ART: Adaptive Reasoning Trees for Explainable Claim Verification

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are powerful candidates for complex decision-making, leveraging vast encoded knowledge and remarkable zero-shot abilities. However, their adoption in high-stakes environments is hindered by their opacity; their outputs lack faithful explanations and cannot be effectively contested to correct errors, undermining trustworthiness. In this paper, we propose *ART (Adaptive Reasoning Trees)*, a hierarchical method for claim verification. The process begins with a root claim, which branches into supporting and attacking child arguments. An argument's strength is determined bottom-up via a pairwise tournament of its children, adjudicated by a judge LLM, allowing a final, transparent and contestable verdict to be systematically derived which is missing in methods like Chain-of-Thought (CoT). We empirically validate ART on multiple datasets, analyzing different argument generators and comparison strategies. Our findings show that ART's structured reasoning outperforms strong baselines, establishing a new benchmark for explainable claim verification which is more reliable and ensures clarity in the overall decision making step.

1 Introduction

The advent of Large Language Models (LLMs) has marked a significant milestone, primarily due to their advanced reasoning capabilities (Grattafiori et al., 2024; OpenAI et al., 2024). This has enabled novel solutions to a wide array of complex tasks, with their impact being especially profound in specialized domains requiring nuanced judgment, such as medicine (Maity and Saikia, 2025; Zhang et al., 2023) and law (Kant et al., 2025). Despite the rapid advancement of their reasoning capabilities, a significant and persistent challenge in Large Language Models (LLMs) is their propensity for hallucination (Huang et al., 2025b; Agrawal et al., 2024; Guerreiro et al., 2023). This phenomenon produces content that appears credible but is inaccurate or illogical. Such unreliability poses a critical barrier to their deployment in truth-sensitive applications. For instance, in automated claim verification, a hallucinating model could incorrectly validate misinformation (Augenstein et al., 2023a), while in generating counter-narratives to combat hate speech, it could inadvertently introduce falsehoods, undermining the very purpose of the intervention (Wadhwa et al., 2024). A further challenge is the inherent opacity of LLM reasoning, which conflicts with the demands of modern decision-making. Processes in critical domains require a high degree of transparency to allow for auditing, accountability, and the ability for users to contest an outcome (Leofante et al., 2024; Henin and Le Métayer, 2022). Because the internal logic of LLMs is largely inscrutable, they fail to meet this requirement, creating a fundamental barrier to their responsible deployment in high-stakes environments.

To address these challenges, we propose ***ART (Adaptive Reasoning Trees)***, a novel framework designed to enhance the transparency of automated claim verification. ART decomposes a claim into a hierarchy of supporting and attacking arguments, which then compete in a pairwise tournament adjudicated by a judge LLM. To ensure objectivity, our framework employs a dedicated LLM-as-a-Judge (Rad et al., 2025; Adlakha et al., 2024a) to perform pairwise comparisons and determine the relative strength of competing arguments. This role is deliberately separated from argument generation to prevent the inherent self-affirmation bias of a model evaluating its own outputs, which is critical for the impartiality and robustness of the final verdict.

In short, our contributions are three-fold:

- *ART* employs a novel tree-based structure. It uses specialized ***Attack*** and ***Support*** LLMs to generate arguments for and against a claim,

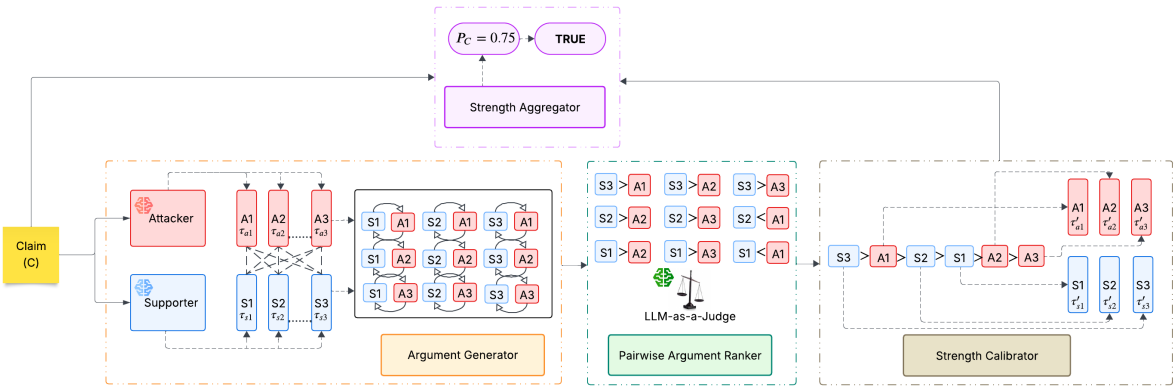

Figure 1: Overview of the ART (Adaptive Reasoning Trees) framework with a tree of *depth*=1 and *breadth*=3 for illustration. Each breadth adds b support and attack arguments. A claim is first processed by the **Argument Generator**, which creates a tree of supporting and attacking arguments. Arguments of opposing stances are then pitted against each other in a pairwise tournament. The **Pairwise Argument Ranker**, an LLM-as-a-judge, evaluates these pairs to dynamically update the strength of each argument. Finally, the **Strength Aggregator** consolidates these updated scores into a final probability for the claim's veracity.

assigning each an initial strength score.

- To the best of our knowledge, we are the first to introduce a tournament-style framework for claim verification where every support argument is pitted against every attack argument in a series of pairwise competitions. The outcomes are then used to dynamically update each argument's strength score via the *Bradley-Terry*[1] model, which informs the final prediction.
- ART achieves state-of-the-art results on multiple claim verification datasets when tested on a variety of open-source LLMs as compared to other baselines like *ArgLLM* (Freedman et al., 2025), Direct Prompting and Chain-of-thought (CoT) (Wei et al., 2022).

Note. The terms "claim" and "argument" are used interchangeably.

2 Related Work

2.1 Explainable LLM-based Decision Making

Large language models have demonstrated impressive reasoning abilities, but explaining their decisions in a trustworthy manner remains challenging. **Chain-of-Thought (CoT)** prompting is a common approach to induce step-by-step explanations from LLMs, yielding intermediate reasoning steps rather than just final answers (Wei et al., 2022; Kojima et al., 2022). While CoT can improve performance on complex tasks, its generated rationales are not guaranteed to be *faithful* to the model's actual decision process or even factually correct. Models often produce convincing but *hallucinated* explanations that violate factual context (Zhao et al., 2024; Huang et al., 2025a). This unreliability undermines trust, especially in high-stakes domains. Moreover, slight variations in prompts or instructions can significantly alter LLM outputs, showing brittleness in instruction-following (Singh et al., 2024). Another challenge is evaluating the correctness of LLM-generated explanations. Human evaluation is costly, and having a model judge its own reasoning introduces bias. Recent work proposes using separate LLMs or fine-tuned modules as judges (Chan et al., 2023; Adlakha et al., 2024b). While this direct usage of **LLM-as-a-Judge** paradigm improves consistency, judges drawn from the same model family may still inherit the generator's biases unless a proper separation of roles and multiple perspectives are enforced. To address this, ART explicitly separates the roles of *solver* and *evaluator*: supporting and attacking arguments are generated by dedicated models, and their conflicts are adjudicated by an independent Judge LLM. By structuring reasoning as a tree of opposing arguments and resolving disputes through impartial judging, ART ensures that each step of the decision process can be transparently verified and, if necessary, contested.

2.2 Automated Claim Verification

Automated fact-checking has been studied extensively in NLP. Early approaches retrieved evidence from Wikipedia or knowledge bases and

[1] https://tinyurl.com/4x393csa

applied classifiers to label claims as supported or refuted (Thorne et al., 2018; Nakov et al., 2021; Shi et al., 2016). Retrieval-augmented generation (RAG) further improved factual grounding by conditioning on external sources (Lewis et al., 2020; Chen et al., 2023). However, these systems often provide labels without detailed justifications (Guo et al., 2021).

Recent work has turned to LLMs for claim verification, aiming to improve accuracy over the traditional three-step pipeline by leveraging their generative ability to produce justifications (Guo et al., 2021; Vladika et al., 2025). Zero-shot or few-shot CoT prompting allows models like GPT-3/4 to generate verdicts with rationale (Lee et al., 2023), but these rationales may contain hallucinations or omit critical counter-evidence (Augenstein et al., 2023b). Argument mining approaches aim to improve transparency by generating pro/con arguments (Hidey et al., 2018; Atanasova et al., 2020). Our work differs in that ART structures reasoning as a hierarchical argumentation tree. Each leaf is an evidence-backed argument for or against the claim, and the verdict emerges through systematic tournament-style evaluation.

2.3 Multi-Agent Debate

Ensembling multiple reasoning paths or models improves robustness. Self-consistency decoding aggregates multiple CoT samples to reduce individual errors (Wang et al., 2022). Multi-agent systems extend this by having multiple LLMs collaborate or compete. The idea of AI debate was introduced by (Irving et al., 2018), where two agents argue and a judge makes the decision. Recent implementations show that multi-agent debate improves factual accuracy (Du et al., 2023; Khan et al., 2024). Debate has also been used for evaluation, e.g., ChatEval (Chan et al., 2023) and Debate Helps (Michael et al., 2023).

Another line of work uses specialized roles or personas for different LLMs. Wang et al. (Wang et al., 2023) explored multi-persona collaboration, while Jung et al. (Jung et al., 2022) proposed critic LLMs to identify flaws. These role-based ensembles demonstrate the benefit of diversity in reasoning styles. ART builds on this idea with distinct *Supporter*, *Attacker*, and *Judge* roles, and introduces a structured tournament that ensures balanced evaluation between opposing views.

2.4 Comparison with ArgLLMs

Our work is related to the recent proposal of *Argumentative LLMs (ArgLLMs)* (Freedman et al., 2025), which embeds LLM-generated pro/con arguments into formal *quantitative bipolar argumentation frameworks* (QBAFs). Within this paradigm, argument strengths are fixed and aggregated using deterministic semantics (e.g., DF-QuAD), yielding provable properties of contestability and logical soundness. This line of work makes an important contribution by grounding LLM outputs in formal argumentation theory. ART, rather than imposing static semantics, adopts a *dynamic pairwise tournament* mechanism: every support argument is directly compared against every attack argument, with outcomes adjudicated by an independent Judge LLM. This shift enables reasoning to adapt flexibly to the relative persuasiveness of arguments, rather than relying on pre-defined aggregation rules. Moreover, ART enforces *role specialization* by explicitly separating the Supporter, Attacker, and Judge models. This avoids the *self-affirmation bias* inherent in ArgLLMs, where a single model family is often reused for both argument generation and evaluation. By diversifying roles across models, ART enhances impartiality and robustness.

In summary, ART builds on the intuition that argumentative prompting improves transparency, but diverges from ArgLLMs by introducing a dynamic, role-separated, and empirically validated framework. This makes ART better suited for real-world claim verification tasks, where adaptability and benchmarked performance are as critical as theoretical guarantees.

3 Methodology

ART structures the verification process as a binary reasoning tree in which the root node, referred to as the *claim node*, represents the input claim under examination, and all other nodes are *argument nodes* that recursively support or attack their parent. This claim verification process is explicit with each comparison in the tree contributing to the overall decision. ART operates in four main phases: *argument generation*, *pairwise argument ranking*, *strength calibration* and *strength aggregation*. Figure 1 presents an overview of ART showing an example tree with a depth of one and a breadth of three.

3.1 Argument Generation

Each argument node is instantiated by a debater LLM prompted with a specific persona of either *supporter* (S) or *attacker* (A). The supporter produces arguments in favor of the parent node, while the attacker generates counterarguments against it. Although in principle the reasoning process could grow without bound, we cap the maximum tree depth at *two* in this work to balance reasoning depth and computational efficiency. **Note:** Same LLM is used for both support and attack.

3.2 Pairwise Argument Ranking

Once generated, arguments are evaluated through a comparison procedure by an LLM-as-a-judge (J) (Rad et al., 2025; Adlakha et al., 2024a). For each pair of argument nodes, the scoring LLM selects which of the supporting or attacking child argument is more persuasive given the parent node argument. A complete traversal of the tree yields $N = \frac{(2b)^{d+1}-1}{2b-1}$ nodes of depth d and breadth b with $\frac{(2b)^{d}-1}{2b-1} * b^2$ pairwise comparisons (see Appendix E). Importantly, ART is *adaptive*: subtrees whose comparisons fall below a confidence threshold can be selectively pruned during *argument verdict* to improve the accuracy of the final decision.

Let p be a parent argument node in the tree, with k supporting and attacking children defined as the sets: $\mathcal{S}(p) = \{S_1, S_2, \ldots, S_k\}$, and $\mathcal{A}(p) = \{A_1, A_2, \ldots, A_k\}$.

For each pair (a, b) where $a \in \mathcal{S}(p)$ and $b \in \mathcal{A}(p)$, a scoring LLM, J, selects the more persuasive argument:

$$\text{outcome}(a, b) = J(a, b \mid p) \in \{a, b\}$$

3.3 Strength Calibration

For each argument $i \in \mathcal{S}(p) \cup \mathcal{A}(p)$, the judge J also calculates its intrinsic strength, denoted τ_i, referring to how strongly its supports or attacks its parent claim similar to (Freedman et al., 2025). While an intrinsic strength score, τ_i, provides a baseline measure for an argument, this absolute evaluation, performed in isolation by a judge model J, is insufficient for a robust analysis. Such isolated scores struggle to handle redundancy, are susceptible to inconsistent scaling and bias from the judge model, and fail to establish a clear relative ranking.

In contrast, pairwise scoring addresses these limitations by reframing the task as a more constrained and reliable relative judgment: determining which of two arguments is stronger, a simpler task for an LLM (Carterette et al., 2008). This comparative approach naturally produces a robust ranking that surfaces nuanced qualitative differences and accounts for semantic overlap, which absolute scoring methods cannot capture. This presents a trade-off between the fast but potentially biased intrinsic scores and the robust but computationally expensive pairwise comparisons. To balance this, our approach uses the **Bradley-Terry (BT) model to calibrate the initial intrinsic strengths**. The BT model analyzes the tournament outcomes to derive a purely relative performance score for each argument. This data-driven score then acts as a scaling factor, adjusting the initial intrinsic value τ_a up or down based on its competitive success. This hybrid approach retains the baseline evaluation while refining it with robust, comparative data.

Bradley–Terry Calibration. Each child argument $i \in \mathcal{S}(p) \cup \mathcal{A}(p)$ has a latent strength $\theta_i > 0$. For $a \in \mathcal{S}(p)$ and $b \in \mathcal{A}(p)$, the probability that a is judged stronger than b is

$$P(a \succ b) \;=\; \frac{\theta_a}{\theta_a + \theta_b} \tag{1}$$

Let $U(p) = \mathcal{S}(p) \cup \mathcal{A}(p)$. We estimate θ by fixed-point iteration using only cross-pairs $(a, b) \in \mathcal{S}(p) \times \mathcal{A}(p)$, then normalize:

$$\begin{aligned}
\theta_a^{(t+1)} &= \frac{\sum_{b \in \mathcal{A}(p)} \tau_{ab}}{\sum_{b \in \mathcal{A}(p)} \frac{\tau_{ab}+\tau_{ba}}{\theta_a^{(t)}+\theta_b^{(t)}} + \varepsilon}, \quad a \in \mathcal{S}(p), \\
\theta_b^{(t+1)} &= \frac{\sum_{a \in \mathcal{S}(p)} \tau_{ba}}{\sum_{a \in \mathcal{S}(p)} \frac{\tau_{ab}+\tau_{ba}}{\theta_a^{(t)}+\theta_b^{(t)}} + \varepsilon}, \quad b \in \mathcal{A}(p), \\
\sum_{i \in U(p)} \theta_i^{(t+1)} &= 1 \text{ s.t. } \theta_i \in [0, 1]
\end{aligned} \tag{2}$$

Here, τ_{ab} counts judgments where supporter a is stronger than attacker b (and τ_{ba} is the reverse), and $\varepsilon > 0$ prevents division by zero. Eq. 2 is an *Majorize–Minimize* (MM) step for the Bradley–Terry log-likelihood, ensuring monotone ascent and scale-invariance to pairwise counts (duplicates or batch reweighting leave strengths unchanged up to normalization). These properties justify early stopping and yield stable, reproducible calibration (Proposition 2; Algorithm 1).

$$\begin{aligned}
\tau_i' = \text{clip}_{[0,1]}\big((1-\lambda)\,\tau_i + \lambda\,\theta_i\big), \\
i \in U(p), \lambda \in [0, 1]
\end{aligned} \tag{3}$$

We update each argument's weight by *blending* its current value with its BT score (Eq. 3). Specifically, the blend parameter $\lambda \in [0, 1]$ controls how much we trust the BT score—larger λ moves the weight closer to θ_i—and we clip the result to $[0, 1]$ to avoid extremes. This convex update keeps values bounded, smooths noisy estimates, and still reflects the pairwise outcomes.

3.4 Strength Aggregation

The outcomes of the local comparisons are aggregated upward through the tree until a final decision is reached at the claim node. This final stage produces both a binary classification label and an interpretable reasoning trace, showing how the balance of supporting and attacking arguments led to the system's conclusion.

Final Claim Score. After BT calibration step, the strength of each argument p is updated in topological order. For p with attackers $\mathrm{A}(p)$ and supporters $\mathrm{S}(p)$, the ProductAggregation term is calculated as

$$\alpha_p = \prod_{a \in \mathrm{S}(p)} (1 - s_a) - \prod_{b \in \mathrm{A}(p)} (1 - s_b), \quad (4)$$

where s_a and s_b are the propagated strengths of the respective children.

Ultimately, the final score of the parent argument node becomes

$$s_p = \begin{cases} \tau'_p - \alpha_p * \tau'_p, & \alpha_p > 0, \\ \tau'_p - \alpha_p * (1 - \tau'_p), & \alpha_p \leq 0, \end{cases} \quad (5)$$

where τ'_p is the BT-calibrated intrinsic weight of p.

The score of the root claim C is calculated as

$$P(C) = s_C \quad (6)$$

Argument strengths propagate via a DF-QuAD–style quantitative semantics (Rago et al., 2016): attacker/supporter effects are aggregated (Eq. 4) and node strengths updated (Eq. 5) deterministically, yielding smooth, discontinuity-free propagation (see Proposition 3). The final claim score $P(C)$ (Eq. 6) is labeled **True** if $P(C) > 0.5$, else **False**. We set the classification *threshold* to be 0.5.

4 Experiments

4.1 Datasets

MedQA (Jin et al., 2020) dataset comprises multiple-choice medical examination questions in English, Simplified Chinese, and Traditional Chinese. Each example contains a clinical vignette, a question, and five answer choices, with exactly one correct answer. We utilize the test split of the English subset. Since the original dataset is in multiple-choice format with answer choices typically expressed as short phrases, prior work has converted these into declarative statements: each answer choice is rewritten as a complete sentence, with the statement corresponding to the correct answer labeled as true (Freedman et al., 2025). The processed test set contains 250 positive and 250 negative examples.

StrategyQA (Geva et al., 2021) dataset consists of open-domain questions requiring multi-hop reasoning over common sense and factual knowledge. Unlike MedQA, each example in StrategyQA contains only a question without additional context, with answers provided as simple phrases. Similar to MedQA, prior work has processed StrategyQA by converting each question into a declarative statement with an associated truth label (Freedman et al., 2025). The processed test set is balanced with 250 positive and 250 negative examples.

TruthfulQA (Lin et al., 2022) dataset comprises of 250 positive and 250 negative claims respectively and is specifically used to evaluate LLMs to gauge if they can correctly identify truthful answers without being deceived by common misconceptions and falsehoods.

4.2 Baselines and Models

Models. ART is flexible to the choice of underlying language model. In this paper, we focus on widely used open-source LLMs that combine strong performance across diverse tasks with easy replicability of results. Specifically, we experiment with *Llama 3.1 3B-Instruct* and *Llama 3.1 8B-Instruct*, as well as the larger *Llama 3.3 70B-Instruct* (Grattafiori et al., 2024), together with *Mistral-7B Instruct-v0.2* (Jiang et al., 2023) and *Mixtral-8x7B-Instruct-v0.1* (Jiang et al., 2024). Unless otherwise specified, all models are queried with the following decoding parameters: `"temperature": 0.2, "max_new_tokens": 512, "top_p": 0.95`. All models are served through the OpenAI-compatible vLLM[2] API and batch executed on 1-4 NVIDIA A100 GPUs depending on model sizes.

[2] https://docs.vllm.ai/en/latest/

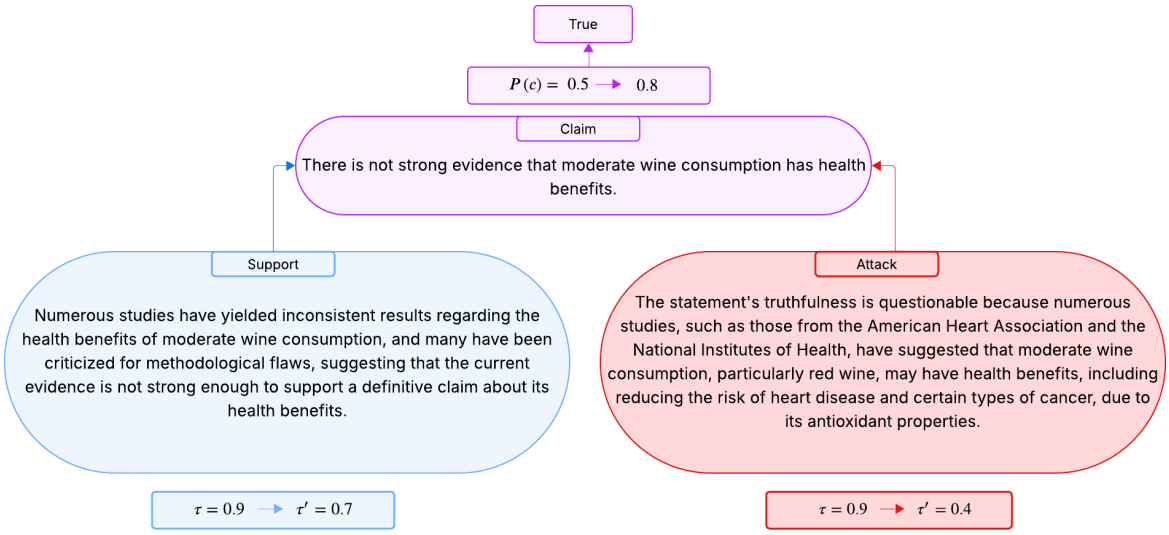

Figure 2: Example from *MedQA* with *Llama 3.3 70B Instruct*. Pairwise tournaments calibrate argument strengths by blending intrinsic weights τ with BT scores θ to yield τ'. Here, support and attack start with equal intrinsic weights but calibrate to 0.7 and 0.4, shifting the claim's probability from 0.5 (False) to 0.8 (True). Without the pairwise and BT calibration steps, the claim would be misclassified as False.

Baselines. We evaluate our approach against three baselines. **Direct Prompting**: the LLM is directly prompted to verify a claim with a *Yes/No* answer. **Chain-of-Thought (CoT)** (Wei et al., 2022): the LLM is additionally asked to generate reasoning alongside its Yes/No decision. **ArgLLM** (Freedman et al., 2025): this method constructs a tree-like structure but without pairwise comparisons or calibrations. The differences between ART and ArgLLM are discussed in Section 2.4 in more detail. All the prompts used for Direct Prompting, CoT, arguments generation, intrinsic strength calculation, pairwise comparisons, are shown in Appendix F.

5 Results

We benchmark ART against Direct prompting, CoT prompting, and ArgLLM (Freedman et al., 2025) in Table 1, using tree depth and breadth fixed at 1 across datasets and LLMs. In this ART configuration, each root claim is evaluated through a single supporting argument and a single attacking argument competing to determine the claim's validity.

Evaluator Settings: We evaluate two evaluator settings. In the *Single-LLM* setup (*Eval=Self*), the same LLM performs argument generation, intrinsic strength computation, pairwise comparisons, and final judgment. This setup directly tests the model's ability to identify relatively stronger arguments, but it is also prone to self-affirmation bias since generation and adjudication rely on the same LLM. In contrast, the *Multi-LLM* setup (*Eval=70B*) separates these roles: one model generates arguments, while a stronger judge LLM performs pairwise comparisons and intrinsic strength calculation for ART nodes, thereby reducing the bias present in the *Single-LLM* configuration. We use *Llama 3.3 70B-Instruct* model as the judge model due to its strong performance in complex problems. The *Multi-LLM* configuration delivers substantially better results than *Single-LLM* in our studies.

Bradley–Terry Calibration with Intrinsic Strengths: We employ the Bradley–Terry (BT) model to derive argument strengths from pairwise tournaments. In the *w/o intr.* setting, λ is set to 1 and only the BT-derived score θ is used (Eq. 3) to achieve the final score s (Eq. 6), while in the *intr.* setting, λ is set to 0.5 so that the BT score is further blended with the intrinsic strength τ, yielding the adjusted value τ' to calculate s. We believe intrinsic strength helps regularize the BT estimates by encoding prior plausibility/quality, reducing variance when pairwise evidence is sparse or noisy, and preventing brittle swings from a few contested comparisons. On average, we achieve a 3% relative accuracy improvement in the *intr.* setting over the *w/o intr.* setting having aggregated over both the evaluator settings. **Note.** If not explicitly specified, ART should be considered with the *intr.* and *Multi-LLM* (*Eval=70B*) configuration.

Dataset	Model	Direct	CoT	ArgLLM (Eval=70B)	ART (w/o intr.) (Eval=70B)	ART (intr.) (Eval=70B)	ArgLLM (Eval=Self)	ART (w/o intr.) (Eval=Self)	ART (intr.) (Eval=Self)
MedClaim	Llama 3.1 3B Instruct	51.4	57.9	58.7	63.1	**65.0**	52.2	**54.3**	53.7
	Llama 3.1 8B Instruct	54.5	64.6	62.9	**64.6**	63.6	54.9	53.7	**56.0**
	Mistral-7B Instruct-v0.2	51.4	50.4	54.3	54.8	**57.7**	47.1	50.1	**52.8**
	Mixtral-8x7B Instruct-v0.1	58.7	53.5	62.3	**65.3**	61.7	**61.5**	56.2	57.7
	Llama 3.3 70B Instruct	76.0	74.3	67.7	68.8	**69.0**	–	–	–
StrategyQA	Llama 3.1 3B Instruct	58.6	67.0	61.4	62.4	**62.6**	55.6	**58.6**	57.6
	Llama 3.1 8B Instruct	58.2	73.6	64.4	69.8	**71.4**	58.8	57	**59.2**
	Mistral-7B Instruct-v0.2	59.4	55.4	65.6	64.4	**70.0**	55.2	**58.6**	58.4
	Mixtral-8x7B Instruct-v0.1	55.6	68.4	64.4	**70.2**	68.8	62	61.2	**63.6**
	Llama 3.3 70B Instruct	81.0	82.4	68.4	74.2	**75.8**	–	–	–
TruthfulQA	Llama 3.1 3B Instruct	62.2	68	65.2	66.9	**67.0**	60.6	**65.0**	63.8
	Llama 3.1 8B Instruct	63.0	70.4	67.8	69.4	**75.6**	59.8	52.8	**68.2**
	Mistral-7B Instruct-v0.2	70.6	57.2	68.2	65.6	**69.2**	57.2	54.4	**59.8**
	Mixtral-8x7B Instruct-v0.1	66.6	62.8	71.4	76.4	**78.0**	**62.8**	60.4	61.6
	Llama 3.3 70B Instruct	81.6	83.6	68.4	**80.6**	79.4	–	–	–

Table 1: Evaluation results on MedClaim, StrategyQA and TruthfulQA comparing Direct prompting, CoT, ArgLLM, and ART with *depth*=1 and *breadth*=1. Results are shown for two evaluator settings: (i) *Eval = 70B*, where judgments are made by *Llama 3.3 70B Instruct*, and (ii) *Eval = Self*, where the evaluator matches the main model. The *w/o intr.* setting uses only the BT score θ by setting $\lambda = 1$, while *intr.* combines θ with intrinsic strength τ with $\lambda = 0.5$ (Eq. 3). See Section 3.3 for details. Bold indicates the best *Accuracy* score, while underlining marks the second best within the two evaluator settings. Blue color text indicates cases where CoT or Direct achieves the best performance.

5.1 Comparison with Direct and CoT

We observe that ART outperforms the Direct Prompting methodology overall, which is expected since direct prompting requires the LLM to evaluate the claim in a single step without any structured decomposition of arguments. In contrast, ART explicitly organizes the reasoning process into supporting and attacking arguments, enabling more reliable adjudication of the root claim. The main gap appears on larger models: CoT often performs better because they exploit rich step-by-step reasoning, while ART's multiple argument branches can add noise and slightly reduce accuracy. Overall, while ART not only achieves at par performance over CoT, its key advantages lie beyond accuracy. ART produces faithful, contestable explanations structured as arguments, from which final decisions are deterministically derived. By contrast, CoT traces offer post-hoc rationalizations that may not reflect the model's actual stochastic reasoning process (Turpin et al., 2023). ART therefore provides greater transparency and verifiability, as each decision can be directly traced to its underlying arguments and semantics.

5.2 ART vs ArgLLM

When $\lambda = 0$, the update reduces to the ArgLLM baseline: no BT contribution is applied and each weight stays at its pre-BT value (i.e., $\tau'_i = \tau_i$). This recovers the original ArgLLM setting and serves as a clean ablation point for comparisons. We observe that in most cases (see Table 1 and 2), ART outperforms ArgLLM. This can be attributed to how strength is computed for each node: ArgLLM estimates strengths in isolation, without considering sibling arguments, which limits the robustness of its scoring. In contrast, ART employs a pairwise tournament in which arguments of opposite polarity directly compete, with strengths estimated through the Bradley–Terry (BT) model. This ensures that scores are both relative and comparable within the same latent space, producing more reliable outcomes. Furthermore, ART's calibration step grounds final decisions in argumentation semantics, enhancing consistency and interpretability compared to ArgLLM's uncalibrated node-level scores. ArgLLM is a special case of ART obtained by setting $\lambda = 0$: the update ignores pairwise outcomes and applies no score calibration, relying solely on the intrinsic strengths of the arguments.

Figure 2 shows a qualitative example where two arguments that begin with equal intrinsic weights can calibrate to 0.7 (support) versus 0.4 (attack), shifting the root claim's probability from 0.5 to 0.8 and flipping a mistaken rejection into a correct acceptance. In practice, this pairwise-and-calibrate step addresses ArgLLM's main failure mode—uncalibrated, non-comparable scores—while preserving transparent, contestable reasoning traces. ART moves all arguments into a shared latent scale and corrects over/under-weighting.

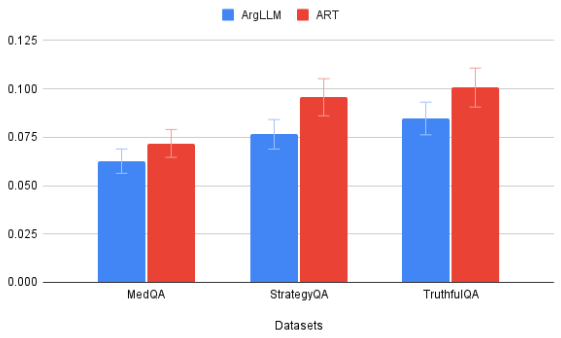

Figure 3: Variance of strengths distribution under ArgLLM ($\lambda = 0$) vs. ART ($\lambda = 0.5$). of the root claim nodes. ART yields higher variance—scores pushed away from 0.5—indicating more decisive, calibrated strengths.

Dataset	Model	ArgLLM (D=1, B=2)	ArgLLM (D=2, B=1)	ART (D=1, B=2)	ART (D=2, B=1)
MedClaim	Llama 3.1 3B Instruct	57.9	56.8	**60.8**	53.3
	Llama 3.1 8B Instruct	62.5	61.9	**62.9**	53.7
	Mistral-7B Instruct-v0.2	55.6	52.8	**58.3**	50.1
	Mixtral-8x7B Instruct-v0.1	**66.1**	62.7	65.7	55.8
	Llama 3.3 70B Instruct	67.1	67.1	**67.4**	56.0
StrategyQA	Llama 3.1 3B Instruct	63.2	61.8	**64.4**	57.6
	Llama 3.1 8B Instruct	64.8	66.2	**72.2**	53.4
	Mistral-7B Instruct-v0.2	66.8	67.0	**70.2**	59.2
	Mixtral-8x7B Instruct-v0.1	62.6	**67.0**	65.2	54.4
	Llama 3.3 70B Instruct	69.4	68.2	**75.4**	58.2
TruthfulQA	Llama 3.1 3B Instruct	68.0	67.4	**69.0**	60.8
	Llama 3.1 8B Instruct	69.6	67.0	**73.0**	55.0
	Mistral-7B Instruct-v0.2	69.6	68.6	**72.2**	51.4
	Mixtral-8x7B Instruct-v0.1	72.4	76.0	**79.4**	62.2
	Llama 3.3 70B Instruct	71.0	74.8	**80.2**	63.0

Table 2: Accuracy of ArgLLM and ART on datasets with different tree settings (*Eval=70B*, *intr.*). Best scores are in bold and second best are underlined. **D** = depth, **B** = breadth.

Bias Variance Tradeoff: We observe that BT–calibrated strengths lead to higher variance in the final claim prediction probabilities as compared to ArgLLM strengths (see Figure 3). ART increases variance by decompressing intrinsic scores and incorporating BT comparison evidence, but improves ranking calibration. ART helps with bias reduction via pairwise comparisons as opposed to isolated strength calculations.

5.3 Tree Variations

We analyze performance across different tree sizes in terms of depth and breadth in Table 2. To balance accuracy with latency, we restrict our exploration to a maximum depth and breadth of two. Our results show that increasing the breadth of the tree improves the performance of models with different size across benchmarks. Overall, breadth makes more difference than the height as each argument in the tree is now validated with more support and attack child nodes giving more context to the evaluator model J. ART outperforms ArgLLM in this setup as well.

Having shown that widening a single tree is effective, we next examined whether an ensemble of separate, simpler trees could be a competitive alternative. We evaluate this multi-tree ensemble—along with its computational tradeoffs—in Appendix D. Empirically, while greater breadth for pairwise comparisons remains a strong general-purpose strategy, a minimalist configuration (D=1, B=1) coupled with a strong evaluator (e.g., Llama 3.3 70B) can be unexpectedly potent, particularly under compute constraints.

6 Conclusion and Future Work

We introduce Adaptive Reasoning Trees (ART) as a principled framework for transparent and contestable claim verification. ART systematically decomposes a claim into a hierarchy of supporting and attacking arguments, forming a tree structure where each argument may itself branch into further pro and con arguments. These arguments engage in pairwise tournaments, with their relative strengths estimated using the Bradley–Terry model, ensuring that scores are calibrated within a shared latent space. Through a bottom-up aggregation process, ART computes the overall strength of the root claim to decide whether it should be accepted or rejected. ART is robust across benchmarks and configurations (varying tree breadth and height), producing faithful, interpretable traces and consistently outperforming Direct Prompting, Chain-of-Thought, and ArgLLM in accuracy and robustness. It delivers a transparent, contestable verification framework in which final decisions arise from the balanced aggregation of all arguments.

Future work can explore grounding-based argument generation using knowledge bases like *Wikipedia* for evidence-backed support and attack generation. Additionally, tournaments could weigh comparisons by evidence quality, recency, and source reliability for provenance-aware, auditable decisions. Moreover, while Bradley–Terry offers a solid calibration method, sophisticated methods leveraging neural networks that can learn the non-linear cues and interactions could be a potential avenue. Lastly, ART could be augmented with a reinforcement learning objective that rewards each component for the correct collective decision.

7 Limitations

While ART offers a transparent, contestable and traceable framework for claim verification, it has some limitations. As compared to Direct and CoT prompting methods that make a single LLM call, ART makes multiple LLM calls - for support/attack generation, intrinsic scoring, and pairwise comparisons which could lead to higher time complexity. However, with multiple parallel asynchronous calls, this time could be brought down significantly. Additionally, LLM calls could be reduced by optimizing LLMs to generate both support and attacks at the same time. While ART offers good results on the three datasets used in this paper, it may still face challenges with claims that are more complex and uncertain and require extensive world knowledge, and domain expertise. Such cases can amplify error propagation across the tree and stress the judge's calibration. LLM judges can be prompt sensitive or biased towards a certain set of arguments, which could reduce the reliability on fair judgment and make the overall process less trustworthy. Despite these caveats, ART lays the groundwork for claim verification—fostering explainable, auditable, and contestable reasoning.

References

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. 2024a. Evaluating correctness and faithfulness of instruction-following models for question answering. *Preprint*, arXiv:2307.16877.

Vaibhav Adlakha and 1 others. 2024b. Evaluating correctness and faithfulness of llm explanations. *arXiv preprint arXiv:2401.04895*.

Ayush Agrawal, Mirac Suzgun, Lester Mackey, and Adam Kalai. 2024. Do language models know when they're hallucinating references? In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 912–928, St. Julian's, Malta. Association for Computational Linguistics.

Pepa Atanasova and 1 others. 2020. Generating fact-checking explanations. In *ACL*.

Isabelle Augenstein, Timothy Baldwin, Meeyoung Cha, Tanmoy Chakraborty, Giovanni Luca Ciampaglia, David Corney, Renee DiResta, Emilio Ferrara, Scott Hale, Alon Halevy, Eduard Hovy, Heng Ji, Filippo Menczer, Ruben Miguez, Preslav Nakov, Dietram Scheufele, Shivam Sharma, and Giovanni Zagni. 2023a. Factuality challenges in the era of large language models. *Preprint*, arXiv:2310.05189.

Isabelle Augenstein and 1 others. 2023b. Challenges in factuality for llms. In *EMNLP*.

Ben Carterette, Paul Bennett, Max Chickering, and Susan Dumais. 2008. Here or there: Preference judgments for relevance. In *Proceedings of ECIR 2008*. Springer. ECIR 2018 Test of Time Honorable Mention.

Alvin Chan and 1 others. 2023. Chateval: Towards better llm-based evaluators through multi-agent debate. In *ACL*.

Xinya Chen and 1 others. 2023. Benchmarking large language models in rag settings. *arXiv preprint arXiv:2311.09383*.

Nan Du and 1 others. 2023. Improving factual accuracy via multi-agent debate. In *NeurIPS*.

Gabriel Freedman, Adam Dejl, Deniz Gorur, Xiang Yin, Antonio Rago, and Francesca Toni. 2025. Argumentative large language models for explainable and contestable claim verification. *Preprint*, arXiv:2405.02079.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Preprint*, arXiv:2101.02235.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Nuno M. Guerreiro, Duarte Alves, Jonas Waldendorf, Barry Haddow, Alexandra Birch, Pierre Colombo, and André F. T. Martins. 2023. Hallucinations in large multilingual translation models. *Preprint*, arXiv:2303.16104.

Zhijiang Guo, M. Schlichtkrull, and Andreas Vlachos. 2021. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Clément Henin and Daniel Le Métayer. 2022. Beyond explainability: justifiability and contestability of algorithmic decision systems. *AI Soc.*, 37(4):1397–1410.

Christopher Hidey and 1 others. 2018. Unsupervised argument mining from online discussions. In *ACL*.

Jie Huang and 1 others. 2025a. Hallucinations in large language models: Taxonomy, evaluation, and mitigation. *arXiv preprint arXiv:2501.00001*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting

Liu. 2025b. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Geoffrey Irving and 1 others. 2018. Ai safety via debate. *arXiv preprint arXiv:1805.00899*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Preprint*, arXiv:2009.13081.

Youngsoo Jung and 1 others. 2022. Critic llms: Self-improvement through critical feedback. *arXiv preprint arXiv:2212.12345*.

Manuj Kant, Sareh Nabi, Manav Kant, Roland Scharrer, Megan Ma, and Marzieh Nabi. 2025. Towards robust legal reasoning: Harnessing logical llms in law. *Preprint*, arXiv:2502.17638.

Ahmad Khan and 1 others. 2024. Debate as persuasion: Improving llm truthfulness. *arXiv preprint arXiv:2402.10011*.

Takeshi Kojima, Shixiang Gu, Alistair Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Jooyoung Lee and 1 others. 2023. Prompt-and-rank for fact verification with llms. *arXiv preprint arXiv:2305.12345*.

Francesco Leofante, Hamed Ayoobi, Adam Dejl, Gabriel Freedman, Deniz Gorur, Junqi Jiang, Guilherme Paulino-Passos, Antonio Rago, Anna Rapberger, Fabrizio Russo, Xiang Yin, Dekai Zhang, and Francesca Toni. 2024. Contestable ai needs computational argumentation. *Preprint*, arXiv:2405.10729.

Patrick Lewis and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. *Preprint*, arXiv:2109.07958.

S. Maity and M. J. Saikia. 2025. Large Language Models in Healthcare and Medical Applications: A Review. *Bioengineering (Basel)*, 12(6):631.

Julian Michael and 1 others. 2023. Debate helps: Multi-llm collaboration for evaluation. In *ICLR*.

Preslav Nakov and 1 others. 2021. Automated fact-checking for assisting human fact-checkers. *Communications of the ACM*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Melissa Kazemi Rad, Huy Nghiem, Andy Luo, Sahil Wadhwa, Mohammad Sorower, and Stephen Rawls. 2025. Refining input guardrails: Enhancing llm-as-a-judge efficiency through chain-of-thought fine-tuning and alignment. *Preprint*, arXiv:2501.13080.

A. Rago, F. Toni, M. Aurisicchio, and P. Baroni. 2016. Discontinuity-free decision support with quantitative argumentation debates. In *Proceedings of the Fifteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'16, page 63–72. AAAI Press.

Bichen Shi and 1 others. 2016. Fact checking in community forums. In *WWW*.

Abhinav Singh and 1 others. 2024. Rethinking interpretability in the era of llms. *arXiv preprint arXiv:2404.09999*.

James Thorne and 1 others. 2018. Fever: a large-scale dataset for fact extraction and verification. In *NAACL*.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Preprint*, arXiv:2305.04388.

Juraj Vladika, Ivana Hacajov'a, and Florian Matthes. 2025. Step-by-step fact verification system for medical claims with explainable reasoning. *ArXiv*, abs/2502.14765.

Sahil Wadhwa, Chengtian Xu, Haoming Chen, Aakash Mahalingam, Akankshya Kar, and Divya Chaudhary. 2024. Northeastern uni at multilingual counterspeech generation: Enhancing counter speech generation with llm alignment through direct preference optimization. *Preprint*, arXiv:2412.15453.

Xuezhi Wang and 1 others. 2022. Self-consistency improves chain-of-thought reasoning in llms. *arXiv preprint arXiv:2203.11171*.

Yuntao Wang and 1 others. 2023. Synergy of multiple personas in llm collaboration. *arXiv preprint arXiv:2311.06510*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Haodi Zhang, Jiahong Li, Yichi Wang, and Yuanfeng Song. 2023. Integrating automated knowledge extraction with large language models for explainable medical decision-making. In *2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1710–1717.

Wayne Xin Zhao and 1 others. 2024. Explainability in large language models: A survey. *arXiv preprint arXiv:2402.04644*.

Appendix

A ART Algorithm

ART is leveraged for objectively calibrating the persuasive weights of arguments within a hierarchical tree. ART identifies points of contention where a claim is defended by "supporters" and challenged by "attackers." It then leverages a large language model (LLM) as an impartial judge to conduct pairwise comparisons between every competing supporter and attacker. By aggregating the results of these head-to-head contests, the algorithm uses the Bradley-Terry statistical model to compute a "strength score" for each argument based on its performance. This score is then used to intelligently update the argument's initial weight, ensuring the final structure reflects the relative quality of its components as determined by the LLM judge.

Algorithm 1 Calibration via LLM Pairwise Judgments + Bradley–Terry (bipartite)

Input: Argument tree T (root id `db0`), judge LLM $\mathcal{J}$, pairwise prompt Π, stabilizer $\varepsilon > 0$, $\lambda \in [0, 1]$
Output: Calibrated tree T
I. Identify calibration parents.
$root \leftarrow T[\texttt{db0}]$ **if** $root = \varnothing$ **then**
  return T
$C \leftarrow \{p \in T \mid p.\texttt{supporters} \neq \varnothing \wedge p.\texttt{attackers} \neq \varnothing\}$ **if** $C = \varnothing$ **then**
  return T
II. Per-parent setup and pairwise counts.
foreach *parent* $p \in C$ **do**
  $S \leftarrow p.\texttt{supporters}$; $A \leftarrow p.\texttt{attackers}$;
    $U_p \leftarrow S \cup A$;
    $E_p \leftarrow S \times A$ `// Cross-pairs only`
  Initialize sparse $W_p[x, y] \leftarrow 0$ for all $(x, y) \in E_p \cup \{(y, x) : (x, y) \in E_p\}$
  foreach $s \in S$ **do**
    foreach $a \in A$ **do**
      $prompt \leftarrow \Pi(p,\ s,\ a)$
      $winner \leftarrow \mathcal{J}(prompt)$
      if $winner =$ `SUPPORT` **then**
        $W_p[s, a] \leftarrow W_p[s, a] + 1$
      else if $winner =$ `ATTACK` **then**
        $W_p[a, s] \leftarrow W_p[a, s] + 1$
III. BT fitting and rescaling (per parent).
foreach *parent* $p \in C$ **do**
  if $E_p = \varnothing$ **then**
    continue
  Initialize $\theta_u \leftarrow 1$ for each $u \in U_p$
  repeat
    foreach $u \in U_p$ **do**
      if $u \in S$ **then**
        $wins \leftarrow \sum_{a \in A} W_p[u, a]$
        $den \leftarrow \sum_{a \in A} \dfrac{W_p[u, a] + W_p[a, u]}{\theta_u + \theta_a}$
      else `//` $u \in A$
        $wins \leftarrow \sum_{s \in S} W_p[u, s]$
        $den \leftarrow \sum_{s \in S} \dfrac{W_p[u, s] + W_p[s, u]}{\theta_u + \theta_s}$
      $\theta_u \leftarrow \dfrac{wins}{den + \varepsilon}$
    Normalize $\sum_{u \in U_p} \theta_u = 1$
  until *convergence*;
  foreach $u \in U_p$ **do**
    $u.\texttt{initial_weight} \leftarrow \text{clip}\big((u.\texttt{initial_weight} \cdot (1 - \lambda) + \lambda \cdot \theta_u),\ 0,\ 1\big)$
return T

B Bradley–Terry

MM view and notation (bipartite A–B). Let A be supporters, B attackers, and $E := A \times B$ the set of cross-pairs. For $(a, b) \in E$, let τ_{ab} be wins of a over b, $n_{ab} = \tau_{ab} + \tau_{ba}$, and $S_{ab}^{(t)} = \theta_a^{(t)} + \theta_b^{(t)}$ (pair-sum at iterate t). Define

$$\begin{aligned} \mathrm{wins}_a &= \sum_{b \in B} \tau_{ab}, & \mathrm{wins}_b &= \sum_{a \in A} \tau_{ba}, \\ c_a^{(t)} &= \sum_{b \in B} \frac{n_{ab}}{S_{ab}^{(t)}}, & c_b^{(t)} &= \sum_{a \in A} \frac{n_{ab}}{S_{ab}^{(t)}}. \end{aligned}$$

Linearizing $-\log(\theta_a + \theta_b)$ at $S_{ab}^{(t)}$ yields a separable lower bound, maximized at $\tilde{\theta}_u = \mathrm{wins}_u / c_u^{(t)}$ for $u \in A \cup B$; normalizing to $\sum_u \theta_u^{(t+1)} = 1$ recovers Eq. (2) with $\varepsilon = 0$, restricted to cross-pairs.

Proposition 1 (Scale invariance under bipartite comparisons)**.** *For any $c > 0$, replacing all cross-pair counts by $c\,\tau_{ab}$ (hence $n_{ab} \mapsto c\,n_{ab}$) leaves the mapping $\theta^{(t)} \mapsto \theta^{(t+1)}$ in Eq. (2) unchanged after normalization; thus the fixed point $\hat{\theta}$ (up to scale) is unchanged.*

Proof. Each wins_u and each term of $c_u^{(t)}$ scales by c, so $\mathrm{wins}_u / c_u^{(t)}$ is invariant; normalization then gives the same $\theta^{(t+1)}$. Equivalently, the bipartite BT log-likelihood

$$\begin{aligned} \mathcal{L}(\theta) = &\sum_{(a,b) \in E} \Big[\tau_{ab} \log \theta_a + \tau_{ba} \log \theta_b\Big] \\ &- \sum_{(a,b) \in E} n_{ab} \log\big(\theta_a + \theta_b\big). \end{aligned} \tag{7}$$

is multiplied by c and has the same maximizer up to normalization. □

Proposition 2 (MM minorization and monotone ascent for $\varepsilon = 0$ (bipartite))**.** *With the notation above, the update $\tilde{\theta}_u = wins_u / c_u^{(t)}$ for $u \in A \cup B$, followed by normalization, is a Majorize–Minimize step for $\mathcal{L}$ and satisfies $\mathcal{L}(\theta^{(t+1)}) \geq \mathcal{L}(\theta^{(t)})$, with equality iff $\theta^{(t+1)} = \theta^{(t)}$.*

Proof. By convexity of $-\log x$, with $x = \theta_a + \theta_b$ and $x_0 = S_{ab}^{(t)}$, $-\log(\theta_a + \theta_b) \geq -\log S_{ab}^{(t)} - \frac{\theta_a + \theta_b - S_{ab}^{(t)}}{S_{ab}^{(t)}}$. Multiplying by n_{ab} and summing over E yields

$$\begin{aligned} \mathcal{L}(\theta) \geq &\sum_{(a,b) \in E} \big[\tau_{ab} \log \theta_a + \tau_{ba} \log \theta_b\big] \\ &- \sum_{(a,b) \in E} n_{ab} \log S_{ab}^{(t)} - \sum_{(a,b) \in E} \tfrac{n_{ab}}{S_{ab}^{(t)}} \big(\theta_a + \theta_b - S_{ab}^{(t)}\big) \\ = &\sum_{u \in A \cup B} \mathrm{wins}_u \log \theta_u - \sum_{u \in A \cup B} \theta_u\, c_u^{(t)} + \mathrm{const}\big(\theta^{(t)}\big) \\ =: &\; Q(\theta \,|\, \theta^{(t)}) + \mathrm{const}. \end{aligned} \tag{8}$$

Q is tight at $\theta^{(t)}$ and strictly concave on $\{\theta > 0\}$; the first-order conditions give $\mathrm{wins}_u / \theta_u - c_u^{(t)} = 0$, i.e., $\tilde{\theta}_u = \mathrm{wins}_u / c_u^{(t)}$. Since $\mathcal{L}(c\theta) = \mathcal{L}(\theta)$ for any $c > 0$, normalizing $\tilde{\theta}$ to $\sum_u \theta_u^{(t+1)} = 1$ does not change $\mathcal{L}$; hence

$$\begin{aligned} \mathcal{L}(\theta^{(t+1)}) &\geq Q(\theta^{(t+1)} \,|\, \theta^{(t)}) + \mathrm{const} \\ &\geq Q(\theta^{(t)} \,|\, \theta^{(t)}) + \mathrm{const} = \mathcal{L}(\theta^{(t)}), \end{aligned} \tag{9}$$

with equality iff $\theta^{(t+1)} = \theta^{(t)}$. □

Proposition 3 (DF-QuAD combination via signed product gap)**.** *Let p be a node with supporters $\mathrm{S}(p)$ and attackers $\mathrm{A}(p)$. Define*

$$\alpha_p \;=\; \prod_{a \in \mathrm{S}(p)} (1 - s_a) \;-\; \prod_{b \in \mathrm{A}(p)} (1 - s_b), \tag{10}$$

where $s_a, s_b \in [0, 1]$ are propagated strengths. Let $\tau'_p \in [0, 1]$ be the BT-calibrated intrinsic weight. Then the parent score

$$s_p \;=\; \begin{cases} \tau'_p \;-\; \alpha_p\, \tau'_p, & \alpha_p > 0, \\ \tau'_p \;-\; \alpha_p\, (1 - \tau'_p), & \alpha_p \leq 0, \end{cases} \tag{11}$$

is exactly the DF-QuAD update for p and satisfies $s_p \in [0, 1]$, is continuous in α_p, and is nonincreasing in α_p.

Proof. Define the DF-QuAD noisy-OR aggregates for support and attack:

$$v_s \;=\; 1 - \prod_{a \in \mathrm{S}(p)} (1 - s_a), \qquad v_a \;=\; 1 - \prod_{b \in \mathrm{A}(p)} (1 - s_b)$$

Let the signed net-support gap be $\delta := v_s - v_a \in [-1, 1]$. By (10),

$$\begin{aligned} \delta &:= v_s - v_a \\ &= \left(1 - \prod_{a \in \mathrm{S}(p)} (1 - s_a)\right) - \left(1 - \prod_{b \in \mathrm{A}(p)} (1 - s_b)\right) \\ &= \prod_{b \in \mathrm{A}(p)} (1 - s_b) - \prod_{a \in \mathrm{S}(p)} (1 - s_a) \;=\; -\alpha_p. \end{aligned}$$

DF-QuAD's combination function C maps (τ'_p, v_a, v_s) to

$$C(\tau'_p, v_a, v_s) = \begin{cases} \tau'_p + (1-\tau'_p)\,\delta, & \delta \geq 0, \\ \tau'_p + \tau'_p\,\delta, & \delta < 0 \end{cases}$$

Substituting $\delta = -\alpha_p$ yields exactly (11): if $\alpha_p \leq 0$ (i.e., $\delta \geq 0$), then $s_p = \tau'_p - (1-\tau'_p)\alpha_p$; if $\alpha_p > 0$ (i.e., $\delta < 0$), then $s_p = \tau'_p - \tau'_p\alpha_p$.

Range. Since both products in (10) lie in $[0,1]$, we have $\alpha_p \in [-1,1]$. For $\alpha_p > 0$, $s_p = \tau'_p(1-\alpha_p) \in [0, \tau'_p] \subseteq [0,1]$. For $\alpha_p \leq 0$, $s_p = \tau'_p + (1-\tau'_p)(-\alpha_p) \in [\tau'_p, 1] \subseteq [0,1]$.

Continuity. At $\alpha_p = 0$ both branches give $s_p = \tau'_p$, hence s_p is continuous in α_p.

Monotonicity. For $\alpha_p > 0$, $\frac{\partial s_p}{\partial \alpha_p} = -\tau'_p \leq 0$; for $\alpha_p \leq 0$, $\frac{\partial s_p}{\partial \alpha_p} = -(1-\tau'_p) \leq 0$. Thus s_p is nonincreasing in α_p, so larger attack dominance (larger α_p) never increases s_p. □

C Examples

In this section, we provide some examples where pairwise tournaments and calibrations helped with correct predictions. Here, ART leverages *Llama 3.3 70B Instruct* model is used with the *intr.* setting. Increasing number of attack and support arguments in the tournament leads to more stable results with lower variance at the cost of higher latency. In Figure 4 we provide more examples where ART guides the final decision in the right direction using two support and attack arguments.

D Multi-Tree Ensemble

In our exploration of tree-based reasoning structures, we hypothesized that an ensemble method might yield more robust and accurate results. To this end, we designed an experiment comparing three distinct configurations: a simple single-tree $(D = 1, B = 1)$, a single-tree with a pairwise tournament $(D = 1, B = 2)$, and a multi-tree ensemble. The ensemble method involved generating two independent $(D = 1, B = 1)$ Argumentative Reasoning Trees (ART) and averaging their final scores. This allowed us to test whether a direct, internal comparison of arguments is more effective than aggregating scores from isolated reasoning paths.

The results of this experiment, presented in Table 3, reveal a surprisingly nuanced outcome where no single configuration is universally superior. The single-tree $(D = 1, B = 2)$ approach, which facilitates a direct pairwise tournament, proves to be the most robust baseline, achieving the highest accuracy in the majority of test cases across all datasets. This confirms that forcing a direct comparison between arguments is a powerful and generally effective strategy.

However, a deeper analysis shows that this advantage is not absolute and is **highly dependent on the capability of the evaluator model**. This dependency creates clear and interesting patterns in the results. For instance, when a state-of-the-art model like **Llama 3.3 70B** acts as the evaluator, simpler structures often become the top performers. We see the multi-tree ensemble winning with the Llama 3.3 70B generator on MedClaim, and the basic $(D = 1, B = 1)$ tree winning with the same generator on StrategyQA. This suggests that a powerful evaluator may not need a complex reasoning structure to accurately assess a claim's validity.

This leads to our central insight: **the optimal reasoning architecture is not independent but is co-dependent on the components of the evaluation pipeline**. While increasing tree breadth for direct argument comparison remains a strong general-purpose strategy, our findings demonstrate that simpler architectures and ensemble methods can be unexpectedly potent. Their effectiveness, particularly when paired with highly capable evaluators, confirms that the interplay between the generator, the evaluator, and the reasoning structure is a critical factor in achieving maximum performance.

Crucially, a practical consideration is the trade-off between accuracy and computational cost. We observe that in the specific scenarios where the Multi-Tree Ensemble outperforms other methods, the simpler Single-Tree $(D = 1, B = 1)$ configuration is often a close second-place contender. Given that the ensemble method effectively doubles the number of LLM calls and the associated overhead, the marginal performance gain frequently fails to justify the significant increase in cost and latency. This efficiency analysis provides a compelling final argument, reinforcing our conclusion that pursuing broader single-tree structures is a more promising and practical direction than adopting multi-tree ensembles.

E Time Complexity of ART

Proposition 4 (Counts on a full $2b$-ary tree of depth d)**.** *Let each parent have b supporters and b at-*

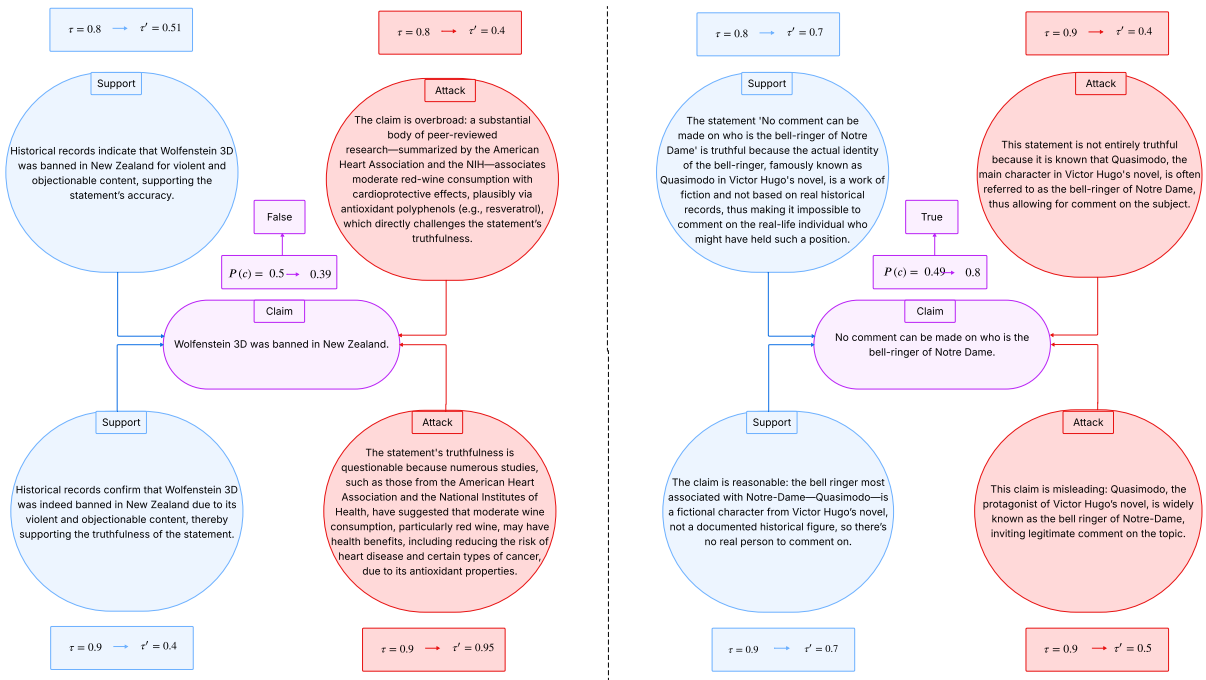

Figure 4: Examples where ART improves the prediction results due to calibration that happens as a result of pairwise tournaments. Increasing the number of support and attack arguments in the tournament provides greater stability to the estimated strengths.

Dataset	Model	Evaluator	ART Single-Tree (D=1, B=1)	ART Single-Tree (D=1, B=2)	ART Multi-Tree Ensemble(D=1, B=1)
MedClaim	Llama 3.1 3B Instruct	Llama 3.1 3B Instruct	53.68	**56.42**	53.68
	Llama 3.1 3B Instruct	Llama 3.3 70B Instruct	**65.05**	60.84	61.2
	Llama 3.1 8B Instruct	Llama 3.1 8B Instruct	56	**56.21**	55.37
	Llama 3.1 8B Instruct	Llama 3.3 70B Instruct	63.57	62.97	**63.67**
	Mistral-7B Instruct-v0.2	Mistral-7B Instruct-v0.2	52.84	**54.31**	52.58
	Mistral-7B Instruct-v0.2	Llama 3.3 70B Instruct	57.68	**58.31**	57.89
	Mixtral-8x7B Instruct-v0.1	Mixtral-8x7B Instruct-v0.1	56.21	**61.05**	60.63
	Mixtral-8x7B Instruct-v0.1	Llama 3.3 70B Instruct	65.26	**65.68**	64.5
	Llama 3.3 70B Instruct	Llama 3.3 70B Instruct	69.05	67.36	**69.26**
StrategyQA	Llama 3.1 3B Instruct	Llama 3.1 3B Instruct	57.6	**59.4**	59
	Llama 3.1 3B Instruct	Llama 3.3 70B Instruct	62.4	**64.4**	61.8
	Llama 3.1 8B Instruct	Llama 3.1 8B Instruct	59.2	**64.6**	60.2
	Llama 3.1 8B Instruct	Llama 3.3 70B Instruct	71.4	**72.2**	70.8
	Mistral-7B Instruct-v0.2	Mistral-7B Instruct-v0.2	58.4	61.2	**61.6**
	Mistral-7B Instruct-v0.2	Llama 3.3 70B Instruct	70	**70.2**	68.6
	Mixtral-8x7B Instruct-v0.1	Mixtral-8x7B Instruct-v0.1	**63.6**	61	61.2
	Mixtral-8x7B Instruct-v0.1	Llama 3.3 70B Instruct	68.8	65.2	**69.2**
	Llama 3.3 70B Instruct	Llama 3.3 70B Instruct	**75.8**	75.4	74.6
TruthfulQA	Llama 3.1 3B Instruct	Llama 3.1 3B Instruct	63.8	**64.8**	64.6
	Llama 3.1 3B Instruct	Llama 3.3 70B Instruct	67	**69**	68.2
	Llama 3.1 8B Instruct	Llama 3.1 8B Instruct	**68.2**	62	60
	Llama 3.1 8B Instruct	Llama 3.3 70B Instruct	**75.6**	73	75
	Mistral-7B Instruct-v0.2	Mistral-7B Instruct-v0.2	59.8	60	**60.8**
	Mistral-7B Instruct-v0.2	Llama 3.3 70B Instruct	69.2	**72.2**	70.6
	Mixtral-8x7B Instruct-v0.1	Mixtral-8x7B Instruct-v0.1	61.6	**62**	50.2
	Mixtral-8x7B Instruct-v0.1	Llama 3.3 70B Instruct	78	**79.4**	71.4
	Llama 3.3 70B Instruct	Llama 3.3 70B Instruct	79.4	**80.2**	80

Table 3: Accuracy comparison between single-tree (D=1, B=1), single-tree (D=1, B=2) and multi-tree ensemble ART configurations with intrinsic strength calibration, broken down by generator and evaluator models . The results show that the preferred method is highly dependent on the evaluator's capability. Best scores in each row are in bold.

tackers (branching factor $2b$*), with root at depth* 0*. Then the total nodes* (N) *and cross-group (support×attack) pairs* (M) *are*

$$N = \sum_{j=0}^{d} (2b)^j = \frac{(2b)^{d+1} - 1}{2b - 1},$$

$$M = \sum_{j=0}^{d-1} (2b)^j \, b^2 = b^2 \, \frac{(2b)^d - 1}{2b - 1} = \frac{b}{2} \, (N - 1),$$

ART (pairwise + Bradley–Terry + DF-QuAD). Per parent there are b^2 cross-group pairs; across all internal nodes this is M. With a fixed cap $k{=}100$ BT iterations (constant), one BT solve per parent costs $O(b^2)$ per iteration, so the total BT work is $O(kM) = O(M)$. Pairwise evaluation is $O(M)$ where an LLM judge is used to compare all cross pair arguments, and bottom-up DF-QuAD aggregation is $O(bN)$ ($O(2b)$ per internal node). Since $M = \Theta(bN)$,

$$\begin{aligned} T_{\text{ART}} &= O(M) + O(M) + O(bN) + O(N) \\ &= \Theta(bN) = \Theta\big(b(2b)^d\big). \end{aligned}$$

Space: $O(N{+}M) = \Theta(bN)$ if storing all pairwise tallies τ_{ab}; with streaming BT statistics, peak space can be reduced to $O(N)$.

ArgLLM (no pairwise, no Bradley–Terry). ArgLLM performs tree generation, intrinsic scoring, and DF-QuAD only. Aggregation is $O(bN)$ overall; other steps are $O(N)$:

$$\begin{aligned} T_{\text{ArgLLM}} &= O(bN) + O(N) \\ &= \Theta(bN) \;=\; \Theta\big(b(2b)^d\big). \\ \textit{Space} &= O(N) \end{aligned}$$

Comparison. Asymptotically both are $\Theta(bN)$ (BT iterations capped), but ART carries a larger leading constant due to pairwise competitions and BT calculations and typically higher space. so ART is slower by a constant factor (and uses $\Theta(bN)$ space if pairwise tallies are retained), whereas ArgLLM is lighter in both time constant and space.

ART parallelizes naturally: each supporter–attacker pair can be scored independently, and each parent's Bradley–Terry solve runs independently as well. We process the tree level-by-level (all nodes at the same depth in parallel) and batch pair evaluations on *A100* GPU. With this task- and data-parallel structure, ART achieves runtime comparable with ArgLLM.

F Prompts

This section details the prompts employed in our experiments. We utilized distinct templates for several key tasks: Direct Questioning, Chain-of-Thought, generating supporting and attacking arguments, assigning initial scores via a dedicated scoring prompt, and conducting the pairwise comparisons for calibration.

Direct Questioning Prompt The following prompt was used to generate the direct question for a given claim.

Direct Questioning Prompt

You are an analyst evaluating whether statements are
true or false. For the statement:
Statement: "[Your Claim Here]"
please determine whether the statement is correct, accurate and truthful. Please give the result in the form
'Result: True' if the statement is true and
'Result: False' if the statement is false.
Result:

Chain-of-Thought Prompt The following prompt was used to generate the Chain-of-Thought based reasoning response for a given claim.

Chain-of-Thought Prompt

You are an analyst evaluating the validity of statements. For the statement:
Statement: "[Your Claim Here]"
please determine whether the statement is correct,
accurate and truthful. Please think step by step before
providing the final answer.
Let's think step by step.
{Reasoning/Output from previous step}
Therefore, the final answer (**true or false**) is:

Dynamic Argument Generation Prompt The following prompt is designed to dynamically generate either a supporting or an attacking argument for a given claim.

Support Argument Prompt

Please provide a single short argument supporting the following claim. Construct the argument so it refers to the truthfulness of the claim. Only provide an argument if you think there is a valid and convincing support for this claim (there is a non-zero probability that the claim is true), otherwise return: N/A.
Claim:[Your Claim Here]
Now take a deep breath and respond with a JSON object. The JSON object should have a single key, 'argument'.
Example: {'argument':'your argument'}

The above example is for support argument generation and similarly for the attack argument 'supporting' is changed into 'attacking' and 'support' into 'attack'.

Dynamic Uncertainty Evaluator Prompt. The following prompt is designed to dynamically generate Uncertainty scores using LLM as a judge for a support or attack argument. The prompt demonstrates a support Uncertainty Evaluation Prompt, where the language is framed in terms of an argument being in favour of a claim and how it supports the parent statement. For constructing the corresponding attack version, the wording is systematically adjusted: the phrase "in favour of" is replaced with "against" to reflect opposition, and "supports" is replaced with "refutes" to indicate contradiction. This ensures that the structure of the prompt remains consistent while capturing the intended argumentative stance.

Uncertainty Evaluator Prompt (Support)

You are an analyst evaluating the validity and relevance of arguments. For the argument:
Argument: "[Argument Text]"
Please give your confidence that the argument presents a compelling case in favour of the statement:
Statement: "[Claim Text]"
Your assessment should be based on how well the argument supports the considered statement, as well as the correctness, accuracy, and truthfulness of the given argument.

Provide your confidence as a score between 0.0 to 1.0. Your estimates should be well-calibrated. Respond only with a JSON object conforming to the schema:

```
{
  "title": "ScoreSchema",
  "type": "object",
  "properties": {
  "likelihood": {
  "title": "Likelihood",
  "description": "Intrinsic strength of the argument, a value between 0.0 and 1.0.",
  "type": "number"
  }
},
  "required": [
    "likelihood"
  ]
}
```

Pairwise Argument Comparison Prompt The following prompt is designed to compare supporting and attacking arguments for a child node in ART. The judge LLM J receives the original statement (the parent node) along with its two child nodes—one supporting and one attacking—and determines which argument is stronger. If both arguments are deemed equally strong, the model should return TIE.

Pairwise Argument Comparison

You are an analyst comparing two arguments with respect to a statement. Decide which argument is stronger specifically with regard to the statement's truth. Your response should be chosen out of the options: "ATTACK_STRONGER", "SUPPORT_STRONGER", "TIE". If they are orthogonal or insufficiently supported, choose TIE. The choice is then used to calibrate the intrinsic strength of each node in the tree.

STATEMENT:
"[Statement Text]"
SUPPORT ARGUMENT:
[Supporting Argument Text]
ATTACK ARGUMENT:
[Attacking Argument Text]

Return JSON ONLY with keys: choice ∈ {ATTACK_STRONGER, SUPPORT_STRONGER, TIE}.
Example:{"choice":"ATTACK_STRONGER"}