LeNS: A Neuro-Symbolic Framework with Logic-Guided Abductive Feedback for Defeasible Legal Reasoning

Anonymous ACL submission

Abstract

Legal reasoning inherently exhibits significant non-monotonicity due to statutory exceptions and dynamically evolving facts. Pure LLMs suffer from severe hallucinations when performing such complex reasoning tasks. Furthermore, traditional one-way "formalize-then-solve" neuro-symbolic frameworks are highly vulnerable to cascading errors caused by fact omissions during the preliminary formalization stage. To overcome these bottlenecks, this paper proposes a bidirectional neuro-symbolic framework named LeNS equipped with an abductive mechanism for legal reasoning. The framework generates an initial formal language context based on abstracted facts and rules, and then utilizes a logic-guided abductive feedback loop to dynamically supplement missing factual premises. Additionally, a multi-dimensional coherence ranking strategy autonomously resolves legal indeterminacy caused by the open texture of law to select the optimal conclusion. Extensive empirical evaluations across four datasets demonstrate that LeNS achieves highly competitive performance exceeding the best baselines by an average of 10.4%. Notably, LeNS exhibits exceptional robustness on the statute shifts benchmark LawShift outperforming the best baselines by an average of 33.9%.[1]

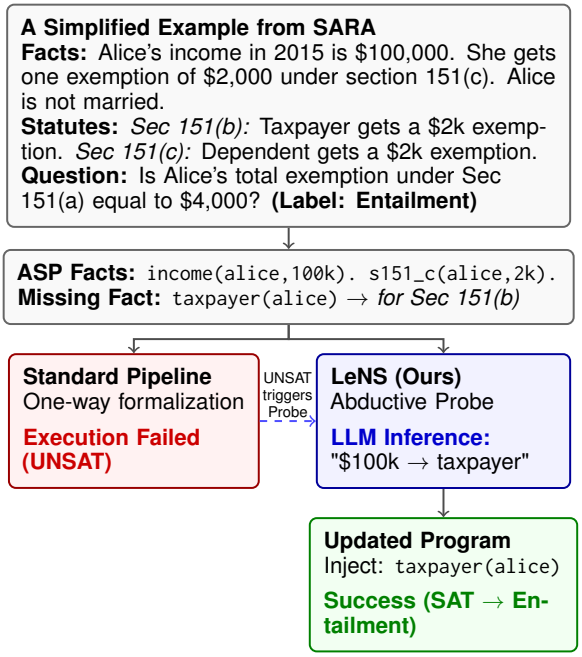

Figure 1: Comparison of the standard pipeline and the proposed LeNS framework. LeNS overcomes the failure of standard methods by using logical execution failures to trigger an abductive feedback loop, dynamically injecting missing premises back into the ASP programs.

1 Introduction

Recently, the application of LLMs to automate legal reasoning tasks has attracted considerable attention, but the logical and non-monotonic nature of legal reasoning also present challenges. For instance, Katz et al. (2024) and Cui et al. (2023) demonstrate the remarkable proficiency of pure LLMs in passing human bar exams and providing general legal consultations. However, compared to general domains, legal reasoning intrinsically demands rigorous logical reasoning and exhibits significant non-monotonicity, where previously established conclusions can be invalidated by statutory exceptions and new information (Sartor, 1993; Prakken and Sartor, 2004). Blair-Stanek et al. (2023) and Dahl et al. (2024) reveal that LLMs are highly susceptible to severe hallucinations when resolving legal reasoning tasks. Deng et al. (2023) and Feng et al. (2022) highlight the necessity of meticulously aligning facts with nested rules for judicial adjudication, and suggest that the probabilistic nature of LLMs severely limits their capacity for complex legal reasoning.

For more reliable legal reasoning, the autoformalization-base neuro-symbolic methods for logical reasoning have gained attention. Holzen-

[1]The core data and code are available at https://anonymous.4open.science/r/LeNS-34B2.

berger and Van Durme (2023) formalized legal reasoning tasks into programs of Prolog (Colmerauer and Roussel, 1993) by information extraction to translate unstructured case facts while relying on existing manually formalized statutes. Kant et al. (2025) and Jurayj et al. (2026) further utilized LLMs to formalize both case facts and statutory rules into Prolog programs for tax and insurance reasoning, respectively. As long as the formalizations are correct, resorting to symbolic solvers can guarantee faithfulness of reasoning. However, existing works on auto-formalization for legal reasoning suffer from two limitations. Firstly, these works adopt a one-way "formalize-then-solve" pipeline which is highly vulnerable to any omissions or other errors in the formalization stage. As illustrated in Figure 1, omitting the crucial factual premise of "Alice is a taxpayer" during the initial formalization phase causes the symbolic solver to fail. These one-way frameworks also deviate from real-world judicial practice, which requires adjudicators to perform multiple rounds of abductive analysis between facts and legal norms. Secondly, existing works adopt Prolog as the formal language, but the inherent monotonic nature of Prolog prevents it from effectively modeling defeasible legal exceptions (Ashley, 2017). Due to the non-monotonicity caused by statutory exceptions and the semantic ambiguity brought by the open texture of legal text, a single case might yield multiple logically valid results. This calls for mechanisms to infer such multiple possibilities and identify the most plausible outcome.

To effectively handle defeasible legal reasoning, this paper proposes LeNS, an autoformalization-based neuro-symbolic framework equipped with logic-guided abductive feedback. The framework formalizes legal reasoning into Answer Set Programming (ASP) (Gelfond and Lifschitz, 1988), as ASP employs the stable model semantics for non-monotonic reasoning, and ASP solvers can be used to generate multiple answer sets to reflect multiple logically valid conclusions. Inspired by (Cotnareanu et al., 2026) which explores abductive logic in commonsense reasoning tasks, LeNS is equipped with logic-guided abductive feedback. The framework first generates an initial formal context from abstracted facts and rules. Upon encountering symbolic execution failures caused by omitted premises as illustrated in Figure 1, LeNS triggers a logic-guided abductive feedback loop. This loop derives candidate abductive targets from the formalized statutory rules and utilizes LLMs to review the case narrative for verification. Once verified, the framework dynamically injects the missing factual premises back into the solver to complete the logical chain. Furthermore, to select from multiple answer sets, inspired by multi-dimensional verification techniques recently utilized in text generation (Dhuliawala et al., 2024), LeNS employs a multi-dimensional coherence ranking strategy evaluating textual consistency, legal plausibility, and simplicity.

Extensive empirical evaluations across four challenging legal benchmarks demonstrate the effectiveness of LeNS. Specifically, our framework achieves highly competitive performance by outperforming the best baselines by an average of 10.4%. Furthermore, LeNS exhibits exceptional robustness on the statute shifts benchmark LawShift by exceeding the best baselines by an average of 33.9%, which effectively overcomes the parametric memory limitations of pure LLMs.

2 Related Work

LLMs in Legal Reasoning. Automated legal reasoning based on LLMs attracts widespread attention. Initially, Chalkidis et al. (2022) and Dai et al. (2025) construct LexGLUE and LAiW benchmarks, respectively, to evaluate LLMs' legal text processing abilities. Furthermore, Holzenberger et al. (2020) and Goebel et al. (2024) introduce the SARA and COLIEE datasets to strictly assess, respectively, formal statutory reasoning and open texture entailment. Moreover, Fei et al. (2024) and Han et al. (2026) propose LawBench and LawShift, respectively, to evaluate prison term prediction and robustness against statute shifts. To enhance the legal reasoning capabilities of the models, Huang et al. (2023) and Colombo et al. (2024) develop fine-tuned legal LLMs such as Lawyer LLaMA and SaulLM, respectively. However, these evaluations reveal that pure LLMs frequently produce logical errors. These limitations motivate the exploration of neuro-symbolic architectures to achieve reliable legal reasoning.

Neuro-Symbolic AI in Law. Researchers increasingly integrate symbolic logic into legal AI to overcome the inherent limitations of LLMs. Initially, Satoh et al. (2010) and Merigoux et al. (2021) develop pure symbolic systems such as PROLEG and Catala for legal reasoning tasks which require manual formalization. To overcome the manual

translation bottleneck, Pan et al. (2023) and Yang et al. (2023) pioneer autoformalization in general domains by coupling LLM translation with symbolic solvers. Inspired by this paradigm, Kant et al. (2025) and Jurayj et al. (2026) construct one-way "formalize-then-solve" pipelines for insurance and tax adjudication, respectively, by translating legal texts into Prolog programs. To address the vulnerability of these one-way pipelines to fact omissions, the proposed LeNS framework utilizes ASP to support defeasible reasoning actively recover omitted facts via logic-guided abductive feedback.

3 Preliminaries

A standard symbolic vocabulary provides the theoretical foundation to formally represent unstructured legal texts. We define a finite set of predicates $\mathcal{P}$ for legal relations and a set of entities $\mathcal{E}$ for case subjects to form atomic formulas $p(c_1, \ldots, c_n)$ with $p \in \mathcal{P}$ and arguments $c_i \in \mathcal{C}$. A literal is subsequently defined as an atom or its negation such as `taxpayer(alice)` or `-taxpayer(alice)`. To effectively characterize the non-monotonicity of legal reasoning caused by exceptions we formally represent the legal text using a default theory derived from the foundational concepts of Lawsky (2017) and Prakken and Sartor (2004). A default rule is formulated as $r = \frac{\alpha:\beta_1,\ldots,\beta_m}{\gamma}$ where the prerequisite α is a conjunction of literals while the justifications $\beta_1, \ldots, \beta_m$ form a set of literals and the conclusion γ is a single literal. Consequently, the system logically deduces the conclusion γ if case facts satisfy the prerequisite α and the justifications $\beta_1, \ldots, \beta_m$ are consistent.

Answer Set Programming serves as a highly efficient declarative logic programming paradigm to computationally solve these default theories through strong default negation. Gelfond and Lifschitz formally defined the solutions of an ASP program as answer sets representing minimal consistent sets of facts satisfying all rules. Furthermore, Baral and Gelfond (1994) proved that literal-based default theories seamlessly compile into such executable ASP programs. Consequently, we ultimately translate the default logic into executable ASP sentences formulated as $\gamma \leftarrow \alpha, \text{not } \neg\beta_1, \ldots, \text{not } \neg\beta_m$ where `not` denotes default negation and "$\neg$" denotes strong negation. Symbolic solvers subsequently execute these compiled programs to compute the corresponding answer sets. Due to the inherent legal indeterminacy caused by statutory exceptions the symbolic solver frequently generates multiple valid answer sets denoted as $\{E_1, E_2, \ldots, E_n\}$. Each valid answer set rigorously encapsulates a distinct and logically sound judicial interpretation of the underlying legal narrative.

4 The LeNS Framework

4.1 Task Definition

The legal reasoning task essentially constitutes a mathematical mapping from unstructured inputs to specific judicial outcomes. Given a set of unstructured case facts F and relevant statutory rules R, the primary objective is to learn a predictive function mapping (F, R) to the target judgment Y. To accommodate diverse judicial applications, the target judgment Y is represented as either a discrete categorical label $Y \in \mathcal{Y}$ or a continuous numerical value $Y \in \mathcal{V}$. To achieve reliable legal reasoning, this work reconceptualizes this predictive mapping as a rigorous reasoning process. Consequently, the proposed framework translates the natural language inputs into an executable logic program and deduces the ground-truth judgment Y^* based on the generated answer sets.

4.2 Architecture Overview

This section presents the bidirectional Legal Neuro-Symbolic framework. The proposed framework formulates legal adjudication as a dynamic reasoning process. First, an abstraction and autoformalization module grounds unstructured legal texts into initial ASP programs. Subsequently, a logic-guided abductive engine recursively recovers missing factual premises to complete the logical chain. Finally, a multi-dimensional coherence ranking mechanism selects the optimal answer set to derive the final judgment. Prompt templates for all modules are available in Appendix B.3.

Abstraction and Autoformalization. This module transforms unstructured case facts and statutes into an executable formalized context through semantic abstraction and symbolic autoformalization. The LeNS framework utilizes prompt-based LLMs to parse raw case descriptions into atomic fact statements F and decouple legal statutes into structured conditional rules R. To guarantee extraction accuracy, the system forces LLMs to output corresponding source sentences to evaluate factual consistency and legal plausibility with default thresholds of 0.6 and 0.7 respectively. These two met-

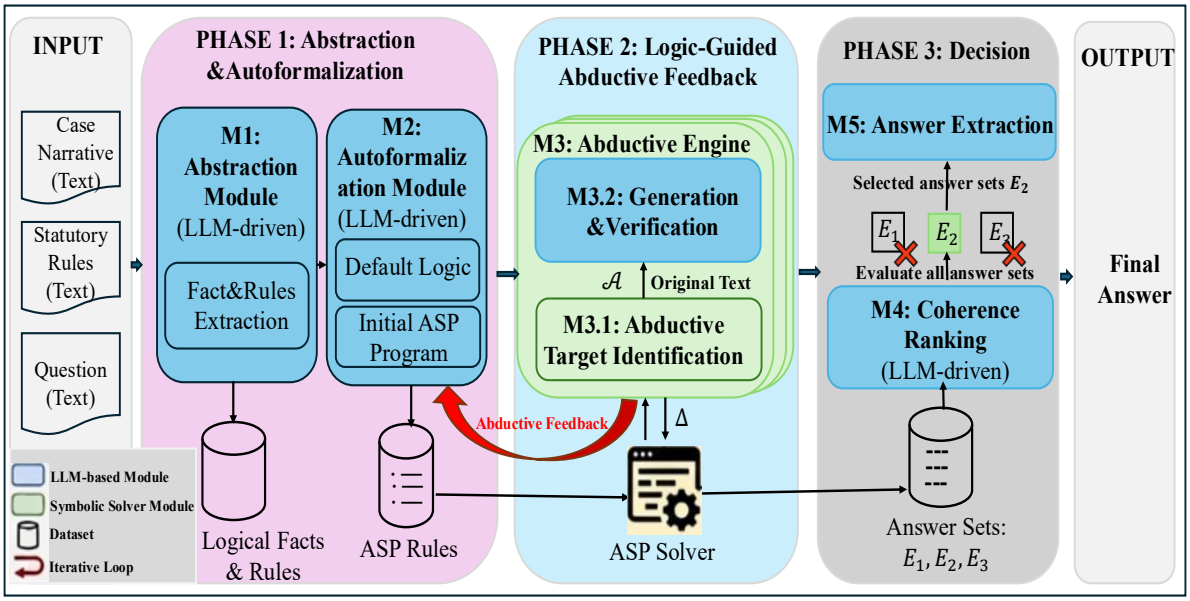

Figure 2: Overview of the proposed LeNS framework. The execution workflow systematically consists of three phases including abstraction and autoformalization, logic-guided abductive feedback, and the final decision.

rics respectively guarantee the textual grounding of the extracted statements within the source narrative and their strict alignment with legal commonsense. The module subsequently formalizes these validated sentences step-by-step into default logic and automatically converts them into ASP programs. For example as illustrated in Figure 1, an extracted atomic fact translates into an ASP fact "income(alice, 100000, 2015)." while a structured statute transforms into an ASP rule "exemption(alice, 2000, 2015) ← taxpayer(alice, 2015),not ¬ exemption(alice, 2000, 2015).". The symbolic solver strictly verifies syntactic correctness and logical satisfiability while safely discarding sentences failing a three-attempt correction loop to construct the final formalized context.

Logic-Guided Abductive Feedback. The logic-guided abductive feedback mechanism leverages symbolic feedback to mine omitted factual premises. This module comprises two sub-steps consisting of abductive target identification alongside generation and verification. First based on the generated initial ASP program the framework invokes the symbolic solver Clingo[2] to compute the initial answer set backbone B_0. When reasoning based on the initial backbone fails, the system extracts the facts absent from the backbone but present in the prerequisites and justifications of the formalized rules as candidate abductive facts. The framework subsequently scores and sorts these extracted facts to construct a unified set of abductive targets $\mathcal{A}$ by assigning higher weights to rule prerequisites and prioritizing facts utilized earlier in the reasoning process. Guided by the natural language semantics of these targets the module utilizes LLMs to mine potentially omitted facts ΔF from the unstructured text. The framework dynamically injects the abduced facts into the ASP program only when they successfully pass the dual validation of factual consistency and legal plausibility. Finally the symbolic solver recomputes the updated answer set backbone B_{i+1} iteratively up to three times to determine the final conclusion. Appendix B.2 provides the detailed pseudocode of the logic-guided abductive feedback algorithm.

Coherence Ranking. The framework computes a coherence score for each answer set E_i formulated as $Score(E_i) = w_1 \cdot S_{textual}(E_i, T) + w_2 \cdot S_{legal}(E_i) + w_3 \cdot S_{simplicity}(E_i)$. These three metrics $S_{textual}$, S_{legal}, and $S_{simplicity}$, respectively, evaluate the textual consistency against the original narrative T, the legal plausibility based on legal principles and the simplicity preference for minimal hypotheses within the candidate answer set. The module sets the hyperparameters w_1, w_2 and w_3 to 0.4, 0.4 and 0.2, respectively, to balance these dimensions. It ultimately selects the highest scoring answer set as the optimal logical backbone.

[2] https://potassco.org/clingo/

Answer Extraction. This module derives the final judicial decision through a predicate-driven symbolic matching mechanism. Specifically, the module retrieves answer set atoms sharing identical predicate with the query atom. For classification tasks the framework outputs Entailment given exact entity matches and Contradiction if the answer set contains the negated query atom. For regression tasks, the module directly extracts numerical parameters from the matched atom as the final prediction. When symbolic predicate matching fails the framework automatically relies on LLMs for direct prediction to guarantee overall pipeline robustness.

5 Experiments

5.1 Experimental Setup

This section establishes rigorous comparative baselines across diverse LLMs to comprehensively evaluate the proposed framework. Specifically, the experimental setup employs four advanced LLMs as base models including DeepSeek-V3.2 (DeepSeek-AI, 2024), GPT-5.4-mini (Singh et al., 2025), o4-mini (OpenAI, 2025), and Gemini-3-flash (Team et al., 2023). The evaluation compares the LeNS framework against two primary reasoning paradigms namely zero/few-shot prompt-based LLMs (Wei et al., 2022) and Parse2ASP representing the traditional one-way "formalize-then-solve" neuro-symbolic pipeline based on the Logic-LM framework (Pan et al., 2023). Maintaining identical backbone LLMs across these baselines ensures a fair assessment of the architectural advantages introduced by the logic-guided abductive feedback mechanism. Appendix B.1 provides the detailed hyperparameter configurations.

This section evaluates the reasoning capabilities across diverse legal systems by conducting experiments on four datasets including the SARA v3 dataset (Holzenberger et al., 2020), the LawBench dataset (Fei et al., 2024), the LawShift dataset (Han et al., 2026) and the COLIEE Task 4 dataset (Goebel et al., 2024). The SARA v3 dataset focuses on US tax law and contains a 276-sample classification subtask alongside a 100-sample numerical tax calculation subtask. We respectively utilize the F1-score and Numerical Accuracy (NumAcc) to assess these two subtasks. For the tax prediction task, we consider the model prediction correct when the relative deviation of the predicted value $\hat{y}$ from the ground truth y satisfies $\frac{|y-\hat{y}|}{\max(0.1y,500)} < 1$. The LawBench dataset focuses on Chinese Criminal Law. We select 100 samples from the ID 3-5 subset for prison term prediction. We manually augment them with nine category labels and consider the prediction correct when the true prison term months fall within the predicted category. The LawShift dataset introduces statute shifts to evaluate model robustness. We use the Term-up and Term-down subsets to calculate accuracy on original and poisoned samples alongside a strict pass rate requiring correct predictions for both versions. The COLIEE Task 4 dataset provides a statutory entailment task. We randomly sample 100 cases for entailment classification evaluated by the F1-score. Appendix A provides the detailed dataset statistics and data samples.

5.2 Main Results

Table 1 demonstrates that the proposed LeNS framework significantly outperforms both prompt-based LLMs and traditional one-way formalization pipelines across diverse complex legal reasoning scenarios. Prompt-based models exhibit severe limitations in tasks requiring complex mathematical calculations and dynamic legal condition matching. For instance, on the LawShift dataset evaluating robustness against statute shifts, the zero-shot prompt-based DeepSeek-V3.2 plummets to 14.2% and 14.9% accuracy on the Term-up and Term-down subsets, respectively. Conversely, the proposed LeNS framework elevates these scores to a substantial 95.7% and 93.0%, respectively. Furthermore, in the SARA numerical tax calculation subtask, the proposed framework consistently maintains a numerical accuracy exceeding 75% across diverse base models to effectively mitigate the numerical hallucination issues inherent in prompt-based approaches. Additionally, the Parse2ASP baseline utilizing Gemini-3-flash attains only 5.6% and 68.0% accuracy on the LawShift Term-up and LawBench datasets, respectively, due to its inability to dynamically recover missing factual premises. Conversely, the proposed framework autonomously supplements the omitted factual premises and substantially elevates the corresponding accuracies to 83.3% and 77.0%, respectively. These results verify that the logic-guided abductive feedback mechanism effectively overcomes the inherent defects of traditional one-way formalization pipelines.

The performance on the COLIEE Task 4 benchmark uncovers the boundaries of logical formalization when processing extreme semantic diversity.

Table 1: Overall performance across four datasets. "Class." and "Tax" denote statutory entailment classification and numerical tax calculation, respectively. Bold values indicate the best results per base model.

Model	Method	SARA		LawShift		LawBench		COLIEE
		Class.	Tax	Term-up	Term-down	Months	Category	Task 4
DeepSeek-V3.2	Zero-shot	70.9	13.0	14.2	14.9	73.0	58.0	**78.9**
	Few-shot	66.7	15.0	17.7	16.7	73.0	63.0	77.8
	Parse2ASP	73.0	**77.8**	8.8	59.5	68.0	63.0	50.2
	LeNS (Ours)	**77.4**	75.0	**95.7**	**93.0**	**78.0**	**71.0**	63.0
GPT-5.4-mini	Zero-shot	51.8	29.0	10.2	8.9	69.0	53.0	64.0
	Few-shot	53.3	37.0	13.9	33.9	69.0	57.0	**68.7**
	Parse2ASP	74.4	48.0	19.7	19.1	69.0	61.0	61.3
	LeNS (Ours)	**75.0**	**78.3**	**78.1**	**63.7**	**74.0**	**65.0**	62.0
o4-mini	Zero-shot	92.0	69.0	21.9	32.5	73.0	63.0	**77.9**
	Few-shot	94.6	**90.0**	57.6	55.0	72.0	64.0	77.8
	Parse2ASP	91.4	56.0	31.8	64.6	67.8	55.4	64.6
	LeNS (Ours)	**94.9**	88.0	**70.9**	**81.5**	**79.1**	**73.1**	63.6
Gemini-3-flash	Zero-shot	88.0	55.0	5.12	12.1	**80.0**	61.0	87.0
	Few-shot	**88.4**	57.0	72.5	68.3	79.0	64.0	**88.0**
	Parse2ASP	85.4	90.0	5.6	41.1	68.0	55.8	73.8
	LeNS (Ours)	86.9	**93.3**	83.3	**98.6**	77.0	**69.0**	73.0

The zero-shot or few-shot prompt-based DeepSeek-V3.2 and Gemini-3-flash models achieve the highest F1 scores of 78.9% and 88.0%, respectively. However, the proposed LeNS framework only attains corresponding F1 scores of 63.0% and 73.0%. This anomaly primarily stems from the inherent open texture of the Japanese Civil Code alongside the severe semantic entanglement between sample questions and case facts. This deep fusion obscures the boundaries between factual premises and legal queries to significantly exacerbate the formalization challenges for neuro-symbolic approaches. Conversely, prompt-based LLMs frequently exploit semantic shortcuts and leverage pre-trained parametric memory to guess the entailment labels instead of performing rigorous logical deduction.

5.3 In-depth Analysis

5.3.1 Executability and Abductive Feedback

Table 2 presents the number of abduced facts and execution rates generated by the LeNS framework. The evaluation defines the execution rate as the proportion of samples successfully completing the symbolic matching process and the execution accuracy as the prediction accuracy calculated upon this executable foundation. First, on the SARA classification dataset, the execution rate fails to exceed 50.0%, whereas the framework achieves an execution rate exceeding 90.0% across most models on the LawShift Term-up subset which demonstrates that the inherent complexity of legal tasks causes significant variance in execution rates. Second, employing o4-mini on the SARA dataset generates only 0.5 abduced facts on average, resulting in a lower execution rate of 41.7% but attaining a perfect 100.0% execution accuracy. Conversely,

Table 2: Executability results of the proposed LeNS framework across four datasets. "Exec." and "Exec-Acc" denote the execution rates and the execution accuracy under the execution rate, respectively. "#Abd." indicates the average number of abduced facts per case. Models are abbreviated as DS-V3.2 (DeepSeek-V3.2), o4-m (o4-mini), GPT-5.4m (GPT-5.4-mini), and Gem-3F (Gemini-3-flash).

Dataset	Model	Exec.(%)	#Abd.	Exec-Acc(%)	LeNS(%)
SARA (Class.)	DS-V3.2	25.2	1.0	63.8	77.4
	o4-m	41.7	0.5	**100.0**	**94.9**
	GPT-5.4m	46.7	0.7	55.4	74.9
	Gem-3F	**50.0**	6.8	72.1	86.9
SARA Tax	DS-V3.2	42.0	5.7	78.6	75.0
	o4-m	66.7	0.5	73.3	88.0
	GPT-5.4m	43.8	0.6	78.6	65.6
	Gem-3F	**71.1**	0.8	**96.9**	**93.3**
LawBench (3-5)	DS-V3.2	60.0	0.9	78.3	78.0
	o4-m	75.0	0.6	**84.3**	**79.1**
	GPT-5.4m	63.0	1.5	77.0	74.0
	Gem-3F	**76.0**	0.7	81.6	77.0
LawShift (Term-up)	DS-V3.2	95.6	0.1	**95.6**	**95.7**
	o4-m	90.9	0.7	70.1	70.9
	GPT-5.4m	97.7	0.2	78.1	78.1
	Gem-3F	**99.5**	0.1	83.6	83.3
LawShift (Term-down)	DS-V3.2	98.6	0.2	92.9	93.0
	o4-m	85.2	0.7	81.5	81.5
	GPT-5.4m	98.6	0.25	63.7	63.7
	Gem-3F	**99.5**	0.1	**98.6**	**98.6**
COLIEE (Task 4)	DS-V3.2	48.0	1.2	47.9	63.0
	o4-m	56.6	1.0	60.0	62.0
	GPT-5.4m	63.0	1.0	60.7	63.6
	Gem-3F	**65.0**	2.3	**67.7**	**68.0**

employing Gemini-3-flash generates an average of 6.8 facts to achieve a maximum execution rate of 50.0%, but the introduced semantic noise reduces the execution accuracy to 72.1%. This comparison indicates that an increased number of abduced facts improves the model execution rate but inevitably leads to a decrease in execution accuracy. Finally, when evaluated with DeepSeek-V3.2, the joint mechanism for answer extraction effectively compensates for the low 25.2% execution rate to ultimately elevate the overall accuracy to 77.4%. This result reveals the critical importance of integrating LLM reasoning within the answer extraction module to rescue samples failing in symbolic matching.

5.3.2 Logical Consistency Analysis

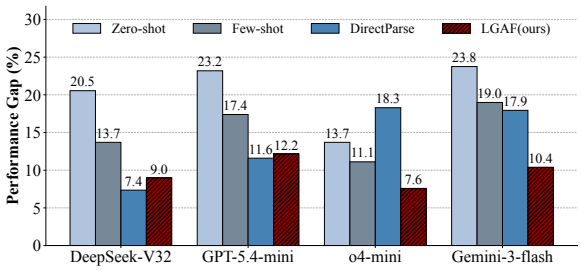

Figure 3: Relative performance gap between the prison term month accuracy and the category accuracy on the LawBench.

The evaluation calculates relative performance gap between the prison term month accuracy (MonthAcc) and the category accuracy (CategoryAcc) to measure this logical consistency on the LawBench dataset. The metric defines this relative gap as the percentage difference between the two accuracies calculated via $\frac{MonthAcc - CategoryAcc}{MonthAcc} \times 100\%$, and Figure 3 presents the corresponding results where a smaller gap value indicates a closer alignment between MonthAcc and CategoryAcc to demonstrate a higher degree of logical consistency in the reasoning process. The prompt-based LLMs incur a massive performance drop, as evidenced by a severe gap of 23.8% for the zero-shot Gemini-3-flash model. This degradation stems from the inherent flaw of generative paradigms relying on semantic guessing rather than rigorous logical deduction. Conversely, employing DeepSeek-V3.2 within the proposed LeNS framework achieves a MonthAcc of 78.0% and a CategoryAcc of 71.0%, yielding the highest absolute performance while significantly constraining the relative gap to 9.0%. This result verifies that integrating a symbolic solver successfully enforces strict logical consistency and mitigates semantic illusions.

5.3.3 Robustness to Statute Shifts

The robustness analysis against statute shifts demonstrates that the proposed LeNS framework possesses exceptional adaptability to counterfactual legal rules. The evaluation verifies this robustness by comparing prediction accuracies between the original and modified statutes on the LawShift dataset. Figure 4 and Figure 18 in the appendix C illustrate that baseline methods suffer from severe performance degradation when processing modified statutes that contradict real-world laws. Specifically, the accuracy of zero-shot prompt-based Gemini-3-flash model drops from 88.8% on the original statutes to 14.4% on the shifted versions. This degradation primarily occurs because prompt-based LLMs over-rely on pre-trained parametric memory restricting their ability to strictly follow counterfactual legal rules. Conversely, LeNS utilizing DeepSeek-V3.2 achieves an accuracy of 95.7% on the original statutes and a perfect 100.0% on the shifted statutes to exhibit strong resilience against statute shifts. This contrast confirms that integrating a symbolic solver successfully enforces strict logical derivation based entirely on the provided case facts and statutes.

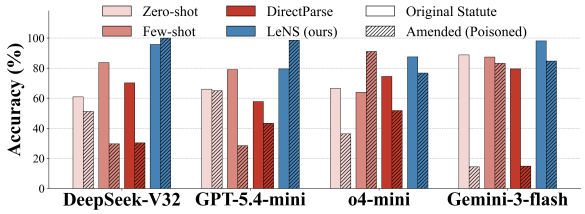

Figure 4: Accuracy comparison on the LawShift (Term-up) dataset before and after statute shifts.

5.3.4 Ablation Study

Extensive ablation studies on both the SARA classification and the LawBench datasets quantify the individual contributions of the core modules. Table 3 and Table 5 in the appendix C demonstrate that removing the abstraction module (w/o Abstraction) drops the execution rate to zero because the formal solver cannot directly process unstructured narratives. Consequently, the framework relies entirely on LLM reasoning and suffers a 5.5% accuracy drop for DeepSeek-V3.2. Furthermore, eliminating the abductive module (w/o Abductive) restricts the framework to strict logical reasoning without sup-

plementing missing facts. This limitation reduces the execution rate and severely degrades the overall accuracy by 9.6% for DeepSeek-V3.2. Finally, discarding the coherence ranking module (w/o Coherence) impairs the capability to filter the optimal logical backbone from multiple valid answer sets and degrades the overall performance across all foundation models

Table 3: Ablation study results on the SARA classification dataset. Values in red indicate the performance degradation compared to the full proposed framework.

Model	Variant	Exec.(%)	Exec-Acc(%)	LeNS(%)
DS-V3.2	LeNS (Full)	25.2	63.8	77.4
	w/o Abductive	21.3	63.3	67.8 (↓ 9.6)
	w/o Abstraction	0.0	0.0	71.9 (↓ 5.5)
	w/o Coherence	25.6	72.5	76.9 (↓ 0.5)
Gem-3F	LeNS (Full)	50.0	72.1	86.9
	w/o Abductive	45.3	75.8	84.3 (↓ 2.6)
	w/o Abstraction	0.0	0.0	86.5 (↓ 0.4)
	w/o Coherence	46.0	79.4	85.1 (↓ 1.8)

5.4 Error Analysis and Case Studies

The error analysis of 144 incorrect predictions on the COLIEE dataset reveals that translating unstructured narratives into formal representations constitutes the primary bottleneck. Figure 5 and Table 6 in Appendix D demonstrate that the abstraction and formalization modules cause 22% and 45% of the errors, respectively. Specifically, translating unstructured texts frequently introduces predicate mismatch and question target misalignment which completely alters the essential conditions required for unification. Furthermore, the abductive and answer extraction modules account for the remaining 18% and 15% of the errors where the LLM arbitrarily invents unsupported premises and rigid symbolic matching fails to unify synonymous predicates. Ultimately, these failures stem from the inherited semantic mismatch which underscores the necessity of developing flexible semantic unification mechanisms for future neuro-symbolic pipelines.

6 Conclusions

This paper proposes LeNS, a bidirectional neuro-symbolic framework equipped with logic-guided abductive feedback for defeasible legal reasoning. The abstraction and autoformalization modules effectively extract key facts and statutory rules from unstructured contexts to generate executable ASP programs. To overcome the severe vulnerability

Figure 5: Fine-grained error distribution across the four core modules of the proposed framework on the COLIEE dataset.

of traditional one-way pipelines, the abductive mechanism proactively recovers omitted factual premises while the multi-dimensional coherence ranking strategy autonomously resolves legal indeterminacy. Extensive empirical evaluations across multiple benchmarks demonstrate that this framework exhibits superior effectiveness in complex numerical calculations and exceptional robustness against statute shifts.

Transforming unstructured legal narratives into executable formal representations remains the primary bottleneck of our framework. Since the logic-guided abductive mechanism depends on the semantic accuracy of the initial rule formalization, semantic hallucinations in formalized rules inevitably propagate into the abductive module to cause unfounded generations of unsupported factual premises. Our error analysis indicates that the autoformalization and abductive modules account for 45% and 18% of the total errors, respectively. Researchers increasingly employ retrieval-augmented generation (RAG) techniques to mitigate these cascading hallucination issues across different reasoning stages. Nigam et al. (2025) demonstrate the effectiveness of integrating RAG techniques to retrieve relevant statutory rules. Hu et al. (2025) propose a RAG framework to effectively enhance autoformalization for general logical reasoning. Consequently, in future work, we plan to consider utilizing RAG to enhance autoformalization for defeasible legal reasoning.

Limitations

A primary limitation of the proposed LeNS framework is its strict reliance on the explicit provision of all reasoning-related legal articles. Real-world statutory laws constitute a massive knowledge base necessitating a prerequisite retrieval step to extract case-relevant provisions. The current methodology depends entirely on the locally given context and assumes that all applicable articles are comprehensively presented to the system beforehand. Consequently, this underlying assumption restricts the deployment of LeNS in open-ended real-world judicial scenarios.

A critical limitation of the logic-guided abductive feedback mechanism is its strict reliance on the accuracy of the initial rule formalization. Although the iterative probing loop successfully recovers missing factual premises, this abductive process originates entirely from the logical constraints of the extracted rules. Consequently, flawed statutory rules generated by LLMs during the preliminary abstraction phase misdirect the symbolic solver to search for incorrect factual assumptions. This inability to self-correct fundamental rule extraction errors ultimately leaves the framework vulnerable when processing complex semantics.

Guaranteeing the semantic correctness of intermediate autoformalized sentences remains a significant challenge for the proposed LeNS framework. Since the current methodology relies on language model assessments and symbolic solver executions for indirect verification, it fails to formally prove the strict semantic equivalence between the generated logic programs and the original natural language. Although a formalized program might be syntactically executable and yield the expected judgment, its internal representations can still harbor subtle semantic deviations. Consequently, ensuring strict semantic alignment during the autoformalization process constitutes a critical hurdle for future neuro-symbolic research.

References

Kevin D Ashley. 2017. *Artificial intelligence and legal analytics: new tools for law practice in the digital age*.

Chitta Baral and Michael Gelfond. 1994. Logic programming and knowledge representation. *J. Log. Program.*, 19/20:73–148.

Andrew Blair-Stanek, Nils Holzenberger, and Benjamin Van Durme. 2023. Can GPT-3 perform statutory reasoning? In *ICAIL 2023, Braga, Portugal, June 19-23, 2023*, pages 22–31. ACM.

Ilias Chalkidis, Abhik Jana, and et al. 2022. Lexglue: A benchmark dataset for legal language understanding in english. In *ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4310–4330.

Alain Colmerauer and Philippe Roussel. 1993. The birth of prolog. In *HOPL-II, Preprints, Cambridge, Massachusetts, USA, April 20-23, 1993*, pages 37–52. ACM.

Pierre Colombo, Telmo Pessoa Pires, and et al. 2024. Saullm-7b: A pioneering large language model for law. *CoRR*, abs/2403.03883.

Joseph Cotnareanu, Didier Chetelat, and et al. 2026. A balanced neuro-symbolic approach for commonsense abductive logic. *arXiv preprint arXiv:2601.18595*.

Jiaxi Cui, Zongjian Li, and et al. 2023. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *CoRR*, abs/2306.16092.

Matthew Dahl, Varun Magesh, and et al. 2024. Large legal fictions: Profiling legal hallucinations in large language models. *CoRR*, abs/2401.01301.

Yongfu Dai, Duanyu Feng, and et al. 2025. Laiw: A chinese legal large language models benchmark. In *COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 10738–10766.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

Wentao Deng, Jiahuan Pei, and et al. 2023. Syllogistic reasoning for legal judgment analysis. In *EMNLP 2023, Singapore, December 6-10, 2023*, pages 13997–14009. Association for Computational Linguistics.

Shehzaad Dhuliawala, Mojtaba Komeili, and et al. 2024. Chain-of-verification reduces hallucination in large language models. In *ACL 2024*, pages 3563–3578.

Zhiwei Fei, Xiaoyu Shen, and et al. 2024. Lawbench: Benchmarking legal knowledge of large language models. In *EMNLP, Miami, FL, USA, November 12-16, 2024*, pages 7933–7962.

Yi Feng, Chuanyi Li, and Vincent Ng. 2022. Legal judgment prediction via event extraction with constraints. In *ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 648–664.

M Gelfond and V Lifschitz. The stable model semanticsfor logic programming. In *5th Conference on Logic Programming*, pages 1070–1080.

Michael Gelfond and Vladimir Lifschitz. 1988. The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, USA, August 15-19, 1988 (2 Volumes)*, pages 1070–1080. MIT Press.

Randy Goebel, Yoshinobu Kano, and et al. 2024. Overview of benchmark datasets and methods for the legal information extraction/entailment competition (coliee) 2024. In *JSAI International Symposium on Artificial Intelligence*, pages 109–124. Springer.

Zhuo Han, Yi Yang, and et al. 2026. Lawshift: Benchmarking legal judgment prediction under statute shifts. *Advances in Neural Information Processing Systems*, 38.

Nils Holzenberger, Andrew Blair-Stanek, and Benjamin Van Durme. 2020. A dataset for statutory reasoning in tax law entailment and question answering. In *SIGKDD & (KDD 2020), Virtual Workshop, August 24, 2020*, CEUR Workshop Proceedings, pages 31–38. CEUR-WS.org.

Nils Holzenberger and Benjamin Van Durme. 2023. Connecting symbolic statutory reasoning with legal information extraction. In *Proceedings of the Natural Legal Language Processing Workshop 2023*, pages 113–131.

Ruikang Hu, Shaoyu Lin, Yeliang Xiu, and Yongmei Liu. 2025. LTRAG: enhancing autoformalization and self-refinement for logical reasoning with thought-guided RAG. In *ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, Findings of ACL, pages 2483–2493.

Quzhe Huang, Mingxu Tao, and et al. 2023. Lawyer llama technical report. *CoRR*, abs/2305.15062.

William Jurayj, Nils Holzenberger, and Benjamin Van Durme. 2026. Language models and logic programs for trustworthy tax reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 38688–38698.

Manuj Kant, Sareh Nabi, and et al. 2025. Towards robust legal reasoning: Harnessing logical llms in law. *arXiv preprint arXiv:2502.17638*.

Daniel Martin Katz, Michael James Bommarito, and et al. 2024. Gpt-4 passes the bar exam. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 382(2270).

Sarah B Lawsky. 2017. A logic for statutes. *Fla. Tax Rev.*, 21:60.

Denis Merigoux, Nicolas Chataing, and Jonathan Protzenko. 2021. Catala: a programming language for the law. *Proc. ACM Program. Lang.*, 5(ICFP):1–29.

Shubham Kumar Nigam, Balaramamahanthi Deepak Patnaik, and et al. 2025. Nyayarag: Realistic legal judgment prediction with RAG under the indian common law system. In *IJCNLP-AACL 2025, Mumbai, India, December 20-24, 2025*, pages 1709–1726.

OpenAI. 2025. Openai o3 and o4-mini system card. [Online]. Accessed April 22, 2025.

Liangming Pan, Alon Albalak, and et al. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *EMNLP 2023, Singapore, December 6-10, 2023*, Findings of ACL, pages 3806–3824.

Henry Prakken and Giovanni Sartor. 2004. The three faces of defeasibility in the law. *Ratio Juris*, 17(1):118–139.

Giovanni Sartor. 1993. A simple computational model for nonmonotonic and adversarial legal reasoning. In *ICAIL '93, Amsterdam, The Netherlands, June 15-18, 1993*, pages 192–201. ACM.

Ken Satoh, Kento Asai, and et al. 2010. PROLEG: an implementation of the presupposed ultimate fact theory of japanese civil code by PROLOG technology. In *New Frontiers in Artificial Intelligence - JSAI-isAI 2010 Workshops, LENLS, JURISIN, AMBN, ISS, Tokyo, Japan, November 18-19, 2010, Revised Selected Papers*, Lecture Notes in Computer Science, pages 153–164. Springer.

Aaditya Singh, Adam Fry, and et al. 2025. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*.

Gemini Team, Rohan Anil, and et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Jason Wei, Xuezhi Wang, and et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Zhun Yang, Adam Ishay, and Joohyung Lee. 2023. Coupling large language models with logic programming for robust and general reasoning from text. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5186–5219.

A Datasets Details

A.1 Dataset Statistics

This section introduces the statistical characteristics and data distribution conditions of the four distinct legal reasoning datasets employed in the experiments. The SARA dataset originates from United States tax law. It contains two distinct sub-tasks including classification and numerical tax prediction. The classification task comprises 276 samples balanced between entailment and contradiction. The tax prediction task contains 100 tax calculation cases. The LawBench dataset is a Chinese dataset derived from Chinese criminal law cases. The experiment randomly extracts 100 samples from the ID 3-5 prison term prediction sub-dataset and manually constructs a prison term range classification for each sample involving 9 imbalanced prison term category labels. The LawShift benchmark is also a Chinese dataset designed to evaluate robustness under Chinese criminal law. The experiment selects two subsets including Term-up and Term-down for evaluation. Both subsets contain 215 prison term classification samples with 3-class labels. Finally the COLIEE Task 4 dataset evaluates entailment based on Japanese civil law. The experiment randomly extracts a subset of 100 samples with an equal label distribution.

Table 4 summarizes the macro-level statistical information across all datasets and their respective sub-tasks. The table delineates the jurisdiction, language, task formulation, sample size, average token length, and output space.

Dataset	Jurisdiction	Lang.	Task Type	Size	Avg. Len.	Output Space
COLIEE (Task 4)	Japanese Civil Law	EN	Binary Classification	100	280.1	Balanced (50 Yes / 50 No)
LawBench	Chinese Criminal Law	ZH	Regression & 9-Class Cls.	100	1,325.8	Months & 9 Categories (Imbalanced)
SARA (Classification)	U.S. Tax Law	EN	Binary Classification	276	675.2	Balanced (138 Ent. / 138 Contra.)
SARA (Regression)	U.S. Tax Law	EN	Numerical Regression	100	7,298.4	Continuous Tax Values
LawShift (Term-up)	Chinese Criminal Law	ZH	3-Class Classification	215	1,841.7	3 Categories (0-3, 3-10, >10 yrs)
LawShift (Term-down)	Chinese Criminal Law	ZH	3-Class Classification	215	1,841.7	3 Categories (0-3, 3-10, >10 yrs)

Table 4: Macro-level statistics of the evaluation datasets and specific sub-tasks. LawBench includes an imbalanced 9-class categorization (e.g., 0-24 months, 24-84 months, etc.) alongside numerical month regression. Both COLIEE and SARA classification evaluations utilize strictly balanced subsets randomly sampled from their original corpora.

A.2 Dataset Samples

This section provides an example for each dataset.

Task Type: Binary Classification (Entailment)
Case Narrative: Alice's income in 2015 is $100000. She gets one exemption of $2000 for the year 2015 under section 151(c). Alice is not married.
Statutes: section 151(a):
(a) Allowance of deductions
In the case of an individual, the exemptions provided by this section shall be allowed as deductions in computing taxable income.
section 151(b):
(b) Taxpayer and spouse
An exemption of the exemption amount for the taxpayer; and an additional exemption of the exemption amount for the spouse of the taxpayer if a joint return is not made by the taxpayer and his spouse, and if the spouse, for the calendar year in which the taxable year of the taxpayer begins, has no gross income and is not the dependent of another taxpayer.
section 151(d)(1):
(1) In general
Except as otherwise provided in this subsection, the term "exemption amount" means $2,000.
section 151(c):
(c) Additional exemption for dependents
An exemption of the exemption amount for each individual who is a dependent (as defined in section 152) of the taxpayer for the taxable year.
Question: Alice's total exemption for 2015 under section 151(a) is equal to $4000.
Label: `Entailment`

Figure 6: An entailment classification sample from the SARA dataset under United States tax law.

Task Type: Numerical Regression (Tax Calculation)
Case Narrative: Alice was paid $1200 in 2019 for services performed in jail. Alice was committed to jail from January 24, 2015 to May 5th, 2019. From May 5th 2019 to Dec 31st 2019, Alice was paid $5320 in remuneration. Alice takes the standard deduction.
Statutes: section 1(a):
(a) Married individuals filing joint returns and surviving spouses
There is hereby imposed on the taxable income of-
(1) every married individual (as defined in section 7703) who makes a single return jointly with his spouse, and
(2) every surviving spouse (as defined in section 2(a)),
a tax determined in accordance with the following:
(i) 15% of taxable income if the taxable income is not over $36,900;
(ii) $5,535, plus 28% of the excess over $36,900 if the taxable income is over $36,900 but not over $89,150;
(iii) $20,165, plus 31% of the excess over $89,150 if the taxable income is over $89,150 but not over $140,000;
(iv) $35,928.50, plus 36% of the excess over $140,000 if the taxable income is over $140,000 but not over $250,000;
(v) $75,528.50, plus 39.6% of the excess over $250,000 if the taxable income is over $250,000.
...
Question: How much tax does Alice have to pay in 2019?
Label: `$0`

Figure 7: A continuous numerical regression sample from the SARA dataset under United States tax law.

Task Type: Binary Classification (Entailment)
Article: Article 698
If a manager engages in benevolent intervention in another's business in order to allow a principal to escape imminent danger to the principal's person, reputation, or property, the manager is not liable to compensate for damage resulting from this unless the manager has acted in bad faith or with gross negligence.
query: In cases where an individual rescues another person from getting hit by a car by pushing that person out of the way, causing the person's luxury kimono to get dirty, the rescuer does not have to compensate damages for the kimono.
Label: `Y` (Entailment)

Figure 8: A binary entailment sample from the COLIEE Task 4 dataset under Japanese Civil Law.

Task Type: Numerical Regression & 9-Class Categorization **Fact Description:** The prosecution alleges that between 2011 and 2014, the defendant Zheng, while serving as the director of a township health center, took advantage of his official position to illegally accept kickbacks from various suppliers during the procurement of medical supplies, vaccines, and medical equipment. Zheng received a total of 121,960 RMB in kickbacks. Subsequently, Zheng's relatives refunded 172,429 RMB of illicit funds on his behalf. **Charge:** Accepting Bribes **Label:** `18 Months` (Category: `Under 3 Years`)	任务类型**:** 数值回归与9分类 事实描述**:** 检察院起诉书指控：2011年至2014年，被告人郑某在担任某乡卫生院院长期间，利用职务便利，在采购医药用品、狂犬疫苗及医疗器械设备过程中，先后多次收受供应商给予的回扣共计人民币121,960元。案发后，被告人郑某亲属替其退缴违纪资金人民币172,429 元。 罪名**:** 受贿罪 真实标签**:** `18` 个月 (类别: 三年以下)

Figure 9: A prison term prediction sample from the LawBench dataset under Chinese Criminal Law.

Original Example	原始样本
Task Type: 3-Class Classification (Prison Term) **Fact Description:** The prosecution alleges that on November 25, 2019, the defendant Yang had a conflict with the victim and repeatedly hacked the victim with a kitchen knife. Upon another person's plea, Yang ceased the attack and fled the scene. Forensic appraisal concluded that the victim sustained minor injuries. **Original Statute:** Article 232 (Intentional Homicide): ... for relatively minor circumstances, the sentence shall be between 3 and 10 years of imprisonment. **Label:** `42 Months` (Category: `3-10 Years`)	任务类型**:** 3分类(刑期预测) 事实描述**:** 公诉机关指控：2019年11月25日，被告人羊某与被害人发生冲突，持菜刀连续砍击被害人。在他人哀求下，被告人停止加害并逃离现场。经鉴定，被害人伤势构成轻伤。 原始法条**:** 《刑法》第二百三十二条（故意杀人罪）：.....情节较轻的，处三年以上十年以下有期徒刑。 真实标签**:** `42`个月 (类别: `3-10`年)

Poisoned Example	修改样本
Fact Description: [Identical to the original facts] **Poisoned Statute:** Article 232 (Intentional Homicide) was deliberately poisoned: ... for relatively minor circumstances, the statutory penalty is manipulated to <u>over 10 years</u> of imprisonment. **Ground Truth Label:** `Over 10 Years` (Forcibly upgraded)	事实描述**:** [与原始事实完全一致] 修改法条**:** 《刑法》第二百三十二条被蓄意篡改：.....情节较轻的，法定刑期被强制上调<u>至十年以上</u>有期徒刑。 真实标签**:** `10`年以上 (强制升级)

Figure 10: A statute shift sample from the LawShift (Term-up) dataset based on Chinese Criminal Law. / 基于中国刑法的LawShift (Term-up) 数据集下法条变动的样本。

Original Example 原始样本

Task Type: 3-Class Classification (Prison Term)
Fact Description: The prosecution alleges that on November 25, 2019, the defendant Yang had a conflict with the victim and repeatedly hacked the victim with a kitchen knife. Upon another person's plea, Yang ceased the attack and fled the scene. Forensic appraisal concluded that the victim sustained minor injuries.
Original Statute: Article 232 (Intentional Homicide): ... for relatively minor circumstances, the sentence shall be between 3 and 10 years of imprisonment.
Label: `42 Months` (Category: `3-10 Years`)

任务类型: 3分类(刑期预测)
事实描述: 公诉机关指控：2019年11月25日，被告人羊某与被害人发生冲突，持菜刀连续砍击被害人。在他人哀求下，被告人停止加害并逃离现场。经鉴定，被害人伤势构成轻伤。
原始法条: 《刑法》第二百三十二条（故意杀人罪）：.....情节较轻的，处三年以上十年以下有期徒刑。
真实标签: `42`个月 (类别: `3-10`年)

Poisoned Example 修改样本

Fact Description: [Identical to the original facts]
Poisoned Statute: Article 232 (Intentional Homicide) was deliberately poisoned: ... for relatively minor circumstances, the statutory penalty is manipulated to <u>under 3 years</u> of imprisonment.
Ground Truth Label: `0-3 Years` (Forcibly downgraded)

事实描述: [与原始事实完全一致]
修改法条: 《刑法》第二百三十二条被蓄意篡改：.....情节较轻的，法定刑期被强制下调至<u>三年以下</u>有期徒刑。
真实标签: 三年以下 (强制降级)

Figure 11: A statute shift sample from the LawShift (Term-down) dataset based on Chinese Criminal Law. / 基于中国刑法的LawShift (Term-down) 数据集下法条变动的样本。

B Implementation Details and Prompts

B.1 Hyperparameter Configurations

This section details the hyperparameter configurations and execution environments in the experiments. All experiments utilize a fixed random seed of 42. The large language models employ a dynamic temperature strategy based on specific module requirements. The fact extraction, logical formalization and symbolic-to-text translation tasks utilize a decoding temperature of 0.0 while ensemble voting and coherence ranking apply a temperature of 0.7. Finally all modules set the maximum generation length to 2048 tokens.

B.2 Algorithm of Abductive Feedback

The execution workflow of the proposed logic-guided abductive feedback mechanism is formally outlined in Algorithm 1. The system initially invokes the symbolic solver to compute the initial backbone from the given logic program. The algorithm explicitly illustrates how the framework iteratively aggregates missing prerequisites and justifications from the blocked rules if the entailment status of the query remains undetermined. The system subsequently scores and sorts these abductive targets into a unified descending list by prioritizing rule prerequisites over justifications and favoring shallower reasoning depths. Following this prioritization the framework invokes large language models in a single batch-processing inference step to evaluate the sorted targets to generate potentially omitted facts. The framework utilizes these models to directly calculate the textual consistency score against the unstructured case facts for each abduced fact alongside its legal plausibility score. Finally the system accumulates the valid abduced facts to inject them into the logic program and subsequently invokes the symbolic solver to recompute the updated backbone. This iterative reasoning process strictly terminates after three rounds or prematurely concludes when two consecutive backbones remain identical to ensure computational efficiency.

Algorithm 1 Logic-Guided Abductive Feedback

Require: Logic Program Π, Query Q, Unstructured Case Facts F, Threshold τ
Ensure: Final Logic State, Updated Backbone B_i

```
Initialize i ← 0
B_0 ← SymbolicSolver(Π)                                   ▷ Generate initial backbone
while i < 3 do
                                                          ▷ M3.1 Abductive Target Identification
    if Q ∈ B_i return (Entailment, B_i)
    if ¬Q ∈ B_i return (Contradiction, B_i)
    R_rel ← {r ∈ Π | r is blocked}; PQ ← ∅
    for each rule r ∈ R_rel do
        P_miss ← {p ∈ body^+(r) | p ∉ B_i}                ▷ Missing prerequisites
        N_miss ← {n | not n ∈ body^-(r) ∧ n ∉ B_i}        ▷ Missing justifications
        for each target t ∈ P_miss ∪ N_miss do
            ω ← (if t ∈ P_miss then w_pre else w_just)    ▷ Prioritize prerequisites (w_pre > w_just)
            Score(t) ← W_1 · ω + W_2/(Depth(t, Π) + 1)    ▷ Prioritize shallower depth
            PQ.push(t, Score(t))
    if PQ is empty break
    A ← SortDescending(PQ)                                ▷ Unified list of sorted abductive targets
                                                          ▷ M3.2 Hypothesis Generation & Verification
    ΔF ← LLM(F, B_i, prompt(A))                           ▷ Batch inference returns potentially omitted facts
    ΔF_valid ← ∅
    for each abduced fact f ∈ ΔF do                       ▷ Dual validation
        if S_textual(f, F) > τ and S_legal(f) > τ then
            ΔF_valid ← ΔF_valid ∪ {f}                     ▷ Accumulate valid facts
    if ΔF_valid == ∅ break                                ▷ No new facts abduced
    Π ← Π ∪ ΔF_valid                                      ▷ Inject new facts into the logic program
    B_{i+1} ← SymbolicSolver(Π)                           ▷ Recompute the backbone
    if B_{i+1} == B_i break                               ▷ Terminate if backbone converges
    i ← i + 1
return (Unknown, B_i)
```

B.3 Generalized Prompt Templates

The generalized prompt engineering paradigm across the six core modules employs a dataset-agnostic design utilizing structural placeholders instantiated with domain-specific inputs. Figure 12 through Figure 17 present the independent prompts for these modules. The figures instantiate all prompt templates using a representative binary entailment classification case drawn from the SARA dataset.

Prompt for Module 1: Fact and Rule Extraction

System Instruction: Task Given the context (case text), statute (legal rule), and question, extract atomized facts and DEFAULT RULES for logical reasoning.
Important Guidelines
1. Information not related to reasoning can be simplified or omitted.
2. Each extracted fact and rule should be atomic (one relationship per statement).
3. The extracted facts CANNOT contradict each other.
4. CRITICAL: Rules often have EXCEPTIONS. You MUST identify and preserve them!
5. INDEPENDENCE RULE: All inferred facts must be derivable from the case facts and statutes ALONE, WITHOUT assuming the question's conclusion is correct. Do NOT infer or list any fact that would only be true if the question statement were already assumed to be true. The question is a claim to VERIFY, not a premise to use.
6. **ABDUCTIVE REASONING**: You MUST infer HIDDEN FACTS that are necessary to answer the question but not explicitly stated: - Apply statute definitions to the case facts (e.g., if statute says "taxpayer gets exemption", and Alice is a taxpayer, infer "Alice gets taxpayer exemption") - Derive implicit conditions from explicit facts (e.g., "Alice is not married" implies "Alice cannot file joint return") - For NON-NUMERIC logical inferences (relationships, eligibility, status), write them as '[INFERRED]' facts directly. Example: "[INFERRED from statute 152(d)(2)(C)] Bob bears a relationship to Alice under §152(d)(2)(C)." - For NUMERIC CALCULATIONS (tax amounts, totals, percentages, reductions): Show the full step-by-step calculation process, then write the final result as an '[INFERRED]' fact using the section-specific predicate name from the question. Use the exact computed value — the system will handle rounding automatically.
Strict Fidelity Principle
When extracting rules, sentence structure and phrasing may be rewritten (encouraged to rewrite into default rule format, e.g., "If … then … unless … "), but you must **strictly preserve the exact numeric values of each sentencing/amount tier in the original statute**. Do NOT use legal common sense or convention to alter the conclusion of a statute rule. Even if a statutory provision appears inconsistent with typical legal knowledge (e.g., a "minor circumstance" clause leads to a higher penalty), you must preserve that conclusion unchanged in the rewritten rule.
Definitions
Fact: A specific, unconditional statement about particular entities. **Default Rule:** A conditional rule that may have EXCEPTIONS.
- Format: "If CONDITION, then CONCLUSION unless EXCEPTION" - IMPORTANT: Legal rules almost always have implicit exceptions. Identify them!
Input Format:
The facts are: [case facts] The statutes are: [statute content] The question is: [question text]
Output Format (JSON):
"Extracted facts": ["fact1", "fact2"], "Extracted rules": ["rule1", ...], "Extracted question": ["simplified question statement"]
Example 1 :
The facts are: Alice's income in 2015 is $100000. She gets one exemption of $2000 for the year 2015 under section 151(c). Alice is not married.
The statutes are: section 151(a):
(a) Allowance of deductions
In the case of an individual, the exemptions provided by this section shall be allowed as deductions in computing taxable income.
section 151(b):
(b) Taxpayer and spouse
An exemption of the exemption amount for the taxpayer; and an additional exemption of the exemption amount for the spouse of the taxpayer if a joint return is not made by the taxpayer and his spouse.
section 151(c):
(c) Additional exemption for dependents
An exemption of the exemption amount for each individual who is a dependent (as defined in section 152) of the taxpayer for the taxable year.
The question is: Alice's total exemption for 2015 under section 151(a) is equal to $6000.
Extracted facts: ["Alice has an exemption of $2000 for the year 2015 under section 151(c).", "[INFERRED from statute 151(b)] Alice as a taxpayer gets $2000 personal exemption under section 151(b).", "[INFERRED] Total exemption under section 151(a) = 151(b) $2000 + 151(c) $2000 = $4000."]
Extracted rules: ["If exemptions exist under section 151(b) and 151(c), the total is allowed as a deduction under section 151(a)."] Extracted question: ["Alice's total exemption for 2015 under section 151(a) is $6000."]
Output only the JSON, nothing else.

Figure 12: Prompt template for the fact and rule extraction module.

Prompt for Module 2: Logical Formalization (NL to ASP)

Instruction: Task Translate the given natural language sentence into Default Logic (Reiter format).
Default Logic Syntax: Default Logic is a non-monotonic reasoning formalism introduced by Raymond Reiter. It distinguishes between two types of statements: facts and default rules.
A fact is an unconditional ground statement represented as ‘predicate(entity1, entity2).‘ where all arguments are lowercase constants. For instance, ‘married(alice, bob).‘ asserts that Alice is married to Bob, and ‘income(alice, 50000, 2015).‘ states that Alice’s income in 2015 is 50000.
A default rule follows the Reiter format ‘prerequisite : justification / conclusion‘, which reads as: "If the prerequisite holds and the justification is consistent with what is known, then conclude the conclusion." The justification component is the key mechanism for handling exceptions. When encountering natural language patterns like "A unless B", the exception B becomes the justification with classical negation: ‘A : -B / conclusion‘. Here, the ‘-‘ symbol denotes classical negation, meaning B is definitively false, which translates to ‘not B‘ (negation as failure) in Answer Set Programming.
When there is no exception in a conditional statement such as "If A, then C", the justification is simply ‘TRUE‘, yielding ‘A : TRUE / C.‘ as the translation. For statements with multiple exceptions like "If A, then C unless B1 or B2", each exception appears negated in the justification: ‘A : -B1, -B2 / C.‘
In terms of naming conventions, variables are written in UPPERCASE (such as X, Y, Person, Year) while constants are written in lowercase (such as alice, bob, 2015). Default rules with conditions must use the Reiter format, whereas simple ground facts can be written directly without the Reiter structure.
Variable Safety All variables in the CONCLUSION must appear in the PREREQUISITE, not in the justification. Variables in the justification (negated part) do NOT bind variables in ASP.
- WRONG: ‘individual(X) : -exception(X, Year) / result(X, Year).‘ → Year only in justification, NOT SAFE!
- CORRECT: ‘individual(X, Year) : -exception(X, Year) / result(X, Year).‘ → Year in prerequisite
If a variable cannot be bound by a prerequisite predicate, use a specific constant instead.
Predicate Consistency
When translating a sentence, keep the predicate naming STYLE and argument STRUCTURE consistent with the context, but you MUST faithfully preserve the SEMANTICS of the natural language sentence being translated. Check the provided "extracted predicates from context" and "translated context" carefully:
- If the context uses ‘s151_d1_exemption(Person, Amount, Year)‘ with 3 arguments, follow the SAME naming style and argument structure
- If the context uses ‘exemption(Person, Amount, Year)‘, do NOT invent ‘exemption_amount(Amount)‘ with different argument structure
- Match the predicate NAME and the NUMBER/ORDER of arguments for consistency
> VALUE PRESERVATION — ABSOLUTE RULE > The numeric values, names, and years in the **sentence you are translating** are SACRED. > NEVER copy a value from the context examples into your output if the sentence itself states a different value. > This rule applies equally whether you are translating a fact, a rule, or a question/hypothesis. > The context examples exist only to tell you the predicate NAME and argument ORDER — nothing else.
Examples of WRONG vs CORRECT translations: - Context has: ‘s151_a(alice, 4000, 2015)‘ (derived from earlier facts) - Sentence to translate: "Alice’s total exemption for 2015 under section 151(a) is equal to $6000."
- WRONG: ‘s151_a(alice, 4000, 2015).‘ ← Copied $4000 from context — this destroys the sentence’s meaning!
- CORRECT: ‘s151_a(alice, 6000, 2015).‘ ← Predicate name/order from context, but value $6000 from the sentence itself
- Context has: ‘s151_d1_exemption(alice, 2000, 2015)‘ - Sentence: "The exemption amount is $3000"
- WRONG: ‘exemption_amount(3000).‘ ← Different predicate name, fewer arguments
- CORRECT: ‘s151_d1_exemption(alice, 3000, 2015).‘ ← Same predicate style, but preserves the sentence’s own value $3000

Input Format
The extracted predicates from context are: [predicates list] **The extracted entities from context are:** [entities list] **The translated context are:** NL:"", DL:"". **The natural language sentence to be translated is:** "[sentence]"
Output Format (JSON)
"predicates": ["predicate1", "predicate2"], "entities": ["entity1", "entity2"], "default logic": "the default logic statement"
Example 1:
Input: - The extracted predicates from context are: "None" - The extracted entities from context are: "None" - The translated context are: "(No additional context)" - The natural language sentence to be translated is: "Alice has an exemption of $2000 for the year 2015 under section 151(c)."
Output: "predicates": ["s151_c"], "entities": ["alice", "2000", "2015"], "default_logic": "s151_c(alice, 2000, 2015)."

Output only JSON, nothing else."""

Figure 13: Prompt template for the logical formalization module.

Prompt for Module 3: Abductive Probing (Missing Premise Inference)

Instruction: You are an expert legal reasoning analyst specializing in U.S. federal tax law. Based on "Question + Abstract Context + Formalized Context + Original Context", perform alignment-based gap-filling: identify facts or rules that may affect the tax entailment judgment but were missed by the extraction/formalization pipeline.
Input Schema:
You will receive the following 4 sections in the user message: 1. Original Case Text (‘context‘) - The original case text, the primary source of factual evidence.
2. Question to Answer (‘question‘) - A tax-related statement to verify (Entailment or Contradiction), determining the focus of this round of probing.
3. Abstract Context (Already Extracted Information) (‘current_items_str‘) - Extracted facts and rules (usually natural language summaries).
4. Formalized Context (ASP / Formal Rules) (‘asp_context‘) - Formalized rules and atoms (logical-layer representation).
Core Principles:
1. Question-driven: Only supplement information that could affect the answer (tax amount or entailment judgment).
2. Original text anchoring: missing_facts / missing_rules must have explicit support in the Original Case Text, or be a highly stable legal semantic inference.
3. Dual-context cross-checking: - Abstract context: check whether semantics are already covered. - Formalized context: check whether logic is already encoded. - Only supplement if both are missing AND the original text provides evidence.
4. Avoid hallucination and redundancy: - Do not add information with no support in the original text. - Do not repeat already extracted/formalized semantically equivalent information.
5. Atomic expression: Each missing_fact contains only one atomic relation; missing_rules must be directly usable by the formalization module.
Suggested Checking Workflow:
Step 1: Identify required information slots around the question e.g., filing status, income amounts, deduction eligibility, exemption conditions, statute applicability thresholds, taxable income calculation.
Step 2: Cross-check Abstract Context Determine whether these slots are already covered by existing extraction; mark "abstract layer suspected gap" if not.
Step 3: Cross-check Formalized Context Determine whether key conditions are already encoded in predicates/rules; mark "formalized layer suspected gap" if not.
Step 4: Return to original text evidence Retain only gaps supported by the original text, output to missing_facts / missing_rules.
Priority Check Areas (Tax-Domain-Specific) - Filing status and its impact on tax brackets - Gross income vs. exclusions (e.g., §3306(c)(21) jail wages) - Deduction eligibility (standard vs. itemized, phase-out rules like §68) - Exemption amounts (§151) and exemption phase-out (§151(d)(5)) - Tax bracket thresholds and rate applicability - Statute section references (§1, §63, §151, §152, etc.)
Output Format (JSON): "analysis": "Alignment analysis based on question-abstract context-formalized context-original context", "has_missing": true/false, "missing_facts": ["Missing facts (each with a single atomic relation)"], "missing_rules": ["Missing rules"]
Output Constraints: - Output JSON only, no extra explanation. - If no gaps: ‘has_missing=false‘ and ‘missing_facts=[]‘, ‘missing_rules=[]‘.

Figure 14: Prompt template for the Abductive Probing module.

Prompt for Module 4: Symbolic-to-Text Translation

Instruction: Convert the following ASP (Answer Set Programming) predicates to natural language.
ASP Syntax:
Answer Set Programming is a declarative logic programming paradigm. An ASP program consists of facts, rules, and constraints.
A fact is a ground atom that is unconditionally true, written as ‘predicate(arg1, arg2).‘ with a period at the end. For instance, ‘married(alice, bob).‘ means "Alice is married to Bob", and ‘income(alice, 50000, 2015).‘ means "Alice’s income in 2015 is $50000".
A rule derives new facts from existing ones using the format ‘head :- body.‘, which reads as "head is true if body is true". For example, ‘single(X) :- person(X), not married(X, _).‘ means "A person is single if they are a person and are not married to anyone".
Negation in ASP uses the keyword ‘not‘, which represents negation as failure. The expression ‘not p(X)‘ means "there is no evidence that p(X) is true", which is different from classical negation. When translating negated predicates to natural language, use explicit negative phrasing: ‘not joint_return(alice, bob, 2015)‘ becomes "Alice and Bob did NOT file a joint return in 2015", and ‘not married(alice)‘ becomes "Alice is NOT married".
An **integrity constraint** has the form ‘:- condition.‘ and rejects any model where the condition is true. For example, ‘:- married(X,Y), single(X).‘ means "It cannot be the case that someone is both married and single".
Examples:
ASP: ‘joint_return(alice, bob, 2015)‘ → NL: "Alice and Bob filed a joint return in 2015"
ASP: ‘dependent(bob, carol)‘ → NL: "Bob is Carol’s dependent"
ASP: ‘blind(alice)‘ → NL: "Alice is blind"
ASP: ‘exemption_blocked(alice, bob, 2015)‘ → NL: "Alice’s exemption for Bob in 2015 is blocked"
ASP: ‘not spouse_exemption_allowed(alice, bob, 2015)‘ → NL: "Spouse exemption for Bob is NOT allowed for Alice in 2015"
ASP: ‘spouse_exemption(A,B) :- married(A,B), not joint_return(A,B,_).‘ → NL: "Spouse exemption is allowed if married AND no joint return is filed"
CRITICAL: A predicate WITHOUT the ‘not‘ keyword is ALWAYS a positive/affirmative statement. Only predicates explicitly prefixed with ‘not‘ should be translated as negative.
Output Format: Output only natural language descriptions, one per line. No other content."""

Figure 15: Prompt template for the symbolic-to-text translation module.

Prompt for Module 5a: Coherence Ranking (Textual consistency Scoring)

Instruction: You are a textual consistency expert. Determine if facts in Hypothesis can be derived from Premise.
Scoring Guidelines (0-100, use continuous scale) Provide a precise score based on the degree of textual consistency: -
95-100: All facts are explicitly stated or directly derivable from the premise
- 85-94: Most facts are derivable, with minor reasonable inferences
- 70-84: Majority derivable, but some facts require moderate inference
- 50-69: Partially derivable, several facts lack textual support
- 30-49: Mostly speculative, limited textual grounding
- 0-29: Largely unsupported, likely hallucination

IMPORTANT: Use the full range of scores. Avoid clustering around common values like 50, 70, or 100. Small differences matter - distinguish between similar but not identical cases (e.g., 72 vs 78).

Output Format (JSON): "score": 76, "reason": "Scoring reason"

Prompt for Module 5b: Coherence Ranking (Legal plausibility Scoring)

Instruction: You are a legal expert evaluating the legal reasonableness of explanations. Evaluate if the answer set explanation conforms to legal principles and common sense.
Scoring Guidelines (0-100, use continuous scale): Provide a precise score based on legal soundness:
- 95-100: Fully conforms to legal principles, logically impeccable
- 85-94: Strong legal reasoning with minor stylistic issues
- 70-84: Generally reasonable, some arguable interpretations
- 50-69: Contains legal logic gaps or questionable assumptions
- 30-49: Significant legal reasoning problems
- 0-29: Seriously violates legal principles or defies common sense

IMPORTANT: Use the full range of scores. Avoid clustering around common values like 50, 70, or 100. Small differences matter - distinguish between similar but not identical cases (e.g., 72 vs 78).

Output Format (JSON): "score": 76, "reason": "Scoring reason"

Figure 16: Prompt templates for textual consistency and legal plausibility scoring in the coherence ranking module.

Prompt for Module 6: LLM-based Direct Reasoning (Baseline & Fallback)

Instruction: You are a legal reasoning expert specializing in tax law entailment. The symbolic reasoning system could not determine the answer. Based on the extracted facts and rules, determine whether the question statement holds (Entailment or Contradiction).
- Use the extracted facts and rules as the primary reasoning basis.
- For failed formalization rules listed under "Rules That Failed Formalization", apply them manually in natural language — do NOT default to Contradiction simply because formalization failed.
- If the symbolic status note indicates a strong signal (e.g., SYMBOLIC MATCH or SYMBOLIC CONFLICT), treat it as supporting evidence.

Decision Rules:
1. Numeric questions (highest priority): Check whether the value derivable from the extracted facts matches the question value. Rounding tolerance is ±$1 — differences within $1 count as a match (Entailment). Differences greater than $1 are a mismatch (Contradiction).
2. Non-numeric questions (relationships, eligibility, status): Determine whether the extracted facts and rules explicitly support or contradict the question statement.
3. When evidence is ambiguous: Reason from legal common sense based on the extracted facts, and state your reasoning clearly.

Input Format:
- **Extracted Facts and Rules:** Structured facts and rules extracted from the case, including any rules that failed formalization.
- **NL_Question:** The legal question to verify.
- **Symbolic Reasoning Status:** A note explaining why the symbolic system could not determine the answer.

Example 1: Entailment (numeric match)
Extracted Facts and Rules:
- [FACT] Alice's income in 2015 is $100000.
- [FACT] Alice gets one exemption of $2000 under section 151(c) for 2015.
- [INFERRED] Alice as a taxpayer gets $2000 personal exemption under section 151(b).
- [INFERRED] Total exemption under section 151(a) = $2000 + $2000 = $4000. NL_Question**: Alice's total exemption for 2015 under section 151(a) is equal to $4000.

Output: "answer": "Entailment", "explanation": "The extracted facts show Alice has a personal exemption of $2000 under 151(b) and a dependent exemption of $2000 under 151(c), totaling $4000. This exactly matches the question value, so the statement holds."

Figure 17: Prompt template for the LLM-based direct reasoning module.

C Additional Experimental Results

C.1 Robustness on LawShift Term-down

Figure 18 illustrates the robustness results of the four base models on the adversarial LawShift (Term-down) dataset. Prompt-based LLMs suffer from severe performance degradation due to their reliance on the parametric memory of outdated pre-training knowledge when statutes undergo dynamic amendments to demand shorter prison terms. The proposed LeNS framework consistently maintains high accuracy reaching 98.6% with Gemini-3-flash while the Zero-shot prompt-based LLMs and DirectParse baselines exhibit catastrophic reasoning failures.

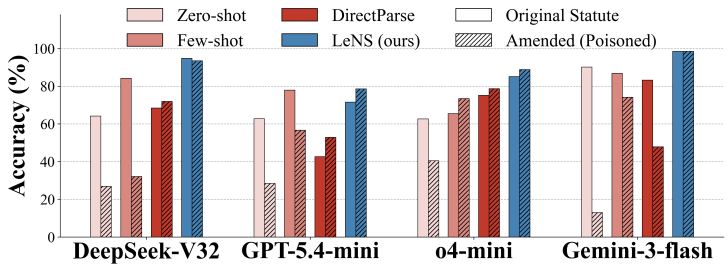

Figure 18: Accuracy comparison on the LawShift (Term-down) dataset before and after statute shifts.

C.2 Ablation Study on the LawBench Dataset

Table 5 presents the ablation study of the proposed LeNS framework on the LawBench dataset. When the abstraction module (w/o Abstraction) is removed the symbolic solver cannot directly process natural language causing the executability to drop to zero. The framework consequently relies entirely on LLMs for direct prediction which leads to a 4.0% decrease in category accuracy for DeepSeek-V32. When the coherence ranking module (w/o Coherence) is removed the LeNS framework based on Gemini-3-flash drops by 4.0% in prison term month accuracy. Finally, when the abductive module (w/o Abductive) is removed the framework fails to dynamically supplement missing legal premises which causes the LeNS framework based on DeepSeek-V32 to decrease by 1.2% in prison term month accuracy.

Table 5: Ablation study results on the LawBench dataset. Values in parentheses indicate the performance change compared to the LeNS framework.

Model	Variant	Exec.(%)	Exec-Acc(%)		LeNS(%)	
			R-Acc	C-Acc	R-Acc	C-Acc
DeepSeek-V32	LeNS (Full)	60.0	78.3	73.3	**78.0**	**71.0**
	w/o Abductive	63.0	77.8	73.0	76.8 (↓ 1.2)	72.5 (↑ 1.5)
	w/o Abstraction	0.0	0.0	0.0	76.0 (↓ 2.0)	67.0 (↓ 4.0)
	w/o Coherence	63.0	81.0	76.2	78.0 (-)	70.0 (↓ 1.0)
Gemini-3-flash	LeNS (Full)	76.0	81.6	73.7	77.0	**69.0**
	w/o Abductive	77.8	81.8	70.1	**78.0** (↑ 1.0)	68.0 (↓ 1.0)
	w/o Abstraction	0.0	0.0	0.0	75.0 (↓ 2.0)	68.0 (↓ 1.0)
	w/o Coherence	91.0	71.4	64.8	73.0 (↓ 4.0)	66.0 (↓ 3.0)

D Fine-Grained Error Analysis Examples

Table 6 reveals the result analysis of the 144 error samples generated by the LeNS framework on the COLIEE dataset. These errors distribute across the four core modules of the framework. The abstraction module accounts for 22% of the total errors. The primary error types of this module include question target misalignment implicit fact omission and over-literal extraction. For instance the language model misaligns the question target by incorrectly extracting the perfection requirements of real rights when queried about their priority. The formalization stage introduces the highest proportion of errors at 45%. The main error types in this stage consist of predicate misalignment logical inconsistency and variable grounding failure. For instance, the model generates mismatched predicates such as a cancellation action by a mortgagor when the formalized rule strictly defines a borrower canceling a mortgage. The abductive module contributes 18% of the errors. The principal error types encompass probing exhaustion and probing hallucination. A specific case demonstrates the model hallucinating a third party consent fact without any explicit textual evidence. Finally the answer extraction module produces the remaining 15% of the error proportion. These errors mainly include symbolic matching errors and semantic fallback reasoning errors. For example the rigid symbolic matching fails to unify a query atom demanding contract cancellation with an answer set atom allowing contract cancellation.

Table 6: Error taxonomy and representative case studies of the 144 incorrect predictions across four base models on the COLIEE dataset. The red text indicates the key clauses for each error type.

Module	Error Type	Prop.	Representative Case and Phenomenon
Abstraction	Question Target Misalignment	12%	**Input** The real rights that formed earlier in time take priority. **Output** Real rights do not require requirements of perfection in this context.
	Implicit Fact Omission	6%	**Input** The obligee exercises the credit of sale value vested in the obligor. **Output** Failed to infer the implicit fact that the obligee is exercising a subrogor's right.
	Over-literal Extraction	4%	**Input** The person believes a fictitious explanation of the counterparty. **Output** Illegally hallucinates the conclusion that the manifestation was induced by fraud.
Formalization	Predicate Misalignment	20%	**Rule** `borrower_may_cancel_contract(mortgage)` **Question** `cancel_contract(mortgagor)`
	Logical Inconsistency	15%	**Rule 1** `voidable(Act) :- juridical_act(Act), performed_by(Act, adult_ward).` **Rule 2** `-voidable(Act) :- gift_contract(Act).`
	Variable Grounding Failure	10%	**Input** If the possessor has exercised the necessary care to prevent the damage, the owner must compensate for the damage. **Output** Variables fail to map correctly to specific entities (e.g., `liable_for_damage(X, _) :- exercised_necessary_care(X)`).
Abductive	Probing Exhaustion	12%	**Phenomenon** Probe results return UNKNOWN for three consecutive rounds resulting in exhausted probing with no facts supplemented.
	Probing Hallucination	6%	**Question** The third party consents to the arrangement. **Addition** Hallucinates `consents(third_party)` without textual evidence.
Answer Ext.	Symbolic Matching Error	8%	**Query Atom** `demand_cancellation_of_contract(obligor)` **Answer Set** `may_cancel_contract(obligor)`
	LLM Reasoning Error	7%	**Phenomenon** The semantic fallback model generates flawed reasoning steps when processing the raw text of complex cases.