Journal Title
XX(X):2–39

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Sound and Complete Neurosymbolic Reasoning with LLM-Grounded Interpretations

Bradley P. Allen[1], Prateek Chhikara[2], Thomas Macaulay Ferguson[3], Filip Ilievski[4] and Paul Groth[1]

Abstract

Large language models (LLMs) have demonstrated impressive capabilities in natural language understanding and generation, but exhibit problems with logical consistency in their output. How can we harness LLMs' broad-coverage parametric knowledge in formal reasoning despite their inconsistency? We present a method for directly integrating an LLM into the interpretation function of the formal semantics for a paraconsistent logic. We evaluate the method empirically using datasets derived from the short-form factuality benchmarks GPQA and SimpleQA, showing that bilateral factuality evaluation improves macro-F1 over a unilateral baseline by roughly 6 percentage points on both benchmarks (at the cost of reduced coverage, as abstention is triggered on inconsistent or uncertain cases). We further describe a proof-of-concept tableau reasoner implementing the method, and apply it to a medication-safety knowledge base of 228 asserted and 712 inferred statements: the system detects 92 gluts corresponding to medically significant errors (e.g., opioids inferred as non-addictive, beta-blockers inferred as safe in asthma) while remaining satisfiable, demonstrating that contradictions are localized rather than causing logical explosion. Unlike prior work, our method offers a theoretical framework with a practical implementation for neurosymbolic reasoning that leverages an LLM's knowledge while preserving the underlying logic's soundness and completeness properties.

Keywords

Neurosymbolic AI, Large Language Models, Paraconsistent Logic, Factuality Evaluation, Knowledge Representation

Prepared using sagej.cls *[Version: 2017/01/17 v1.20]*

1 Introduction

Applications involving commonsense reasoning and biomedical knowledge, where inconsistencies and incomplete information are commonplace, remain challenging frontiers for AI systems. While LLMs encode vast parametric knowledge (Petroni et al. 2019), they also suffer from inconsistency and incompleteness (Cheng et al. 2025), limiting their use as knowledge bases for such applications. Efforts to connect logical reasoning with LLMs relying on prompting strategies and external symbolic solvers (Wei et al. 2022; Cheng et al. 2025) show potential but exhibit significant shortcomings as well (Hoppe et al. 2025), lacking formal frameworks for managing LLM knowledge inconsistency and incompleteness.

Paraconsistent logics (Priest et al. 2025) are multi-valued non-classical logics that handle inconsistent information without logical explosion, where contradictions would make everything provable. *Belnap computers* (Belnap 1977a,b) are theoretical constructions described by Nuel Belnap to model contexts in which machines are responsible for reasoning in the face of incomplete or inconsistent information.

We propose a Belnap computer using an *LLM judge* as an external knowledge source (Figure 1). LLM judges scale factuality evaluation of LLM output for short- or long-form question answering tasks, returning truth valuations for statements in the LLM's output (Li et al. 2024). In our approach, an LLM judge responds to a query for the valuation of an atomic formula in the context of a paraconsistent reasoner with a *generalized truth value* (Shramko and Wansing 2025, 2011). Generalized truth values allow the LLM judge to provide information to the reasoner not only about the degree of truth of an atomic formula given the LLM's parametric knowledge, but also with respect to the degree of knowledge the LLM has about the formula.

Section 2 discusses related work, and Section 3 defines ***a bilateral factuality evaluation function (contribution i)***, which separately assesses whether an LLM can verify and whether it can refute a claim made by an atomic formula. Unlike standard factuality evaluation, which yields a single truth value, bilateral evaluation surfaces the LLM's epistemic state: consistent knowledge, inconsistency (where the LLM both affirms and denies), or uncertainty (where the LLM can do neither). This is valuable as an evaluation method in and of itself, providing actionable information beyond that provided by current LLM judge approaches to factuality evaluation. Section 4

[1]University of Amsterdam, Amsterdam, The Netherlands
[2]University of Southern California, Los Angeles, CA, United States
[3]Rensselaer Polytechnic Institute, Troy, NY, United States
[4]Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

Corresponding author:
Bradley P. Allen
Email: b.p.allen@uva.nl

Prepared using sagej.cls

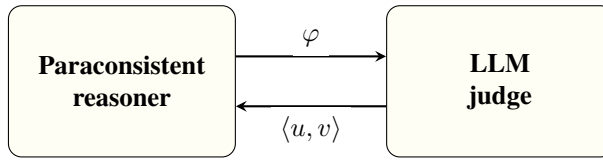

Figure 1. A Belnap computer using an LLM judge as a source of knowledge. Let $\mathcal{L}$ be the object language for a paraconsistent logic, and let $\mathcal{L}_{AT}$ be the set of atomic formulas. A paraconsistent reasoner (left) sends an atomic formula $\varphi \in \mathcal{L}_{AT}$ to the LLM judge (right), which returns a generalized truth value $\langle u, v \rangle$, such that u indicates if the LLM judge was able to verify φ, and v indicates if the LLM judge was able to refute φ.

shows how bilateral evaluation integrates naturally into the formal semantics of a paraconsistent logic through ***an LLM-grounded interpretation (contribution ii)*** that preserves the soundness and completeness of analytic tableau systems for reasoning in the logic, enabling principled reasoning over LLM parametric knowledge despite its inherent inconsistency and incompleteness. Section 5 provides ***empirical evidence for practical implementation of bilateral factuality evaluation (contribution iii)***, presenting evaluation findings using benchmarks derived from short-form factuality benchmarks and discussing limitations. Section 6 describes ***a demonstration of a scalable proof-of-concept implementation of a Belnap computer as a paraconsistent reasoner with LLM integration (contribution iv)*** as described in Section 3 and 4, and demonstrates our approach's feasibility and scalability. Section 7 discusses limitations of our approach, and Section 8 concludes with a summary and discussion of future work.*

2 Related work

Logical reasoning with LLMs Current approaches to reasoning with LLMs (Hoppe et al. 2025; Cheng et al. 2025) appear in Figure 2. In a *prompt-based* approach (*a*), an LLM is prompted with a verbalization $\delta(\Gamma) \in \Sigma^*$ of a set of formulas Γ, and performs natural language reasoning to produce a verbalization $\delta(\varphi)$ of φ (Wei et al. 2022; Kojima et al. 2022; Dhuliawala et al. 2023; Yao et al. 2023). In a *solver-based* approach (*b*), an LLM is prompted with a verbalization of a set of formulas and produces a set of formulas Γ in an object language $\mathcal{L}$, which a reasoner uses to deduce φ (Pan et al. 2023; Olausson et al. 2023; Callewaert et al. 2025). In an approach based on *pre-training or fine-tuning* (*c*), a reasoner provides a training set of proofs Π, and the LLM learns from that to reason as in (*a*) (Jiao et al. 2023; Morishita et al. 2024; Feng et al. 2024; Liu et al. 2025). Unlike approaches (*b*) and (*c*) that use LLMs alongside reasoning systems, in our proposed *interpretation-based* approach (*d*) we integrate LLMs directly into the formal semantics of the reasoner's logic itself, by using an LLM to implement an interpretation function $\mathcal{I}$ to be used by the reasoner in inferring φ. This allows us to provide formal guarantees

*We note that this article is an extended version of Allen et al. (2025); we express our deep appreciation to the two anonymous reviewers for valuable feedback which contributed to this version.

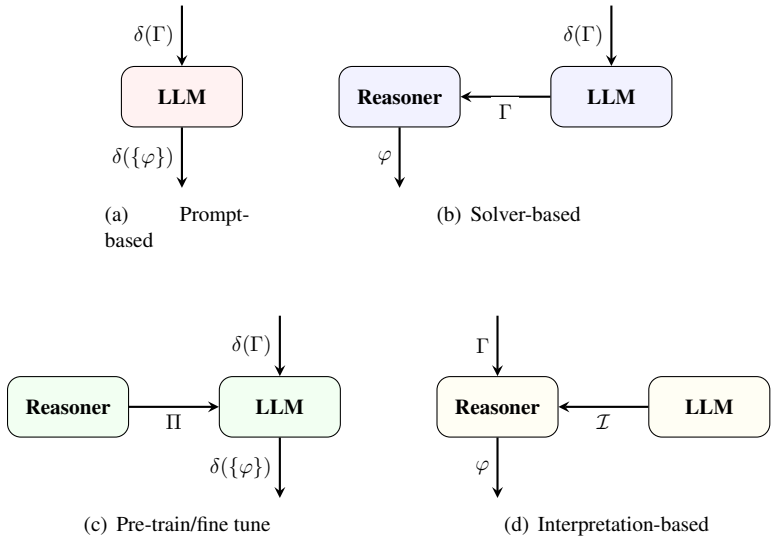

Figure 2. Approaches to logical reasoning with LLMs. Let $\mathcal{L}$ be a first-order language, Γ be a set of statements in $\mathcal{L}$, φ be a statement in $\mathcal{L}$, $\mathcal{I}$ be an interpretation for $\mathcal{L}$, Π be a set of proofs of statements in $\mathcal{L}$, and $\delta : \mathcal{P}(\mathcal{L}) \to \Sigma^*$ be a verbalization function that takes a set of formulas in $\mathcal{L}$ and returns a natural language translation of the formulas. In each approach, we show how reasoning is performed in the context of generating a formal or natural language proof showing that $\Gamma \vdash \varphi$.

about the soundness and completeness of the reasoning process that incorporates the LLM. In contrast with approaches (*a*), (*b*), and (*c*), instead of trying to get an LLM to reason using logic, we get a logic to reason using an LLM.

This architectural distinction has important methodological implications. Systems such as Logic-LM (Pan et al. 2023) and LINC (Olausson et al. 2023) fall within configurations (b) or (c): they use neural components to generate or transform logical representations, which are then processed by symbolic reasoners. DeepProbLog (Manhaeve et al. 2018) shares our interpretation-based architecture but operates within classical probabilistic semantics, which cannot accommodate the contradictions LLMs routinely produce. Our approach addresses a different problem than do these systems: how to formally integrate unreliable knowledge sources (in our case, LLMs) into reasoning systems while preserving soundness and completeness guarantees. The relevant comparison is not benchmark accuracy on clean reasoning tasks, but robustness to the inconsistencies that real-world knowledge sources exhibit. Our contribution is that bilateral evaluation surfaces the epistemic state of the LLM, and paraconsistent semantics provides principled machinery for reasoning over that state.

Factuality evaluation using LLM judges Factuality evaluation is the assessment of whether the output of a language model is factually correct (Wang et al. 2023; Bang

et al. 2025). Recent work has focused on LLM judges (Zheng et al. 2023; Zhu et al. 2023) as a means to scale factuality evaluation, by prompting an LLM to produce a truth value assignment of the output of an LLM that is being evaluated, specifically a short- (Wei et al. 2024) or long-form (Jacovi et al. 2025) answer to a question. We apply LLM judges to assign generalized truth values to natural language translations of atomic formulas, taking a single-answer grading approach (Zheng et al. 2023). In contrast with current approaches to factuality evaluation, the use of generalized truth values provides information as to the LLM's epistemic stance towards the formula in question.

Generalized truth values Generalized truth values offer significant advantages over truth valuations provided by current approaches to factuality evaluation using LLM judges, presenting a more sophisticated framework for handling complex evidential scenarios. These systems enable systematic distinctions between different types of evidence, as demonstrated by the theoretical work of Shramko and Wansing (2011) and Ferguson (2021b), allowing evaluators to categorize and weight various forms of factual support rather than treating all evidence as equivalent. Additionally, generalized truth values provide principled methods for handling inconsistent and incomplete information, a critical advantage given the messy nature of real-world factual claims where evidence may be contradictory or absent, as explored in the foundational work by Shramko and Wansing (2011), the extensions by Szmuc (2019) and Ferguson (2021b), and the theoretical groundwork laid by Correia (2010). Perhaps, most importantly, for practical evaluation applications, these frameworks enhance the explanatory transparency of logical valuations through their structured nature, as noted by Shramko and Wansing (2011), Ferguson (2021b), and Fine (2016), providing clear logical foundations that make evaluation decisions more interpretable and allowing researchers to understand not just whether a claim is deemed factual, but why that determination was reached and what types of evidence contributed to the final assessment. Our work, following Ferguson (2021b), uses truth values that are elements of the bilattice $\mathcal{NINE}$ (Arieli and Avron 1998).

Bilateralism Bilateralism in logic (Rumfitt 2000) holds that understanding a proposition requires grasping both the conditions under which it can be asserted and the conditions under which it should be denied. Meaning isn't just about knowing when something is true, but also explicitly understanding when it is false. This philosophical view contrasts with unilateral approaches where only conditions for truth or assertion are primary, and falsity or denial is treated as derivative (just the absence or negation of truth). Bilateralists argue this misses something fundamental about meaning and inference, and that having explicit roles for both verification and refutation leads to better logical reasoning and clearer understanding. Our work empirically validates Rumfitt's philosophical position: treating assertion and denial as primitive speech acts (rather than reducing one to the other) provides superior analytical power.

Multi-valued logics for paraconsistent reasoning Work that builds on Belnap's four-valued semantics has led to the development of a range of non-classical paraconsistent logics. Patel-Schneider (1989) showed how this idea could be applied to make terminological logics capable of performing subsumption correctly in the presence of

contradictory knowledge. Kamide (2010), Ma et al. (2007), and Maier et al. (2013) expanded on this work to define a number of paraconsistent description logics. More recently, Ferguson (2017b) has proposed a computational interpretation of versions of first degree entailment (**FDE**) and Richard Angell's logic of analytic containment (**AC**). This interpretation models reasoners using these logics as Belnap computers, and leads to a formal framework for **FDE** and **AC** as bilateral logics with sound and complete analytic tableau systems. We show how an LLM judge can be used to provide an interpretation for **AC** that preserves the soundness and completeness of the tableau system **ACrQ**, with applications to description logics as described in (Ferguson 2021a).

Interpreting symbols in logical statements as natural language terms Previous efforts have explored the degree to which meaning can be captured through the use of natural language terms as symbols in a formal knowledge representation language. Google distance between symbol names has been used to weight ontology matches, explicitly leveraging that ontology concepts are often named with meaningful words (Gligorov et al. 2007). Semantic distances between natural language representations of concepts have been used to reason with inconsistent ontologies, selecting maximally consistent subsets based on linguistic similarity (Huang and van Harmelen 2008). The degree to which symbol names carry social meaning in knowledge graphs has been quantified, providing an information-theoretic framework for measuring the effectiveness of this symbol-language correspondence (de Rooij et al. 2016). Most recently, work on context-aware clause selection in theorem proving has shown that symbol names can provide valuable heuristics even in purely formal reasoning tasks (Schon 2025). Our approach builds on these efforts, using the linguistic processing capabilities of LLMs to interpret the predicate and constant symbols in atomic formulas in **AC** as terms in natural language. At first glance, this may seem to fly in the face of traditional approaches to logic; however, traditional formal systems have always required human judgment to determine what constitutes truth for atomic formulas. Our contribution makes this interpretative act explicit by using LLMs to operationalize how humans assess claims through language.

3 Bilateral factuality evaluation of an atomic formula using an LLM

AC is a *conceptivist* logic (Ferguson 2017a) that addresses hierarchical relationships between concepts. While in this work we focus on **AC** as a paraconsistent logic, it is also *paracomplete*, i.e., it rejects the law of the excluded middle. The combination of those two properties makes it particularly suitable for applications involving vague predicates, incomplete information, or situations where classical logic's demands for both consistency and completeness are too strong. Appendix A provides a definition of **AC** with restricted quantification, which is necessary to support concept subsumption and existential quantification of roles when used as a description logic (Ferguson 2021b). This definition treats **AC** as a *bilateral* logic, i.e., a logic which manages values for both the truth and falsity of a formula separately. Bilateralism in philosophical logic (Rumfitt

2000) holds that understanding a proposition requires grasping both the conditions under which it can be asserted and the conditions under which it should be denied. We operationalize this principle by evaluating the factuality of atomic formulas using an LLM judge.

First, we generate a natural language verbalization of an atomic formula, then prompt the LLM to generate two statements on the verifiability and refutability of the assertion. The statements are then mapped into the set of truth values $\mathcal{V}_3$ used in weak Kleene logic (Kleene 1952; Szmuc 2019), i.e., $\mathfrak{t}$ (true), $\mathfrak{e}$ (undefined), and $\mathfrak{f}$ (false). Weak Kleene truth values allow us to formalize the semantics of LLM-judge output. For example, the SimpleQA grader used in the SimpleQA benchmark (Wei et al. 2024) grades an LLM's answer to a question as either "CORRECT", "INCORRECT", or "NOT ATTEMPTED"; the PreciseWikiQA Question Answerability Prompt in the HalluLens benchmark (Bang et al. 2025) uses "UNVERIFIABLE" instead of "NOT ATTEMPTED". We equate "CORRECT" with $\mathfrak{t}$ and "INCORRECT" with $\mathfrak{f}$; equating $\mathfrak{e}$ with "NOT ATTEMPTED" or "UNVERIFIABLE" is consistent with Kleene's original statement that it indicates "an absence of information" that a given formula is either $\mathfrak{t}$ or $\mathfrak{f}$ (Kleene 1952, p. 333).

Finally, we pair the weak Kleene truth value u for verifiability with the weak Kleene truth value v for refutability to yield a generalized truth value $\langle u, v\rangle \in \mathcal{V}_3 \times \mathcal{V}_3$. Generalized truth values offer significant advantages over truth values provided by current approaches to factuality evaluation using LLM judges: they enable systematic distinctions between different types of evidence (Shramko and Wansing 2011; Ferguson 2021b), provide principled methods for handling inconsistent and incomplete information (Shramko and Wansing 2011; Szmuc 2019; Ferguson 2021b; Correia 2010), and enhance the explanatory transparency of logical valuations through their structured nature (Shramko and Wansing 2011; Ferguson 2021b; Fine 2016). Figure 3 shows an example of this process in action; a longer example is provided in Appendix C.

Preliminaries Let Σ be a countable set of tokens and Σ^* be the set of finite sequences of tokens $\ulcorner t_0 \dots t_k \urcorner$, where $t_{0\leq i\leq k} \in \Sigma$, $k \in \mathbb{N}$. Given sequences $\sigma, \sigma' \in \Sigma^*$, $\sigma \prec \sigma'$ iff σ is a *proper contiguous subsequence* of σ'. Let $L_{\mathfrak{C}}$ be an LLM trained on a corpus $\mathfrak{C} \in \mathcal{P}(\Sigma^*)$.

Definition 3.1. *A* verbalization function $\delta : \mathcal{P}(\mathcal{L}) \to \Sigma^*$ *is a total function that maps a set of formulas to a sequence of tokens.*

The verbalization function δ thus serves as a critical bridge between formal logic and natural language in our framework, and makes the assumption that symbols in our logical language can meaningfully correspond to words or phrases that an LLM can interpret. In practice, δ can be implemented through three distinct approaches:

1. **Direct syntactic mapping**: Logical formulas are provided directly in their formal syntax (e.g., `yearOfDiscovery(America, 1492)`). This assumes the LLM can parse and understand logical notation. This is the approach used in the implementation of bilateral factuality evaluation used in the evaluation in Section 5 and the tableau reasoning system described in Section 6.

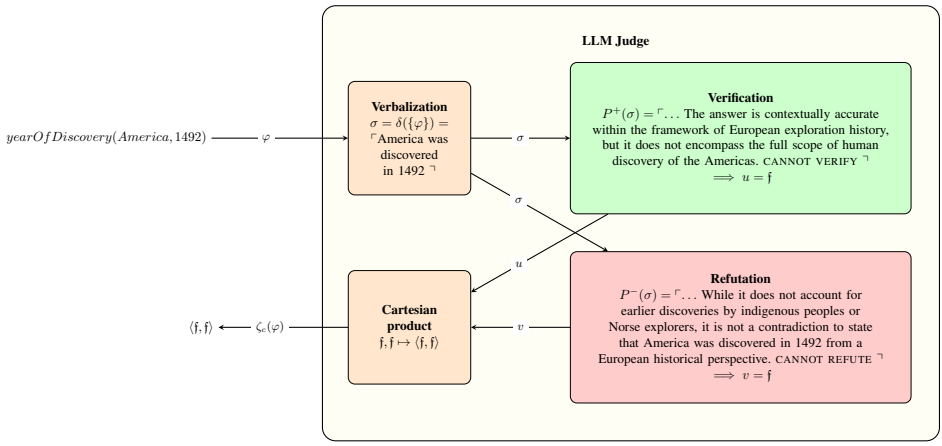

Figure 3. An example of bilateral factuality evaluation ζ_c as performed by the LLM judge shown in Figure 1. Let $\varphi \in \mathcal{L}_{AT}$ be an assertion that the discovery of America occurred in the year 1492. $\sigma = \delta(\{\varphi\})$ is the verbalization of φ (Definition 3.1). The bilateral factuality evaluation of φ by an LLM judge (Definitions 3.2, 3.3, and 3.4) generates the truth value $\zeta_c(\varphi) = \langle \mathfrak{f}, \mathfrak{f} \rangle$. The LLM has in effect identified *incompleteness* in its knowledge — based on differing perspectives on who discovered America, it can neither verify nor refute φ.

2. **Template-based transformation**: Formulas are converted using predefined templates (e.g., "America was discovered in 1492"). This approach provides more natural language but requires manual template creation for each predicate type.
3. **LLM-generated verbalization**: An LLM generates the natural language representation of the formula. This is the most flexible approach, but it introduces another layer of potential error and interpretation.

Definition 3.2. *A* verification function $P^+ : \Sigma^* \to \Sigma^*$ *prompts* $L_{\mathfrak{C}}$ *to take a verbalization of an atomic formula* $\varphi \in \mathcal{L}_{AT}$ *and generate a token sequence* σ^+ *that states that* φ *is verified or that it cannot be verified.*

Definition 3.3. *A* refutation function $P^- : \Sigma^* \to \Sigma^*$ *prompts* $L_{\mathfrak{C}}$ *to take a verbalization of an atomic formula* $\varphi \in \mathcal{L}_{AT}$ *and generate a token sequence* σ^- *that states that* φ *is refuted or that it cannot be refuted.*

Definition 3.4. *A* bilateral factuality evaluation function $\zeta : \mathcal{L}_{AT} \to \mathcal{V}_3 \times \mathcal{V}_3$ *is a total function that given an atomic formula* $\varphi \in \mathcal{L}_{AT}$ *yields a pair* $\langle u, v \rangle$ *where:*

$$u = \begin{cases} \mathfrak{t} & \text{if } \ulcorner \mathsf{VERIFIED} \urcorner \prec P^+(\delta(\{\varphi\})) \\ \mathfrak{f} & \text{if } \ulcorner \mathsf{CANNOT\ VERIFY} \urcorner \prec P^+(\delta(\{\varphi\})) \\ \mathfrak{e} & \text{otherwise} \end{cases}$$

$$v = \begin{cases} \mathfrak{t} & \textit{if } \ulcorner \mathsf{REFUTED} \urcorner \prec P^{-}(\delta(\{\varphi\})) \\ \mathfrak{f} & \textit{if } \ulcorner \mathsf{CANNOT\ REFUTE} \urcorner \prec P^{-}(\delta(\{\varphi\})) \\ \mathfrak{e} & \textit{otherwise} \end{cases}$$

The "otherwise" cases in Definition 3.4 reflect LLMs failing to output tokens indicating verification or refutation state, e.g., by failing to follow instructions or timing out during API calls. We use repeated sampling with majority vote (Brown et al. 2024) to determine truth value components; other hallucination mitigation approaches, such as chain-of-verification (Dhuliawala et al. 2023), are also admissible.

The definition of ζ leaves open the possibility that multiple calls over time might return different truth values. This violates the assumption of analytic tableau reasoning that atomic formulas have stable truth values within the scope of the reasoning process. We ensure this by using a caching version of the bilateral evaluation function ζ_c where valuations for atomic formulas are persistently and immutably stored. This type of caching is consistent with the range of optimization techniques used in description logics tableau reasoners (Goré and Nguyen 2007; Nguyen 2009).

Definition 3.5. *A* caching bilateral factuality evaluation function $\zeta_c : \mathcal{L}_{AT} \to \mathcal{V}_3 \times \mathcal{V}_3$ *is a total function defined as:*

$$\zeta_c(\varphi) = \begin{cases} c(\varphi) & \textit{if } \varphi \in \textit{dom}(c) \\ \zeta(\varphi) & \textit{otherwise, and } c := c \cup \{(\varphi, \zeta(\varphi))\} \end{cases}$$

where c is a persistent and immutable cache mapping atomic formulas to truth value pairs, and $dom(c)$ denotes the domain of c, i.e., the set of atomic formulas for which a valuation has already been stored: $\mathrm{dom}(c) = \{\varphi \in L_{AT} \mid \exists \langle u, v\rangle \in \mathcal{V}_3 \times \mathcal{V}_3$ *such that* $(\varphi, \langle u, v\rangle) \in c\}$.

Having established a bilateral evaluation, we now show how this enables LLM judges to implement interpretation functions directly.

4 LLM-grounded interpretations

We formalize how the bilateral evaluation function ζ_c is integrated into the definition of an interpretation for **AC**, show that for every LLM-grounded **AC** interpretation there is an equivalent standard **AC** interpretation, and then show that soundness and completeness of the tableau-style analytic calculus **ACrQ** defined in Definition 18 of Ferguson (2021b) is preserved when we adopt an LLM-grounded interpretation.

These results are metalogical: they concern the reasoner's treatment of atomic valuations, not the quality of the valuations themselves, which we address empirically in Section 5. They assume stability (guaranteed via caching; Definition 3.5, Lemma 4.1) and faithful verbalization of atomic formulas by δ (Section 7). Throughout this section we write $\Gamma \models_{\mathcal{I}} \varphi$ to denote validity relative to the class of interpretations over which $\mathcal{I}$ is understood to range (either all standard **AC** interpretations or all LLM-grounded **AC** interpretations, as disambiguated by context), rather than model-checking in a single interpretation.

Definition 4.1. *An* LLM-grounded **AC** interpretation $\mathcal{I} = \langle \mathbf{C}^{\mathcal{I}}, \mathbf{R}^{\mathcal{I}} \rangle$ *is an* **AC** *interpretation such that for every function* $R^{\mathcal{I}} \in \mathbf{R}^{\mathcal{I}}$ *and* $c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}} \in \mathbf{C}^{\mathcal{I}}$*:*

$$R^{\mathcal{I}}(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}}) = \zeta_c(R(c_1, \ldots, c_n))$$

Lemma 4.1. Stability of LLM-grounded interpretations. *For any LLM-grounded* **AC** *interpretation* $\mathcal{I}$ *and atomic formula* $\varphi \in \mathcal{L}_{AT}$*:*

1. $\mathcal{I}(\varphi)$ *is well-defined and yields exactly one pair* $\langle u, v \rangle \in \mathcal{V}_3 \times \mathcal{V}_3$
2. *Once computed,* $\mathcal{I}(\varphi)$ *remains constant throughout the reasoning process*

Proof. The stability follows directly from Definition 3.5 of ζ_c. Let t_0 be the time at which ζ is first called to set $c(\varphi)$. If $\zeta_c(\varphi) = \langle u, v \rangle$ at time t_0, then for all subsequent calls at $t > t_0$, $\zeta_c(\varphi) = \langle u, v \rangle$. This ensures that once an atomic formula φ has been evaluated and the returned pair $\langle u, v \rangle \in \mathcal{V}_3 \times \mathcal{V}_3$ has been cached, all subsequent evaluations will return the same pair from the cache, guaranteeing stability.

Lemma 4.1 states that the logical validity of derivations depends on logical operator structure rather than atomic proposition content, a fundamental principle in formal logic. The tableau method only depends on the truth-functional behavior of the logical connectives, which remains unchanged between standard and LLM-grounded interpretations.

Lemma 4.2. LLM-grounded to standard interpretation mapping. *For any LLM-grounded* **AC** *interpretation* $\mathcal{I} = \langle \mathbf{C}^{\mathcal{I}}, \mathbf{R}^{\mathcal{I}} \rangle$*, there exists a standard* **AC** *interpretation* $\mathcal{I}'$ *that preserves the semantic behavior of* $\mathcal{I}$ *on all formulas.*

Proof. Given an LLM-grounded **AC** interpretation $\mathcal{I} = \langle \mathbf{C}^{\mathcal{I}}, \mathbf{R}^{\mathcal{I}} \rangle$, we define a standard **AC** interpretation $\mathcal{I}' = \langle \mathbf{C}^{\mathcal{I}'}, \mathbf{R}^{\mathcal{I}'} \rangle$ such that:

1. $\mathbf{C}^{\mathcal{I}'} = \mathbf{C}^{\mathcal{I}}$
2. $\mathbf{R}^{\mathcal{I}'} = \mathbf{R}^{\mathcal{I}}$
3. For all $R \in \mathbf{R}$, $R^{\mathcal{I}'}(c_1^{\mathcal{I}'}, \ldots, c_n^{\mathcal{I}'}) = R^{\mathcal{I}}(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}})$
4. For all $c \in \mathcal{C}'$, $c^{\mathcal{I}'} = c^{\mathcal{I}}$

We show that for any formula $\varphi \in \mathcal{L}$, $\mathcal{I}(\varphi) = \mathcal{I}'(\varphi)$ by induction on the complexity of φ:

- For $\varphi = R(c_1, \ldots c_n) \in \mathcal{L}_{AT}$, then by Definition 4.1, $\mathcal{I}(\varphi) = \mathcal{I}(R(c_1, \ldots, c_n)) = R^{\mathcal{I}}(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}}) = R^{\mathcal{I}'}(c_1^{\mathcal{I}'}, \ldots, c_n^{\mathcal{I}'}) = \mathcal{I}'(R(c_1, \ldots, c_n)) = \mathcal{I}'(\varphi)$.
- For $\varphi = \neg\psi$, by Definition A.5, $\mathcal{I}(\neg\psi) = \langle \mathcal{I}_1(\psi), \mathcal{I}_0(\psi) \rangle$ and $\mathcal{I}'(\neg\psi) = \langle \mathcal{I}'_1(\psi), \mathcal{I}'_0(\psi) \rangle$. By the inductive hypothesis, $\mathcal{I}(\psi) = \mathcal{I}'(\psi)$. Therefore $\mathcal{I}(\neg\psi) = \mathcal{I}'(\neg\psi)$.
- For $\varphi = \psi \wedge \chi$, by Definition A.5, $\mathcal{I}(\psi \wedge \chi) = \langle \mathcal{I}_0(\psi) \dot{\wedge} \mathcal{I}_0(\chi), \mathcal{I}_1(\psi) \dot{\vee} \mathcal{I}_1(\chi) \rangle$ and similarly for $\mathcal{I}'$. By the inductive hypothesis, $\mathcal{I}(\psi) = \mathcal{I}'(\psi)$ and $\mathcal{I}(\chi) = \mathcal{I}'(\chi)$. Therefore $\mathcal{I}(\psi \wedge \chi) = \mathcal{I}'(\psi \wedge \chi)$.

The same arguments apply in the cases of disjunction, restricted universal quantification, and restricted existential quantification, again following Definition A.5 in Appendix A.

Lemma 4.3. Standard to LLM-grounded interpretation mapping. *For any standard* **AC** *interpretation* $\mathcal{I} = \langle \mathbf{C}^{\mathcal{I}}, \mathbf{R}^{\mathcal{I}} \rangle$*, there exists an LLM-grounded* **AC** *interpretation* $\mathcal{I}'$ *that preserves the semantic behavior of* $\mathcal{I}$ *on all formulas.*

Proof. Let $\mathcal{I} = \langle \mathbf{C}^{\mathcal{I}}, \mathbf{R}^{\mathcal{I}} \rangle$ be a standard **AC** interpretation. Define a key-value store $c_{\mathcal{I}}$ such that for all atomic $R(c_1, ..., c_n) \in \mathcal{L}_{AT}$, $c_{\mathcal{I}}(R(c_1, ..., c_n)) = R^{\mathcal{I}}(c_1^{\mathcal{I}}, ..., c_n^{\mathcal{I}})$. Since $c_{\mathcal{I}}$ is defined on every atomic formula, $\mathrm{dom}(c_{\mathcal{I}}) = \mathcal{L}_{AT}$, and so by Definition 3.5, $\zeta_{c_{\mathcal{I}}}$ returns $c_{\mathcal{I}}(\varphi)$ directly for each $\varphi \in \mathcal{L}_{AT}$, without invoking ζ. The induced LLM-grounded **AC** interpretation $\mathcal{I}'$ therefore agrees with $\mathcal{I}$ on all atoms by construction. An induction on formula complexity, parallel to that in Lemma 4.2, extends this agreement to all formulas.

Corollary 4.1. Extensional equivalence of validity. $\Gamma \vDash_{\mathbf{AC}} \varphi$ *holds if and only if the inference from* Γ *to* φ *is valid over all LLM-grounded* **AC** *interpretations.*

Proof. Immediate from Lemmas 4.2 and 4.3. This extensional equivalence is what allows the metalogical properties of **ACrQ** to transfer to the LLM-grounded setting, as the following two theorems make explicit.

Theorem 4.1. Preservation of soundness. *Let* Γ *be a finite set of formulas and* φ *a formula in* **AC***. If* $\Gamma \vdash_{\mathbf{ACrQ}} \varphi$*, then* $\Gamma \models_{\mathcal{I}} \varphi$ *is valid for all LLM-grounded* **AC** *interpretations* $\mathcal{I}$*.*

Proof. By Theorem 3 of Ferguson (2021b), if $\Gamma \vdash_{\mathbf{ACrQ}} \varphi$, then $\Gamma \models_{\mathbf{AC}} \varphi$. Therefore $\Gamma \models_{\mathcal{I}} \varphi$, as is the case for any **AC** interpretation, standard or LLM-grounded.

Theorem 4.2. Preservation of completeness. *Let* Γ *be a finite set of formulas and* φ *a formula in* **AC***. Then if for all LLM-grounded* **AC** *interpretations* $\mathcal{I}$*,* $\Gamma \models_{\mathcal{I}} \varphi$*, then* $\Gamma \vdash_{\mathbf{ACrQ}} \varphi$*.*

Proof. Let $\mathcal{I}'$ be an arbitrary standard **AC** interpretation. By Lemma 4.3, there exists an LLM-grounded **AC** interpretation $\mathcal{I}$ that agrees with $\mathcal{I}'$ on all formulas. By hypothesis, $\Gamma \models_{\mathcal{I}} \varphi$, and therefore $\Gamma \models_{\mathcal{I}'} \varphi$. Since $\mathcal{I}'$ was arbitrary, $\Gamma \models_{\mathcal{I}'} \varphi$ holds for every standard **AC** interpretation. Then by Theorem 4 of Ferguson (2021b), $\Gamma \vdash_{\mathbf{ACrQ}} \varphi$.

Section 6 presents our implementation of these theoretical concepts in a complete tableau reasoning system, demonstrating the practical realizability of LLM-grounded interpretations.

		Bilateral (ζ)		Unilateral	
Dataset	**Model Type**	**Macro F1**	**Coverage**	**Macro F1**	**Coverage**
GPQA	flagship	0.699 (0.010)	0.589 (0.008)	0.633 (0.007)	1.000 (0.000)
	distilled	0.608 (0.011)	0.504 (0.008)	0.559 (0.008)	1.000 (0.000)
SimpleQA	flagship	0.736 (0.009)	0.584 (0.008)	0.657 (0.008)	1.000 (0.000)
	distilled	0.624 (0.011)	0.499 (0.008)	0.570 (0.008)	1.000 (0.000)

Table 1. Summary macro F1 (given abstention) and coverage metrics for the bilateral factuality evaluation function ζ and a baseline unilateral factuality evaluation function.

5 Factuality evaluation of LLM-grounded interpretations

To validate the factuality of LLM-grounded interpretations, we evaluate the bilateral evaluation function ζ that underlies them, using question/answer pairs from two short-form factuality benchmarks. Given our theoretical results above, this focus on the practicality of atomic formula valuation within our framework provides a proof-of-concept that a Belnap computer of the type described above is feasible.[†]

Data and metrics Question/answer pairs in short-form factuality benchmarks are typically factoids providing a question together with a short answer (Figure 3). We use these as an approximation to the verbalization $\delta(\varphi)$ of an atomic formula φ. We used the short-form factuality benchmarks GPQA (Rein et al. 2023) and SimpleQA (Wei et al. 2024) to create two test datasets (each with N=400) for our experiments, each balanced between positive and negative examples. Test data preparation and experimental setup are discussed in Appendix D. We evaluated the performance of ζ over the two datasets using two metrics: *macro F1* against question/answer pairs that the judge did not abstain from, i.e., where the judge provided a valuation of $\zeta(\varphi) = \langle \mathfrak{t}, \mathfrak{f} \rangle$ (i.e., verified and not refuted) or $\zeta(\varphi) = \langle \mathfrak{f}, \mathfrak{t} \rangle$ (i.e., not verified and refuted), and *coverage*, which is the percentage of the total set of question/answer pairs where the judge did not abstain. We also measured the *time taken per evaluation*, and the *number of tokens used per evaluation*.

LLM judges We used three flagship LLMs (Llama 4 Maverick, GPT-4o, and Claude 3.5 Sonnet), and three distilled LLMs (Llama 4 Scout, GPT-4o Mini, and Claude 3.5 Haiku). Each LLM was evaluated using three different pairs of prompts: direct prompts that asked for a verification or refutation for the QA pair, zero-shot chain-of-thought prompts, and few-shot chain-of-thought prompts. As a baseline, we also used the six models and three prompt types to perform *unilateral* factuality evaluation, which asks an LLM to simply determine whether a question/answer pair is $\mathfrak{t}$ or $\mathfrak{f}$. The prompt templates used are provided in Appendix B. Standard errors (in parentheses) presented in tables in this section and in Appendix D were estimated by bootstrap resampling: 1000 subsamples of size N=100 were drawn from the classification results within each model category (Politis and Romano 1994).

[†]Code and data for the experiment is available at https://github.com/bradleypallen/bilateral-factuality-evaluation.

Results Table 1 compares macro F1 and coverage between the unilateral and bilateral evaluations across the two datasets, grouped by whether a model's type was flagship or distilled, and Table 2 does the same for mean time of execution and mean tokens used. Table 3 summarizes the distribution of truth values produced in bilateral evaluation. Detailed breakouts are shown in the tables in Appendix D. Our key findings are as follows.

1. ***Bilateral evaluation macro F1 outperforms unilateral evaluation*** $(p < 0.01)$ ***at the cost of lower coverage***. The mean difference between bilateral and unilateral macro F1 is 0.062 and for coverage is -0.456.
2. ***Flagship models outperform distilled models*** $(p < 0.01)$. Table 1 shows that this is the case for both unilateral and bilateral approaches, though the difference is more pronounced with bilateral evaluation (0.091 on the GPQA dataset, 0.112 on the SimpleQA dataset) versus unilateral evaluation (0.074 on the GPQA dataset, 0.087 on the SimpleQA dataset).
3. ***Bilateral evaluation is more expensive than unilateral evaluation*** $(p < 0.01)$. Table 2 shows that bilateral evaluation takes roughly twice as much time and twice as many tokens as unilateral evaluation. However, evaluation times vary widely, with GPT-4o Mini using direct prompting taking a mean of 2.5 seconds, and Llama 4 Scout using zero-shot prompting taking a mean of 43.4 seconds. Token consumption scales predictably, from up to a mean of 2,008.6 tokens with direct prompting, and up to a mean of 6,704.7 tokens with few-shot prompting.
4. ***Inconsistency occurs significantly more frequently than incompleteness*** $(p < 0.05)$. Table 3 shows that bilateral judge models more frequently abstain by assigning $\langle \mathfrak{t}, \mathfrak{t} \rangle$ (both verified and refuted) as opposed to $\langle \mathfrak{f}, \mathfrak{f} \rangle$ (neither verified nor refuted).

We note that in the experiments reported here, the $\mathfrak{e}$ component of the weak Kleene truth value never occurred: every bilateral evaluation produced a pair $\langle u, v \rangle \in \{\mathfrak{t}, \mathfrak{f}\} \times \{\mathfrak{t}, \mathfrak{f}\}$. The otherwise clauses of Definition 3.4 — which assign $\mathfrak{e}$ when neither the verification nor the refutation marker tokens appear in the LLM output — were therefore never triggered. We attribute this to two factors. First, the verification and refutation prompt templates (Appendix B) instruct the LLM to terminate its response with a single line drawn from a two-element set ($\{\ulcorner\mathsf{VERIFIED}\urcorner, \ulcorner\mathsf{CANNOT\ VERIFY}\urcorner\}$ and $\{\ulcorner\mathsf{REFUTED}\urcorner, \ulcorner\mathsf{CANNOT\ REFUTE}\urcorner\}$, respectively), which strongly constrains the output distribution. Second, all six models evaluated — three flagship (Llama 4 Maverick, GPT-4o, Claude 3.5 Sonnet) and three distilled (Llama 4 Scout, GPT-4o Mini, Claude 3.5 Haiku) — are highly reliable at following such terminal-line instructions, and we observed no API timeouts or malformed responses over the $2 \times 400 \times 18 = 14,400$ calls underlying the results in Tables 1-3. The $\mathfrak{e}$ component nevertheless remains a principled and necessary part of the formalism: it absorbs genuine instruction-following failures, API errors, and timeouts without corrupting the reasoner's state, and it is likely to be exercised in longer-running deployments, under rate-limit pressure, or with smaller local models whose instruction-following reliability is lower. We retain it in Definition 3.4 precisely for those settings.

		Bilateral (ζ)		Unilateral	
Dataset	**Model Type**	**Time (s)**	**Tokens**	**Time (s)**	**Tokens**
GPQA	flagship	36.747 (0.281)	4781.663 (45.878)	12.411 (0.140)	2212.766 (27.182)
	distilled	34.771 (0.224)	4731.672 (41.661)	13.439 (0.095)	2532.153 (26.615)
SimpleQA	flagship	32.789 (0.274)	4163.807 (37.704)	12.256 (0.140)	2100.465 (25.634)
	distilled	30.310 (0.253)	3964.037 (39.991)	11.424 (0.120)	2167.419 (28.044)

Table 2. Summary execution time (in seconds) and total tokens used for the bilateral factuality evaluation function ζ and a baseline unilateral factuality evaluation function.

Dataset	**Model Type**	$\langle \mathfrak{t}, \mathfrak{t} \rangle$	$\langle \mathfrak{t}, \mathfrak{f} \rangle$	$\langle \mathfrak{f}, \mathfrak{t} \rangle$	$\langle \mathfrak{f}, \mathfrak{f} \rangle$
GPQA	flagship	0.301 (0.008)	0.211 (0.007)	0.378 (0.008)	0.110 (0.005)
	distilled	0.299 (0.007)	0.192 (0.006)	0.312 (0.007)	0.197 (0.006)
SimpleQA	flagship	0.310 (0.007)	0.228 (0.007)	0.357 (0.008)	0.106 (0.005)
	distilled	0.301 (0.007)	0.233 (0.007)	0.266 (0.007)	0.200 (0.006)

Table 3. Summary truth value distributions for the bilateral factuality evaluation function ζ. Of note is the fact that the models evaluated did not produce any truth values where $u = \mathfrak{e}$ or $v = \mathfrak{e}$ during the evaluation.

The accuracy-coverage trade-off A key finding from our experimental results is that bilateral evaluation achieves higher macro F1 than unilateral evaluation at the cost of lower coverage (Table 1). This trade-off is not a limitation but a feature: the increased visibility into the LLM's epistemic state that bilateral evaluation provides enables more conservative and reliable predictions.

The coverage reduction occurs because bilateral evaluation abstains whenever the LLM's epistemic state is indefinite—either inconsistent ($\langle \mathfrak{t}, \mathfrak{t} \rangle$) or ignorant ($\langle \mathfrak{f}, \mathfrak{f} \rangle$). Rather than forcing a commitment when evidence is conflicting or absent, the system acknowledges uncertainty. This epistemic humility is precisely what produces the accuracy improvement: by abstaining on difficult or ambiguous cases, the system's definite verdicts are more reliable.

This pattern aligns with the well-established framework of *selective classification* (also known as classification with abstention or the reject option) (El-Yaniv and Wiener 2010; Geifman and El-Yaniv 2017). Selective classification is motivated by applications where conservative prediction is critical—the cost of an incorrect prediction outweighs the cost of abstention. Medical diagnosis is a canonical example: a system that abstains on ambiguous cases and defers to human judgment is preferable to one that confidently misdiagnoses. The medication safety example in Section 6 illustrates this directly: the system's abstention on `Cardiosafe(aspirin)` (a genuinely contested claim) and its glut detection on `Nonaddictive(valium)` (a claim the LLM can refute) both represent appropriate epistemic caution in a domain where errors carry significant consequences.

Bilateral evaluation provides a principled mechanism for selective classification that goes beyond simple confidence thresholding. Standard approaches to selective

classification typically use model confidence scores to decide when to abstain, but such scores are notoriously miscalibrated in LLMs (Kadavath et al. 2022). Our approach instead grounds abstention in the structure of evidence: abstention occurs when verification and refutation evidence are both present (glut) or both absent (gap). This provides interpretable diagnostics—a downstream system can distinguish between "the LLM is conflicted" and "the LLM lacks relevant knowledge," enabling different handling strategies for each case.

The connection to verbalization (Section 7) is direct: gap valuations ($\langle \mathfrak{f}, \mathfrak{f} \rangle$) arise when the verbalized formula fails to activate relevant LLM knowledge. Improving verbalization strategies would shift some gaps to definite valuations, increasing coverage while maintaining accuracy. Glut valuations ($\langle \mathfrak{t}, \mathfrak{t} \rangle$), by contrast, reflect genuine LLM inconsistency that no verbalization strategy can resolve—these represent cases where the LLM's parametric knowledge is itself contradictory.

Bilateral truth value distributions across subject domains We note that our experimental benchmarks (GPQA and SimpleQA) span diverse subject matters, and the distribution of truth values likely varies across domains. Highly technical domains may produce more gaps due to verbalization challenges; domains with contested claims may produce more gluts. Table 14 presents the distribution of bilateral truth values across ten topic categories in the SimpleQA dataset using zero-shot evaluation, revealing systematic variation in epistemic states both across models and subject domains.

Three distinct behavioral patterns emerge among model families. GPT models exhibit a pronounced tendency toward contradiction, with glut rates ($\langle \mathfrak{t}, \mathfrak{t} \rangle$) averaging 59.0% and near-zero gap rates ($\langle \mathfrak{f}, \mathfrak{f} \rangle \approx 0.1\%$), indicating that these models routinely find both supporting and refuting evidence for the same claim. In contrast, Claude models display greater epistemic caution, producing substantially lower glut rates (19.6%) and higher gap rates (30.4%), suggesting a willingness to withhold judgment when evidence is inconclusive. Llama models occupy an intermediate position with high verification rates ($\langle \mathfrak{t}, \mathfrak{f} \rangle = 29.8\%$) but also elevated glut rates (52.9%).

Notably, topic difficulty varies considerably: Video games and Sports exhibit the highest glut rates (63.6% and 47.3%, respectively), while History shows the lowest (35.7%) and the most consistent cross-model behavior ($\sigma = 15.0$). These findings demonstrate that bilateral evaluation exposes model-specific epistemic signatures that unilateral TRUE/FALSE judgments would collapse, and suggest that certain knowledge domains—particularly entertainment and popular culture—present greater challenges for consistent factuality assessment than more extensively documented domains such as History.

6 A Belnap computer with paraconsistent reasoning and LLM integration

To evaluate the practical utility of our theoretical framework, we implemented a reasoning system for $\mathbf{ACrQ}$ in Python with type annotations and a modular architecture separating formula representation, semantic evaluation, and tableau construction.

System architecture The architecture supports two complementary reasoning modes:

- **Forward-chaining inference** derives new atomic formulas by instantiating universally quantified rules over ground assertions. Given a theory containing rules of the form $t\!:\![\forall X\, P(X)]Q(X)$ and ground assertions $t\!:\!P(a)$, the forward-chaining procedure derives $t\!:\!Q(a)$ and iterates until no new facts can be inferred. Each newly derived atomic formula is then evaluated by the LLM judge using the caching bilateral factuality evaluation function ζ_c, which may introduce additional assertions (confirming the inference) or their duals (indicating LLM refutation), potentially surfacing gluts.
- **Tableau-based satisfiability checking** determines whether the current theory is satisfiable under $\mathbf{ACrQ}$ semantics, extending the tableau calculus defined by Ferguson (2021b). The tableau procedure decomposes signed formulas according to $\mathbf{ACrQ}$ rules until branches either close (indicating unsatisfiability) or remain open (indicating satisfiability). For theories containing gluts—formulas φ where both $t\!:\!\varphi$ and $t\!:\!\varphi^*$ hold—the paraconsistent semantics prevents logical explosion, allowing the reasoner to identify exactly which conclusions are contested while maintaining overall satisfiability.

Together, these two modes support an iterative workflow: users assert formulas in natural language, the system translates them to $\mathbf{AC}$ syntax, forward-chaining derives consequences and queries the LLM for validation, and satisfiability checking identifies any gaps (neither verified nor refuted) or gluts (both verified and refuted) in the resulting theory.

The system adds to the set of tableau rules in Definition 18 in (Ferguson 2021b) an additional LLM evaluation rule that queries external knowledge sources during tableau construction. The LLM evaluation rule is formally specified as:

$$\frac{\mathrm{LLM}(P(a)) = \langle u, v\rangle}{\sigma : P(a) \vdash \Delta_{\mathrm{LLM}}(\sigma, P(a), u, v)}\ \text{(llm-eval)}$$

where Δ_{LLM} generates conclusions based on the bilateral truth value returned by the LLM. This rule is applied with lowest priority, after all deterministic tableau rules have been exhausted. The implementation of the tableau procedure uses the caching bilateral factuality evaluation function ζ_c where valuations for atomic formulas are persistently and immutably stored, ensuring the stability of truth values throughout the reasoning process, consistent with optimization techniques used in description logic tableau reasoners (Goré and Nguyen 2007). The tableau maintains full tree connectivity by chain-connecting initial formulas as siblings, ensuring all properties remain observable and verifiable throughout the reasoning process. The system preserves $\mathbf{ACrQ}$'s paraconsistent properties, handling contradictions (or "knowledge gluts") without logical explosion. For example, when the LLM returns $\langle \mathbf{t}, \mathbf{t}\rangle$, indicating both positive and negative evidence, the tableau construction procedure generates nodes for both $\mathbf{t} : P(a)$ and $\mathbf{t} : P^*(a)$, maintaining the glut without causing branch closure, applying Norihiro Kamide's method (Kamide 2010) of encoding bilateral valuations

for atomic formulas with a predicate P using a predicate P^* that corresponds to P's *refutability* value (Maier et al. 2013; Ferguson 2021b).

Claude Code (Anthropic 2025) was used to accelerate implementation while maintaining correctness through rigorous verification: the system is supported by extensive test suites (311 tests covering compliance with Ferguson (2021b), semantic correctness, and edge cases) with all features validated through code review by two of the authors (BA and TF) before integration.[‡]

To support interactive use of the reasoner, the implementation includes a dialogue management system, Theory Manager, that bridges natural language input and formal reasoning. Natural language assertions are entered by the user, translated into formulas in **AC**, and added to a set of assertions stored persistently by the reasoner as a theory. The user can issue commands to perform forward-chaining inference, and to perform satisfiability checking of the current theory, including gap and glut detection.

As an illustration of the reasoner in action, Figure 4 shows a trace of an interactive session with the Theory Manager. In this example, the user first specifies the use of OpenAI's GPT-4o model with the session, and then makes two assertions in natural language, the canonical example of all humans are mortal and Socrates is a human, which the Theory Manager translates into the **AC** syntax. The user then invokes the prover to apply rules in a forward-chaining fashion, yielding the inference that Socrates is a mortal. Given the appearance of atomic statements in the tableau being constructed, whether directly asserted or inferred, the `llm-eval` rule is being applied; given it agrees with the atomic formulas added to this point, the tableau construction procedure does not add any additional nodes. Then the user asserts that Socrates is a pig, and this time the `llm-eval` rule yields an assertion that Socrates is not a pig. Satisfiability checking invoked by the user identifies that an inconsistency now exists in the set of assertions in the theory; however, as this does not result in explosion per the definition of **AC**, the theory is still satisfiable.

A medication safety reasoning use case To illustrate the system's capabilities for a use case where inconsistency tolerance is practically motivated, we present a medication safety example. Medical knowledge bases frequently contain errors, outdated information, or conflicting guidelines that can lead to incorrect inferences (da Costa and Subrahmanian 1989). Traditional approaches to managing such inconsistencies in medical ontologies focus on detecting and repairing problematic axioms through techniques such as axiom pinpointing (Schlobach et al. 2007; Kalyanpur 2006; Baader and Suntisrivaraporn 2008). However, these methods require that inconsistencies manifest as logical contradictions within the ontology itself; they cannot detect axioms that are logically consistent but factually incorrect given external medical knowledge. Our approach addresses this gap: by validating inferred conclusions against LLM-encoded medical knowledge, we can surface errors (gluts) that would otherwise remain

[‡]Code and data for the reasoner is available at https://github.com/bradleypallen/wkrq; in addition, the Python package used by the reasoner for the implementation of ζ and ζ_c is available at https://github.com/bradleypallen/bilateral-truth.

```
1  theory> llm openai gpt-4o
2  [OK] LLM evaluator enabled: openai
3
4  theory> assert all humans are mortal
5  [OK] Asserted: S0001
6    Formula: t:[forall X Human(X)]Mortal(X)
7
8  theory> assert socrates is a human
9  [OK] Asserted: S0002
10   Formula: t:Human(socrates)
11
12 theory> infer
13 [OK] Inferred 1 new statement(s):
14   I0003: t:Mortal(socrates)              ← logical inference
15
16 theory> assert socrates is a pig
17 [OK] Asserted: S0004
18   Formula: t:Pig(socrates)
19
20 theory> infer
21 [OK] Inferred 1 new statement(s):
22   E0005: t:Pig*(socrates)                ← LLM refutation
23
24 theory> check
25 [OK] Satisfiable
26 [!] Found 1 glut (conflicting evidence):
27   S0004: t:Pig(socrates)                 ← glut detected
28   E0005: t:Pig*(socrates)                ← but satisfiable
```

Figure 4. Example interaction with the Theory Manager demonstrating paraconsistent reasoning. The system correctly infers `Mortal(socrates)` through logical deduction (line 13), then queries the LLM when a false claim is asserted. The LLM returns $\zeta_c(\text{Pig(socrates)}) = \langle \mathfrak{f}, \mathfrak{t} \rangle$ (cannot verify, can refute), resulting in `Pig*(socrates)` (line 22). Despite the glut between the user assertion and LLM evidence (lines 24-25), the system remains satisfiable, demonstrating paraconsistent handling of contradictions.

hidden, while the paraconsistent semantics allows reasoning to continue even when conflicts are detected. This example is deliberately artificial—it includes intentionally incorrect rules to test glut detection—but it demonstrates a capability that could complement existing ontology debugging tools in domains where knowledge bases must be validated against evolving empirical evidence.

The medication safety example knowledge base contains 228 asserted statements encoding 28 universally quantified rules and 200 ground assertions across 16 drug classes. Table 4 summarizes the results of asserting the statements in the knowledge base, running the forward chaining reasoner, and then running the tableau satisfiability checker to detect gluts. Forward-chaining derived 712 new atomic formulas, yielding a theory of 940 total statements. The system detected 92 gluts corresponding to medically

Metric	Value	Glut Category	Count
Asserted statements	228	Opioids nonaddictive	25
Inferred statements	712	Antipsychotics metabolically neutral	15
Satisfiability	SAT	Beta-blockers safe in asthma	11
Gluts detected	92	Benzodiazepines nonaddictive	11
Gaps detected	2	Other categories	30

Table 4. Scaled evaluation results (940 statements total).

Formula	KB	LLM	Status
Nonaddictive(valium)	$\mathfrak{t}$	$\langle \mathfrak{f}, \mathfrak{t} \rangle$	Glut
Cardiosafe(ibuprofen)	$\mathfrak{t}$	$\langle \mathfrak{f}, \mathfrak{t} \rangle$	Glut
Sedative(valium)	$\mathfrak{t}$	$\langle \mathfrak{t}, \mathfrak{f} \rangle$	Agreement
Analgesic(ibuprofen)	$\mathfrak{t}$	$\langle \mathfrak{t}, \mathfrak{f} \rangle$	Agreement
Cardiosafe(aspirin)	$\mathfrak{t}$	$\langle \mathfrak{f}, \mathfrak{f} \rangle$	Gap

Table 5. LLM-grounded evaluation of derived formulas in the medication safety example. The KB column shows the truth value from symbolic inference; the LLM column shows $\zeta_c(\varphi) = \langle u, v \rangle$.

significant errors: all 25 opioid medications yielded gluts for $\mathsf{Nonaddictive}(X)$ because the LLM correctly identifies opioids as highly addictive. The system also correctly identified that beta-blockers are contraindicated in asthma patients (11 gluts) and that fluoroquinolones pose risks for children (7 gluts). Despite 92 contradictions, the theory remained satisfiable. We encode rules about drug classifications, some correct and some deliberately erroneous:

$$\mathsf{S0001} : \mathfrak{t} : [\forall X\, \mathsf{Benzodiazepine}(X)]\mathsf{Nonaddictive}(X) \quad \text{(incorrect)}$$
$$\mathsf{S0002} : \mathfrak{t} : [\forall X\, \mathsf{Benzodiazepine}(X)]\mathsf{Sedative}(X) \quad \text{(correct)}$$
$$\mathsf{S0003} : \mathfrak{t} : [\forall X\, \mathsf{Nsaid}(X)]\mathsf{Analgesic}(X) \quad \text{(correct)}$$
$$\mathsf{S0004} : \mathfrak{t} : [\forall X\, \mathsf{Nsaid}(X)]\mathsf{Cardiosafe}(X) \quad \text{(incorrect)}$$

Rule S0001 is medically false: benzodiazepines carry significant dependence risks. Rule S0004 is also false: NSAIDs increase cardiovascular risk. We add ground assertions about specific medications (e.g., $\mathfrak{t} : \mathsf{Benzodiazepine}(\mathsf{valium})$, $\mathfrak{t} : \mathsf{Nsaid}(\mathsf{ibuprofen})$). Forward-chaining derives consequences such as $\mathsf{Nonaddictive}(\mathsf{valium})$ and $\mathsf{Cardiosafe}(\mathsf{ibuprofen})$. The LLM judge (GPT-4o) then evaluates each derived formula bilaterally.

Table 5 shows the results of the evaluation. The deliberately erroneous rules produced gluts: the LLM correctly identifies that Valium carries dependence risks and that ibuprofen increases cardiovascular risk. Where rules are accurate, the LLM confirmed the derived conclusions. The gap for $\mathsf{Cardiosafe}(\mathsf{aspirin})$ reflects genuine medical ambiguity—low-dose aspirin has cardioprotective effects in some populations

but bleeding risks in others. Despite multiple gluts, the system remained satisfiable. A classical reasoner would face explosion; our paraconsistent semantics localizes contradictions, allowing identification of exactly which conclusions are contested while maintaining valid inferences from uncontested premises.

Computational complexity The forward-chaining phase computes the deductive closure of the asserted knowledge base under the rule set, producing a finite ground theory. Let R denote the number of rules, C the number of constants, and k the maximum number of distinct variables in any rule body. Standard Datalog complexity analysis applies (Dantsin et al. 2001): each iteration considers $O(R \cdot C^k)$ candidate instantiations, and since each iteration must add at least one new ground atom or terminate, the fixpoint is reached in at most $O(|P| \cdot C^a)$ iterations, where $|P|$ is the number of predicates and a the maximum arity. For fixed schema parameters, forward-chaining is polynomial in the number of constants, consistent with Datalog's PTIME data complexity for ground query answering. Following the definition of the bilateral factuality evaluation function, each atom requires n queries to assess support and n queries to assess refutation where n is the size of the repeated sample, yielding $2n$ LLM calls per inference. For F inferred atoms with mean call latency L, total interpretation time is $O(nFL)$ sequentially, or $O(nFL/P)$ with parallelism degree P.

The tableau procedure checks satisfiability of the ground theory under ACrQ semantics via the signed tableau calculus of Ferguson (2021b), which provides a sound and complete proof procedure for AC with restricted quantifiers. Because forward-chaining eliminates quantifiers by grounding, the tableau operates over a quantifier-free theory of N ground atoms. The key complexity observation, noted by Ferguson (2021b), is that AC validity is polynomial-time reducible to classical propositional validity via a translation that preserves formula structure. This translation doubles the signature (introducing $\overline{R}$ for each predicate R to track refutation) but preserves formula size up to a constant factor. For ground theories, tableau satisfiability thus reduces to propositional satisfiability over $O(N)$ atoms—NP-complete in general (Cook 1971), though the conjunctive structure of theories produced by forward-chaining often admits more efficient decision procedures. The tableau invokes $\mathcal{I}_{LLM}$ whenever a ground atomic formula appears as a node in the tableau tree; letting A denote the number of distinct atoms encountered during expansion, semantic interpretation requires $O(nAL)$ time. Unlike classical description logics, where TBox reasoning ranges from EXPTIME-complete for $\mathcal{ALC}$ (Schild 1991) to 2NEXPTIME-complete for $\mathcal{SROIQ}$ (Kazakov 2008), ACrQ's grounding strategy yields complexity comparable to ABox-only reasoning. The cost of paraconsistency is not computational but rather the overhead of LLM-based semantic interpretation.

Table 6 reports timing from the medication safety use case, with $R = 28$ rules, $C = 200$ constants, $k = 2$ maximum variables per rule, $N = 940$ ground statements after closure, and $n = 3$ samples per verification/refutation direction. These results confirm that LLM latency dominates execution time: forward-chaining logic completes in under 10ms, while semantic interpretation accounts for over 99% of total runtime. This cost structure indicates that moderately-sized knowledge bases are tractable with the current proof-of-concept implementation—the medication safety knowledge base,

Phase	Time	LLM Calls	Notes
Forward-chaining (logic)	0.006s	—	Grounding
Forward-chaining (LLM)	463s	4,272	712×6 calls at $\sim$0.11s
Tableau (logic + LLM)	297s	varies	Expansion over 940 statements
Total	**760s**		

Table 6. Timing from the medication safety evaluation. LLM interpretation dominates computational cost.

with 200 entities and 940 ground statements, completes in under 13 minutes. More significantly, the dominant costs are amenable to optimization strategies that can plausibly scale throughput by one to two orders of magnitude. First, the $2n$ LLM calls per atom are trivially parallelizable across independent atoms; with parallelism degree $P = 50$ (readily achievable with standard API rate limits), the 463s forward-chaining LLM phase would reduce to under 10s. Second, mature DL reasoners have developed sophisticated optimization techniques—including absorption, lazy unfolding, dependency-directed backjumping, and semantic branching (Horrocks 2003)—that substantially reduce tableau expansion; adapting these techniques to ACrQ would decrease both logical overhead and the number of LLM-interpreted atoms A. Third, local LLM deployment eliminates network latency entirely: current quantized models achieve sub-10ms inference on consumer hardware, which would reduce per-call latency by an order of magnitude relative to API-based inference. Combined, these optimizations suggest that knowledge bases with 10^4–10^5 ground statements—comparable to medium-scale industrial ontologies—are within reach of production deployment.

7 Discussion and limitations

We have provided a theoretical framework for integrating an LLM with a paraconsistent reasoner, demonstrated the feasibility of providing LLM-grounded valuations of atomic formulas, and demonstrated a Belnap computer that implements the framework. In this section we discuss the broader implications of these results and the limitations of our approach with respect to verbalization, computational complexity, error propagation, and belief revision.

Verbalization The bilateral factuality evaluation function ζ depends critically on the verbalization function δ, which translates atomic formulas into natural language for LLM evaluation. This translation determines whether the LLM can provide a meaningful valuation or must return $\langle \mathfrak{f}, \mathfrak{f} \rangle$, indicating epistemic gap. An atomic formula $\varphi = R(c_1, \ldots, c_n)$ is evaluable when two conditions hold: (i) the arguments $c_1, \ldots, c_n$ refer to entities about which the LLM has parametric knowledge, and (ii) the predicate R expresses a property or relation the LLM can assess for those entities. When both conditions are met, the LLM can draw on its training data to provide verification or refutation evidence. When either condition fails, the LLM lacks grounds for evaluation and returns $\langle \mathfrak{f}, \mathfrak{f} \rangle$. Consider the contrast between

`Addictive(valium)` and `Addictive(compound7)`. The former is evaluable: "Valium" refers to a well-documented pharmaceutical (diazepam), and "addictive" is a property extensively discussed in medical literature. The LLM can assess this claim. The latter is not evaluable: "compound7" is an opaque identifier about which the LLM has no knowledge. The LLM cannot verify or refute claims about unknown entities, so it returns $\langle \mathfrak{f}, \mathfrak{f} \rangle$. Similarly, predicate interpretability matters. The formula `Sedative(valium)` verbalizes naturally to "Valium is a sedative," a claim the LLM can evaluate. But a domain-specific predicate like `Cyp3a4Inhibitor(valium)` may or may not be evaluable depending on whether the LLM's training data includes sufficient pharmacological detail about cytochrome P450 interactions. Highly technical or abbreviated predicates may fail to activate relevant LLM knowledge even when that knowledge is present. When the LLM cannot evaluate a formula, the returned value $\langle \mathfrak{f}, \mathfrak{f} \rangle$ represents a legitimate epistemic state: the LLM has neither evidence for nor evidence against the claim. This is not a system failure. The paraconsistent semantics of **AC** handles gaps naturally—the formula's valuation remains underdetermined by the LLM, and reasoning proceeds based on symbolic inference and explicit assertions. The LLM-grounded interpretation thus acts as a *partial oracle*, validating what it can and remaining silent otherwise. This contrasts with approaches that force binary verdicts. A unilateral evaluation function that must return true or false cannot represent ignorance; it must either guess or abstain entirely. The bilateral approach using $\mathcal{NINE}$-valued semantics provides a middle path: the LLM contributes what epistemic signal it can, and the logic handles the rest. As the medication safety example in Section 6 illustrates, formulas with transparent predicate names referring to well-documented properties (e.g., `Analgesic(ibuprofen)`) yield definite valuations, while formulas with less evocative names (e.g., `Monitored(warfarin)`) may return gaps—not necessarily because the LLM lacks the relevant knowledge, but because the verbalization fails to activate it. The implementation described in Section 3 uses direct syntactic mapping, providing the formula's surface syntax to the LLM (e.g., `Nonaddictive(valium)` verbalized as "Nonaddictive(valium)"). This minimal approach relies on the LLM's ability to parse logical notation and on implicit grounding through symbol names that happen to correspond to natural language terms. While sufficient for formulas with transparent symbol names, this approach has limitations:

1. *Opaque symbol names*: Knowledge bases often use abbreviated or systematic naming conventions (e.g., `rxn_4521`, `hasCADRisk`) that do not transparently convey meaning.
2. *Predicate-argument structure*: The relationship between predicate and arguments may require explicit verbalization. The formula `Treats(aspirin, headache)` is more naturally evaluated as "Aspirin treats headaches" than as "Treats(aspirin, headache)."

Alternative verbalization strategies can address these limitations. *Template-based transformation* uses predefined patterns for each predicate: `Treats(X, Y)` → "X treats Y"; `Contraindicated(X, Y)` → "X is contraindicated for patients with Y".

This provides more natural language input but requires manual template creation. *LLM-generated verbalization* uses a language model to produce natural language renderings of formulas, offering flexibility at the cost of introducing another potential source of error. Investigating alternative implementations of δ remains future work.

A complementary strategy for mitigating both epistemic gaps and verbalization mismatches is retrieval-augmented generation (RAG) (Gao et al. 2023). When an atomic formula refers to entities or predicates that fall outside the LLM's parametric knowledge (for example, specialized axioms drawn from a proprietary pharmacovigilance database or a domain-specific ontology), the verbalization function δ can be extended to retrieve relevant supporting documents from an external corpus and include them in the prompt seen by the verification and refutation functions P^+ and P^-. This effectively supplies the LLM with evidence it would not otherwise possess, converting what would have been valuations of $\langle \mathfrak{f}, \mathfrak{f} \rangle$ into definite valuations of $\langle \mathfrak{t}, \mathfrak{f} \rangle$ or $\langle \mathfrak{f}, \mathfrak{t} \rangle$ while preserving the bilateral nature of ζ. Because retrieval is implemented inside δ (and potentially, P^+ and P^-), the theoretical framework in Section 4 is unaffected, and the metalogical results of Theorems 4.1 and 4.2 apply unchanged, since they rely solely on the stability of the cached valuations (Lemma 4.1), not on how those valuations are produced. The integration of RAG into δ, and the development of retrieval strategies that preserve the bilateral approach by treating retrieval for verification and refutation separately, is a promising direction for future work, particularly in the context of biomedical domains, where authoritative knowledge is under constant revision and highly specialized.

These considerations suggest guidelines for knowledge base design when LLM-grounded evaluation is intended:

1. *Use transparent symbol names*: Choose constant and predicate names that correspond to natural language terms the LLM will recognize. Prefer `ibuprofen` over `drug_0042`; prefer `CausesLiverDamage` over `hepTox`.
2. *Favor well-documented entities*: Formulas involving entities with substantial presence in LLM training data (approved drugs, named diseases, public figures, established concepts) will be more reliably evaluable than those involving obscure or private entities.
3. *Expect gaps for domain-specific content*: Formulas involving proprietary identifiers, internal codes, or specialized terminology will likely yield $\langle \mathfrak{f}, \mathfrak{f} \rangle$. Plan for the reasoner to handle these through symbolic inference alone.
4. *Consider verbalization at design time*: When defining predicates, consider how they will verbalize. A predicate that produces awkward or ambiguous verbalizations will be harder for the LLM to evaluate.

A natural extension would enrich the verbalization function to incorporate contextual information from the theory itself. If the knowledge base contains assertions about an otherwise-unknown entity (e.g., `HasCondition(patient42, RenalImpairment)`), this context could be included in the prompt when evaluating formulas involving that entity. The LLM would then evaluate not in isolation but in light of what the theory asserts about the entity. We leave exploration of such context-aware verbalization strategies to future work.

Feature	ACrQ	Description Logics
Quantification	Restricted $[\forall X\ P(X)]Q(X)$	Guarded (role restrictions)
Truth values	3 (t, f, e) + 3 meta-signs	2 (classical)
Contradictions	Tolerated (gluts)	Cause explosion
Decidability	Semi-decidable	Decidable (by design)
Primary use	Paraconsistent reasoning	Ontology reasoning

Table 7. Feature comparison between ACrQ and description logics.

Scalability The moderately-sized knowledge base used in the medication safety example raises natural questions about how ACrQ-based reasoning compares to mature description logic (DL) reasoners in terms of scalability, and what optimizations might enable further scaling. Description logics are decidable fragments of first-order logic with well-characterized complexity. DL reasoners achieve decidability by carefully restricting expressivity, while ACrQ preserves full restricted quantification and tolerates contradictions. Mature DL reasoners (HermiT (Glimm et al. 2014), Pellet (Sirin et al. 2007), FaCT++ (Tsarkov and Horrocks 2006)) employ sophisticated optimizations developed over decades, many of which are directly transferable to our paraconsistent reasoner. These include *lazy unfolding*, which unfolds definitions only when needed for clash detection (yielding 10–100× speedup for sparse theories); *semantic branching*, which chooses branch order based on likelihood of closure (2–10× speedup); *dependency-directed backjumping*, which skips irrelevant choice points (exponential gains in pathological cases); *satisfiability caching for subformulas* (10–1000× for theories with repeated patterns); *formula indexing via hash-based lookup* to reduce $O(N^2)$ to $O(N)$ average case; *incremental reasoning through dependency tracking* for partial recomputation; and *parallel branch exploration* for near-linear speedup with available cores (Horrocks 2003). Beyond these transferable DL techniques, ACrQ's paraconsistent nature enables unique optimizations that, to our knowledge, have not been previously explored, and which we leave to future work: *glut-aware pruning*, whereby known gluts are recorded and excluded from closure checking; *bilateral predicate indexing*, which maintains dual index structures for P and P^* to enable $O(N)$ glut/gap detection; and *sign-stratified processing*, which handles definite signs before branching signs.

A full cost–benefit analysis for production deployment — covering LLM API costs versus local deployment, the break-even point at which caching amortizes interpretation cost, and the sensitivity of downstream decision quality to the sample size n used in repeated sampling — is beyond the scope of the present theoretical contribution but would provide actionable guidance for practitioners. We regard such an analysis, anchored in a concrete application domain such as clinical decision support or ontology curation, as a valuable companion to the present work.

Error Propagation A natural concern with LLM-grounded interpretations is error propagation: if the LLM judge returns an incorrect valuation for an atomic formula, does that error propagate through the reasoning system and corrupt downstream inferences?

The bilateral architecture mitigates this risk in two ways. First, errors in definite valuations ($\langle \mathsf{t}, \mathsf{f} \rangle$ or $\langle \mathsf{f}, \mathsf{t} \rangle$) behave identically to errors in classical interpretations—they produce unsound conclusions, but no worse than reasoning over any other knowledge base containing incorrect facts. Second, and more importantly, the indefinite valuations that bilateral evaluation surfaces—gaps ($\langle \mathsf{f}, \mathsf{f} \rangle$) and gluts ($\langle \mathsf{t}, \mathsf{t} \rangle$)—act as natural error containment. When the LLM is uncertain or conflicted, this uncertainty propagates conservatively through the paraconsistent semantics rather than forcing a potentially incorrect commitment. The reasoner can identify which conclusions depend on definite versus indefinite premises, enabling downstream systems to assign different confidence levels accordingly. For instance, in the medication safety example of Section 6, the gluts detected for `Nonaddictive(valium)` and `Cardiosafe(ibuprofen)` flag exactly those conclusions where the knowledge base's rules conflict with LLM knowledge, enabling a downstream system to treat these conclusions with appropriate caution. Full error analysis, including empirical measurement of how verbalization failures and LLM miscalibration affect end-to-end reasoning accuracy, remains future work.

Belief Revision The current implementation supports cache invalidation with replay: when external evidence indicates a cached valuation may be stale, the entry is cleared and the formula is re-evaluated on subsequent queries. This approach is orthogonal to the paraconsistent semantics: where paraconsistency tolerates contradictions within a knowledge state, belief revision addresses changes between knowledge states over time. Recent work on paraconsistent belief revision (Testa et al. 2017; Coniglio et al. 2024) has adapted AGM-style postulates (Alchourrón et al. 1985) to logics of formal inconsistency, but retains the AGM motivation of restoring consistency when new information contradicts prior beliefs. Our framework takes a different stance: gluts ($\langle \mathsf{t}, \mathsf{t} \rangle$) are valid epistemic states reflecting genuine LLM inconsistency, not errors requiring repair. Whether and how to integrate revision mechanisms that respect this tolerance, or to explore alternatives such as truth maintenance systems (Forbus and de Kleer 1993), remains future work.

8 Conclusion and future work

We described a novel approach to logical reasoning using LLMs with several key contributions. We defined a bilateral approach to factuality evaluation that identifies gaps and contradictions in LLM parametric knowledge. We introduced the concept of LLM-grounded interpretations that integrate an LLM directly into the formal semantics of the underlying logic while preserving its soundness and completeness. We provided empirical evidence that bilateral factuality evaluation outperforms unilateral approaches, and demonstrated through concrete examples how the system handles contradictory information without logical explosion. And finally, we implemented a Belnap computer that demonstrates the practical feasibility of this approach, including an interactive dialogue management interface that bridges natural language and formal reasoning. Our evaluation with an example medication safety knowledge base of 940 statements (228 asserted, 712 inferred) demonstrates that LLM-grounded reasoning

maintains its theoretical properties at realistic knowledge base sizes. The system successfully identified 92 gluts where deductively inferred conclusions conflicted with LLM medical knowledge—including medically significant errors such as claiming opioids are non-addictive, beta-blockers are safe in asthma, and fluoroquinolones are safe for children. Despite these contradictions, the paraconsistent semantics preserved satisfiability and allowed identification of exactly which rules conflicted with empirical medical knowledge. Computational analysis reveals that the primary bottleneck is LLM API latency rather than logical reasoning, suggesting that standard optimization techniques from description logic reasoners—including batching, parallelization, and caching—could enable scaling to knowledge bases of over one hundred thousand axioms.

In addition to the future work described in Section 7, we plan to conduct a user evaluation of the usability of the Theory Manager for ontology and knowledge graph construction, and to extend our approach based on work in the area of generalized truth values (Shramko and Wansing 2005; Hornischer 2025) to provide a multi-valued semantics for modeling factuality in LLMs, with the goal of improving the theoretical framework and metrics for factuality evaluation. A further valuable direction for future work would be an empirical comparison of paraconsistent and classical reasoning over LLM-grounded knowledge. Such a study could quantify the practical value of tolerating gluts rather than treating them as inconsistencies requiring repair, by measuring (i) how often classical reasoners over the same LLM-grounded atoms are forced into trivialization or require ad hoc repair, and (ii) the downstream utility of the glut and gap diagnostics surfaced by the paraconsistent reasoner for tasks such as ontology debugging, knowledge-base curation, and safety-critical decision support. Finally, a complementary application-focused study reporting a detailed cost–benefit analysis in a production setting (including monetary cost per inference, latency budgets, caching behavior, and the operational value of the glut and gap diagnostics) would translate the engineering directions sketched in Section 7 into concrete guidance for real-world deployments.

Acknowledgements

This work was partially supported by the EU's Horizon Europe research and innovation programme within the ENEXA project (grant Agreement no. 101070305). Particular thanks are due to Frank van Harmelen for his valuable feedback based on a close reading of an earlier version of this article, and to Fabian Hoppe, Levin Hornischer, Jan-Christoph Kalo, and Lise Stork for discussions and perceptive observations that enriched our research.

References

Alchourrón CE, Gärdenfors P and Makinson D (1985) On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic* 50(2): 510–530. DOI: 10.2307/2274239.

Allen BP, Chhikara P, Ferguson TM, Ilievski F and Groth P (2025) Sound and complete neurosymbolic reasoning with llm-grounded interpretations. In: H Gilpin L, Giunchiglia E, Hitzler P and van Krieken E (eds.) *Proceedings of The 19th International Conference on Neurosymbolic Learning and Reasoning*, *Proceedings of Machine Learning Research*,

volume 284. PMLR, pp. 392–419. URL https://proceedings.mlr.press/v284/allen25a.html.

Anthropic (2025) Claude code overview. URL https://docs.claude.com/en/docs/claude-code/overview. URL accessed: 2025-10-14.

Arieli O and Avron A (1998) The value of the four values. *Artificial Intelligence* 102(1): 97–141.

Baader F and Suntisrivaraporn B (2008) Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In: *Proceedings of the 3rd International Conference on Knowledge Representation in Medicine (KR-MED'08)*, *CEUR Workshop Proceedings*, volume 410. pp. 1–7.

Bang Y, Ji Z, Schelten A, Hartshorn A, Fowler T, Zhang C, Cancedda N and Fung P (2025) HalluLens: LLM Hallucination Benchmark. *arXiv preprint arXiv:2504.17550* .

Belnap N (1977a) How a computer should think. In: Ryle G (ed.) *Contemporary aspects of philosophy*. Oriel Press, pp. 30–55.

Belnap N (1977b) A useful four-valued logic. In: *Modern uses of multiple-valued logic*. Springer, pp. 5–37.

Brown B, Juravsky J, Ehrlich R, Clark R, Le QV, Ré C and Mirhoseini A (2024) Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787* .

Callewaert B, Vandevelde S and Vennekens J (2025) VERUS-LM: a Versatile Framework for Combining LLMs with Symbolic Reasoning. *arXiv preprint arXiv:2501.14540* .

Cheng F, Li H, Liu F, van Rooij R, Zhang K and Lin Z (2025) Empowering LLMs with Logical Reasoning: A Comprehensive Survey. *arXiv preprint arXiv:2502.15652* .

Coniglio ME, Figallo M and Testa RR (2024) A logic for paraconsistent belief revision based on epistemic entrenchment. *arXiv preprint arXiv:2412.06117* .

Cook SA (1971) The complexity of theorem-proving procedures. In: *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC)*. pp. 151–158. DOI:10.1145/800157.805047.

Correia F (2010) Grounding and truth-functions. *Logique et Analyse* 53(211): 251–279.

da Costa NC and Subrahmanian V (1989) Paraconsistent logics as a formalism for reasoning about inconsistent knowledge bases. *Artificial Intelligence in Medicine* 1(4): 167–174.

Dantsin E, Eiter T, Gottlob G and Voronkov A (2001) Complexity and expressive power of logic programming. *ACM Computing Surveys* 33(3): 374–425. DOI:10.1145/502807.502810.

de Rooij S, Beek W, Bloem P, van Harmelen F and Schlobach S (2016) Are names meaningful? Quantifying social meaning on the Semantic Web. In: *International Semantic Web Conference*. Springer, pp. 184–199.

Dhuliawala S, Komeili M, Xu J, Raileanu R, Li X, Celikyilmaz A and Weston J (2023) Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495* .

El-Yaniv R and Wiener Y (2010) On the Foundations of Noise-free Selective Classification. *Journal of Machine Learning Research* 11(5).

Feng J, Xu R, Hao J, Sharma H, Shen Y, Zhao D and Chen W (2024) Language models can be deductive solvers. In: *Findings of the Association for Computational Linguistics: NAACL*

2024. pp. 4026–4042.

Ferguson TM (2017a) A computational interpretation of conceptivism. In: *Meaning and Proscription in Formal Logic: Variations on the Propositional Logic of William T. Parry*. Springer, pp. 73–105.

Ferguson TM (2017b) Faulty Belnap Computers and Subsystems of **FDE**. In: *Meaning and Proscription in Formal Logic: Variations on the Propositional Logic of William T. Parry*. Springer, pp. 107–131.

Ferguson TM (2021a) Modeling Intentional States with Subsystems of ALC. In: *Proceedings of the 34th International Workshop on Description Logics (DL 2021)*, *CEUR Workshop Proceedings*, volume 2954. pp. 1–12.

Ferguson TM (2021b) Tableaux and restricted quantification for systems related to weak Kleene logic. In: *International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*. Springer, pp. 3–19.

Fine K (2016) Angellic content. *Journal of Philosophical Logic* 45: 199–226.

Forbus KD and de Kleer J (1993) *Building Problem Solvers*. Cambridge, MA: MIT Press. ISBN 978-0262061575.

Gao Y, Xiong Y, Gao X, Jia K, Pan J, Bi Y, Dai Y, Sun J and Wang H (2023) Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* .

Geifman Y and El-Yaniv R (2017) Selective classification for deep neural networks. *Advances in neural information processing systems* 30.

Gligorov R, Ten Kate W, Aleksovski Z and Van Harmelen F (2007) Using google distance to weight approximate ontology matches. In: *Proceedings of the 16th International Conference on World Wide Web*. ACM, pp. 767–776.

Glimm B, Horrocks I, Motik B, Stoilos G and Wang Z (2014) HermiT: An OWL 2 reasoner. *Journal of Automated Reasoning* 53(3): 245–269. DOI:10.1007/s10817-014-9305-1.

Goré R and Nguyen LA (2007) EXPTIME tableaux with global caching for description logics with transitive roles, inverse roles and role hierarchies. In: *International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*. Springer, pp. 133–148.

Hoppe F, Ilievski F and Kalo JC (2025) Investigating the robustness of deductive reasoning with large language models. *arXiv preprint arXiv:2502.04352* .

Hornischer L (2025) Iterating Both and Neither: With Applications to the Paradoxes. *Notre Dame Journal of Formal Logic* 1(1): 1–43.

Horrocks I (2003) Implementation and optimization techniques. In: Baader F, Calvanese D, McGuinness DL, Nardi D and Patel-Schneider PF (eds.) *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, pp. 306–346.

Huang Z and van Harmelen F (2008) Using semantic distances for reasoning with inconsistent ontologies. In: *International Semantic Web Conference*. Springer, pp. 178–194.

Jacovi A, Wang A, Alberti C, Tao C, Lipovetz J, Olszewska K, Haas L, Liu M, Keating N, Bloniarz A et al. (2025) The FACTS Grounding Leaderboard: Benchmarking LLMs' Ability to Ground Responses to Long-Form Input. *arXiv preprint arXiv:2501.03200* .

Jiao F, Teng Z, Ding B, Liu Z, Chen NF and Joty S (2023) Exploring self-supervised logic-enhanced training for large language models. *arXiv preprint arXiv:2305.13718* .

Kadavath S, Conerly T, Askell A, Henighan T, Drain D, Perez E, Schiefer N, Hatfield-Dodds Z, DasSarma N, Tran-Johnson E et al. (2022) Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221* .

Kalyanpur A (2006) *Debugging and repair of OWL ontologies*. University of Maryland, College Park.

Kamide N (2010) Paraconsistent description logics revisited. In: *23rd International Workshop on Description Logics DL2010*. p. 197.

Kazakov Y (2008) RIQ and SROIQ are harder than SHOIQ. In: *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR)*. pp. 274–284.

Kleene SC (1952) *Introduction to metamathematics*. Wolters-Noordhoff Publishing and North-Holland Publishing Company.

Kojima T, Gu SS, Reid M, Matsuo Y and Iwasawa Y (2022) Large language models are zero-shot reasoners. *Advances in neural information processing systems* 35: 22199–22213.

Li H, Dong Q, Chen J, Su H, Zhou Y, Ai Q, Ye Z and Liu Y (2024) LLMs-as-judges: a comprehensive survey on LLM-based evaluation methods. *arXiv preprint arXiv:2412.05579* .

Liu J, Fan Y, Jiang Z, Ding H, Hu Y, Zhang C, Shi Y, Weng S, Chen A, Chen S et al. (2025) Synlogic: Synthesizing verifiable reasoning data at scale for learning logical reasoning and beyond. *arXiv preprint arXiv:2505.19641* .

Ma Y, Hitzler P and Lin Z (2007) Algorithms for paraconsistent reasoning with OWL. In: *European Semantic Web Conference*. Springer, pp. 399–413.

Maier F, Ma Y and Hitzler P (2013) Paraconsistent OWL and related logics. *Semantic Web* 4(4): 395–427.

Manhaeve R, Dumancic S, Kimmig A, Demeester T and De Raedt L (2018) Deepproblog: Neural probabilistic logic programming. *Advances in neural information processing systems* 31.

Morishita T, Morio G, Yamaguchi A and Sogawa Y (2024) Enhancing reasoning capabilities of LLMs via principled synthetic logic corpus. *Advances in Neural Information Processing Systems* 37: 73572–73604.

Nguyen LA (2009) An efficient tableau prover using global caching for the description logic ALC. *Fundamenta Informaticae* 93(1-3): 273–288.

Olausson TX, Gu A, Lipkin B, Zhang CE, Solar-Lezama A, Tenenbaum JB and Levy R (2023) LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. *arXiv preprint arXiv:2310.15164* .

Pan L, Albalak A, Wang X and Wang WY (2023) Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295* .

Patel-Schneider PF (1989) A four-valued semantics for terminological logics. *Artificial Intelligence* 38(3): 319–351.

Petroni F, Rocktäschel T, Lewis P, Bakhtin A, Wu Y, Miller AH and Riedel S (2019) Language models as knowledge bases? *arXiv preprint arXiv:1909.01066* .

Politis DN and Romano JP (1994) Large sample confidence regions based on subsamples under minimal assumptions. *The Annals of Statistics* : 2031–2050.

Priest G, Tanaka K and Weber Z (2025) Paraconsistent Logic. https://plato.stanford.edu/archives/spr2025/entries/logic-paraconsistent/.

Rein D, Hou BL, Stickland AC, Petty J, Pang RY, Dirani J, Michael J and Bowman SR (2023) GPQA: A Graduate-Level Google-Proof Q&A Benchmark. *arXiv preprint arXiv:2311.12022* .

Rumfitt I (2000) 'Yes' and 'No'. *Mind* 109(436): 781–823.

Schild K (1991) A correspondence theory for terminological logics: Preliminary report. In: *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI)*. pp. 466–471.

Schlobach S, Huang Z, Cornet R and van Harmelen F (2007) Debugging incoherent terminologies. *Journal of automated reasoning* 39(3): 317–349.

Schon C (2025) Context-Aware Clause Selection Using Symbol Name Meanings in Theorem Proving. In: *International Symposium on Frontiers of Combining Systems*. Springer, pp. 306–323.

Shramko Y and Wansing H (2005) Some useful 16-valued logics: How a computer network should think. *Journal of Philosophical Logic* 34: 121–153.

Shramko Y and Wansing H (2011) *Truth and falsehood: An inquiry into generalized logical values*, volume 36. Springer Science & Business Media.

Shramko Y and Wansing H (2025) Truth Values. https://plato.stanford.edu/archives/spr2025/entries/truth-values/.

Sirin E, Parsia B, Grau BC, Kalyanpur A and Katz Y (2007) Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics* 5(2): 51–53.

Szmuc DE (2019) An epistemic interpretation of paraconsistent weak Kleene logic. *Logic and Logical Philosophy* 28(2): 277–330.

Testa RR, Coniglio ME and Ribeiro MM (2017) AGM-like paraconsistent belief change. *Logic Journal of the IGPL* 25(4): 632–672. DOI:10.1093/jigpal/jzx010.

Tsarkov D and Horrocks I (2006) FaCT++ description logic reasoner: System description. In: *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR 2006)*, *Lecture Notes in Computer Science*, volume 4130. Springer, pp. 292–297. DOI: 10.1007/11814771_26.

Wang C, Liu X, Yue Y, Tang X, Zhang T, Jiayang C, Yao Y, Gao W, Hu X, Qi Z et al. (2023) Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521* .

Wei J, Karina N, Chung HW, Jiao YJ, Papay S, Glaese A, Schulman J and Fedus W (2024) Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368* .

Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, Le QV, Zhou D et al. (2022) Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems* 35: 24824–24837.

Yao S, Yu D, Zhao J, Shafran I, Griffiths TL, Cao Y and Narasimhan K (2023) Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601* .

Zheng L, Chiang WL, Sheng Y, Zhuang S, Wu Z, Zhuang Y, Lin Z, Li Z, Li D, Xing E et al. (2023) Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. *Advances in Neural*

Information Processing Systems 36: 46595–46623.

Zhu L, Wang X and Wang X (2023) JudgeLM: Fine-tuned Large Language Models are Scalable Judges. *arXiv preprint arXiv:2310.17631* .

A Angell's logic of analytic containment AC

Below we summarize the definitions of the object language, truth functions, and interpretations for the version of **AC** presented in greater detail in Ferguson (2021b).

A.1 Object language

Definition A.1. *Let $\mathcal{L}$ be a first-order language built from a countable set $\mathcal{C}$ of constants, a countable set of variables $\mathcal{V}$, a countable set $\mathcal{R}$ of relation symbols, the Boolean connectives $\neg$, $\wedge$, and $\vee$, restricted universal and existential quantifiers $\forall$ and $\exists$, and round parentheses (as used in complex formulas) and square brackets (as used in quantified formulas) as auxiliary symbols. If $R \in \mathcal{R}$ and $c_1, \ldots, c_n \in \mathcal{C}$, then $R(c_1, \ldots, c_n)$ is an atomic formula. Let $\mathcal{L}_{AT}$ be the set of atomic formulas. The formulas of $\mathcal{L}$ are the elements of $\mathcal{L}_{AT}$, together with the following, where $\varphi, \psi \in \mathcal{L}$ and $x \in \mathcal{V}$:*

$$\neg\varphi \mid (\varphi \wedge \psi) \mid (\varphi \vee \psi) \mid [\forall x \varphi(x)]\psi(x) \mid [\exists x \varphi(x)]\psi(x)$$

A.2 Truth functions

Definition A.2. *The* weak Kleene truth tables *over the set of truth values $\mathcal{V}_3 = \{\mathfrak{t}, \mathfrak{e}, \mathfrak{f}\}$ are:*

$\neg$	
$\mathfrak{t}$	$\mathfrak{f}$
$\mathfrak{e}$	$\mathfrak{e}$
$\mathfrak{f}$	$\mathfrak{t}$

$\wedge$	$\mathfrak{t}$	$\mathfrak{e}$	$\mathfrak{f}$
$\mathfrak{t}$	$\mathfrak{t}$	$\mathfrak{e}$	$\mathfrak{f}$
$\mathfrak{e}$	$\mathfrak{e}$	$\mathfrak{e}$	$\mathfrak{e}$
$\mathfrak{f}$	$\mathfrak{f}$	$\mathfrak{e}$	$\mathfrak{f}$

$\vee$	$\mathfrak{t}$	$\mathfrak{e}$	$\mathfrak{f}$
$\mathfrak{t}$	$\mathfrak{t}$	$\mathfrak{e}$	$\mathfrak{t}$
$\mathfrak{e}$	$\mathfrak{e}$	$\mathfrak{e}$	$\mathfrak{e}$
$\mathfrak{f}$	$\mathfrak{t}$	$\mathfrak{e}$	$\mathfrak{f}$

The weak Kleene truth tables for conjunction and disjunction induce the truth functions $\dot{\wedge}$ and $\dot{\vee}$, respectively.

Definition A.3. *The* restricted Kleene quantifier functions $\dot{\forall}$ *and* $\dot{\exists}$ *are mappings from sets of truth values to truth values such that:*

$$\dot{\exists}(X) = \begin{cases} \mathfrak{t} & \text{if } \langle \mathfrak{t}, \mathfrak{t} \rangle \in X \\ \mathfrak{e} & \text{if for all } \langle u, v \rangle, \text{ either } u = \mathfrak{e} \text{ or } v = \mathfrak{e} \\ \mathfrak{f} & \text{if } \langle \mathfrak{t}, \mathfrak{t} \rangle \notin X \text{ and for some } \langle u, v \rangle \in X,\ u \neq \mathfrak{e} \text{ and } v \neq \mathfrak{e} \end{cases}$$

$$\dot{\forall}(X) = \begin{cases} \mathfrak{t} & \text{if } \langle \mathfrak{t}, \mathfrak{f} \rangle, \langle \mathfrak{t}, \mathfrak{e} \rangle \notin X \text{ and for some } \langle u, v \rangle \in X, u \neq \mathfrak{e} \text{ and } v \neq \mathfrak{e} \\ \mathfrak{e} & \text{if for all } \langle u, v \rangle \in X, \text{ either } u = \mathfrak{e} \text{ or } v = \mathfrak{e} \\ \mathfrak{f} & \text{if } \{\langle \mathfrak{t}, \mathfrak{t} \rangle, \langle \mathfrak{t}, \mathfrak{e} \rangle\} \cap X \neq \emptyset \text{ and for some } \langle u, v \rangle \in X, \text{ either } u = \mathfrak{e} \text{ or } v = \mathfrak{e} \end{cases}$$

A.3 Interpretations

Definition A.4. *An* $\mathbf{AC}$ interpretation $\mathcal{I}$ *is a pair* $\langle \mathbf{C}^{\mathcal{I}}, \mathbf{R}^{\mathcal{I}} \rangle$ *where* $\mathbf{C}^{\mathcal{I}}$ *is a domain of individuals and* $\mathbf{R}^{\mathcal{I}}$ *is a set of functions where* $\mathcal{I}$ *assigns:*

- *every constant* $c \in \mathbf{C}$ *an individual* $c^{\mathcal{I}} \in \mathbf{C}^{\mathcal{I}}$
- *every n-ary predicate* R *a function* $R^{\mathcal{I}} : (\mathbf{C}^{\mathcal{I}})^n \to \mathcal{V}_3 \times \mathcal{V}_3$

Definition A.5. *An* $\mathbf{AC}$ *interpretation* $\mathcal{I}$ *induces a map* $\mathcal{I} : \mathcal{L} \to \mathcal{V}_3 \times \mathcal{V}_3$ *as follows, where* $\mathcal{I}_0$ *and* $\mathcal{I}_1$ *project the first and second coordinates respectively:*

- *For atomic formulas* $R(c_1, \ldots, c_n) \in \mathcal{L}_{AT}$, $\mathcal{I}(\varphi) = R^{\mathcal{I}}(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}})$
- $\mathcal{I}(\neg\varphi) = \langle \mathcal{I}_1(\varphi), \mathcal{I}_0(\varphi) \rangle$
- $\mathcal{I}(\varphi \wedge \psi) = \langle \mathcal{I}_0(\varphi) \dot{\wedge} \mathcal{I}_0(\psi), \mathcal{I}_1(\varphi) \dot{\vee} \mathcal{I}_1(\psi) \rangle$
- $\mathcal{I}(\varphi \vee \psi) = \langle \mathcal{I}_0(\varphi) \dot{\vee} \mathcal{I}_0(\psi), \mathcal{I}_1(\varphi) \dot{\wedge} \mathcal{I}_1(\psi) \rangle$
- $\mathcal{I}([\forall x \varphi(x)]\psi(x)) = \langle \dot{\forall}(\{\mathcal{I}_0(\varphi(c)), \mathcal{I}_0(\psi(c)) \mid c \in \mathcal{C}\}), \dot{\exists}(\{\mathcal{I}_0(\varphi(c)), \mathcal{I}_1(\psi(c)) \mid c \in \mathcal{C}\}) \rangle$
- $\mathcal{I}([\exists x \varphi(x)]\psi(x)) = \langle \dot{\exists}(\{\mathcal{I}_0(\varphi(c)), \mathcal{I}_0(\psi(c)) \mid c \in \mathcal{C}\}), \dot{\forall}(\{\mathcal{I}_0(\varphi(c)), \mathcal{I}_1(\psi(c)) \mid c \in \mathcal{C}\}) \rangle$

Definition A.6. *Given an* $\mathbf{AC}$ *interpretation* $\mathcal{I}$*, validity with respect to* $\mathcal{I}$ *is defined as truth preservation, i.e.*

$$\Gamma \models_{\mathcal{I}} \varphi \text{ if for all instances of } \mathcal{I} \text{ such that } \forall \psi \in \Gamma\, \mathcal{I}_0(\psi) = \mathfrak{t}, \mathcal{I}_0(\varphi) = \mathfrak{t}.$$

B Prompts

B.1 Direct verification template

```
Determine whether the following answer to the given question is correct.
Conclude with a single line containing ONLY one of these two phrases:
VERIFIED
CANNOT VERIFY

Question: {question}
Proposed answer: {answer}
```

B.2 Direct refutation template

```
Determine whether the following answer to the given question can be refuted.
Conclude with a single line containing ONLY one of these two phrases:
REFUTED
CANNOT REFUTE

Question: {question}
Proposed answer: {answer}
```

B.3 Zero-shot verification template

```
I'll provide you with a question and its proposed answer.
Your task is to verify whether this answer is correct by following these steps:

1. Analyze the exact meaning of both the question and answer,
identifying any key terms that need clarification.
2. Establish specific conditions that would make this answer true for this question.
```

```
3. Provide direct evidence supporting the answer, including specific facts, examples,
or authoritative references that confirm its accuracy.
4. Test if the answer remains valid across all contexts where the question applies,
noting any limitations or exceptions.
5. Check for consistency with established knowledge in the relevant domain.

Based on your analysis, determine whether the answer is verified and explain
your reasoning with specific supporting evidence.
Your goal is not to find fault but to determine if positive
evidence exists to confirm the answer.

After your complete analysis, conclude with a single line containing
ONLY one of these two phrases:
VERIFIED
CANNOT VERIFY

Question: {question}
Proposed answer: {answer}
```

B.4 Zero-shot refutation template

```
I'll provide you with a question and its proposed answer.
Your task is to determine if this answer can be refuted by following these steps:

1. Analyze the exact meaning of both the question and the proposed answer.
2. Identify what specific conditions would need to be true for this answer to be false
(not merely the absence of evidence).
3. Search for direct counterexamples or contradicting evidence that
actively demonstrates why the answer is incorrect.
4. Construct specific scenarios where the answer fails to hold true,
even if the question's premises are accepted.
5. Identify any logical inconsistencies, factual errors, or category mistakes
within the answer.

Focus on building an affirmative case for why the answer is incorrect,
rather than simply noting a lack of supporting evidence.
Provide specific counterevidence and explain precisely
how it contradicts the proposed answer.

After your complete analysis, conclude with a single line containing
ONLY one of these two phrases:
REFUTED
CANNOT REFUTE

Question: {question}
Proposed answer: {answer}
```

B.5 Few-shot verification template

```
I'll provide you with a question and its proposed answer.
Your task is to verify whether this answer is correct by following these steps:

1. Analyze the exact meaning of both the question and answer, identifying any key terms that need clarification.
2. Establish specific conditions that would make this answer true for this question.
3. Provide direct evidence supporting the answer, including specific facts, examples, or authoritative references
      that confirm
its accuracy.
4. Test if the answer remains valid across all contexts where the question applies, noting any limitations or
      exceptions.
5. Check for consistency with established knowledge in the relevant domain.

Based on your analysis, determine whether the answer is verified and explain your reasoning with specific supporting
      evidence.
Your goal is not to find fault but to determine if positive evidence exists to confirm the answer.

Here are some examples of how to approach verification:

Example 1:
Question: Who was the first man to walk on the Moon?
Proposed answer: Neil Armstrong
Let's verify this answer systematically:
Key Terms Analysis:
"First man to walk on the Moon" requires precise historical verification
Specific date and mission details are crucial
Conditions for Verification:
Must be the first human to physically step onto lunar surface
Must have documented evidence of this historic moment
Must be part of a specific NASA mission
```

```
Direct Evidence:
Neil Armstrong stepped onto the Moon's surface on July 20, 1969
During Apollo 11 mission
Famous quote: "That's one small step for man, one giant leap for mankind"
Extensive NASA documentation confirms this
Broadcast live on global television
Verified by multiple independent sources including NASA, scientific records, and historical archives
Context and Limitations:
Specifically refers to first human, not first spacecraft
Specifically lunar surface, not orbital passage
Confirmed during Cold War space race
Part of NASA's Apollo program
Consistency Check:
Matches scientific and historical consensus
No credible alternative claims exist
Extensively documented in multiple international records
This answer meets all verification criteria with robust, multi-source evidence.
VERIFIED

Example 2:
Question: Chocolate is made from material
Proposed answer: Sugar
Let's analyze this systematically:
Key Term Analysis:
"Chocolate" refers to a food product derived from cacao/cocoa beans
"Material" suggests the fundamental ingredient or component
"Sugar" is a sweetener, not the primary material of chocolate
Conditions for Accuracy:
Sugar is an ingredient in chocolate
But sugar is NOT the primary material of chocolate
Chocolate's primary material is cocoa/cacao solids
Direct Evidence:
Chocolate is primarily made from cocoa beans processed into cocoa solids
Cocoa solids come from cacao tree seeds/beans
Sugar is added as a sweetener, not the base material
Chocolate composition typically includes:

Cocoa solids (primary material)
Cocoa butter
Sugar (secondary ingredient)
Milk (in milk chocolate)

Context Testing:
In all chocolate production processes, cocoa is the fundamental material
Sugar is always a supplementary ingredient, not the base material
Domain Consistency:
Culinary and food science consistently define cocoa/cacao as chocolate's primary material
Based on comprehensive analysis, the proposed answer is incorrect.
CANNOT VERIFY

After your complete analysis, conclude with a single line containing ONLY one of these two phrases:
VERIFIED
CANNOT VERIFY

Question: {question}
Proposed answer: {answer}
```

B.6 Few-shot refutation template

```
I'll provide you with a question and its proposed answer.
Your task is to determine if this answer can be refuted by following these steps:

1. Analyze the exact meaning of both the question and the proposed answer.
2. Identify what specific conditions would need to be true for this answer to be false (not merely the absence of
       evidence).
3. Search for direct counterexamples or contradicting evidence that actively demonstrates why the answer is incorrect
       .
4. Construct specific scenarios where the answer fails to hold true, even if the question's premises are accepted.
5. Identify any logical inconsistencies, factual errors, or category mistakes within the answer.

Focus on building an affirmative case for why the answer is incorrect, rather than simply noting a lack of supporting
       evidence.
Provide specific counterevidence and explain precisely how it contradicts the proposed answer.

Here are some examples of how to approach refutation:

Example 1:
Question: Are penguins birds?
Proposed answer: No
Let's analyze this systematically:
Meaning Analysis:
Question asks about the taxonomic classification of penguins
```

```
Proposed answer claims penguins are NOT birds
Conditions for Falsity:
Penguins must meet standard biological criteria for birds
Must share key avian characteristics
Counterevidence:
Penguins have ALL classic bird characteristics:

Feathered body
Lay eggs
Warm-blooded
Have beaks
Descended from dinosaur lineage
Classified in scientific taxonomy under Class Aves
Specifically, penguins belong to the order Sphenisciformes, which is a recognized bird order
Biological and genetic evidence conclusively places penguins within bird classification

Specific Scenarios Contradicting Answer:
Penguins have wing-like flippers adapted for swimming
They have respiratory and skeletal structures identical to other bird species
Genetic sequencing confirms their bird lineage
Logical Inconsistencies:
Rejecting penguins as birds would require rejecting fundamental biological classification systems
No scientific basis exists for excluding penguins from bird category
REFUTED

Example 2:
Question: Who was the first man to walk on the Moon?
Proposed answer: Neil Armstrong
Let's analyze this systematically:
Meaning Analysis:
Question seeks the definitive first human male to set foot on lunar surface
Proposed answer: Neil Armstrong (Apollo 11 mission, July 20, 1969)
Potential Conditions for Falsity:
Documented evidence of another person walking on Moon before Armstrong
Proof that Armstrong was not actually the first
Historical record showing a different individual preceded him
Counterevidence Search:
No credible historical evidence exists contradicting Armstrong's first Moon walk
NASA records and global documentation consistently confirm Armstrong as first
Extensive photographic and video evidence supports this claim
Scenario Testing:
No alternative scenarios emerge that could plausibly replace Armstrong's achievement
Extensive verification by multiple nations and independent researchers confirms his primacy
Logical Consistency Check:
Armstrong's Moon walk is extensively documented
Multiple witnesses and technological records corroborate the event
No logical inconsistencies detected in the claim
The proposed answer is completely accurate and supported by overwhelming historical evidence.
CANNOT REFUTE

After your complete analysis, conclude with a single line containing ONLY one of these two phrases:
REFUTED
CANNOT REFUTE

Question: {question}
Proposed answer: {answer}
```

B.7 Prompt template for generating negative examples for SimpleQA-derived benchmark

```
You are an expert synthetic data generator. Your task is to generate three plausible but
incorrect answers to a given question that will serve as challenging distractors.

Guidelines for generating high-quality wrong answers:
1. Each answer must be factually incorrect but highly plausible within the context
   - Draw from the same domain/topic as the correct answer
   - Use answers that could reasonably be mistaken for the truth
   - Avoid obviously wrong or nonsensical options

2. Strictly match the answer type and format
   - For dates: Use the same date format and plausible timeframe
   - For people: Match profession, era, and relevance
   - For numbers: Stay within reasonable orders of magnitude
   - For places: Use locations of similar type/scale

3. Ensure clear differentiation
   - Make each wrong answer distinct from the correct answer
   - Avoid overlap between wrong answers
   - Space out numerical answers appropriately

4. Maintain consistent specificity
```

```
   - Match the level of detail in the correct answer
   - If the answer is detailed, provide equally detailed wrong options
   - If the answer is brief, keep wrong answers similarly concise

5. Optimize for realism and difficulty
   - Wrong answers should feel natural, not contrived
   - Target common misconceptions when possible
   - Make the options challenging enough to test real knowledge
   - Avoid answers that are too obviously incorrect

Example 1:
Question: Who was the first president of the United States?
Answer: George Washington
Wrong Answers:
- John Adams
- Thomas Jefferson
- Benjamin Franklin
Reason: All are founding fathers but not the first president

Example 2:
Question: What is the largest planet in our solar system?
Answer: Jupiter
Wrong Answers:
- Saturn
- Neptune
- Uranus
Reason: All are gas giant planets, but smaller than Jupiter

Example 3:
Question: Who wrote Romeo and Juliet?
Answer: William Shakespeare
Wrong Answers:
- Christopher Marlowe
- Ben Jonson
- John Webster
Reason: All are prominent Elizabethan playwrights

Return only three wrong answers as a list in JSON format with the following requirements:
- Each wrong answer should be a string
- The output should be a single JSON object with key "negative_answers"
- The value should be an array of exactly 3 wrong answers
- No explanations or additional text should be included
- The answers should maintain consistent formatting with the correct answer

Example format:
{{
    "negative_answers": ["Wrong Answer 1", "Wrong Answer 2", "Wrong Answer 3"]
}}
```

B.8 Direct prompt template for unilateral factuality evaluation

```
Determine whether the following answer to the given question is TRUE or FALSE.

Conclude with a single line containing ONLY one of these two words:
TRUE
FALSE

Question: {question}
Proposed answer: {answer}
```

B.9 Zero-shot prompt template for unilateral factuality evaluation

```
I'll provide you with a question and its proposed answer.
Your task is to evaluate whether this answer is correct by following these steps:

1. Analyze the exact meaning of both the question and answer, identifying any key terms that need clarification.
2. Establish specific conditions that would make this answer true for this question.
3. Provide direct evidence supporting the answer, including specific facts, examples, or
authoritative references that confirm its accuracy.
4. Test if the answer remains valid across all contexts where the question applies, noting any limitations or
      exceptions.
5. Check for consistency with established knowledge in the relevant domain.
6. Search for direct counterexamples or contradicting evidence that actively demonstrates why the answer is incorrect
      .
7. Construct specific scenarios where the answer fails to hold true, even if the question's premises are accepted.
8. Identify any logical inconsistencies, factual errors, or category mistakes within the answer.

After analyzing the question and answer, provide a single line containing ONLY one of these two words:
```

```
TRUE
FALSE

Question: {question}
Proposed answer: {answer}
```

B.10 Few-shot prompt template for unilateral factuality evaluation

```
I'll provide you with a question and its proposed answer.
Your task is to evaluate whether this answer is correct by following these steps:

1. Analyze the exact meaning of both the question and answer, identifying any key terms that need clarification.
2. Establish specific conditions that would make this answer true for this question.
3. Provide direct evidence supporting the answer, including specific facts, examples, or
authoritative references that confirm its accuracy.
4. Test if the answer remains valid across all contexts where the question applies, noting any limitations or
      exceptions.
5. Check for consistency with established knowledge in the relevant domain.
6. Search for direct counterexamples or contradicting evidence that actively demonstrates why the answer is incorrect
      .
7. Construct specific scenarios where the answer fails to hold true, even if the question's premises are accepted.
8. Identify any logical inconsistencies, factual errors, or category mistakes within the answer.

Here are examples of how to approach evaluation:

Example 1:
Question: Who was the first man to walk on the Moon?
Proposed answer: Neil Armstrong
Analyze the question and answer:
Question: "Who was the first man to walk on the Moon?" This is a straightforward factual question seeking the
      identity of
the first human to set foot on the lunar surface.
Proposed answer: "Neil Armstrong" This is a name, presumably offered as the answer to the question.
Establish conditions for truth:
The answer is true if Neil Armstrong was indeed the first human to walk on the Moon.
Provide supporting evidence:
Historical records, NASA documentation, and countless reliable sources confirm that Neil Armstrong was the first
      person to
walk on the Moon on July 20, 1969, during the Apollo 11 mission.
Test validity across contexts:
The answer holds true in all historical contexts related to the first Moon landing.
Check for consistency with established knowledge:
The answer aligns perfectly with established historical and scientific knowledge.
Search for counterexamples:
There are no credible counterexamples. No other individual is historically recognized as the first person to walk on
      the Moon.
Construct failure scenarios:
There are no scenarios where the answer fails, assuming the question refers to the generally accepted historical
      event.
Identify logical inconsistencies:
There are no logical inconsistencies or factual errors.
TRUE

Example 2:
Question: What is the main ingredient in chocolate?
Proposed answer: Sugar
Analyze the question and answer:
Question: "Chocolate is made from material" - This is an incomplete sentence. The question is implicitly asking "What
       material is
chocolate made from?" or "What is a key material used to make chocolate?".
Proposed answer: "Sugar" - This suggests that sugar is the material chocolate is made from.
Establish conditions for truth:
The answer would be true if sugar was the only ingredient in chocolate, or
if the question was interpreted as "Is sugar a material used to make chocolate?".
Provide supporting evidence:
Sugar is a common and significant ingredient in most chocolate recipes.
Test validity across contexts:
This answer fails in many contexts. Chocolate is not only made from sugar.
Check for consistency with established knowledge:
Chocolate is made from cacao beans, sugar, and often other ingredients like milk solids, cocoa butter, lecithin, and
      flavorings.
Search for counterexamples:
Dark chocolate often contains a higher percentage of cacao and less sugar.
Sugar-free chocolate exists, using artificial sweeteners instead.
Cacao beans are essential for chocolate, and chocolate cannot be made without them.
Construct failure scenarios:
Imagine a recipe for 100% cacao chocolate. It would contain no sugar.
Imagine a sugar-free chocolate bar. It would contain no sugar.
Identify logical inconsistencies:
The answer implies sugar is the only ingredient, which is false.
FALSE
```

```
Question: {question}
Proposed answer: {answer}
```

C Example

To illustrate bilateral evaluation, we present an example with statements about penguins in the context of a knowledge base with a universally quantified statement that all birds can fly:

1. $\mathbf{C} = \{penguin, eagle, sparrow, ...\}$
2. $\varphi_0 = [\, \forall x\, bird(x)\,]\, flies(x)$
3. $\varphi_1 = \text{bird}(\text{penguin})$
4. $\varphi_2 = \text{flies}(\text{penguin})$
5. $\varphi_3 = \neg\varphi_2$
6. $\delta(\varphi_1) = \ulcorner\text{Penguins are birds}\urcorner$
7. $P^+(\delta(\varphi_1)) = \ulcorner\ldots$ Penguins are scientifically classified as birds. They belong to the family Spheniscidae . . . Conclusion: VERIFIED$\urcorner$
8. $P^-(\delta(\varphi_1)) = \ulcorner\ldots$ All evolutionary biologists classify penguins as birds. This is supported by molecular evidence, fossil records, and anatomical features. There is no reasonable alternative classification. Conclusion: CANNOT REFUTE$\urcorner$
9. $\zeta(\varphi_1) = \langle t, f\rangle$
10. $\delta(\varphi_2) = \ulcorner\text{Penguins fly}\urcorner$
11. $P^+(\delta(\varphi_2)) = \ulcorner\ldots$ While penguins have wings, they cannot achieve aerial flight. Their wings are adapted for swimming rather than flying. They flap their wings underwater to "fly" through water. From a strict biological perspective, penguins do not fly through air. Conclusion: CANNOT VERIFY$\urcorner$
12. $P^-(\delta(\varphi_2)) = \ulcorner\ldots$ Penguins are flightless birds. Their wings have evolved into flippers for aquatic propulsion rather than aerial flight. This is well-established in ornithology. Conclusion: REFUTED$\urcorner$
13. $\zeta(\varphi_2) = \langle f, t\rangle$
14. $\mathcal{I}(\varphi_0) = \langle \dot{\forall}(\{\mathcal{I}_0(\text{bird}(c)), \mathcal{I}_0(\text{flies}(c)) \mid c \in \mathbf{C}\}), \dot{\exists}(\{\mathcal{I}_0(\text{bird}(c)), \mathcal{I}_1(\text{flies}(c)) \mid c \in \mathbf{C}\})\rangle$
    $= \langle f, t\rangle$
15. $\mathcal{I}(\varphi_3) = \langle \mathcal{I}_1(\varphi_2), \mathcal{I}_0(\varphi_2)\rangle = \langle t, f\rangle$

This bilateral evaluation reveals the inconsistency. The universal statement φ_0 evaluates to false when considering penguins, and φ_3 evaluates to true, but both statements can coexist in $\mathbf{AC}$ without causing explosion. This demonstrates how the system handles the classic penguin problem through paraconsistent reasoning.

D Experiments

D.1 Datasets

We used the short-form factuality benchmarks GPQA (Rein et al. 2023) and SimpleQA (Wei et al. 2024) to create the benchmarks for our experiments. GPQA consists of 448

multiple-choice questions, written by domain experts in biology, physics, and chemistry. SimpleQA consists of 4,326 question/answer pairs addressing a range of general topic areas, including history, science and technology, art, geography, TV shows, and video games. From these two benchmarks we created a balanced set of positive and negative examples. From SimpleQA, we sampled without replacement 200 question/answer pairs to be positive examples, and 200 questions to be negative examples, where we substituted false answers synthetically generated using GPT-4o Mini using the prompt shown in Appendix B.7. From GPQA, we sampled 200 existing question/answer pairs as positive examples, and 200 questions paired with the first incorrect answer for that question provided as part of the dataset.

D.2 Experimental setup

Following Wei et al. (2024), we evaluated our implementation of ζ on a selective classification with a binary abstention task (El-Yaniv and Wiener 2010) using the above two datasets, measuring an LLM judge's grading of a given question/answer pair. The standard pattern in LLM judges in factuality evaluation is to prompt the LLM judge to evaluate the answer to the question as either correct, incorrect, or not attempted. This, again, has a natural mapping to the values of $\mathcal{V}_3$; to derive a single truth value $v \in \mathcal{V}_3$ for the evaluation, we use the following projection $p : \mathcal{V}_3 \times \mathcal{V}_3 \to \mathcal{V}_3$ such that for a pair $\langle u, v \rangle$:

$$p(\langle u, v \rangle) = \begin{cases} \mathfrak{t} & \text{if } \langle u, v \rangle = \langle \mathfrak{t}, \mathfrak{f} \rangle \\ \mathfrak{f} & \text{if } \langle u, v \rangle = \langle \mathfrak{f}, \mathfrak{t} \rangle \\ \mathfrak{e} & \text{otherwise} \end{cases}$$

Calls to the public inference APIs for the models used a temperature of 0.1. Repeated sampling (N=3) with majority vote was used in both the verification and refutation processes. Statistical significance was assessed using paired t-tests (`ttest_rel` from the `scipy.stats` Python package) to compare performance metrics between different model and prompt combinations. The experiments were conducted in the first half of May 2025 using calls to the public inference APIs for each of the models.

D.3 Experimental results

Judge Model	Prompt	Macro F1	Coverage	Time (s)	Tokens
Claude 3.5 Sonnet	direct	0.712 (0.023)	0.748 (0.02)	34.22 (6.08)	2914.80 (778.99)
	zero	0.738 (0.029)	0.542 (0.024)	53.91 (6.41)	4863.48 (731.27)
	few	0.716 (0.028)	0.54 (0.023)	52.92 (6.41)	8079.40 (745.71)
Claude 3.5 Haiku	direct	0.578 (0.027)	0.778 (0.02)	30.52 (4.09)	2641.07 (704.48)
	zero	0.648 (0.034)	0.412 (0.023)	43.12 (3.60)	4221.73 (685.85)
	few	0.604 (0.034)	0.438 (0.024)	47.44 (4.69)	7673.44 (700.47)
Llama 4 Maverick	direct	**0.774** (0.021)	**0.852** (0.016)	65.91 (52.53)	6225.19 (1526.74)
	zero	0.765 (0.025)	0.618 (0.022)	75.95 (44.90)	7492.54 (1357.27)
	few	0.751 (0.023)	0.805 (0.018)	65.08 (41.43)	9945.70 (1444.02)
Llama 4 Scout	direct	0.702 (0.025)	0.712 (0.021)	63.24 (28.92)	5403.00 (1833.85)
	zero	0.712 (0.031)	0.46 (0.024)	93.43 (51.27)	7062.05 (1430.96)
	few	0.694 (0.027)	0.642 (0.022)	69.27 (37.27)	9540.89 (1362.86)
GPT-4o	direct	0.592 (0.027)	0.705 (0.021)	**5.61** (8.41)	**1133.92** (556.51)
	zero	0.603 (0.035)	0.48 (0.022)	36.29 (9.44)	6041.24 (1256.13)
	few	0.69 (0.031)	0.518 (0.023)	42.01 (18.48)	8662.78 (1398.93)
GPT-4o Mini	direct	0.536 (0.028)	0.69 (0.022)	16.25 (12.91)	2863.86 (1716.43)
	zero	0.4 (0.025)	0.438 (0.023)	32.88 (7.54)	5851.67 (959.41)
	few	0.428 (0.028)	0.488 (0.023)	47.99 (15.90)	8787.62 (1120.93)

Table 8. Performance metrics for ζ using different judge models and evaluation prompts on GPQA question/answer pairs (N=400).

Judge Model	Prompt	Macro F1	Coverage	Time (s)	Tokens
Claude 3.5 Sonnet	direct	0.667 (0.021)	1.0 (0.0)	14.52 (2.92)	1360.21 (386.44)
	zero	0.664 (0.022)	1.0 (0.0)	19.29 (2.80)	2161.70 (369.97)
	few	0.66 (0.022)	1.0 (0.0)	23.64 (2.85)	5055.46 (382.93)
Claude 3.5 Haiku	direct	0.533 (0.023)	1.0 (0.0)	14.25 (2.23)	1346.54 (377.10)
	zero	0.556 (0.024)	1.0 (0.0)	19.70 (2.44)	2070.33 (328.06)
	few	0.577 (0.023)	0.998 (0.002)	19.48 (1.55)	4829.19 (327.03)
Llama 4 Maverick	direct	**0.722** (0.021)	**1.0** (0.0)	26.83 (21.83)	2919.78 (825.91)
	zero	0.694 (0.02)	0.998 (0.002)	25.20 (10.44)	3462.93 (665.47)
	few	0.717 (0.021)	0.99 (0.005)	24.49 (11.06)	5756.76 (602.70)
Llama 4 Scout	direct	0.636 (0.023)	1.0 (0.0)	30.07 (19.93)	2467.57 (1096.58)
	zero	0.577 (0.024)	0.998 (0.002)	24.74 (18.10)	2689.86 (1180.22)
	few	0.568 (0.023)	1.0 (0.0)	24.38 (18.08)	4955.30 (1124.60)
GPT-4o	direct	0.572 (0.023)	1.0 (0.0)	**2.92** (7.85)	**546.73** (276.40)
	zero	0.497 (0.024)	1.0 (0.0)	3.89 (4.92)	1059.73 (276.40)
	few	0.484 (0.024)	1.0 (0.0)	8.00 (8.50)	3446.44 (343.78)
GPT-4o Mini	direct	0.543 (0.023)	1.0 (0.0)	9.52 (6.61)	1543.24 (848.31)
	zero	0.444 (0.021)	1.0 (0.0)	6.60 (6.96)	1722.51 (945.13)
	few	0.538 (0.024)	1.0 (0.0)	11.41 (2.60)	4949.93 (477.92)

Table 9. Performance metrics for unilateral factuality evaluation using different judge models and evaluation prompts on GPQA question/answer pairs (N=400).

Judge Model	Prompt	Macro F1	Coverage	Time (s)	Tokens
Claude 3.5 Sonnet	direct	0.81 (0.025)	0.502 (0.024)	19.22 (4.49)	1190.43 (174.00)
	zero	0.733 (0.027)	0.615 (0.022)	43.35 (6.44)	3444.22 (171.12)
	few	0.654 (0.031)	0.65 (0.022)	43.19 (5.45)	6704.68 (161.86)
Claude 3.5 Haiku	direct	0.667 (0.033)	0.458 (0.024)	15.81 (3.07)	1014.98 (149.24)
	zero	0.673 (0.036)	0.385 (0.022)	38.92 (3.14)	3095.57 (131.72)
	few	0.653 (0.033)	0.45 (0.023)	40.01 (4.09)	6435.11 (158.40)
Llama 4 Maverick	direct	0.673 (0.026)	**0.768** (0.02)	21.84 (13.56)	2008.57 (754.99)
	zero	0.746 (0.029)	0.528 (0.024)	36.17 (10.85)	4421.40 (448.00)
	few	0.692 (0.026)	0.715 (0.021)	41.29 (31.55)	6665.58 (512.60)
Llama 4 Scout	direct	0.58 (0.031)	0.572 (0.023)	17.02 (9.05)	1392.07 (434.73)
	zero	0.677 (0.038)	0.358 (0.022)	43.41 (16.12)	4049.26 (380.54)
	few	0.558 (0.031)	0.632 (0.023)	36.90 (27.20)	6267.85 (437.70)
GPT-4o	direct	0.67 (0.026)	0.74 (0.021)	5.11 (5.49)	**477.22** (60.34)
	zero	0.738 (0.032)	0.44 (0.024)	22.18 (3.49)	3720.07 (314.42)
	few	**0.833** (0.026)	0.482 (0.024)	21.00 (4.86)	6079.94 (292.95)
GPT-4o Mini	direct	0.604 (0.027)	0.718 (0.021)	**2.53** (0.72)	483.86 (68.15)
	zero	0.525 (0.038)	0.378 (0.023)	23.75 (10.23)	3812.08 (829.15)
	few	0.586 (0.034)	0.472 (0.023)	30.69 (16.02)	6298.49 (254.71)

Table 10. Performance metrics for ζ using different judge models and evaluation prompts on SimpleQA question/answer pairs (N=400).

Judge Model	Prompt	Macro F1	Coverage	Time (s)	Tokens
Claude 3.5 Sonnet	direct	**0.705** (0.022)	**1.0** (0.0)	6.34 (1.59)	453.53 (80.98)
	zero	0.682 (0.022)	1.0 (0.0)	16.28 (5.05)	1499.60 (112.18)
	few	0.661 (0.023)	1.0 (0.0)	16.82 (1.99)	4331.51 (89.63)
Claude 3.5 Haiku	direct	0.595 (0.023)	1.0 (0.0)	6.69 (1.60)	489.73 (91.76)
	zero	0.55 (0.025)	1.0 (0.0)	16.29 (3.87)	1505.90 (93.60)
	few	0.523 (0.024)	1.0 (0.0)	17.02 (1.55)	4332.01 (63.46)
Llama 4 Maverick	direct	0.643 (0.023)	1.0 (0.0)	6.71 (4.20)	814.69 (350.94)
	zero	0.663 (0.023)	0.992 (0.004)	14.88 (8.78)	2097.99 (230.58)
	few	0.648 (0.023)	0.992 (0.004)	16.97 (10.19)	4524.26 (265.16)
Llama 4 Scout	direct	0.578 (0.023)	1.0 (0.0)	5.73 (4.96)	511.43 (286.82)
	zero	0.559 (0.025)	1.0 (0.0)	8.62 (8.44)	1192.79 (516.58)
	few	0.552 (0.024)	1.0 (0.0)	5.08 (6.68)	3306.05 (376.05)
GPT-4o	direct	0.62 (0.023)	1.0 (0.0)	2.55 (5.42)	**220.40** (30.07)
	zero	0.614 (0.024)	1.0 (0.0)	3.49 (3.70)	733.39 (30.07)
	few	0.632 (0.023)	1.0 (0.0)	7.38 (9.60)	3088.39 (30.07)
GPT-4o Mini	direct	0.587 (0.023)	1.0 (0.0)	**1.08** (0.19)	220.58 (30.79)
	zero	0.528 (0.023)	1.0 (0.0)	1.56 (1.52)	788.69 (185.84)
	few	0.572 (0.022)	1.0 (0.0)	6.75 (2.37)	3923.80 (318.41)

Table 11. Performance metrics for unilateral factuality evaluation using different judge models and evaluation prompts on SimpleQA question/answer pairs (N=400).

Judge Model	**Prompt**	$\langle \mathfrak{t}, \mathfrak{t} \rangle$	$\langle \mathfrak{t}, \mathfrak{f} \rangle$	$\langle \mathfrak{f}, \mathfrak{t} \rangle$	$\langle \mathfrak{f}, \mathfrak{f} \rangle$
Claude 3.5 Sonnet	direct	0.202 (0.018)	0.392 (0.022)	0.355 (0.022)	0.05 (0.011)
	zero	0.435 (0.023)	0.218 (0.02)	0.325 (0.021)	0.022 (0.007)
	few	0.448 (0.023)	0.18 (0.018)	0.36 (0.022)	0.012 (0.005)
Claude 3.5 Haiku	direct	0.11 (0.015)	0.53 (0.023)	0.248 (0.02)	0.112 (0.015)
	zero	0.53 (0.023)	0.208 (0.019)	0.205 (0.019)	0.058 (0.011)
	few	0.502 (0.024)	0.212 (0.02)	0.225 (0.02)	0.06 (0.011)
Llama 4 Maverick	direct	0.04 (0.009)	0.442 (0.023)	0.41 (0.023)	0.108 (0.015)
	zero	0.37 (0.022)	0.245 (0.02)	0.372 (0.023)	0.012 (0.005)
	few	0.155 (0.017)	0.438 (0.023)	0.368 (0.022)	0.04 (0.009)
Llama 4 Scout	direct	0.072 (0.013)	0.368 (0.023)	0.345 (0.022)	0.215 (0.019)
	zero	0.525 (0.024)	0.245 (0.021)	0.215 (0.019)	0.015 (0.006)
	few	0.33 (0.022)	0.385 (0.024)	0.258 (0.02)	0.028 (0.007)
GPT-4o	direct	0.082 (0.013)	0.358 (0.022)	0.348 (0.023)	0.212 (0.018)
	zero	0.502 (0.023)	0.108 (0.015)	0.372 (0.022)	0.018 (0.006)
	few	0.468 (0.024)	0.155 (0.017)	0.362 (0.022)	0.015 (0.006)
GPT-4o Mini	direct	0.172 (0.018)	0.145 (0.017)	0.545 (0.023)	0.138 (0.016)
	zero	0.555 (0.023)	0.018 (0.006)	0.42 (0.023)	0.008 (0.004)
	few	0.51 (0.023)	0.028 (0.007)	0.46 (0.022)	0.002 (0.002)

Table 12. Truth value probabilities for ζ using different judge models and evaluation prompts for GPQA.

Judge Model	**Prompt**	$\langle \mathfrak{t}, \mathfrak{t} \rangle$	$\langle \mathfrak{t}, \mathfrak{f} \rangle$	$\langle \mathfrak{f}, \mathfrak{t} \rangle$	$\langle \mathfrak{f}, \mathfrak{f} \rangle$
Claude 3.5 Sonnet	direct	0.052 (0.01)	0.218 (0.02)	0.285 (0.021)	0.445 (0.024)
	zero	0.245 (0.02)	0.152 (0.016)	0.462 (0.022)	0.14 (0.016)
	few	0.278 (0.021)	0.118 (0.014)	0.532 (0.023)	0.072 (0.012)
Claude 3.5 Haiku	direct	0.035 (0.008)	0.282 (0.022)	0.175 (0.018)	0.507 (0.024)
	zero	0.148 (0.016)	0.145 (0.016)	0.24 (0.02)	0.468 (0.023)
	few	0.132 (0.016)	0.162 (0.017)	0.288 (0.021)	0.418 (0.023)
Llama 4 Maverick	direct	0.088 (0.013)	0.588 (0.022)	0.18 (0.018)	0.145 (0.016)
	zero	0.438 (0.024)	0.35 (0.022)	0.178 (0.018)	0.032 (0.008)
	few	0.218 (0.019)	0.525 (0.023)	0.19 (0.019)	0.068 (0.012)
Llama 4 Scout	direct	0.125 (0.015)	0.368 (0.022)	0.205 (0.018)	0.302 (0.021)
	zero	0.62 (0.022)	0.245 (0.02)	0.112 (0.015)	0.022 (0.007)
	few	0.232 (0.02)	0.53 (0.023)	0.102 (0.014)	0.132 (0.016)
GPT-4o	direct	0.058 (0.011)	0.462 (0.023)	0.278 (0.022)	0.202 (0.019)
	zero	0.56 (0.024)	0.105 (0.014)	0.335 (0.023)	0.0 (0.0)
	few	0.51 (0.024)	0.205 (0.018)	0.278 (0.021)	0.008 (0.004)
GPT-4o Mini	direct	0.145 (0.017)	0.228 (0.019)	0.49 (0.023)	0.138 (0.016)
	zero	0.62 (0.023)	0.055 (0.01)	0.322 (0.022)	0.002 (0.002)
	few	0.488 (0.022)	0.272 (0.02)	0.2 (0.018)	0.038 (0.009)

Table 13. Truth value probabilities for ζ using different judge models and evaluation prompts for SimpleQA.

Topic	Model	⟨t, f⟩	⟨f, t⟩	⟨t, t⟩	⟨f, f⟩
Art	GPT-4o	0.127 (0.039)	0.327 (0.056)	0.545 (0.059)	0.000 (0.000)
	GPT-4o-mini	0.036 (0.022)	0.291 (0.054)	0.673 (0.056)	0.000 (0.000)
	Claude 3.5 Sonnet	0.182 (0.045)	0.509 (0.059)	0.182 (0.045)	0.127 (0.039)
	Claude 3.5 Haiku	0.200 (0.047)	0.218 (0.048)	0.109 (0.037)	0.473 (0.060)
	Llama 4 Maverick	0.345 (0.055)	0.145 (0.041)	0.491 (0.058)	0.018 (0.016)
	Llama 4 Scout	0.273 (0.051)	0.055 (0.026)	0.655 (0.054)	0.018 (0.016)
Geography	GPT-4o	0.114 (0.042)	0.409 (0.063)	0.477 (0.067)	0.000 (0.000)
	GPT-4o-mini	0.114 (0.042)	0.273 (0.058)	0.614 (0.064)	0.000 (0.000)
	Claude 3.5 Sonnet	0.159 (0.047)	0.500 (0.065)	0.273 (0.060)	0.068 (0.032)
	Claude 3.5 Haiku	0.182 (0.049)	0.318 (0.059)	0.159 (0.049)	0.341 (0.061)
	Llama 4 Maverick	0.386 (0.062)	0.182 (0.050)	0.409 (0.064)	0.023 (0.019)
	Llama 4 Scout	0.295 (0.059)	0.091 (0.037)	0.614 (0.063)	0.000 (0.000)
History	GPT-4o	0.143 (0.076)	0.357 (0.108)	0.500 (0.109)	0.000 (0.000)
	GPT-4o-mini	0.071 (0.056)	0.500 (0.111)	0.429 (0.112)	0.000 (0.000)
	Claude 3.5 Sonnet	0.214 (0.091)	0.357 (0.108)	0.214 (0.094)	0.214 (0.088)
	Claude 3.5 Haiku	0.214 (0.090)	0.214 (0.094)	0.143 (0.080)	0.429 (0.108)
	Llama 4 Maverick	0.429 (0.111)	0.214 (0.093)	0.357 (0.108)	0.000 (0.000)
	Llama 4 Scout	0.357 (0.107)	0.143 (0.079)	0.500 (0.109)	0.000 (0.000)
Music	GPT-4o	0.000 (0.000)	0.361 (0.069)	0.639 (0.069)	0.000 (0.000)
	GPT-4o-mini	0.028 (0.023)	0.278 (0.066)	0.694 (0.068)	0.000 (0.000)
	Claude 3.5 Sonnet	0.056 (0.033)	0.611 (0.068)	0.194 (0.056)	0.139 (0.049)
	Claude 3.5 Haiku	0.111 (0.044)	0.278 (0.065)	0.083 (0.039)	0.528 (0.072)
	Llama 4 Maverick	0.278 (0.065)	0.167 (0.054)	0.556 (0.072)	0.000 (0.000)
	Llama 4 Scout	0.194 (0.056)	0.167 (0.054)	0.611 (0.072)	0.028 (0.023)
Other	GPT-4o	0.136 (0.044)	0.341 (0.062)	0.523 (0.065)	0.000 (0.000)
	GPT-4o-mini	0.068 (0.032)	0.455 (0.065)	0.477 (0.067)	0.000 (0.000)
	Claude 3.5 Sonnet	0.136 (0.045)	0.455 (0.064)	0.205 (0.052)	0.205 (0.052)
	Claude 3.5 Haiku	0.091 (0.038)	0.227 (0.055)	0.136 (0.045)	0.545 (0.067)
	Llama 4 Maverick	0.295 (0.060)	0.114 (0.041)	0.523 (0.064)	0.068 (0.033)
	Llama 4 Scout	0.295 (0.060)	0.136 (0.044)	0.455 (0.064)	0.114 (0.041)
Politics	GPT-4o	0.131 (0.038)	0.344 (0.052)	0.525 (0.053)	0.000 (0.000)
	GPT-4o-mini	0.066 (0.028)	0.295 (0.051)	0.623 (0.055)	0.016 (0.014)
	Claude 3.5 Sonnet	0.180 (0.043)	0.443 (0.054)	0.197 (0.044)	0.180 (0.042)
	Claude 3.5 Haiku	0.164 (0.041)	0.246 (0.048)	0.148 (0.041)	0.443 (0.055)
	Llama 4 Maverick	0.410 (0.055)	0.262 (0.050)	0.295 (0.052)	0.033 (0.020)
	Llama 4 Scout	0.246 (0.048)	0.148 (0.039)	0.574 (0.055)	0.033 (0.020)
Sci/Tech	GPT-4o	0.101 (0.031)	0.304 (0.047)	0.595 (0.050)	0.000 (0.000)
	GPT-4o-mini	0.038 (0.018)	0.291 (0.046)	0.671 (0.048)	0.000 (0.000)
	Claude 3.5 Sonnet	0.228 (0.043)	0.329 (0.047)	0.367 (0.050)	0.076 (0.026)
	Claude 3.5 Haiku	0.152 (0.036)	0.114 (0.032)	0.228 (0.042)	0.506 (0.049)
	Llama 4 Maverick	0.456 (0.049)	0.203 (0.039)	0.329 (0.047)	0.013 (0.011)
	Llama 4 Scout	0.253 (0.043)	0.127 (0.033)	0.620 (0.049)	0.000 (0.000)
Sports	GPT-4o	0.161 (0.059)	0.258 (0.068)	0.581 (0.078)	0.000 (0.000)
	GPT-4o-mini	0.065 (0.038)	0.387 (0.073)	0.548 (0.073)	0.000 (0.000)
	Claude 3.5 Sonnet	0.065 (0.038)	0.452 (0.081)	0.290 (0.071)	0.194 (0.062)
	Claude 3.5 Haiku	0.129 (0.051)	0.226 (0.066)	0.194 (0.062)	0.452 (0.075)
	Llama 4 Maverick	0.290 (0.071)	0.161 (0.056)	0.452 (0.078)	0.065 (0.036)
	Llama 4 Scout	0.129 (0.051)	0.097 (0.045)	0.774 (0.063)	0.000 (0.000)
TV	GPT-4o	0.040 (0.033)	0.400 (0.082)	0.560 (0.082)	0.000 (0.000)
	GPT-4o-mini	0.000 (0.000)	0.400 (0.084)	0.600 (0.084)	0.000 (0.000)
	Claude 3.5 Sonnet	0.080 (0.047)	0.640 (0.082)	0.080 (0.045)	0.200 (0.069)
	Claude 3.5 Haiku	0.000 (0.000)	0.520 (0.086)	0.040 (0.033)	0.440 (0.086)
	Llama 4 Maverick	0.120 (0.054)	0.160 (0.063)	0.600 (0.081)	0.120 (0.056)
	Llama 4 Scout	0.160 (0.061)	0.080 (0.047)	0.760 (0.073)	0.000 (0.000)
Games	GPT-4o	0.000 (0.000)	0.182 (0.096)	0.818 (0.096)	0.000 (0.000)
	GPT-4o-mini	0.091 (0.073)	0.091 (0.072)	0.818 (0.094)	0.000 (0.000)
	Claude 3.5 Sonnet	0.000 (0.000)	0.455 (0.122)	0.455 (0.125)	0.091 (0.073)
	Claude 3.5 Haiku	0.182 (0.098)	0.273 (0.116)	0.091 (0.071)	0.455 (0.128)
	Llama 4 Maverick	0.182 (0.097)	0.000 (0.000)	0.818 (0.097)	0.000 (0.000)
	Llama 4 Scout	0.182 (0.095)	0.000 (0.000)	0.818 (0.095)	0.000 (0.000)

Table 14. Truth value probablities by model and SimpleQA topic with zero-shot prompting. Values show proportions (SE) for each of the following epistemic states: $\langle t, f \rangle$ (verified, not refuted), $\langle f, t \rangle$ (not verified, refuted), $\langle t, t \rangle$ (contradictory), $\langle f, f \rangle$ (uncertain). Standard errors estimated via subsampling (Politis and Romano 1994).