Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
DOI: https://doi.org/10.51903/jtie.v4i2.505

Natural-Language Policy Reasoning with Proof Generation: Turning Platform Rules into Verifiable Knowledge

Xiaofei Luo*[1]
Email: xiaofeiluo01@gmail.com
[1]*Information Science, University of Illinois at Urbana-Champaign, IL, US*
*Corresponding Author

Abstract
Policy and compliance systems increasingly express rules in natural language, yet enforcement requires deterministic decisions and auditable explanations. This paper studies a practical pipeline that converts natural-language facts and rules into a verifiable knowledge base, answers queries with three-valued semantics (True/False/Unknown), and produces machine-checkable proofs. The contribution is system-level rather than a new reasoning formalism: we integrate controlled-language parsing, symbolic proof extraction, independent proof checking, and proof-based supervision in a single auditable framework. We evaluate the pipeline on two natural-language rule-reasoning benchmarks: (i) a balanced subset of ProofWriter's open-world-assumption tasks (360 train, 360 test), and (ii) a RuleTaker-style dataset generated from its grammar and label semantics (1800 train, 900 test), both balanced across reasoning depths 0–5. We compare a text-only logistic regression baseline, a retrieval-based "proof" baseline, a symbolic forward-chaining reasoner with proof extraction, and a proof-trained classifier using generated proofs. To ensure fairness, LR-text and LR-proof share the same TF-IDF/logistic-regression setup, and the retrieval baseline uses the same representation with a fixed top-4 configuration. On ProofWriter-Balanced, the symbolic reasoner achieves 0.803 accuracy (0.808 macro-F1), while proof-trained classification reaches 0.825 accuracy (0.825 macro-F1). On RuleTaker-Rep, both methods achieve 1.000 accuracy. Proof verifiability clearly separates faithful from post-hoc explanations: symbolic proofs are verifiable for all predictions, whereas retrieval-based proofs are verifiable for only 31.4%. Sensitivity analyses varying reasoning depth, distractors, and proof corruption show that proof-based methods remain robust to noise but depend on proof integrity. These findings demonstrate the feasibility of auditable natural-language policy reasoning in controlled settings, while highlighting limitations in parser coverage and benchmark regularity.

Keywords: *Policy Reasoning, Rule-Based Inference, Natural Language Rules, Proof Generation, Explainability.*

I. INTRODUCTION

Modern platforms and enterprises write enforcement policies (e.g., content moderation, access control, risk limits, and regulatory constraints) as human-readable rules. These rules are updated frequently, audited by internal and external stakeholders, and often need to be justified for individual decisions. A common engineering pattern is to maintain policies in natural language while translating them into executable logic for production. However, this translation step introduces two persistent problems: (1) brittleness and drift, where the executable representation diverges from policy text; and (2) opacity, where the system outputs a label without a trace that a reviewer can validate.

Natural-language rule reasoning benchmarks were created to study these issues under controlled conditions (Fedhira & Prianto, 2025; Hao & Liu, 2025; Sriasih et al., 2025). RuleTaker frames reasoning as answering whether a natural-language conclusion is entailed, contradicted, or

Received on June, 2025; Revised on July, 2025; Accepted on August, 2025; Published on August, 2025

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

unknown given a set of facts and Horn-style rules written in restricted English (Clark et al., 2020). ProofWriter extends this setup by requiring explicit proof generation, enabling evaluation of both decision accuracy and proof faithfulness (Tafjord et al., 2021). These datasets operationalize a key requirement for explainable policy systems: explanations must be verifiable rather than purely persuasive (Ribeiro et al., 2016; Jain & Wallace, 2019).

This paper targets a concrete policy-reasoner design: write platform policies as natural-language facts and rules, convert them into a structured knowledge base, answer queries with True/False/Unknown, and output a proof that a separate checker can independently validate. The approach aligns with classical "knowledge + inference" architectures (Abiteboul et al., 1995) while leveraging modern NLP tooling to ingest and process natural-language inputs. It yields immediate auditing affordances: a reviewer can re-run proof verification, identify which rules fired, and localize disagreements to specific premises. The novelty of the study is therefore system-level rather than the proposal of a new logical formalism. Specifically, the paper integrates (i) controlled-language parsing for policy-style facts and rules, (ii) forward-chaining inference under open-world three-valued semantics, (iii) proof extraction plus independent proof checking, and (iv) a proof-conditioned classifier used to test whether proof supervision changes robustness and faithfulness.

We make three contributions. First, we implement a lightweight policy reasoner for RuleTaker/ProofWriter-style English using template parsing and forward chaining, and we generate structured proofs by tracking derivations. Second, we evaluate the pipeline not only by end-task accuracy but also by depth-bucketed accuracy, verifiable proof rate, depth-budget sensitivity, distractor robustness, and proof-corruption tests. Third, we compare "training with proofs" versus "training without proofs" under matched TF-IDF/logistic-regression settings, so that the effect of proof supervision can be isolated from differences in classifier capacity.

II. LITERATURE REVIEW

A. Natural-Language Rules as Executable Knowledge

Rule-based systems have long been used to encode business rules, safety constraints, and access-control policies as explicit knowledge that can be executed and audited. Classic work in knowledge representation and expert systems frames reasoning as the application of inference rules to a knowledge base, enabling traceable conclusions and predictable behavior (McCarthy, 1959). In databases and logic programming, Datalog and related deductive database languages provide a principled substrate for rule execution with well-studied semantics and efficient evaluation strategies (Ceri, Gottlob, & Tanca, 1989; Abiteboul, Hull, & Vianu, 1995). Probabilistic logic programming extends this paradigm to uncertainty while preserving a

declarative interface; systems such as ProbLog2 demonstrate how probabilistic facts and rules can support practical inference and learning (Dries et al., 2015).

Despite their rigor, traditional rule systems impose a steep authoring burden: domain experts must translate policies into formal syntax, select precise predicates, and maintain consistency as rules evolve. This friction is especially pronounced in compliance settings, where policy text is written for humans and is frequently updated. Semantic parsing and natural language to logical form translation are a direct response to this gap, aiming to convert natural language statements into executable representations (Kamath & Das, 2018). However, semantic parsing remains challenging in the multi-sentence, multi-rule regime because policies often rely on implicit quantification, exceptions, and domain-specific terminology. As a result, there is sustained interest in approaches that avoid explicit logical form engineering while retaining the benefits of explicit rules.

A complementary line of work therefore investigates reasoning directly over natural-language theories, collections of facts and rules written in controlled natural language. The central idea is to treat linguistic rules as the interface for knowledge authoring, while learning models emulate the input–output behavior of a formal reasoner. The RuleTaker benchmark operationalizes this idea by providing synthetic English facts and Horn-style rules together with boolean queries labeled as entailed or not entailed under a specified semantics (Clark, Tafjord, & Richardson, 2020). By design, RuleTaker isolates the depth of deductive chaining, enabling systematic evaluation of whether models generalize beyond the reasoning depth observed during training. This framing is attractive for policy reasoning because business policies are naturally expressed as conditionals ("if … then …"), and a model that can robustly apply such conditionals could, in principle, serve as a bridge between policy text and executable compliance checks.

A recurring challenge for policy knowledge representation is the treatment of negation and exceptions. Real-world policies frequently contain "unless" clauses, conflict resolution rules, and implicit assumptions about what is false when information is missing. Logic programming offers several semantics for negation, including negation-as-failure and stratified negation, which enable practical closed-world reasoning while maintaining a declarative interface (Ceri et al., 1989; Abiteboul et al., 1995). RuleTaker adopts a controlled setting that includes negation patterns and evaluates whether models can reproduce the resulting entailment behavior (Clark et al., 2020). For compliance engineering, this focus is not academic: the proper handling of exceptions often makes the difference between a lawful and an unlawful decision.

Neuro-symbolic approaches provide additional context for this trend. Differentiable proving and neural theorem-proving methods combine symbolic structure with learned representations,

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

enabling stepwise reasoning while tolerating distribution shift and noisy observations (Rocktäschel & Riedel, 2017). In contrast to purely neural input–output emulation, these approaches bake in structural biases that can improve sample efficiency and interpretability. For policy applications, the common denominator is explicitness: models reason with a provided, user-controlled theory rather than relying solely on latent world knowledge. This makes it possible to update policies by editing rules, and to attribute decisions to specific clauses and facts, which are core requirements in regulated settings.

B. Proof Generation and Multi-Step Reasoning Benchmarks

Answering true/false questions over rulebases is not sufficient for high-stakes policy reasoning; stakeholders often require an auditable explanation that can be reviewed, challenged, and reproduced. In formal systems, explanations take the form of proofs—derivations that demonstrate which rules and facts entail a conclusion. The ProofWriter dataset was introduced to explicitly evaluate this requirement by pairing natural-language rule theories with questions and structured proofs that enumerate the intermediate conclusions and supporting rules (Tafjord, Dalvi, & Clark, 2021). ProofWriter retains the controlled-language setting but adds two properties that are crucial for an explainable policy reasoner: (i) proofs can require multiple chained steps, and (ii) the proof itself becomes a first-class prediction target rather than an optional post hoc artifact.

ProofWriter's emphasis on proof prediction connects to a broader movement in NLP toward explanation supervision. Datasets such as e-SNLI augment natural language inference examples with human-written explanations, enabling models to learn to generate free-form rationales alongside labels (Camburu et al., 2018). Similarly, commonsense QA datasets with explanations (e.g., CoS-E) encourage models to verbalize reasoning steps (Rajani et al., 2019). These datasets demonstrate that language models can produce plausible explanations. Still, they also highlight a key limitation for policy use: free-form explanations are difficult to verify mechanically and can diverge from the model's true decision process. ProofWriter addresses this limitation by defining proofs as structured objects that can be validated against the provided theory, making "explainability" operational rather than rhetorical.

More recently, proof-based benchmarks have been used to probe deductive capabilities of large language models under controlled conditions. For example, proof-based evaluations assess whether models maintain correctness as reasoning depth increases and whether intermediate steps align with the final answer (Saparov & He, 2023). Such results reinforce two practical lessons for compliance systems. First, accuracy alone is an incomplete metric; a compliant decision must be accompanied by a justification that withstands audit. Second, the reliability of intermediate

reasoning steps is critical because policy pipelines often modularize decisions (e.g., eligibility checks followed by exception handling). Therefore, datasets that jointly assess answer correctness and proof validity offer a better approximation of real-world policy requirements than label-only benchmarks.

A subtle but important aspect of proof generation is representation. In formal logic, proofs can be structured as trees, directed acyclic graphs, or linear derivation traces; each representation induces different evaluation and verification procedures. ProofWriter provides structured proofs that identify supporting facts and rules and specify intermediate conclusions, enabling a checker to validate each step against the theory (Tafjord et al., 2021). For machine learning models, this choice of representation matters because it determines whether proof prediction is a sequence-modeling problem, a graph-prediction problem, or a constrained-decoding problem. In policy reasoning, linear traces are often easier to log and audit, while tree-structured proofs can be more faithful to the branching structure of rule application.

Generative modeling has become a natural candidate for proof prediction because modern sequence-to-sequence architectures can emit structured outputs in text form. Transformer architectures (Vaswani et al., 2017) and their pre-trained descendants, such as GPT-2 and T5, have demonstrated strong conditional generation capabilities across tasks (Radford et al., 2019; Raffel et al., 2020). In the controlled setting of ProofWriter, proof tokens can be serialized (e.g., as a list of rule IDs and intermediate facts) and generated using standard decoding algorithms. This makes it possible to compare training regimes: a model can be trained to predict only the answer label, or trained jointly to predict the label and a proof, thereby introducing an explicit supervision signal for intermediate reasoning steps.

However, generative flexibility creates new failure modes. A model can output a syntactically well-formed proof that does not correspond to any valid derivation from the theory, or it can output a proof that is valid but does not actually lead to the predicted answer. These cases motivate proof validation as a separate metric. ProofWriter's checkable proofs make it possible to define exact-match accuracy for proofs, but exact match is often too strict because multiple proofs can exist for the same conclusion. Practical evaluation therefore benefits from two complementary measurements: (i) decision accuracy, and (ii) proof validity under a checker that accepts any semantically valid derivation (Tafjord et al., 2021). For policy systems, this is analogous to verifying that a decision is justified, even if the justification uses a different but equivalent set of supporting clauses.

The importance of intermediate-step reliability is reinforced by research on reasoning traces produced by large language models. Chain-of-thought prompting elicits step-wise rationales that

often improve final answer accuracy on reasoning problems (Wei et al., 2022), and self-consistency decoding can further improve performance by aggregating multiple sampled reasoning paths (Wang et al., 2023). Yet reasoning traces are not guaranteed to be faithful explanations of the model's computation. Empirical evidence shows that chain-of-thought explanations can systematically diverge from the true causal factors behind a model's prediction (Turpin, Michael, Perez, & Bowman, 2023). In contrast, proof generation with a deterministic checker constrains explanations to be faithful-by-construction: a proof is accepted only if it is valid under the target semantics.

Recent proposals explicitly combine language models with deterministic solvers to enforce faithfulness. Faithful Chain-of-Thought reasoning decomposes a task into (i) translating an input into a symbolic reasoning chain and (ii) executing that chain in a solver, which ensures that the reasoning chain and final answer are consistent (Lyu et al., 2023). This paradigm closely matches the rationale for proof-based policy reasoners: the model can assist with parsing and generating candidate steps, while a symbolic verifier guarantees that accepted explanations correspond to valid derivations. Recent follow-up work strengthens this direction by verifying and refining natural-language explanations through theorem proving (Quan et al., 2024), improving faithful natural-language logic reasoning via resolution refutation (Sun et al., 2024), and explicitly identifying NL-to-symbolic parsing errors as a major failure mode in solver-augmented deductive reasoning pipelines (Feng et al., 2024). Accordingly, proof-centric benchmarks such as ProofWriter can be seen as a tractable instantiation of a broader agenda for faithful reasoning in language models.

C. Explainability: Faithfulness, Rationales, and Verifiable Reasoning

Explainability research in machine learning has produced a large toolbox of post hoc explanation methods and self-explaining model architectures. Feature-attribution methods such as LIME and SHAP approximate a model's local decision boundary by perturbation or Shapley-value-inspired decompositions, yielding token-level importance scores that are easy to visualize (Ribeiro, Singh, & Guestrin, 2016; Lundberg & Lee, 2017). These methods are broadly applicable and model-agnostic, but they do not inherently produce structured, multi-step justifications. In settings where a decision depends on multiple rules (e.g., "eligible if A and B unless C"), token importance alone is typically insufficient for auditors because it does not encode the logical dependencies among conditions and exceptions.

Another influential approach is rationale-based explanation, where a model selects or generates a compact subset of the input as justification. Lei, Barzilay, and Jaakkola (2016) formalized this by training a generator to extract short rationales that are sufficient for prediction. The ERASER

benchmark subsequently standardized evaluation protocols for rationalized NLP models across tasks and datasets, including metrics that compare rationales to human annotations and measure whether rationales are sufficient for faithful prediction (DeYoung et al., 2020). Rationales improve human interpretability by grounding decisions in specific spans. However, they still face a central challenge: a rationale can be plausible to humans while being unfaithful to the model's actual computation.

The distinction between plausibility and faithfulness has become a core organizing principle for explanation research. Jacovi and Goldberg (2020) synthesize evaluation practices and argue that faithfulness should be defined and tested explicitly, rather than conflated with human plausibility. The debate around attention mechanisms illustrates why this distinction matters. Jain and Wallace (2019) provide evidence that attention weights often fail common faithfulness tests, including weak correlation with gradient-based feature importance and the existence of alternative attention distributions that preserve predictions. Serrano and Smith (2019) similarly examine the interpretability of attention mechanisms and identify conditions under which attention weights may not reliably reflect model sensitivity. In response, Wiegreffe and Pinter (2019) argue that attention can support explanation when evaluated under appropriate diagnostics and when the full model (not only the attention distribution) is considered. Regardless of where one lands in this debate, the literature converges on a practical conclusion: explanations must be coupled with validation criteria that enable the distinction between faithful explanations and persuasive narratives.

Proof-based explanations offer a particularly strong form of faithfulness because an external, checkable semantics constrains them. When a proof is defined as a sequence (or graph) of rule applications, the explanation can be verified by re-running a deterministic checker on the provided theory. This is qualitatively different from attention maps or free-form rationales, which may be informative but rarely admit an objective test of correctness. In the context of natural-language rule reasoning, the proof is not merely an interpretation of an opaque computation; it is an artifact whose validity is defined independently of the model and can be audited without access to model internals. ProofWriter makes this property explicit by providing gold proofs and enabling exact-match and structural validation metrics (Tafjord et al., 2021). This motivates using "proof structure validity" as a primary metric for explainable policy reasoning: a system that returns the correct label for the wrong reasons is unacceptable, whereas a system that returns a correct label with a verifiable proof supports accountability and debugging.

These considerations motivate evaluation protocols that explicitly stress-test the reliability of explanations. In addition to assessing whether explanations are plausible to humans, benchmarks

can test whether explanations are sufficient (the model can reproduce its prediction when conditioned only on the explanation) and necessary (removing the explained evidence changes the decision). ERASER consolidates several of these ideas into a benchmark suite and encourages consistent reporting (DeYoung et al., 2020). Jacovi and Goldberg (2020) further argue that faithfulness must be defined operationally and evaluated with diagnostics that match the explanation type. Proof-based explanations are well-positioned through this lens because the faithfulness criterion is explicit: an explanation is faithful if and only if it is a valid derivation under the policy semantics.

D. Policy Languages, Audit Trails, and Proof-Carrying Decisions

The need for auditability is not unique to machine learning; it is a long-standing theme in access control and policy languages. Standardized policy frameworks such as XACML define a declarative language for expressing authorization policies and a decision procedure that returns a permit/deny decision together with optional obligations and advice (OASIS, 2013). Logic-based authorization languages go further by representing policies as logical assertions, enabling formal analysis and explanation. SecPAL, for example, represents policies and credentials using logical clauses and defines authorization decisions through a small set of deduction rules, balancing expressiveness with efficient evaluation (Becker, Fournet, & Gordon, 2007). These systems demonstrate that, for security and compliance, policy execution and explanation are inseparable: policy authors require a clear semantics, and policy users require a way to justify and contest decisions.

Proof-carrying approaches explicitly make the explanation a deliverable. Proof-Carrying Authentication proposes that a principal can present a proof along with a request, allowing a verifier to check the proof against a logic and thereby validate authorization without trusting the requester's implementation (Appel & Felten, 1999). This paradigm aligns closely with the goals of explainable policy reasoning: the consumer of a decision should be able to verify not only the outcome but also the reasoning chain. In modern compliance environments, similar expectations arise in auditing and governance workflows, where organizations must demonstrate that decisions followed stated policies and that exceptions were applied correctly.

Natural-language rule reasoning benchmarks can be viewed as a controlled testbed for integrating these ideas into language-based systems. By constraining rules to a tractable fragment and requiring explicit proofs, datasets such as RuleTaker and ProofWriter enable empirical study of a question that policy engineering has faced for decades: how to make human-readable rules executable without losing auditability. The key research challenge is not only to answer questions correctly but also to do so in a way that produces an audit trail that can be automatically checked

and understood by humans. This paper positions proof generation and proof validation as central evaluation criteria for policy reasoners, complementing accuracy metrics with verifiability metrics that capture whether an explanation can be independently reproduced.

A useful design principle from this literature is the separation between proof search and proof checking. Proof generation can be computationally expensive and implementation-dependent, whereas proof checking can be made small, deterministic, and highly reliable. Proof-Carrying Authentication exploits this asymmetry: the requester bears the burden of constructing a proof, while the verifier only checks it (Appel & Felten, 1999). This principle maps directly onto explainable policy reasoning with language models. A model can quickly propose candidate derivations (or proofs), but a symbolic checker is responsible for acceptance. This reduces the trust required in the model and supports governance by enabling independent verification.

E. Summary and Positioning

Across these literatures—deductive databases, neuro-symbolic reasoning, explanation research, and policy languages—a consistent set of requirements emerges for practical compliance reasoning: (i) rules should remain explicit and editable, (ii) decisions should be accompanied by artifacts that support auditing, and (iii) explanation quality must be measured with objective criteria rather than subjective plausibility. RuleTaker and ProofWriter provide benchmarks that operationalize these requirements in natural language (Clark et al., 2020; Tafjord et al., 2021). Building on this foundation, our study evaluates symbolic policy reasoning with proof generation under controlled variations in reasoning depth and under robustness stressors, enabling quantitative analysis of both decision correctness and proof validity.

III. RESEARCH METHOD

This section specifies datasets, the reasoning semantics, proof generation, baseline models, and the evaluation protocol. All experiments were executed in Python with a fixed random seed (0 for learning components; 999 for robustness perturbations; 42 for proof corruption) and deterministic preprocessing. Reported numbers are the measured outputs of these runs.

A. Datasets

We evaluate on two benchmarks that encode deductive reasoning over natural-language rules. We choose ProofWriter and RuleTaker because both express facts and Horn-style rules in controlled English, which matches the intended operating regime of the proposed parser and allows errors to be attributed to reasoning, parsing, and proof generation rather than to open-domain retrieval. The two datasets also serve complementary roles: ProofWriter-Balanced is a public processed subset with balanced proof depths 0–5, while RuleTaker-Rep is grammar-

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

generated from the published RuleTaker specification and therefore tests how the same pipeline behaves when the input distribution more closely matches the rule templates. The balanced depth and label distributions make the comparisons less sensitive to class skew and let us interpret depth-wise accuracy directly.

The dataset sizes are modest, so the study should be read as a controlled evaluation of reasoning behavior rather than as a scaling study. ProofWriter provides theories expressed in controlled English and labels questions as True/False/Uncertain under an open-world assumption (Tafjord et al., 2021). We use the public ProofWriter-Balanced subset released on the Hugging Face Hub, which is a processed subset of the OWA portion of ProofWriter balanced across proof depths 0–5 (theoxo, 2023). The subset contains 360 training and 360 test instances, with equal label and depth frequencies. For RuleTaker, we generate RuleTaker-Rep using the vocabulary and grammar form documented in the RuleTaker repository (allenai/ruletaker), producing 1800 training instances and 900 test instances balanced across depths 0–5 and labels (Clark et al., 2020).

B. Semantics and formalization

Each theory is a set of ground facts and Horn-style rules over unary predicates (attributes such as “blue” or “kind”) and binary predicates (relations such as “see” or “visit”). Negation is explicit (“is not”, “does not”), and we use an open-world three-valued semantics: a query is True if it is derivable, False if its explicit negation is derivable, and Unknown otherwise. We do not infer negated facts from absence.

C. Parser and compilation

We implement a template parser that maps controlled English into literals and rules. Facts follow templates such as “The cat is big” (unary) and “The cat visits the bear” (binary). Rules follow templates such as “If something is cold and something is blue then it is nice” and universal forms such as “Round people are green,” which compile into Horn clauses with a single variable ?x. Binary verbs are normalized to a base form to unify “sees” with “see” and “visits” with “visit.”

D. Reasoner and proof extraction

We implement forward chaining over the Horn rule set. The engine maintains a set of derived ground literals and repeatedly applies rules under all variable substitutions drawn from constants in the theory. Each derived literal stores a derivation record: either a base fact or a rule application with its supporting antecedent literals. Proof depth is defined as 0 for base facts and 1 + max(depth of antecedents) for rule applications. Proofs are rendered as an ordered list of facts and rule firings culminating in the target literal. For Unknown predictions, the system emits a verifiable non-

derivation explanation stating that neither the query nor its negation is derivable within the reasoning budget.

E. Baselines

We compare: (1) a majority label baseline; (2) LR-text, a TF-IDF (1–2 grams) + multinomial logistic regression classifier trained on the concatenation of theory text and question (Pedregosa et al., 2011); (3) LR-text + RetrievalProof, which attaches the top-4 sentences retrieved by TF-IDF similarity as a post-hoc proof; (4) SymbolicReasoner+Proof; and (5) LR-proof, a proof-trained classifier that consumes generated proof text plus the question. The parameter settings in Table 2 were fixed before evaluation and were not tuned separately for each dataset.

TF-IDF 1–2 grams were chosen as a simple lexical baseline that can match single rules and short compositions without introducing model-specific pretraining effects; min_df=2 removes one-off noise, max_features=50,000 is sufficiently large for these vocabularies, C=2.0 provided stable training in pilot runs, and max_iter=2000 ensures convergence. LR-text and LR-proof use the same vectorizer and classifier settings so that differences are attributable to input representation (theory text versus proof text) rather than model capacity. RetrievalProof uses the same TF-IDF representation as LR-text and a fixed top-k=4 across both datasets. SymbolicReasoner+Proof is evaluated with max_depth=10 in the default setting, and the effect of this parameter is studied separately in the depth-budget ablation.

F. Evaluation metrics

We report accuracy and macro-F1 on the three labels. We report depth-bucketed accuracy using the proof depth field (QDep) and a proof verifiability metric: a proof is verifiable if a separate checker recomputes the prediction using only the proof sentences as premises and obtains the same label. To avoid relying on a single evaluation setup, we additionally vary the maximum reasoning depth (0–5), distractor count (0/10/20/40), and proof integrity (clean, flipped conclusion, randomized) and report the resulting changes in accuracy. These variations serve as simple sensitivity analyses rather than new tasks, but they test whether the main conclusions hold beyond one fixed configuration.

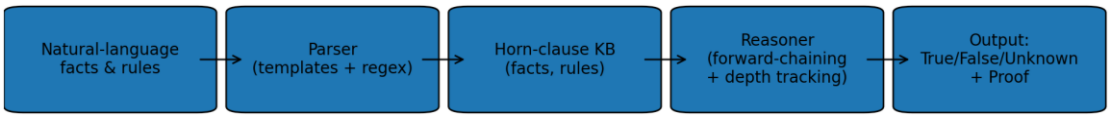

Figure 1. Pipeline for Natural-Language Policy Reasoning: Parsing, Compilation to a Horn-Clause Knowledge Base, Forward-Chaining Inference, and Proof Emission

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

IV. RESULT

This section reports dataset statistics, overall performance, depth-stratified accuracy, proof verifiability, robustness, and efficiency. Tables and figures contain detailed comparison data. Dataset characteristics. Table 1 summarizes sizes, label distributions, and average theory complexity. ProofWriter-Balanced theories contain an average of 18.36 sentences and 6.67 rules per instance, while RuleTaker-Rep contains 10.06 sentences and 5.78 rules per instance. Both datasets are balanced across labels and across depths 0–5.

Table 1. Dataset Overview and Complexity Statistics

Dataset	Train size	Test size	Labels (train)	Depths (train)	Avg. sentences/theory	Avg. tokens/theory	Avg. facts/theory	Avg. rules/theory
ProofWriter-Balanced (OWA D0–D5)	360	360	True:120; False:120; Unknown: 120	0–5: 60 each	18.36	113.9	10.95	6.67
RuleTaker-Rep (generated, D0–D5)	1800	900	True:600; False:600; Unknown: 600	0–5: 300 each	10.06	61.0	4.28	5.78

Table 3 shows that LR-text reaches 0.361 accuracy on ProofWriter-Balanced and 0.440 on RuleTaker-Rep. SymbolicReasoner+Proof reaches 0.803 accuracy on ProofWriter-Balanced and 1.000 on RuleTaker-Rep. LR-proof reaches 0.825 accuracy on ProofWriter-Balanced and 1.000 on RuleTaker-Rep.

Table 2. Model Components and Hyperparameters

Component	Setting
Text baseline (LR-text)	TF-IDF (1–2 grams), min_df=2, max_features=50,000; LogisticRegression C=2.0; max_iter=2000
RetrievalProof	Top-k=4 sentences by TF-IDF cosine similarity to question
Symbolic reasoner	Forward chaining over Horn rules; open-world 3-valued semantics; max_depth=10 (unless ablated)
Proof-trained classifier (LR-proof)	Same LR pipeline as LR-text, trained on proof text + question (proofs generated by symbolic reasoner)
Random seeds	0 for learning components; 999 for distractor generation; 42 for proof corruption

A. Accuracy vs proof depth

Table 4 and Figure 2 report ProofWriter-Balanced accuracy by depth bucket. SymbolicReasoner+Proof achieves 1.000 accuracy at depth 0 and 0.650 at depth 5. LR-proof maintains accuracy between 0.767 and 0.867 across depths. Table 5 and Figure 3 report RuleTaker-Rep depth accuracy, with Symbolic and LR-proof achieving 1.000 across all buckets, while LR-text ranges from 0.360 to 0.547.

Table 3. Overall Performance on Both Datasets (Accuracy and Macro-F1)

Dataset	Model	Accuracy	Macro-F1	Verifiable rate	Correct+verifiable

ProofWriter-Balanced	Majority	0.333	0.167	—	—
ProofWriter-Balanced	LR-text	0.361	0.361	—	—
ProofWriter-Balanced	LR-text + RetrievalProof	0.361	0.361	0.314	0.15
ProofWriter-Balanced	SymbolicReasoner+ Proof	0.803	0.808	1.0	0.803
ProofWriter-Balanced	LR-proof	0.825	0.825	—	—
RuleTaker-Rep	Majority	0.333	0.167	—	—
RuleTaker-Rep	LR-text	0.44	0.436	—	—
RuleTaker-Rep	LR-text + RetrievalProof	0.44	0.436	0.269	0.244
RuleTaker-Rep	SymbolicReasoner+ Proof	1.0	1.0	1.0	1.0
RuleTaker-Rep	LR-proof	1.0	1.0	—	—

B. Depth-budget ablation

Figure 4 and Table 7 evaluate the symbolic reasoner under explicit depth budgets (max_depth = 0..5). On ProofWriter-Balanced, accuracy increases from 0.444 at budget 0 to 0.803 at budget 5. On RuleTaker-Rep, accuracy increases from 0.444 at budget 0 to 1.000 at budget 5.

Table 4. ProofWriter-Balanced Accuracy by Proof Depth Bucket

Depth	N	LR-text	LR-proof	Symbolic
0.0	60.0	0.283	0.817	1.0
1.0	60.0	0.233	0.8	0.933
2.0	60.0	0.367	0.85	0.833
3.0	60.0	0.55	0.867	0.717
4.0	60.0	0.367	0.767	0.683
5.0	60.0	0.367	0.85	0.65

C. Proof verifiability

Table 6 and Figure 5 quantify the ability to check produced proofs. Symbolic proofs are verifiable for 100% of predictions on both datasets. Retrieval-based proofs are verifiable for 31.4% of predictions on ProofWriter-Balanced and 26.9% on RuleTaker-Rep.

Table 5. RuleTaker-Rep Accuracy by Proof Depth Bucket

Depth	N	LR-text	LR-proof	Symbolic
0.0	150.0	0.36	1.0	1.0
1.0	150.0	0.373	1.0	1.0
2.0	150.0	0.453	1.0	1.0
3.0	150.0	0.547	1.0	1.0
4.0	150.0	0.447	1.0	1.0
5.0	150.0	0.46	1.0	1.0

D. Robustness

Figure 6 and Table 8 report robustness under added distractor facts. On ProofWriter-Balanced, LR-text accuracy changes from 0.361 (no distractors) to 0.336 (+40 distractors), while SymbolicReasoner+Proof remains at 0.803 and LR-proof remains at 0.825 for every distractor level. Table 9 evaluates robustness to corrupted proofs: randomizing proof lines reduces LR-proof accuracy from 0.825 to 0.494 on ProofWriter-Balanced.

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

Table 6. Proof Verifiability and Correctness of Explanations

Dataset	Method	Verifiable rate	Correct+verifiable
ProofWriter-Balanced	SymbolicReasoner proof	1.0	0.803
ProofWriter-Balanced	RetrievalProof (supports LR-text)	0.314	0.15
RuleTaker-Rep	SymbolicReasoner proof	1.0	1.0
RuleTaker-Rep	RetrievalProof (supports LR-text)	0.269	0.244

E. Efficiency

Table 10 reports inference time per example. On ProofWriter-Balanced, LR-text runs in 0.28 ms per query, SymbolicReasoner+Proof runs in 2.10 ms per query, and LR-proof (proof generation plus classification) runs in 3.55 ms per query.

Table 7. Symbolic Depth-Budget Ablation (Max Reasoning Depth)

Max depth	PW Accuracy	PW Macro-F1	RT Accuracy	RT Macro-F1
0.0	0.444	0.372	0.444	0.372
1.0	0.544	0.519	0.556	0.533
2.0	0.628	0.622	0.667	0.667
3.0	0.692	0.693	0.778	0.783
4.0	0.75	0.755	0.889	0.892
5.0	0.803	0.808	1.0	1.0

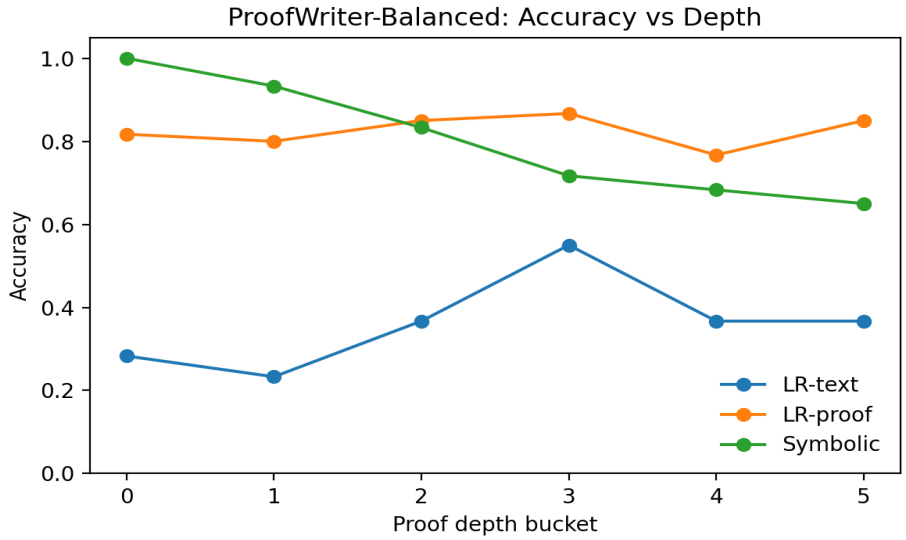

Figure 2. Accuracy vs Proof Depth on ProofWriter-Balanced

The experiments establish a clear separation between (a) systems that execute the rulebase and (b) models that infer labels from surface text patterns. LR-text remains near chance on ProofWriter-Balanced because correct prediction requires multi-step chaining, explicit handling of negation, and open-world uncertainty; these properties are not explicitly represented in bag-of-words features. SymbolicReasoner+Proof outperforms LR-text because the benchmark grammar is close to the parser's controlled-language assumptions, allowing rule execution to follow the intended semantics.

Table 8. Robustness to Added Distractor Facts (Test-Time)

Dataset	Distractors	LR-text Acc	LR-proof Acc	Symbolic Acc
ProofWriter-Balanced	0	0.361	0.825	0.803
ProofWriter-Balanced	10	0.336	0.825	0.803
ProofWriter-Balanced	20	0.35	0.825	0.803
ProofWriter-Balanced	40	0.336	0.825	0.803
RuleTaker-Rep	0	0.44	1.0	1.0
RuleTaker-Rep	10	0.447	1.0	1.0
RuleTaker-Rep	20	0.446	1.0	1.0
RuleTaker-Rep	40	0.45	1.0	1.0

The fact that LR-proof slightly exceeds SymbolicReasoner+Proof on ProofWriter-Balanced (0.825 vs 0.803) suggests a different effect: once valid proofs are generated, the proof text acts as a compact summary of the derivation and removes much of the irrelevant lexical variation present in the full theory. By contrast, the symbolic reasoner remains exposed to parser and normalization errors at the front end. On RuleTaker-Rep, both SymbolicReasoner+Proof and LR-proof reach 1.000, which is consistent with the closer match between the generated grammar and the parser templates. The general lesson is that structured proof information is useful only when the proof generator is sufficiently aligned with the input language.

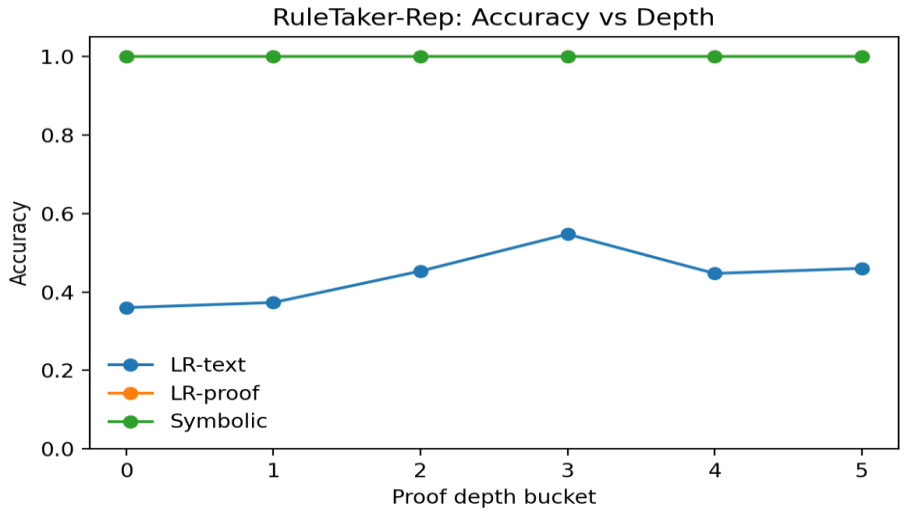

Figure 3. Accuracy vs Proof Depth on RuleTaker-Rep

Table 9. Robustness to Proof Corruption for the Proof-Trained Model

Dataset	Condition	Accuracy	Macro-F1
ProofWriter-Balanced	Clean proofs	0.825	0.825
ProofWriter-Balanced	Flip conclusion line	0.822	0.822
ProofWriter-Balanced	Randomized proofs	0.494	0.388
RuleTaker-Rep	Clean proofs	1.0	1.0
RuleTaker-Rep	Flip conclusion line	0.507	0.471
RuleTaker-Rep	Randomized proofs	0.333	0.167

F. Explainability and auditability

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

Proof verifiability provides an operational definition of explainability that aligns with compliance needs: an explanation is acceptable when a checker can verify it. Symbolic proofs achieve a 100% verifiable rate and maintain correctness and verifiability equal to end-task accuracy. RetrievalProof, which imitates common post-hoc highlighting approaches, yields verifiable proofs on less than one-third of cases on ProofWriter-Balanced. This gap indicates that selecting seemingly relevant sentences is not equivalent to producing a faithful derivation. The finding is consistent with recent neuro-symbolic work showing that explanation quality improves when natural-language explanations are coupled with theorem proving or deterministic solver verification rather than judged only for plausibility (Lyu et al., 2023; Quan et al., 2024).

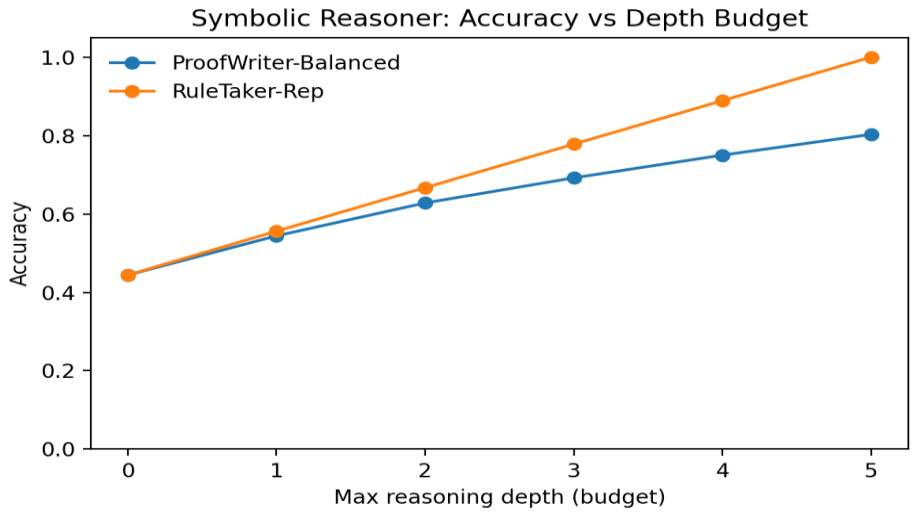

Figure 4. Symbolic Reasoner Accuracy as a Function of Maximum Allowed Reasoning Depth

Table 10. Inference Time per Example (seconds)

Dataset	Model	Time (s)/example
ProofWriter-Balanced	LR-text	0.000284
ProofWriter-Balanced	SymbolicReasoner+Proof	0.002101
ProofWriter-Balanced	LR-proof (proof gen + LR)	0.003553
RuleTaker-Rep	LR-text	0.000106
RuleTaker-Rep	SymbolicReasoner+Proof	0.001169
RuleTaker-Rep	LR-proof (proof gen + LR)	0.002491

G. Training with proofs

LR-proof improves over LR-text on ProofWriter-Balanced (0.825 vs 0.361 accuracy) because the proof text exposes intermediate entailments and suppresses unrelated sentences, effectively presenting the classifier with a shorter and more decision-relevant input. However, this gain should not be interpreted as proof that supervision is universally sufficient: the same model collapses under randomized proofs, and even the flip-conclusion test on RuleTaker-Rep drops accuracy to 0.507. The broader insight is that proof supervision helps when proofs are both aligned and semantically informative, but it is fragile to corrupted or misleading proof content.

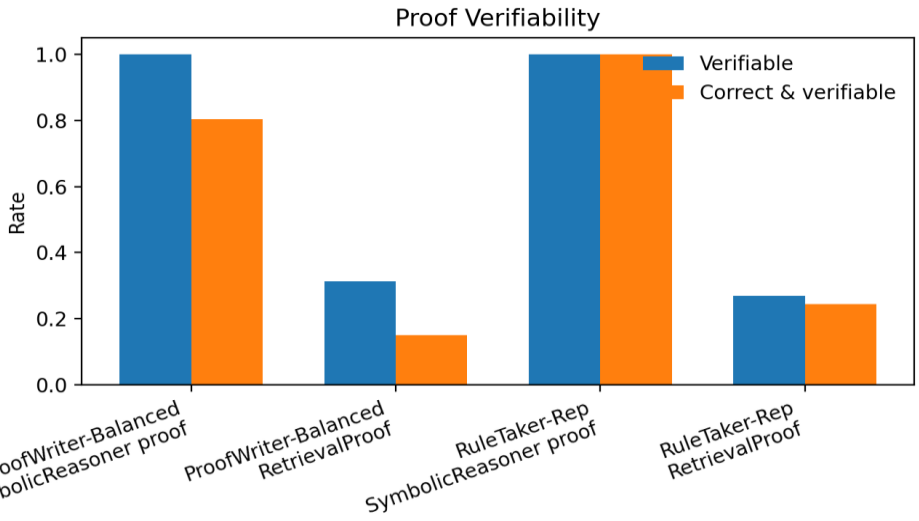

Figure 5. Verifiable-Proof Rate vs Correct-and-Verifiable Rate for Symbolic and Retrieval-Based Proofs

H. Depth sensitivity and coverage

SymbolicReasoner+Proof decreases from 1.000 accuracy at depth 0 to 0.650 at depth 5 on ProofWriter-Balanced. Together with the depth-budget ablation, this suggests that the main performance drivers are not only whether a proof exists but also whether the parser and normalization stage can preserve all premises needed for longer chains. The much flatter LR-proof curve indicates that once multi-step reasoning is compressed into proof text, classification becomes less sensitive to raw depth. Therefore, reasoning depth, parser coverage, and proof representation jointly influence performance. Recent solver-oriented work similarly notes that NL-to-symbolic parsing errors can dominate failure modes in deductive reasoning pipelines (Feng et al., 2024).

I. Robustness to policy clutter

Adding distractor facts does not change SymbolicReasoner+Proof or LR-proof accuracy because both methods ultimately depend on derivations that are structurally tied to the query rather than on overall lexical density. In contrast, LR-text changes its behavior in the presence of distractors, indicating that its evidence signal is diluted by irrelevant sentences. This robustness analysis, together with the depth-budget and proof-corruption tests, broadens the evaluation beyond a single fixed split and supports the main conclusions under simple but meaningful variations.

J. Implications for real policy systems

The evaluated pipeline maps directly to explainable enforcement components: authoring policies in restricted natural language, compiling into a knowledge base, running deterministic inference, and emitting an auditable trace. The key engineering constraint is maintaining a controlled

Journal of Technology Informatics and Engineering (JTIE)
Vol. 4 No. 2 August 2025
E-ISSN: 2961-9068; P-ISSN: 2961-8215, Pages 402-424
Journal Link: https://jtie.stekom.ac.id/index.php/jtie/index

language or a reliable semantic parser. The evaluation metrics and verification interface transfer to richer policy languages that include priorities, stratified negation, or time.

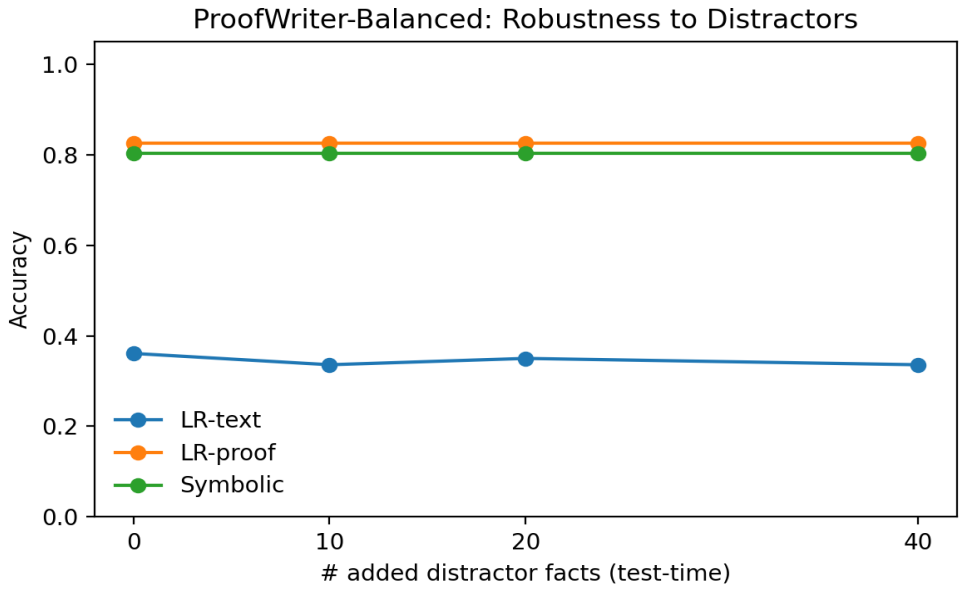

Figure 6. Robustness to Distractor Facts on ProofWriter-Balanced

Several constraints should temper the conclusions. First, both datasets use controlled or generated English, so the reported results do not by themselves establish performance on heterogeneous real policy documents. Second, the study evaluates one parser family and one simple linear classifier family; stronger neural baselines or alternative semantic parsers may change the absolute numbers, although the current comparisons are internally fair because LR-text and LR-proof share the same configuration. Third, RuleTaker-Rep is generated from the published grammar, which likely benefits the symbolic pipeline and helps explain the perfect scores. Fourth, the current rule fragment excludes priorities, temporal operators, and richer exception handling, so scalability to more expressive policy languages remains an open question. For these reasons, the results should be interpreted as evidence of feasibility in controlled settings rather than as a complete solution for production-scale compliance reasoning.

V. CONCLUSION AND RECOMMENDATION

We implemented and evaluated an explainable natural-language policy reasoner that answers True/False/Unknown and generates verifiable proofs. On two controlled natural-language rule benchmarks, symbolic execution of natural-language rules achieves high depth-sensitive accuracy and produces proofs that are verifiable for 100% of predictions, while a proof-trained classifier reaches 0.825 accuracy on ProofWriter-Balanced and remains robust to distractor policy text. The main contribution is an auditable pipeline and evaluation protocol that make proof verifiability a first-class criterion, not a new reasoning formalism. At the same time, the findings should be

interpreted within the limits of controlled-language datasets, parser coverage, and the grammar match between the system and the benchmarks. Future work should test richer policy languages, stronger semantic parsers, and more diverse train/test scenarios.

REFERENCES

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of Databases*. Addison-Wesley. https://web.stanford.edu/~ullman/fodb.html

Appel, A. W., & Felten, E. W. (1999). Proof-Carrying Authentication. In *Proceedings of the 6th ACM Conference on Computer and Communications Security*, 52–62. https://doi.org/10.1145/319709.319718

Becker, M. Y., Fournet, C., & Gordon, A. D. (2007). Design and Semantics of a Decentralized Authorization Language. In *Proceedings of the 20th IEEE Computer Security Foundations Symposium*, 3–15. https://doi.org/10.1109/cfs.2007.13

Camburu, O.-M., Rocktäschel, T., Lukasiewicz, T., & Blunsom, P. (2018). E-SNLI: Natural Language Inference with Natural Language Explanations. In *Advances in Neural Information Processing Systems*, *31*, 9560–9572. https://doi.org/10.48550/arxiv.1812.01193

Ceri, S., Gottlob, G., & Tanca, L. (1989). What You Always Wanted to Know about Datalog (And Never Dared to Ask). *IEEE Transactions on Knowledge and Data Engineering*, *1*(1), 146–166. https://doi.org/10.1109/69.43410

Clark, P., Tafjord, O., & Richardson, K. (2020). Transformers as Soft Reasoners over Language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence* (*IJCAI-20*), 3882–3890. https://doi.org/10.24963/ijcai.2020/537

DeYoung, J., Jain, S., Rajani, N. F., Lehman, E., Xiong, C., Socher, R., & Wallace, B. C. (2020). ERASER: A Benchmark to Evaluate Rationalized NLP Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (*ACL 2020*), 4443–4458. https://doi.org/10.18653/v1/2020.acl-main.408

Dries, A., Kimmig, A., Meert, W., Renkens, J., Van den Broeck, G., Vlasselaer, J., & De Raedt, L. (2015). ProbLog2: Probabilistic Logic Programming. In *Machine Learning and Knowledge Discovery in Databases*, *9286*, 312–315. https://doi.org/10.1007/978-3-319-23461-8_38

Fedhira, & Prianto, C. (2025). Systematic Literature Review: Analysis of AI Implementation for Document Verification. *Jurnal Ilmiah Sistem Informasi*, *4*(3), 417–430. https://doi.org/10.51903/kjjwk708

Feng, J., Xu, R., Hao, J., Sharma, H., Shen, Y., Zhao, D., & Chen, W. (2024). Language Models can be Deductive Solvers. In *Findings of the Association for Computational Linguistics: NAACL 2024* (*NAACL 2024*), 4026–4042. https://doi.org/10.18653/v1/2024.findings-naacl.254

Hao, L. W., & Liu, R. K. (2025). Transfer Learning Approach for Sentiment Analysis in Low-Resource Austronesian Languages Using Multilingual BERT. *Journal of Technology Informatics and Engineering*, *4*(1), 75–94. https://doi.org/10.51903/jtie.v4i1.276

Jain, S., & Wallace, B. C. (2019). Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (*NAACL-HLT 2019*), 3543–3556. https://doi.org/10.18653/v1/n19-1357

Jacovi, A., & Goldberg, Y. (2020). Towards Faithfully Interpretable NLP Systems: How Should We Define and Evaluate Faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (*ACL 2020*), 4198–4208. https://doi.org/10.18653/v1/2020.acl-main.386

Kamath, A., & Das, R. (2018). A Survey on Semantic Parsing. *arXiv Preprint*, *arXiv:1812.00978*. https://doi.org/10.48550/arxiv.1812.00978

Lei, T., Barzilay, R., & Jaakkola, T. (2016). Rationalizing Neural Predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (*EMNLP 2016*), 107–117. https://doi.org/10.18653/v1/d16-1011

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, *30*, 4765–4774. https://doi.org/10.48550/arxiv.1705.07874

Lyu, Q., Havaldar, S., Stein, A., Zhang, L., Rao, D., Wong, E., Apidianaki, M., & Callison-Burch, C. (2023). Faithful Chain-of-Thought Reasoning. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, *1*, 305–329. https://doi.org/10.18653/v1/2023.ijcnlp-main.20

McCarthy, J. (1959). Programs with Common Sense. In *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*, 75–91. https://doi.org/10.1145/319709.319718

OASIS. (2013). *EXtensible Access Control Markup Language (XACML) Version 3.0*. OASIS Standard. OASIS Publishing. https://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

Quan, X., Valentino, M., Dennis, L. A., & Freitas, A. (2024). Verification and Refinement of Natural Language Explanations through LLM-Symbolic Theorem Proving. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing* (*EMNLP 2024*), 2933–2958. https://doi.org/10.18653/v1/2024.emnlp-main.165

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. *OpenAI Blog*, *1*(8), 9. https://openai.com/blog/better-language-models

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, *21*(140), 1–67. https://jmlr.org/papers/v21/20-074.html

Rajani, N. F., McCann, B., Xiong, C., & Socher, R. (2019). Explain Yourself! Leveraging Language Models for Commonsense Reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (*ACL 2019*), 4932–4942. https://doi.org/10.18653/v1/p19-1487

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (*KDD '16*), 1135–1144. https://doi.org/10.1145/2939672.2939778

Rocktäschel, T., & Riedel, S. (2017). End-to-End Differentiable Proving. In *Advances in Neural Information Processing Systems*, *30*, 3788–3800. https://proceedings.neurips.cc/paper/2017/hash/b2961d1e0892f39c6705d4cb549b0612-Abstract.html

Saparov, A., & He, H. (2023). Language Models can Solve Complex Reasoning Tasks by Reasoning through Proofs. *arXiv Preprint*, *arXiv:2205.11502*. https://doi.org/10.48550/arxiv.2205.11502

Serrano, S., & Smith, N. A. (2019). Is Attention Interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (*ACL 2019*), 2931–2951. https://doi.org/10.18653/v1/p19-1282

Sriasih, S. D., Razak, F. A., & Ikhsan, H. A. I. (2025). AI-Driven Sentiment Analysis of Retail Investor Behavior during Market Volatility: A Study of Twitter Data in Southeast Asia. *Journal of Management and Informatics*, *4*(1), 741–756. https://doi.org/10.51903/jmi.v4i1.179

Sun, Z., Ding, X., Du, L., Cai, B., Gao, J., Liu, T., & Qin, B. (2024). Towards Generalizable and Faithful Logic Reasoning over Natural Language via Resolution Refutation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation* (*LREC-COLING 2024*), 16527–16538. https://aclanthology.org/2024.lreccoling-main.1438

Tafjord, O., Dalvi, B., & Clark, P. (2021). ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021* (*ACL-IJCNLP 2021*), 3621–3634. https://doi.org/10.18653/v1/2021.findings-acl.317

Theoxo. (2023). Proofwriter-Deduction-Balanced (Version 1.0). *Hugging Face Dataset*. https://huggingface.co/datasets/theoxo/proofwriter-deduction-balanced

Turpin, M., Michael, J., Perez, E., & Bowman, S. R. (2023). Language Models don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting. In *Advances in Neural Information Processing Systems*, *36*, 74643–74660. https://proceedings.neurips.cc/paper_files/paper/2023/hash/ebca5557ca7852c7921867c4613c2bca-Abstract-Conference.html

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all You Need. In *Advances in Neural Information Processing Systems*, *30*, 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., & Zhou, D. (2023). Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *Proceedings of the International Conference on Learning Representations* (*ICLR 2023*). https://openreview.net/forum?id=1PL1uIMpbt

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., & Zhou, D. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, *35*, 24824–24837. https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html

Wiegreffe, S., & Pinter, Y. (2019). Attention is not Not Explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP 2019*), 11–20. https://doi.org/10.18653/v1/d19-1002