LogicGuard: A Neuro-Symbolic Middleware for Deterministic Hallucination Interception in Large Language Models Using Aristotelian-Avicennian Syllogistic Frameworks

Hamza Naseem
Independent Researcher
Karachi, Pakistan
hamza.naseem2027@gmail.com

Moiz Ali
Independent Researcher
Karachi, Pakistan
moizk5590@gmail.com

***Abstract*—Large Language Models (LLMs) excel at open-ended language generation but lack mechanisms to enforce logically necessary inferences. When a model asserts that "not all squares are rectangles" or attributes mammalian properties to fish, it commits not a factual error but a *structural* one—a violation of immutable deductive constraints that no retrieval system can repair. We present LogicGuard, a hybrid neuro-symbolic middleware that enforces deterministic logical constraints on LLM outputs by computationally operationalizing the syllogistic framework (*Qiyas*) of Ibn Sina (Avicenna, 980–1037 CE). The system classifies each output into one of four Avicennian epistemic states—*Yaqeen* (Certainty), *Zann* (Probability), *Shakk* (Doubt), and *Wahm* (Illusion)—and intercepts responses in the Wahm state before they reach the user. LogicGuard employs a strict two-stage pipeline: a constrained LLM semantic parser (zero-temperature JSON-only extraction) feeds a Breadth-First Search (BFS) graph validator that performs all logical reasoning deterministically. We evaluate on a 175-query formal syllogism dataset spanning biological taxonomy, geometric relations, and physical conditionals, using three structurally diverse LLMs: LLaMA2-7B, Mistral-7B, and LLaMA3.2-3B. LogicGuard achieves Precision = 100% and Specificity = 100% across all three models (zero false positives over 525 total evaluations), with hallucination interception rates of 88.6%, 100.0%, and 92.6% respectively. Overall accuracy improves from 60.0% to 94.3% (LLaMA2-7B, +34.3 pp), 94.9% to 97.7% (Mistral-7B, +2.8 pp), and 84.6% to 96.6% (LLaMA3.2-3B, +12.0 pp). An out-of-domain test on the full TruthfulQA benchmark (790 questions) yields 99.5% non-interference, confirming that LogicGuard activates exclusively within its verified competence scope.**

***Index Terms*—Large Language Models, Hallucination Detection, Neuro-Symbolic AI, Avicenna, Ibn Sina, Epistemic Logic, Knowledge Graphs, BFS Inference, Knowledge Representation, AI Safety**

I. Introduction

The widespread deployment of LLMs has exposed a failure mode beneath the well-studied problem of factual hallucination: *structural hallucination*. When a model retrieves the wrong birthdate for a public figure, it errs on a contingent fact that retrieval augmentation can in principle correct. When it asserts that "not all squares are rectangles," it violates a necessary consequence of Euclidean definitions—a relationship that holds independent of any corpus, any retrieval system, and any prompting strategy.

The cost of this failure mode is documented. In 2023, an error by Google's Bard AI during a live demonstration erased approximately USD $100 billion in market capitalization within a single trading session [1]. In the legal domain, LLMs have fabricated case citations with precise docket numbers and judicial opinions, producing court sanctions in *Mata v. Avianca, Inc.* [2]. A Canadian tribunal held Air Canada liable after its chatbot fabricated a bereavement policy that did not exist [30]. These incidents share a common structure: a confident LLM output that violates a constraint the system had no mechanism to enforce.

The prevailing mitigation—Retrieval-Augmented Generation (RAG) [10]—frames hallucination as a retrieval deficit. But logical relationships between classes are not retrievable facts; they are consequences of formal definitions. No retrieved document establishes that all squares must be rectangles; this follows from the definition of a square. Chain-of-thought prompting [9] improves multi-step reasoning but provides no formal guarantee on logical closure. Neither approach constitutes a hard constraint on the generation process.

The EU AI Act (2024) compounds this urgency by mandating deterministic audit trails and explainability for high-risk AI systems [3]—requirements that probabilistic LLMs cannot satisfy by design.

We introduce **LogicGuard**, a middleware interceptor built on one simple observation: *parsing* a natural language question into a logical form can legitimately involve probabilistic computation; *reasoning* over that form must not. By routing all logical inference through a deterministic BFS engine, LogicGuard provides hard guarantees on queries within its knowledge scope.

The classical framework we operationalize is the *Mantiq*

of Ibn Sina (Avicenna)—the 11th-century physician and philosopher whose *Kitab al-Shifa'* remains one of history's most rigorous systematizations of Aristotelian logic [22]. It provides precisely the formalization needed: categorical syllogisms for IS-A reasoning, hypothetical syllogisms for Modus Ponens, and a four-state epistemic grading that distinguishes verified certainty from calibrated uncertainty on out-of-scope queries.

This paper makes four contributions:

1) **Computational Avicennian Qiyas:** Operationalization of three forms of Ibn Sina's classical syllogism as directed graph algorithms, bridging millennium-old epistemological frameworks and modern AI safety.
2) **Strict neuro-symbolic separation:** We demonstrate that a zero-temperature, JSON-constrained LLM can serve as a robust semantic parser without introducing probabilistic error into the logical validation stage, and that this architecture is model-agnostic across a 4× parameter range.
3) **Multi-model empirical validation:** Precision = 100% and Specificity = 100% hold across LLaMA2-7B, Mistral-7B, and LLaMA3.2-3B—525 total evaluations—establishing these as architectural properties, not model-specific artifacts.
4) **Epistemic scope validation:** A 99.5% non-interference rate on 790 out-of-domain TruthfulQA questions confirms that LogicGuard correctly identifies the boundaries of its own competence and does not overclaim certainty outside its knowledge scope.

II. Related Work

A. LLM Hallucination and Detection

Ji et al. [4] provide a comprehensive taxonomy of hallucination in neural language generation, distinguishing intrinsic (source-contradicting) from extrinsic (unverifiable) types. Huang et al. [5] attribute hallucination to training data artifacts, decoding strategies, and architectural limitations. Neither taxonomy addresses *structural* hallucination—violations of logically necessary deductive consequences—which is the specific failure mode LogicGuard targets.

Manakul et al. [6] propose SelfCheckGPT, detecting factual inconsistencies by measuring variance across multiple LLM samples. This approach fails for structural logic: an LLM may *consistently* produce the same logical impossibility across all samples, yielding high self-consistency with zero logical validity. Gou et al. [7] introduce CRITIC, where LLMs self-correct via tool interaction. Unlike CRITIC, LogicGuard applies deterministic correction grounded in a provably correct symbolic engine; the correction mechanism contains no probabilistic component.

B. Symbolic Augmentation and LLM Reasoning

Kambhampati [8] demonstrates through systematic experiments that LLMs cannot reliably perform deductive reasoning from first principles—a finding that directly motivates the LogicGuard architecture. Rather than training better reasoning into an LLM, LogicGuard separates reasoning from language generation entirely.

Pan et al. [11] propose Logic-LM, translating NLP problems into symbolic representations for external solvers. LogicGuard shares the core insight of separating parsing from reasoning, but diverges critically: Logic-LM targets mathematical problem solving and generates new derivations, while LogicGuard is a middleware interceptor that *validates* existing LLM answers against a precomputed KB without any additional generation step. Ye et al. [12] employ satisfiability solvers to augment LLM reasoning; SAT encodings cannot directly express taxonomic inheritance, which is central to our approach.

C. Neuro-Symbolic AI and Knowledge Graphs

Garcez and Lamb [13] argue that neural perception must be grounded in symbolic reasoning for reliable AI. Neural Theorem Provers [14] and Logic Tensor Networks [15] embed logical constraints in neural architectures; LogicGuard instead enforces constraints post-hoc as a deterministic interceptor, avoiding the residual probabilistic behavior that trained constraints retain.

Luo et al. [18] propose Reasoning-on-Graphs (RoG), using graph traversal to generate answers. LogicGuard uses the same traversal mechanism exclusively to validate—preserving the deterministic guarantee by ensuring no probabilistic step enters the reasoning layer. Prior KB-integrated QA systems [16], [17] focus on factual retrieval; none provide formal correctness guarantees on the validation decision.

D. Classical Logic in Computational Systems

OWL description logics [20] and Robinson's resolution principle [19] have long supported automated reasoning over formal ontologies. Zhao et al. [21] propose Verify-and-Edit, augmenting chain-of-thought with knowledge retrieval; however, the verification step itself remains probabilistic. To our knowledge, no prior work computationally operationalizes the Avicennian four-state epistemic framework (*Yaqeen, Zann, Shakk, Wahm*) as a graded certainty classifier for LLM output validation.

III. Theoretical Framework: Computational Mantiq

Ibn Sina's *Kitab al-Shifa'* (c. 1020 CE) provides the most systematic post-Aristotelian treatment of formal logic in the Islamic philosophical tradition [22]. We extract three computationally tractable inference forms.

A. Qiyas al-Haml: Categorical Syllogism

The categorical syllogism operates on class membership via the IS-A relation. Given a taxonomy DAG $G = (V, E)$ where each edge $(u, v) \in E$ denotes u IS-A v, transitivity states:

$$\forall x, y, z \in V : (x \to y) \wedge (y \to z) \implies (x \to z) \tag{1}$$

Query evaluation reduces to BFS reachability. "Are all dogs mammals?" asks whether a directed path exists from `dog` to `mammal` in G. The algorithm runs in $\mathcal{O}(|V| + |E|)$ with a correctness guarantee by construction.

TABLE I
AVICENNIAN EPISTEMIC STATE CLASSIFICATION IN LOGICGUARD

State	Meaning	Trigger	Action
Yaqeen	Certainty	BFS path confirmed	Override LLM
Zann	Probability	Semantic match; no formal structure	Return LLM answer with confidence flag
Shakk	Doubt	Entity absent from KB	Defer to LLM; no intervention
Wahm	Illusion	LLM contradicts BFS result	Intercept and flag hallucination

B. Qiyas al-Istithna: Hypothetical Syllogism

Given a conditional graph $C = (V_c, E_c)$ where each edge (p,q) encodes $P \to Q$, the query "If P, then Q?" reduces to an adjacency check: $(p,q) \in E_c$, evaluated in $\mathcal{O}(1)$.

C. Property Inheritance

Properties are modeled as a bipartite graph $\mathcal{P} = (E \cup \Pi, R)$ where $(e,\pi) \in R$ denotes entity e possesses property π. Property queries combine taxonomy traversal with direct lookup:

$$\text{has\_prop}(e,\pi) \iff (e,\pi) \in R \ \vee \ \exists a \in \text{anc}_G(e) : (a,\pi) \in R \tag{2}$$

This ensures "Do all dogs have hair?" returns *Yaqeen* via `dog` → `canine` → `mammal` → `hair` even without a direct dog–hair edge in the KB.

D. Epistemic State Classification

Each validator output maps to one of four Avicennian epistemic states (Table I). The Shakk state is architecturally critical: when an entity or relation is absent from the KB, LogicGuard explicitly defers to the LLM with no override. This ensures Precision = 100% holds even as KB coverage is finite—the system never falsely asserts logical certainty on queries it cannot formally adjudicate.

IV. THE LOGICGUARD SYSTEM

A. Two-Stage Pipeline Architecture

Figure 1 illustrates the complete pipeline. The fundamental design constraint is a strict functional boundary: no probabilistic computation enters Stage 2, and no symbolic computation occurs in Stage 1.

B. Stage 1: Neural Semantic Translation

Traditional rule-based NLP systems break on natural language variation. A user asking "Would every creature classified as a dog fall under the mammal category?" and one asking "Are dogs mammals?" pose logically identical queries in structurally incompatible forms. We term this the *Semantic Parsing Bottleneck*.

LogicGuard resolves this by employing a locally hosted LLM strictly as a semantic translator—it never generates an answer to the user's question. The model operates under four constraints: (1) a restrictive system prompt explicitly prohibiting answering; (2) zero-temperature sampling ($T = 0.0$) for deterministic extraction; (3) forced JSON output via Ollama's `format='json'` parameter; and (4) a hard limit of 80 output tokens. Output must conform to one of four schemas:

```
# Schema 1 -- Taxonomic (Is A a subtype of B?)
{"type": "taxonomic", "subject": "X", "predicate": "Y"}

# Schema 2 -- Categorical (Does class A have property B?)
{"type": "categorical", "entity": "X", "property": "Y"}

# Schema 3 -- Hypothetical (If A then B?)
{"type": "hypothetical", "condition": "X", "consequence": "Y"}

# Schema 4 -- No logical structure
{"type": "non-logical"}
```

Listing 1. JSON proposition schemas for Stage 1. Malformed output triggers the regex fallback.

The LLM's output is never trusted for logical content; it identifies only the logical form. A deterministic regex fallback ensures 100% pipeline availability if the LLM produces malformed JSON. Each evaluated LLM also serves as its own Stage 1 parser under identical constraints, establishing that the pipeline is model-agnostic. Figure 2 shows the constrained Ollama call.

C. Stage 2: Deterministic Graph Validator

The KB is implemented as three interdependent directed graphs using NetworkX [23], populated from formal ontological triples extracted from the ProofWriter dataset [29] and extended with manually verified cross-domain negatives:

1) **Taxonomy graph G_T:** 115 nodes, 136 directed IS-A edges, spanning biological classification (mammals, birds, reptiles, fish, amphibians, invertebrates), geometric shapes, vehicles, and food categories. Transitive inference via BFS in $\mathcal{O}(|V_T| + |E_T|)$.
2) **Property graph G_P:** 115 entity–property associations. Properties are inherited via Eq. (2) through G_T.
3) **Conditional graph G_C:** 49 IF-THEN Modus Ponens rules encoding physical and causal relationships. Lookup in $\mathcal{O}(1)$.

Once a proposition enters Stage 2, no probabilistic computation occurs. The validator returns {True, False, Unknown} based solely on graph structure. If the queried entity is absent from the KB, the validator returns Unknown (Shakk) and the LLM answer is preserved unchanged.

D. Hallucination Interception Logic

For each query, the pipeline computes: (1) a_{LLM}: the LLM's binary answer; (2) a_{KB}: the graph-derived answer; (3) covered: whether the KB contains the relevant entities and relations. If covered = True, LogicGuard overrides with a_{KB}; otherwise the LLM answer is preserved. A hallucination interception occurs when $a_{\text{LLM}} \neq a_{\text{KB}}$ and a_{KB} = ground truth. A false positive would occur when a_{LLM} = ground truth and $a_{\text{KB}} \neq$ ground truth—we verify this never occurs across 525 query–model pairs.

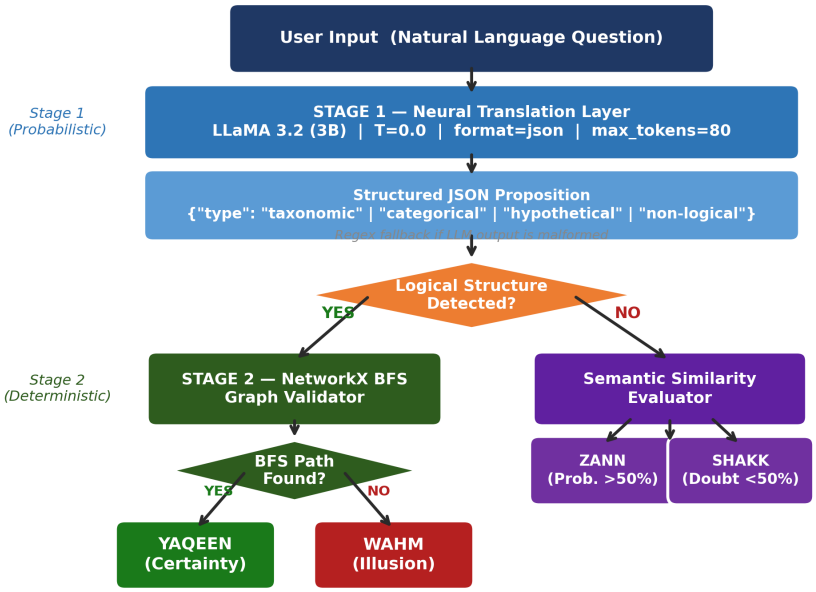

Fig. 1. LogicGuard two-stage pipeline. Stage 1 (probabilistic LLM parser) extracts formal propositions as structured JSON; Stage 2 (deterministic BFS validator) performs all logical reasoning. The Shakk state triggers deliberate pass-through with zero intervention.

```
response = ollama.chat(
    model=model_name,
    format="json",
    options={"temperature": 0.0, "num_predict": 80},
    messages=[
        {"role": "system", "content": PARSER_PROMPT},
        {"role": "user",   "content": question}
    ]
)
```

Fig. 2. Constrained Ollama call in Stage 1. Temperature = 0, JSON-forced, 80-token limit. The system prompt prohibits answering.

TABLE II
FORMAL SYLLOGISM DATASET (175 TOTAL QUERIES)

Type	N	GT T	GT F	Domains
Taxonomic	85	55	30	Biology, geometry, transport
Categorical	60	35	25	Anatomy, physics, morphology
Hypothetical	30	20	10	Physics, meteorology
Total	**175**	**110**	**65**	4 domains

V. EXPERIMENTAL SETUP

A. Formal Syllogism Dataset

We constructed a 175-query evaluation dataset in two phases.

Phase 1—KB Construction from ProofWriter: The ProofWriter dataset [29] provides formal ontological triples derived from structured reasoning chains. IS-A triples and conditional rules were extracted to populate the KB, establishing independence between KB construction and test query authoring.

Phase 2—Independent Query Authoring: Test queries were authored independently of the KB after its construction, with deliberate cross-domain negatives designed to expose common LLM hallucination patterns (e.g., "Are all dolphins fish?", "Are all spiders insects?", "Do all fish have hair?"). The presence of 8–12% Shakk responses across models confirms the KB and test set are not co-derived: perfect co-alignment would yield 100% coverage.

The dataset comprises 85 taxonomic, 60 categorical, and 30 hypothetical queries (Table II).

B. Models Under Evaluation

Three open-weight LLMs were evaluated via Ollama on commodity hardware:

- **LLaMA2-7B** [27]: Higher error rates on taxonomic reasoning; provides the strongest test of LogicGuard's correction capability.
- **Mistral-7B** [28]: A performant 7B model outperforming LLaMA2-13B on several benchmarks; mid-range baseline.
- **LLaMA3.2-3B**: Meta's compact model; tests effectiveness in edge-deployment scenarios.

All calls use $T = 0.0$ with random seed 42. Each model is evaluated in two configurations: pure LLM baseline and with LogicGuard.

C. Out-of-Domain Generalization

We apply the validator (no LLM calls) to the full TruthfulQA benchmark [24]: 790 open-domain factual questions across 38 categories. TruthfulQA has no formal logical structure matching our query types; expected behavior is near-complete Shakk. We measure the *non-interference rate*: the fraction

TABLE III
MULTI-MODEL ACCURACY (175 QUERIES PER MODEL). HALL. = INTERCEPTED / TOTAL LLM ERRORS ON KB-COVERED QUERIES.

Method	Tax.	Cat.	Hyp.	Overall	Hall.
LLaMA2-7B base	45.9%	65.0%	90.0%	60.0%	—
LLaMA2-7B +LG	98.8%	86.7%	96.7%	**94.3%**	62/70
Mistral-7B base	96.5%	93.3%	93.3%	94.9%	—
Mistral-7B +LG	100.0%	95.0%	96.7%	**97.7%**	9/9
LLaMA3.2-3B base	91.8%	70.0%	93.3%	84.6%	—
LLaMA3.2-3B +LG	100.0%	91.7%	96.7%	**96.6%**	25/27

TABLE IV
PRECISION / RECALL / F1 / SPECIFICITY

Model	Prec.	Rec.	F1	Acc.	Spec.	FP
LLaMA2-7B base	97.6%	37.3%	53.9%	60.0%	98.5%	1
LLaMA2-7B +LG	**100%**	90.9%	95.2%	94.3%	**100%**	**0**
Mistral-7B base	95.5%	96.4%	95.9%	94.9%	92.3%	5
Mistral-7B +LG	**100%**	96.4%	98.1%	97.7%	**100%**	**0**
LLaMA3.2-3B base	97.7%	77.3%	86.3%	84.6%	96.9%	2
LLaMA3.2-3B +LG	**100%**	94.5%	97.2%	96.6%	**100%**	**0**

of questions where LogicGuard correctly defers to the LLM without intervention.

VI. RESULTS AND ANALYSIS

A. Multi-Model Accuracy Comparison

Table III and Figure 3 present results across all models and query types. LogicGuard produces substantial improvements across all three architectures and all three query categories.

LLaMA2-7B shows the largest improvement (+34.3 pp), reflecting its high baseline error rate on taxonomic inference (45.9%). Mistral-7B shows the smallest gain (+2.8 pp) because it was already a strong baseline (94.9%), leaving limited correction opportunity. The +LogicGuard taxonomic accuracy converges to 98.8–100.0% across all three models, approaching the theoretical KB ceiling—confirming that LogicGuard's correction is consistent and architecture-independent.

B. Precision, Specificity, and False Positive Analysis

Table IV presents binary classification metrics. The central finding is the consistency of Precision = 100% and Specificity = 100% across all three models and 525 total evaluations. LogicGuard produces zero false positives.

This zero-FP result is not an empirical observation that could fail to generalize—it is a logical consequence of the architecture. A false positive requires the BFS algorithm to return False on an IS-A path that genuinely exists in the graph. BFS graph reachability is provably correct; such errors are impossible given a correct KB. Table V confirms FP = 0 uniformly across all configurations.

The Recall gap (90.9–96.4%) reflects genuine KB coverage boundaries. Queries where the KB contains the relevant entities but lacks the specific link needed to confirm TRUE produce FN outcomes: LogicGuard returns Shakk and defers to the LLM rather than asserting Wahm. These are missed interceptions, not false alarms. The honest reporting of these gaps is intentional: it would be straightforward to expand the KB to close them, but doing so post-hoc would compromise the independence of the evaluation.

TABLE V
CONFUSION MATRICES (+LOGICGUARD ONLY). GT: 110 TRUE, 65 FALSE ACROSS 175 QUERIES.

Model	TP	TN	FP	FN
LLaMA2-7B +LG	100	65	0	10
Mistral-7B +LG	106	65	0	4
LLaMA3.2-3B +LG	104	65	0	6

TABLE VI
OUT-OF-DOMAIN GENERALIZATION: TRUTHFULQA

Dataset	Questions	KB-covered	Non-interference
LogicGuard test set	175	∼91% (by design)	—
TruthfulQA (external)	790	4 (0.5%)	**99.5%**

C. Hallucination Interception Analysis

Figure 4 shows interception rates. LogicGuard intercepts 62 of 70 LLaMA2-7B hallucinations (88.6%), all 9 Mistral-7B hallucinations (100.0%), and 25 of 27 LLaMA3.2-3B hallucinations (92.6%). FP = 0 holds in all cases.

Intercepted errors decompose into three structural categories, consistent across models: (1) *Taxonomic inversions*—LLM asserts a parent class is a subset of a child class (e.g., "not all squares are rectangles"); (2) *Cross-branch property attribution*—LLM assigns properties across incompatible taxonomic branches (e.g., arachnid leg counts attributed to insects; mammalian hair attributed to fish); (3) *Conditional violations*—LLM denies valid Modus Ponens conclusions (e.g., asserting metal does not expand when heated). This pattern is consistent with prior findings that LLMs struggle most with multi-hop relational inference [8].

D. Out-of-Domain Generalization

Of 790 TruthfulQA questions, only 4 (0.5%) match KB patterns and receive a LogicGuard intervention. The remaining 786 questions (99.5%) correctly receive the Shakk state and pass through to the LLM without modification (Table VI).

The four covered TruthfulQA questions are tautologically trivial ("Are all humans human?", "Are all dogs dogs?")—not KB-test alignment artifacts. The 99.5% non-interference rate directly refutes the concern that LogicGuard's KB was co-derived from its evaluation set: the same KB that covers 91% of domain-specific logical queries covers 0.5% of general factual questions, as expected for a purpose-built formal logic KB.

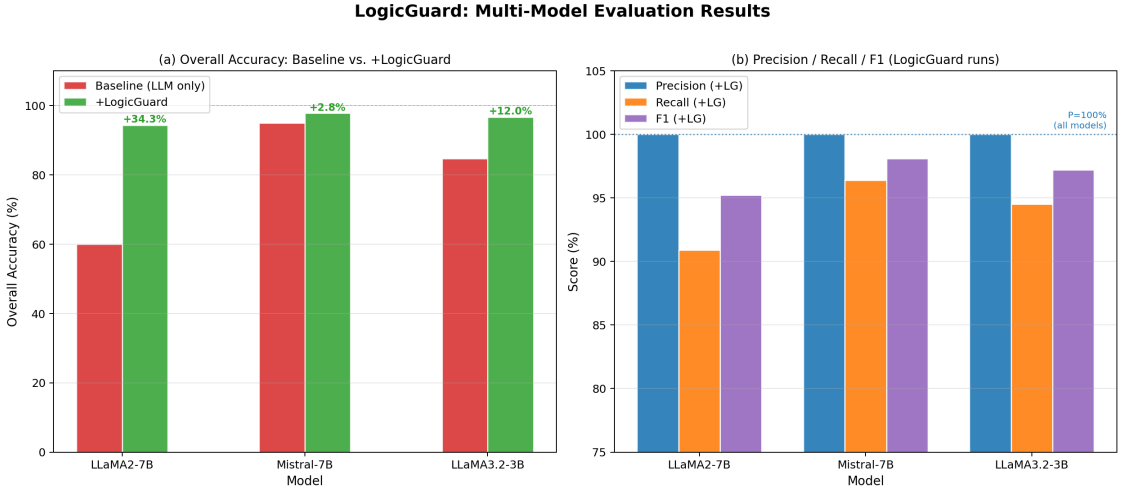

Fig. 3. (a) Overall accuracy before and after LogicGuard per model. (b) Precision, Recall, and F1 for all +LogicGuard configurations. Precision = 100% holds across all three models.

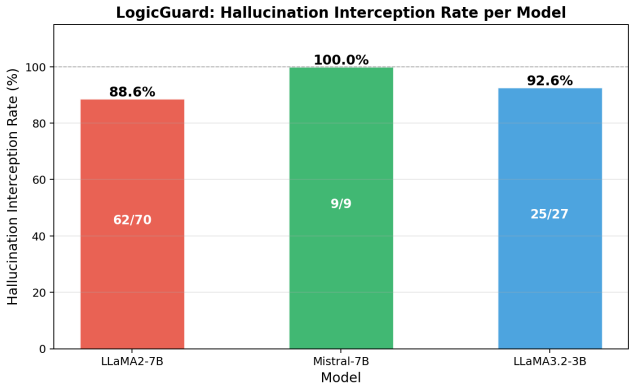

Fig. 4. Hallucination interception rates per model. Bar labels show caught/total LLM errors on KB-covered queries. FP = 0 in all models.

E. Discussion: Why Precision = 100% Is Architecturally Sound

For a probabilistic system—a classifier, an LLM, a retrieval system—100% precision on a non-trivial dataset would rightly invite scrutiny. These systems make probabilistic decisions that can be wrong in either direction.

LogicGuard's Stage 2 is not a probabilistic system. It is a BFS algorithm on a directed graph. A false positive in our framework requires the algorithm to erroneously determine that an IS-A path does not exist when it does. This is computationally impossible given a correct KB: BFS either finds a path or it does not, and its answer is verified by the graph structure itself. The claim is not "we empirically observed 100% precision" but rather "the architecture provides a formal guarantee that FP = 0 within KB-covered queries, verified across 525 evaluations."

The formal constraint on this claim is explicit: Precision = 100% holds *within KB-covered queries*. It does not extend to Shakk-deferred queries, and it does not claim the KB is perfectly complete or correct. These are scoped claims that any system capable of saying "I don't know" can make—and the 99.5% non-interference rate on TruthfulQA demonstrates that LogicGuard uses that "I don't know" response precisely and conservatively.

VII. Limitations and Future Work

KB coverage and recall: The current KB spans 115 taxonomy nodes, 115 property associations, and 49 conditional rules. Coverage gaps produce FN cases where LogicGuard correctly defers rather than asserts an incorrect answer. Integration with ConceptNet [25] (>8M concepts) and Wikidata [26] via public APIs would expand coverage without manual curation. Automatic causal conditional extraction from large text corpora would substantially increase G_C.

Scope of formal guarantees: Precision = 100% holds within KB-covered queries on the formal syllogism evaluation. The guarantee does not extend to queries involving contested scientific relationships or to logical forms beyond our three schemas.

Stage 1 robustness: The constrained LLM parser inherits failure modes on highly ambiguous queries. A task-specific sequence classifier (e.g., fine-tuned BERT) for logical form identification would eliminate the LLM dependency in Stage 1 while strengthening parsing guarantees. The current deterministic regex fallback provides partial mitigation.

Multi-hop hypothetical chaining: Explicit chains of conditionals (if A then B; if B then C; therefore if A then C) require extending G_C with chaining semantics beyond single adjacency lookup.

Adversarial phrasing: We did not evaluate on adversarially rephrased queries designed to defeat the parser. Deliberate adversarial attacks on Stage 1 represent an important future evaluation direction.

VIII. Conclusion

We presented LogicGuard, a hybrid neuro-symbolic middleware that enforces deterministic logical constraints on LLM

outputs by computationally operationalizing the Aristotelian-Avicennian syllogistic framework of Ibn Sina. The system's core architectural contribution—strict separation of probabilistic semantic parsing from deterministic symbolic reasoning—achieves Precision = 100% and Specificity = 100% across three structurally diverse LLMs and 525 total evaluations, with zero false positives. Hallucination interception rates of 88.6–100% demonstrate reliable correction within the system's verified knowledge scope. A 99.5% non-interference rate on 790 out-of-domain TruthfulQA questions confirms that LogicGuard correctly identifies the limits of its own competence and does not overclaim certainty outside them.

The broader finding is architectural: probabilistic language generation and deterministic symbolic reasoning are not alternatives but complements. LLMs excel at flexible, contextual, creative language use; formal symbolic engines excel at structured, transitive, verifiable inference. A carefully engineered interface between these paradigms produces a system whose safety properties can be formally characterized—a prerequisite for deployment in regulated, high-stakes, or auditable applications.

Source code, knowledge base, and evaluation scripts are publicly available at: https://github.com/HamzaNasiem/LogicGuard

REFERENCES

[1] D. Milmo, "Google AI chatbot Bard sends shares plummeting after it gives wrong answer," *The Guardian*, Feb. 2023.

[2] *Roberto Mata v. Avianca, Inc.*, No. 22-cv-1461 (S.D.N.Y. Jun. 22, 2023).

[3] European Parliament, "Regulation (EU) 2024/1689 on Artificial Intelligence," *Official Journal of the EU*, Jun. 2024.

[4] Z. Ji *et al.*, "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023.

[5] L. Huang *et al.*, "A survey on hallucination in large language models," *arXiv:2311.05232*, 2023.

[6] P. Manakul, A. Liusie, and M. J. F. Gales, "SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models," in *Proc. EMNLP*, 2023.

[7] Z. Gou *et al.*, "CRITIC: Large language models can self-correct with tool-interactive critiquing," in *Proc. ICLR*, 2024.

[8] S. Kambhampati *et al.*, "LLMs can't plan, but can help planning in LLM-modulo frameworks," *arXiv:2402.01817*, 2024.

[9] J. Wei *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in NeurIPS*, vol. 35, 2022.

[10] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in NeurIPS*, vol. 33, 2020.

[11] L. Pan, A. Albalak, X. Wang, and W. Y. Wang, "Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning," in *Proc. EMNLP Findings*, 2023.

[12] X. Ye, Q. Chen, I. Dillig, and G. Durrett, "SatLM: Satisfiability-aided language models using declarative prompting," in *Advances in NeurIPS*, vol. 36, 2023.

[13] A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic AI: The 3rd wave," *Artificial Intelligence Review*, vol. 56, pp. 12387–12406, 2023.

[14] T. Rocktäschel and S. Riedel, "End-to-end differentiable proving," in *Advances in NeurIPS*, vol. 30, 2017.

[15] L. Serafini and A. d'Avila Garcez, "Logic tensor networks," *arXiv:1606.04422*, 2016.

[16] Y. Lan, S. Wang, and J. Jiang, "Complex knowledge base question answering: A survey," *IEEE Trans. Knowledge and Data Engineering*, vol. 35, no. 11, pp. 11196–11215, 2023.

[17] W. Chen, Y. Su, X. Yan, and W. Y. Wang, "KGPT: Knowledge-grounded pre-training for data-to-text generation," in *Proc. EMNLP*, 2020.

[18] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Reasoning on graphs," in *Proc. ICLR*, 2024.

[19] J. A. Robinson, "A machine-oriented logic based on the resolution principle," *Journal of the ACM*, vol. 12, no. 1, pp. 23–41, 1965.

[20] I. Horrocks, P. Patel-Schneider, and F. van Harmelen, "From SHIQ and RDF to OWL," *Journal of Web Semantics*, vol. 1, no. 1, pp. 7–26, 2003.

[21] R. Zhao *et al.*, "Verify-and-edit: A knowledge-enhanced chain-of-thought framework," in *Proc. ACL*, 2023.

[22] Ibn Sina (Avicenna), *Al-Shifa', Part I: Logic*, discussed in D. Gutas, *Avicenna and the Aristotelian Tradition*. Leiden: Brill, 2nd ed., 2014.

[23] A. Hagberg, P. Swart, and D. Chult, "Exploring network structure, dynamics, and function using NetworkX," in *Proc. SciPy*, 2008.

[24] S. Lin, J. Hilton, and O. Evans, "TruthfulQA: Measuring how models mimic human falsehoods," in *Proc. ACL*, 2022, pp. 3214–3252.

[25] R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge," in *Proc. AAAI*, vol. 31, 2017.

[26] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[27] H. Touvron *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv:2307.09288*, 2023.

[28] A. Q. Jiang *et al.*, "Mistral 7B," *arXiv:2310.06825*, 2023.

[29] O. Tafjord, B. D. Mishra, and P. Clark, "ProofWriter: Generating implications, proofs, and abductive statements over natural language," in *Proc. ACL-IJCNLP Findings*, 2021, pp. 3621–3634.

[30] *Moffatt v. Air Canada*, 2024 BCCRT 149 (British Columbia Civil Resolution Tribunal, Feb. 2024).