and dropout. The policy head applies two additional linear layers that project the 64-dimensional encoding to the action space, and a softmax is applied to produce the action distribution. At each search step t, a step reward r_t is computed as

$$r_t = w_r \cdot \Delta\text{rel}_t + w_v \cdot |\Delta\text{ver}_t| + w_e \cdot \min(n_t^{\text{new}}, 4) \cdot \delta_e + b_t - \delta_{\text{step}}, \tag{5}$$

where $\Delta\text{rel}_t = r_t^{\text{rel}} - r_{t-1}^{\text{rel}}$ is the change in the node's reliability score, $|\Delta\text{ver}_t|$ is the absolute change in the veracity score, n_t^{new} is the number of newly retrieved evidence items, and b_t is a bonus which added when the current reliability exceeds a high-confidence threshold τ_h or when an authoritative source is retrieved. The term δ_{step} is a fixed per-step penalty that encourages the agent to stop early when further search is unlikely to improve the judgment. The weights w_r, w_v, w_e, and the values of τ_h, δ_{step} are listed in Table 7.

Table 7: Hyperparameters of the policy network.

Parameter	Symbol	Value	Parameter	Symbol	Value
Reward			*GRPO Training*		
Reliability improv. weight	w_r	1.0	Group size	G	8
Veracity change weight	w_v	0.5	Discount factor	γ	0.99
Evidence bonus weight	w_e	0.05	Clipping range	ε	0.2
Per-step penalty	δ_{step}	0.01	Entropy coeff.	α	0.01
High-conf. threshold	τ_h	0.8	KL coeff.	β	0.1
Auth. source bonus	b_{auth}	0.1	Learning rate	—	1e-5
High-reliability bonus	b_{rel}	0.2	Max gradient norm	—	0.5
			Warm-up claims	N_{warm}	8
			Max search steps	—	5

The agent is trained online using Group Relative Policy Optimization (Shao et al., 2024). Completed trajectories are stored in a buffer, and a policy update is triggered whenever the buffer accumulates G trajectories. At each update, the discounted return $R_i = \sum_t \gamma^t r_t$ is computed for each trajectory in the group, and the group-normalized advantage is defined as

$$\hat{A}_i = \frac{R_i - \mu_G}{\sigma_G + \epsilon}, \tag{6}$$

where μ_G and σ_G are the mean and standard deviation of returns within the group. The policy is then updated using a clipped surrogate objective augmented with an entropy bonus and a KL penalty against the old policy, following the standard GRPO formulation. The network is optimized with AdamW, and gradients are clipped by norm before each update. During the first N_{warm} claims, the agent follows a heuristic warm-up policy instead of the learned policy. The heuristic assigns source-type priorities based on claim category and recency: for instance, scientific claims are directed toward Arxiv and Wikipedia, political and health claims toward fact-checking sites, and celebrity or recent-event claims toward social media sources. A stop action is favored once sufficient evidence has been collected or the reliability score exceeds τ_h. This warm-up phase ensures the trajectory buffer is populated with reasonable exploration behavior before the first GRPO update.

E.2 Evidence Evaluation Network

The evaluation network jointly predicts a veracity score and a reliability score for a claim node given its collected evidence. The network takes two inputs: a 12-dimensional claim feature vector and a padded evidence matrix of shape 20×8, accompanied by a binary mask indicating which evidence slots are occupied. The claim feature vector encodes three groups of attributes. The first group is a 6-dimensional one-hot vector for the claim category (historical, scientific, political, health, celebrity, or other). The second group is a 3-dimensional one-hot vector for the locale (Chinese, English, or global). The third group consists of three binary flags indicating whether the claim describes a recent event, a historical fact, or a factual statement. Each evidence item is represented as an 8-dimensional vector. The first three dimensions are one-hot stance indicators (supporting, refuting, or neutral). The next three dimensions are one-hot source-type indicators (official, authoritative, or mainstream). The final two dimensions are continuous scores: a relevance score and a credibility score, both in $[0, 1]$, assigned by the analysis LLM during the evidence extraction stage. Up to 20 evidence items are retained per node; if fewer are available, the remaining slots are zero-padded and masked out. Training data is collected during the fact-checking process. After the retrieval agent finishes collecting evidence for a node, the full evidence set and claim attributes are submitted to an LLM judge, which produces a veracity score $\tilde{v} \in [0, 1]$ and a reliability score $\tilde{r} \in [0, 1]$. The LLM judge evaluates veracity based on the degree to which the collected evidence supports or contradicts the claim, and evaluates reliability based on source authority, evidence quantity, cross-source consistency, and the directness with which the evidence addresses the specific claim. Following the LLM-as-a-Judge paradigm (Li et al., 2024b), samples where the judge's scores contradict the evidence stance distribution or source credibility are discarded prior to training. These LLM-assigned scores serve as soft training targets. Given a target pair $(\tilde{v}, \tilde{r})$, the network is trained to minimize the binary cross-entropy loss:

$$\mathcal{L} = \text{BCE}(\hat{v},\, \tilde{v}) + \text{BCE}(\hat{r},\, \tilde{r}), \tag{7}$$

where $\hat{v}$ and $\hat{r}$ are the network's predicted scores. The network is optimized with Adam (learning rate 10^{-3}) and gradient norm clipping at 1.0. Once sufficient labeled samples have been collected, the network can be fine-tuned offline from the dataset.

E.3 Tree Management and Aggregation Details

Algorithm 1 Tree management.

Input: Initial claim c_0, maximum iterations $T_{\max}$, and thresholds $\tau_r^{\text{exp}}, \tau_r^{\text{dec}}, \tau_v^+, \tau_v^-, \tau_c$.
Output: Final root veracity score v_0 and reliability score r_0.

Initialize tree $\mathcal{T}$ with root node $n_0 \leftarrow c_0$
Initialize queue $\mathcal{Q} \leftarrow \{n_0\}$
for $t = 1, 2, \ldots, T_{\max}$ **do**
  if $\mathcal{Q}$ is empty **then**
    break
  end if
  $n^* \leftarrow \arg\max_{n \in \mathcal{Q}} \textsc{Priority}(n)$
  $\mathcal{Q} \leftarrow \mathcal{Q} \setminus \{n^*\}$
  $\mathcal{E} \leftarrow \textsc{GetEvidence}(n^*)$
  $v_{n^*}, r_{n^*} \leftarrow \textsc{Evaluate}(n^*, \mathcal{E})$
  if $\delta_{n^*} = 1$ **then**
    $\mathcal{S} \leftarrow \textsc{Decompose}(n^*)$
    $\mathcal{T}.\textsc{Add}(n^*, \mathcal{S})$
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \mathcal{S}$
  end if
  $v_0, r_0 \leftarrow \textsc{Aggregate}(\mathcal{T})$
  $\mathcal{T}, \mathcal{Q} \leftarrow \textsc{Prune}(\mathcal{T}, \mathcal{Q})$
  if $\gamma = 1$ **or** $\phi_0 = 1$ **then**
    break
  end if
end for
return (v_0, r_0)

Algorithm 1 summarizes the full procedure of the ToE verification, where three binary flags govern the control flow: the expansion flag δ_n, the decisiveness flag ϕ_n, and the convergence flag γ. These flags are defined in terms of five thresholds: the expansion reliability threshold τ_r^{exp}, the decision reliability threshold τ_r^{dec}, the upper and lower veracity thresholds τ_v^+ and τ_v^-, and the convergence tolerance τ_c. Specifically:

- $\delta_n = 1$ if $r_n^{\text{self}} < \tau_r^{\text{exp}}$, indicating that the node's self-reliability is too low for a confident judgment and further evidence is needed; the root node is always expanded on its first evaluation regardless of this condition.
- $\phi_n = 1$ if $r_n^{\text{aggr}} > \tau_r^{\text{dec}}$ and $v_n^{\text{aggr}} \notin [\tau_v^-, \tau_v^+]$, indicating that the aggregated evidence is sufficiently reliable and the verdict is unambiguously true or false.
- $\gamma = 1$ if the range of v_0 over the last three iterations is below τ_c and $r_0^{\text{aggr}} > \tau_r^{\text{exp}}$, indicating that the root veracity score has stabilized and further iterations are unlikely to change the outcome.

If the expansion flag $\delta_{n^*} = 1$, indicating that the self-reliability r_{n^*} falls below τ_r^{exp} and the node is eligible for expansion, the system invokes the decomposition LLM to generate sub-claims. Following aggregation, the system performs a top-down pruning pass. For any non-leaf node whose decisiveness flag $\phi_n = 1$, meaning its aggregated reliability r_n^{aggr} exceeds τ_r^{dec} and its aggregated veracity v_n^{aggr} falls outside $[\tau_v^-, \tau_v^+]$, all pending descendants are removed from $\mathcal{Q}$. The loop then checks whether the convergence flag $\gamma = 1$ or the root's decisiveness flag $\phi_0 = 1$. Convergence holds when the range of v_0 over the last three iterations drops below τ_c and r_0^{aggr} exceeds τ_r^{exp}. If either condition is met, the system terminates and returns v_0 and r_0 as the final verdict. Otherwise, the next node is selected from $\mathcal{Q}$ according to its priority score, which is computed as a weighted combination of the parent

node's uncertainty and the node's own importance weight assigned at decomposition time, and the next iteration begins.

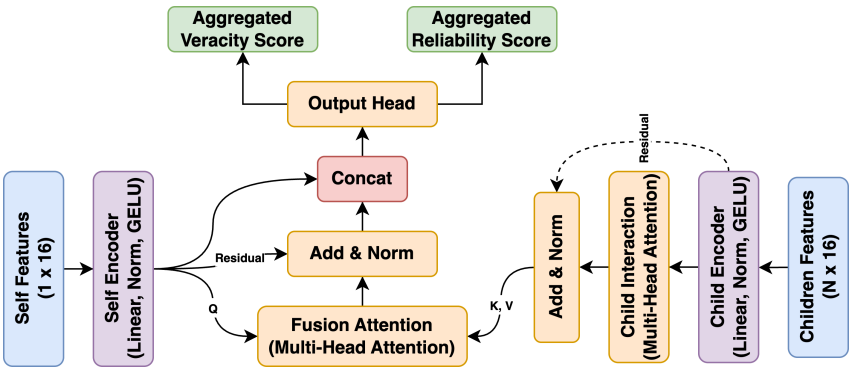


Figure 7: Tree aggregation network.

The aggregation network computes updated veracity and reliability scores for each node in the argument tree by combining information from the node itself and its evaluated children. As illustrated in Fig. 7, the network takes two inputs for any target node: the node's own 16-dimensional feature vector, which encodes basic attributes and a one-hot dimension type, and the 16-dimensional feature vectors of all evaluated child nodes. In the encoding stage, a self-encoder maps the node's own features into a high-dimensional embedding. The self-encoder consists of a linear projection, layer normalization, GELU activation, and dropout. A child encoder of identical structure maps each child's feature vector into a corresponding embedding, producing a sequence of child embeddings. A multi-head self-attention module then processes this sequence to capture cooperative or conflicting relationships among siblings, with the output refined through a residual connection and layer normalization. In the fusion stage, a cross-attention module uses the node's self-embedding as the query and the interaction-refined child representations as keys and values, allowing the node to selectively aggregate information from its children. The attended representation is passed through a residual connection and normalization, then concatenated with the original self-embedding along the channel dimension. The concatenated vector is fed into an output head consisting of two MLP layers with a Sigmoid activation, which simultaneously produces the aggregated veracity score and the aggregated reliability score for the node.

Before neural training begins, veracity and reliability scores are propagated bottom-up through the tree using a rule-based aggregation scheme (Algorithm 2), which serves both as an interpretable fallback and as a source of soft supervision targets. For leaf nodes and unexpanded nodes, the aggregated scores are set equal to their self-assessed values. For internal nodes, the aggregated veracity is computed as a reliability-importance-weighted average of the children's aggregated veracity scores, where each child's weight is proportional to the product of its importance score and its aggregated reliability. The child-side aggregated reliability is derived from the maximum child reliability, further scaled by a consistency factor $\eta = \max(0.5, 1 - \sigma_v)$, where σ_v is the standard deviation of the children's veracity estimates; higher disagreement among children reduces η, thereby lowering the child-side reliability. Finally, the node's aggregated veracity and reliability are each computed as a convex combination of the self-assessed value and the corresponding child-side value, with mixing weights determined by the relative self reliability and child-side reliability. These rule-based scores are stored in the tree and later used as soft targets during neural training.

The network is trained bottom-up on completed argument trees. For leaf nodes and unexpanded nodes, scores are taken directly from their self-assessed values without passing through the network. For all other nodes, the network is applied recursively from the deepest level to the root, so that each node's child embeddings already incorporate aggregated information from lower levels. Training uses a two-component loss. The primary component is a supervised loss on the root node, where the network's predicted veracity and reliability are compared against the ground-truth veracity label $y_v \in \{0, 1\}$ and a reliability target y_r using binary cross-entropy:

$$\mathcal{L}_{\text{root}} = w_{\text{root}} \left[\text{BCE}(\hat{v}_{\text{root}}, y_v) + \text{BCE}(\hat{r}_{\text{root}}, y_r) \right]. \tag{8}$$

Algorithm 2 Rule-based score aggregation.

Input: Node n with self-assessed scores $(v_n^{\text{self}}, r_n^{\text{self}})$, active children $\mathcal{C}$.
Output: Aggregated scores $(v_n^{\text{aggr}}, r_n^{\text{aggr}})$.

$\mathcal{C}' \leftarrow \{c \in \mathcal{C} \mid \text{status}(c) \in \{\text{EXPANDED}, \text{COMPLETED}\}\}$
if $\mathcal{C}' = \emptyset$ **then**
  return $(v_n^{\text{self}},\ r_n^{\text{self}})$
end if
for each $c \in \mathcal{C}'$ **do**
  $w_c \leftarrow \text{imp}(c) \cdot (r_c^{\text{aggr}} + \varepsilon)$
end for
$W \leftarrow \sum_{c \in \mathcal{C}'} w_c$
$v_{\mathcal{C}} \leftarrow \sum_{c \in \mathcal{C}'} \frac{w_c}{W} \cdot v_c^{\text{aggr}}$
$r_{\text{base}} \leftarrow \max_{c \in \mathcal{C}'} r_c^{\text{aggr}}$
if $|\mathcal{C}'| \geq 2$ **then**
  $\bar{v} \leftarrow \frac{1}{|\mathcal{C}'|} \sum_{c \in \mathcal{C}'} v_c^{\text{aggr}}$
  $\sigma_v \leftarrow \sqrt{\frac{1}{|\mathcal{C}'|} \sum_{c \in \mathcal{C}'} \left(v_c^{\text{aggr}} - \bar{v}\right)^2}$
  $r_{\mathcal{C}} \leftarrow r_{\text{base}} \cdot \max(0.5,\ 1 - \sigma_v)$
else
  $r_{\mathcal{C}} \leftarrow r_{\text{base}}$
end if
$\alpha \leftarrow \dfrac{r_n^{\text{self}} + \varepsilon}{r_n^{\text{self}} + r_{\mathcal{C}} + 2\varepsilon}$
$v_n^{\text{aggr}} \leftarrow \alpha \cdot v_n^{\text{self}} + (1 - \alpha) \cdot v_{\mathcal{C}}$
$r_n^{\text{aggr}} \leftarrow \alpha \cdot r_n^{\text{self}} + (1 - \alpha) \cdot r_{\mathcal{C}}$
return $(v_n^{\text{aggr}},\ r_n^{\text{aggr}})$

The secondary component is an imitation loss over all non-root nodes, which encourages the network's predictions to remain consistent with the rule-based aggregation scores. For each active non-root node i, the rule-based scores $(\tilde{v}_i, \tilde{r}_i)$ serve as soft targets:

$$\mathcal{L}_{\text{imit}} = w_{\text{imit}} \cdot \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \left[\text{BCE}(\hat{v}_i,\ \tilde{v}_i) + \text{BCE}(\hat{r}_i,\ \tilde{r}_i)\right], \tag{9}$$

where $\mathcal{N}$ denotes the set of active non-root nodes. The total loss is $\mathcal{L} = \mathcal{L}_{\text{root}} + \mathcal{L}_{\text{imit}}$, with $w_{\text{root}} = 1.0$ and $w_{\text{imit}} = 0.1$. The network is optimized with AdamW (learning rate 10^{-4}, weight decay 10^{-5}) and gradient norm clipping at 1.0. Trees in which the root has no active children are skipped during training.