Towards Robust Legal Reasoning: Harnessing Logical LLMs in Law

Anonymous ACL submission

Abstract

Legal services rely heavily on text processing. While large language models (LLMs) show promise, their application in legal contexts demands higher accuracy, repeatability, and transparency. Logic programs, by encoding legal concepts as structured rules and facts, offer reliable automation but require sophisticated text extraction. We propose a neuro-symbolic approach that integrates LLMs' natural language understanding with logic-based reasoning to address these limitations.

As a legal document case study, we applied neuro-symbolic AI to coverage-related queries in insurance contracts using both closed and open-source LLMs. While LLMs have improved in legal reasoning, they still lack the accuracy and consistency required for complex contract analysis. In our analysis, we tested three methodologies to evaluate whether a specific claim is covered under a contract: a vanilla LLM, an unguided approach that leverages LLMs to encode both the contract and the claim, and a guided approach that uses a framework for the LLM to encode the contract. We demonstrated the promising capabilities of LLM + Logic in the guided approach.

1 Introduction

1.1 Importance of Trustworthy Legal AI

Legal systems rely on rigorous reasoning, explainability, and transparency to ensure fairness and accountability. Unlike many other AI applications, legal decision-making directly affects individuals' rights, obligations, and access to justice. Consequently, AI-driven legal solutions must go beyond surface-level predictions and provide structured, interpretable reasoning.

Expert attorneys engage in complex reasoning beyond pattern recognition. Legal analysis requires System 2 thinking — deliberate and logical reasoning that evaluates statutes, case law, and contracts. Attorneys dissect legal texts, identify principles, and construct arguments based on precedent. Their decisions involve weighing interpretations, assessing nuances, and considering broader implications. Additionally, legal professionals must articulate their reasoning clearly, ensuring their conclusions are defendable against scrutiny from courts, clients, and the opposition.

The sensitive nature of legal queries requires a system that is both correct and interpretable. In the U.S., oversight of AI systems is intensifying, with the Bipartisan House Task Force on Artificial Intelligence (2024) highlighting the need for transparency to prevent deceptive practices and ensure consumer protection. Every legal argument must reference laws, precedents, or contractual clauses, ensuring accountability. Unlike black-box AI, legal reasoning must be auditable, allowing stakeholders to trace conclusions. Without this level of explainability, AI legal tools risk undermining trust and reliability in decision-making.

In parallel, sector-specific supervision in the insurance domain, as in our case study, is evolving; for example, the International Association of Insurance Supervisors (2024) recently published its Draft Application Paper on the Supervision of AI, which calls for rigorous auditability and interpretability standards for AI-driven contract analytics. Under Europe's General Data Protection Regulation, data subjects must be provided "meaningful information about the logic" underlying automated decision-making processes (European Union, 2016). This requirement ensures that individuals can understand, challenge, or seek human intervention regarding algorithmic decisions. Similarly, the proposed EU AI Act mandates that high-risk AI systems be designed with explainability and traceability, ensuring stakeholders can reasonably comprehend the system's functioning and outputs (European Union, 2024).

As AI increasingly integrates into legal work-

flows, the need for trustworthy solutions that embody human-like reasoning, transparency, and explainability becomes more critical. AI must assist in analyzing legal texts and provide justifications that align with established legal reasoning practices. The challenge lies in designing AI systems that generate plausible answers and engage in structured, interpretable decision-making, ensuring they can be trusted in high-stakes legal contexts.

1.2 Challenges in Legal Text Processing

Legal services rely mainly on text-processing capabilities, which can enormously benefit from new advancements in large language models (LLM). Several scientific studies and business initiatives have highlighted the potential and limitations of LLMs in the legal domain. Nevertheless, LLM hallucinations have manifested in critical errors, such as generating nonexistent case law citations and misinterpreting contractual provisions.

A prominent example is *Mata v. Avianca*, where an attorney unknowingly submitted a brief containing fictitious judicial opinions produced by ChatGPT (Aidid, 2024). This event underscores the risks of using LLMs without robust verification mechanisms.

Applying LLMs in the legal domain demands higher accuracy, repeatability, and transparency to achieve a transformative impact. The LLM reasoning abilities have traditionally been too weak to understand the complex logic associated with legal contracts. Considerable progress is still required before these technologies deliver consistent and transparent solutions.

While human lawyers can articulate the reasoning behind their decisions and strategies, LLMs lack this capability to a sufficient degree. Despite progress in methodologies such as retrieval augmented generation - which guides LLMs to retrieve information from credible sources - hallucinations can and do occur, including for citations in the legal domain (Magesh et al., 2024). The auto-regressive nature of these models, which pushes them into greedily generating responses word-by-word rather than upfront planning, may contribute to this limitation (Borazjanizadeh and Piantadosi, 2024).

The recent release of OpenAI o1, which achieved substantially better results on reasoning-based tasks than its predecessors, can change this situation. The subsequent releases of DeepSeek R1 and OpenAI o3-mini, which achieved similar results to OpenAI o1 at substantially lower costs, have demonstrated the potential for "reasoning" LLMs to revolutionize task automation.

Despite these advancements, LLMs (including reasoning models such as OpenAI o1) still have a penchant for hallucinating on tasks that involve applying and interpreting complex rules. OpenAI o1 achieved a score of 77.6% on LegalBench (Guha et al., 2023; Vals.ai, 2025), a benchmark comprising a diverse set of tasks on various legal domains, leaving much room for improvement.

1.3 Proposed Neuro-Symbolic Approach

Unlike LLMs, logic programs, which have proven helpful for formally representing legal concepts as structured code, offer a solution to this ambiguity by reliably automating legal reasoning. Since logic programming fundamentally relies on the interplay of rules and facts, developing computable legal reasoning may depend on a complex information extraction process from written documents (Wang and Pan, 2020; Aitken, 2002).

A neuro-symbolic AI approach of combining LLMs' natural language capabilities with a logic-based reasoning system could eventually offset LLMs' limiting drawbacks to achieve correct, consistent, and explainable text analysis, generation, and manipulation of legalese. Applying this approach raises new questions about a) architecture - how to combine LLMs with logic programs, b) performance - what is the improvement in accuracy and consistency, and c) explainability - is the reasoning more understandable for humans, compared to plain vanilla LLMs.

This paper demonstrates how integrating LLMs with logic programming, particularly by prompting LLMs on legal terms transformed into logic programs, could outperform vanilla LLMs on targeted legal queries. Furthermore, we evaluate the performance gain by measuring the effect of prompting LLMs on legal terms transformed into logic programs compared to applying solely LLMs to query specific legal cases.

The described experiments are based on a predefined and validated set of insurance claim coverage questions and answers from two US health insurance policies: 1) a simplified Chubb Hospital Cash Benefit Policy (see Appendix A.1) and 2) more complex, a Stanford Cardinal Care Aetna Student Health Insurance Plan (Aetna Life Insurance, 2023).

We tested three approaches. In the *vanilla LLM* approach, LLMs answered coverage ques-

tions without any guidance on how to derive the answers. In the *unguided* approach, different LLMs were tasked with converting the insurance contract and claims into logic encoding (Prolog). We then used a Prolog interpreter (SWISH) to determine claim coverage. Finally, in the *guided* approach, we provided the LLM with a structured framework containing basic facts and information necessary for logic encoding.

2 Related Work

2.1 Evaluation of LLM in the Legal Domain

Recent evaluations of LLMs in the legal domain have revealed promising advances and critical limitations. Blair-Stanek and Durme (2025) show that state-of-the-art LLMs exhibit considerable output instability when answering legal questions, with models yielding divergent decisions even under controlled settings. In parallel, Hu et al. (2025) address the prevalent issue of hallucinations in legal question answering by proposing a fine-tuning framework that integrates behavior cloning with a sample-aware iterative direct preference optimization strategy, thereby enhancing factual consistency. Peoples (2025) further underscores that, although LLMs are capable of performing basic legal analysis through a typical chain of thoughts approach such as Issue, Rule, Analysis, and Conclusion (IRAC), their brief and sometimes unreliable outputs raise concerns regarding their adequacy for high-stakes legal reasoning and education.

Complementing these findings, an evaluation reported in the *Journal of Legal Analysis* (Dahl et al., 2024) highlights persistent transparency, ethical compliance, and reliability challenges when deploying LLMs for legal research in practice. The heterogeneous nature of legal language across different jurisdictions often leads to inconsistencies in model outputs, thereby questioning the ability to generalize with the necessary level of accuracy. The study demonstrated that legal hallucinations are pervasive and disturbing: hallucination rates range from 59% to 88% in response to specific legal queries.

Comprehensively the *LegalBench* benchmark introduced by Guha et al. (Guha et al., 2023) provides a collaboratively built suite of tasks that systematically measures various facets of legal reasoning, emphasizing the necessity for domain-specific evaluation metrics. These studies illustrate that while LLMs hold potential for legal applications, careful and targeted methodological improvements are essential to ensure their dependable integration into legal practice. Thus, while reported accuracy metrics are encouraging, they must be evaluated alongside limitations in consistency and transparency to assess the actual applicability of LLMs in the legal domain.

2.2 Advances in Neuro-Symbolic AI

Recent advances in legal language processing have increasingly focused on integrating LLMs with symbolic reasoning to balance the flexibility of neural architectures with the rigor of formal logic. Alonso and Chatzianastasiou (2024) demonstrated that embedding logical rules into neural frameworks can enhance the interpretability and robustness of legal text analysis. Servantez et al. (2024) introduced the *Chain of Logic* prompting method, which decomposes legal reasoning into independent logical steps and recomposes them to form coherent conclusions for rule-based legal evaluation. Similarly, Cummins et al. (2025) presented *InsurLE*. This domain-specific controlled natural language codifies insurance contracts by preserving key syntactic nuances while exposing the underlying formal logic for a computable representation.

Wei et al. (2025) proposed a hybrid neural-symbolic framework that synergizes neural representations with explicit logical rules, thereby improving the rigor of legal reasoning in automated systems. Patil (2025) systematically surveyed methods to enhance reasoning in LLMs and highlighted modular reasoning and retrieval-augmented techniques as promising approaches for bolstering logical consistency in legal applications. Colelough and Regli (2025) provided a comprehensive review of neuro-symbolic AI in the legal domain, identifying substantial progress in learning and inference while noting significant gaps in explainability and understanding derived logic programs.

Calanzone et al. (2024) developed a neuro-symbolic integration approach that enforces logical consistency by incorporating external constraint sets into LLM outputs. Sun et al. (2024) introduced a framework that explicitly learns case-level and law-level logic rules to generate faithful and interpretable explanations for legal case retrieval. Tan et al. (2024) enhanced LLM reasoning through a self-driven Prolog-based chain-of-thought mechanism that iteratively refines logical inferences in legal tasks. Lastly, Vakharia et al. (2024) proposed *ProSLM*, a Prolog-based language model that vali-

dates LLM outputs against a domain-specific legal knowledge base, ensuring higher factual accuracy and interpretability in legal question answering.

Collectively, these studies chart a clear trajectory toward AI systems that harness the complementary strengths of deep learning and logical inference to address the nuanced challenges inherent in legal reasoning.

3 Preliminary Experiments

In this section, we evaluate a range of state-of-the-art reasoning models to benchmark their capabilities in answering coverage-related 'yes/no' claim questions about an insurance policy.

We selected seven LLMs, including O1-preview, DeepSeek-R1, Llama-3.1-405B-Instruct, Claude-3.5-Sonnet, Mistral-Large-Latest, Gemini-1.5-Pro, and GPT-4o-2024-08-06[1]. These models excel in long-context reasoning, mathematical problem-solving, multi-step reasoning, logical consistency, and following policy rules, making them well-suited for analyzing insurance contracts and legal texts. For all models, we set both the temperature and top-p parameters to 1.

The insurance contract used in the experiments in this section is the Simplified Chubb Hospital Cash Benefit policy, referred to as Chubb hereafter. The task is to determine whether nine claims are covered under this insurance policy. The Chubb contract and nine claim queries are provided in Appendix A.1 and A.2, respectively.

We describe the vanilla LLM approach in §3.1 where we directly ask the LLM to answer 'yes/no' claim questions. In Section 3.2, we task the LLM with generating Prolog encodings of the insurance contract and claim queries, which we then manually evaluate using the help of SWISH Prolog interpreter (Contributors to SWI-Prolog, 2024).

3.1 Vanilla LLM Approach

We prompt the selected LLMs to answer nine claim questions about the Chubb insurance policy. We then evaluate the performance of these models across 10 trials and report their average accuracies and standard errors in Table 1. The prompt used for the LLMs is provided in Appendix A.3.1.

The results show that models such as Mistral-large-latest, Gemini-1.5-pro, Claude-3.5-sonnet, Llama-3.1-405B-instruct, and GPT-4o-2024-08-06 achieved a consistent accuracy of 0.78 across all 10 trials, with no variance, even at a temperature setting of 1.0. The error bars in Table 1 represent the Standard Error of the Mean (SEM)[2], indicating the variability of the model across trials.

All models consistently failed to correctly answer two specific questions: Questions 5 and 9 (see Appendix A.2). Question 5 asks whether a self-harm injury is covered if all other conditions are met, while Question 9 concerns coverage for a police officer injured outside of duty (see Appendix A.2 for the exact wording). Clause 1.1 of the policy specifies that hospitalization must result from sickness or accidental injury, meaning the claim in Question 5 is not covered. In Question 9, although "Service in the police" is excluded if the injury arises from it, the injury in this case occurred when the officer's son bit him in the ankle outside of duty. In both cases, the vanilla LLM models struggled to distinguish between being a police officer and being injured outside of service, as well as failing to recognize that punching someone in the face in Question 5 is not classified as sickness or accidental injury.

In contrast, DeepSeek-R1 achieved an average accuracy of 0.81 with an SEM of 0.02, correctly answering Question 5 in 3 out of 10 trials, though it still missed Question 9 in all trials. The O1-preview model performed better, achieving an average accuracy of 0.88 with an SEM of ± 0.02. It correctly answered Question 9 in 9 out of 10 trials but failed to answer Question 5 correctly in 9 out of 10 trials.

As observed, the vanilla LLM approach alone cannot provide answers to claim questions with 100% accuracy and consistency across trials, even when using state-of-the-art reasoning models. Next, we aim to enhance LLMs by leveraging the benefits of logic programming. We will ask them to generate Prolog encodings of the policy and claims and then answer the claim questions by evaluating the generated Prolog encodings.

3.2 Unguided LLM-generated Prolog

We prompted the selected LLMs (from §3) to generate Prolog encodings of the Chubb policy contract and nine claims. We then manually evaluated whether the insurance covered the claims by analyzing the Prolog encodings and using the help of

[1]We included GPT-4o-2024-08-06 along with all aforementioned state-of-the-art reasoning models to assess its performance in legal and contract analysis tasks, given its strong contextual comprehension and broad reasoning ability.

[2]The SEM is calculated by dividing the standard deviation of accuracy scores from 10 trials by the square root of the number of trials

Model	Accuracy ± SEM
Mistral-large-latest	0.78 ± 0.00
Gemini-1.5-pro	0.78 ± 0.00
Claude-3.5-sonnet	0.78 ± 0.00
Llama-3.1-405B-instruct	0.78 ± 0.00
GPT-4o-2024-08-06	0.78 ± 0.00
DeepSeek-R1	0.81 ± 0.02
O1-preview	0.88 ± 0.02

Model	Accuracy ± SEM
Mistral-large-latest	0.50 ± 0.06
Gemini-1.5-pro	0.56 ± 0.05
Claude-3.5-sonnet	0.74 ± 0.03
Llama-3.1-405B-instruct	0.41 ± 0.04
GPT-4o-2024-08-06	0.60 ± 0.07
DeepSeek-R1	0.63 ± 0.03
O1-preview	0.89 ± 0.02

Table 1: Average accuracy of LLMs on the Chubb insurance claim coverage dataset. The ± values represent the Standard Error of the Mean (SEM) across 10 trials. Left: Vanilla LLM; Right: Unguided Prolog Generation.

SWISH Prolog interpreter whenever possible[3].

This process was repeated for 10 trials. In every trial, each LLM generated a policy encoding from the prompt in Appendix A.3.2, and then translated nine claim questions in (given in A.2) into Prolog queries based on the policy encoding. We manually evaluated the policy and claim encodings and recorded the number of correct responses. When the encodings were unambiguous, we confirmed our evaluation using SWISH. We report average accuracies and the standard error of the mean over these 10 trials in Table 1 and provide a qualitative analysis of each LLM for this task below.

The O1-preview model achieved an average accuracy of 0.89 ± 0.02, slightly improving on its vanilla approach in §3.1. O1-preview answered Question 9 correctly in all trials, showing improved reliability over the vanilla approach in distinguishing between an injury caused by a son biting the claimant while the claimant was a police officer—an explicitly covered scenario.

However, similar to its vanilla approach, the O1-preview struggled with Question 5, missing it in 9 out of 10 trials. This question involved self-harm, where the claimant was injured due to a face punch. O1-preview failed to determine whether the scenario fell under an exclusion correctly. While it correctly excluded activities like skydiving, firefighting, and police service (where injuries during these activities are not covered), it failed to identify the primary cause of the claim—whether it was sickness or accidental injury, which are addressed indirectly in the contract. Additionally, O1-preview missed Question 4 in only one trial due to ambiguity in encoding time-based conditions. The vanilla LLM, in comparison, had a slightly lower average accuracy of 0.88 ± 0.02, with errors mostly in Question 5, and it consistently failed to answer Questions 9 and 4.

The DeepSeek-R1 model achieved an average accuracy of 0.63 ± 0.03. In several trials, it generated incorrect logic encodings, particularly in how it handled exclusions. For instance, it sometimes treated exclusions as conjunctive conditions (e.g., both general activity exclusions, such as serving as a firefighter, and the age > 80 condition had to hold simultaneously). However, the policy contract (see Appendix A.1) specifies that exclusions should be interpreted with an OR operator: coverage is denied if sickness or accidental injury results from a listed activity (e.g., skydiving, military service) or if the claimant is 80 years or older at the time of hospitalization. This misinterpretation led to inaccuracies in claim assessments. Some queries were ambiguous, preventing DeepSeek-R1 from determining a final coverage decision. Compared to O1-preview, which achieved 0.89 ± 0.02, DeepSeek-R1 not only had a lower average accuracy (0.63 ± 0.03) but also exhibited more significant variability across trials.

GPT-4o-2024-08-06 achieved an average accuracy of 0.60 ± 0.07. GPT-4o demonstrated errors in encoding policy rules in some trials, and its claim encodings often lacked sufficient information. This led to ambiguity, preventing the determination of essential predicate values required for accurate evaluation. Frequently, a final answer could not be determined due to this ambiguity. Additionally, the model exhibited significant inconsistencies across trials, producing logic encodings of varying quality. In some cases, inaccurate encodings—such as confusion between the claim date and the wellness visit time limit or inconsistent predicate parameters (e.g., passing the claim date instead of the claimants' age to an age exclusion predicate; see Appendix A.1)—led to a low accuracy of 1 out of

[3]The generated encodings often failed to run on SWISH due to ambiguity. In such cases, we manually reasoned through the policy encodings and evaluated the claims.

9 correct answers in one trial, while in another, it correctly answered 8 out of 9 questions.

The encodings of the remaining models were often ambiguous, resulting in inaccurate responses to several questions. The Mistral-large-latest model had an average accuracy of 0.5 ± 0.06, with substantial variability and significant struggles in determining coverage due to ambiguous logic encodings. The Gemini-1.5-Pro model had a slightly better average accuracy of 0.56 ± 0.05 but faced logic ambiguity issues. The Claude-3.5-Sonnet model achieved an average accuracy of 0.74 ± 0.03, with its main struggles being Questions 5 and 9. Finally, the Llama-3.1-405B-instruct model, with an average accuracy of 0.41 ± 0.04, faced frequent ambiguities. All these models performed worse than their vanilla versions, which had a consistent accuracy of 0.78. Overall, the ambiguity and inaccuracy in their encodings led to challenges in accurately responding to specific claim queries. As the next step, we propose expert-guided Prolog encoding generation in §4 to improve accuracy and consistency in LLMs when answering such claims.

4 Expert-Guided Experiments

In what follows, we demonstrate a workflow for leveraging LLMs *through expert guidance* to automate the process of encoding health insurance policies as logic programs (called computable contracts).

We prompted LLMs to encode Prolog rules representing three insurance coverages. The first coverage was the simplified Chubb policy, described in the previous experiment (see Appendix A.1). The latter two have been derived from the Stanford CodeX *Insurance Analyst* (CodeX, 2025a), a deployed, expert-encoded computable contract representing the Stanford Cardinal Care Aetna Student Health Insurance Plan (Aetna Life Insurance, 2023) (Oliver R. Goodenough and Preston J. Carlson, 2023). Specifically, we evaluated LLMs' ability to encode the *Advanced Reproductive Technology* (ART) and the *Comprehensive Infertility* (CI) coverage rules from the *Insurance Analyst*. Each coverage rule in the *Insurance Analyst* evaluates claims to reach coverage decisions, calling helper rules from other parts of the code base in the process (see Figure 1a). The LLMs were prompted to encode their own versions of these coverage rules with 1) the coverage text from the Cardinal Care policy 2) documentation defining a valid claim to the rule, and 3) documentation defining the relevant helper rules which can be called from other parts of the code base (see Figure 1b, Appendix A.3.3). The documentation provided to the LLM constitutes guidance given by an expert. We had each LLM generate an encoding of each coverage 5 times, testing each coverage encoding by querying it with claims and evaluating whether the outputted decisions were correct (see Figure 1c).

Model	Accuracy ± SEM
`GPT-4o`	1.00 ± 0.00
`OpenAI o1`	1.00 ± 0.00
`OpenAI o3-mini`	1.00 ± 0.00
`DeepSeek-R1`	0.73 ± 0.17

Table 2: Evaluation of the LLM-generated simplified Chubb logic program encodings.

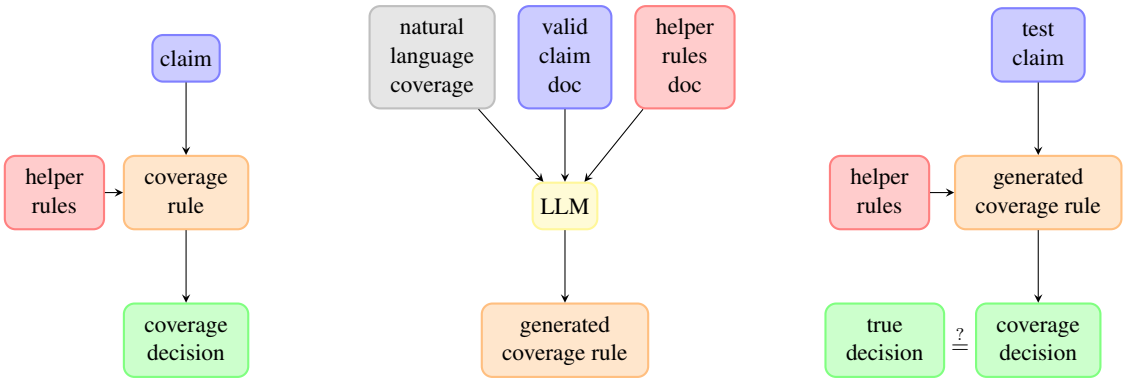
4.1 Guided LLM-generated Prolog for a simplified policy

On the Chubb policy, we prompted LLMs with the text of the policy and documentation about the facts provided in any valid claim (e.g., `claim_hospitalization_reason`, `claim_misrepresentation_occurred`) to be used in generating a representative computable contract. Since this policy is stand-alone, its encoding does not need to integrate into a more extensive code base. Thus, no helper rules (from other parts of the code base) needed to be included in the prompt.

Three of the four LLMs performed well on this task (see Table 2), with each of their 5 generated encodings perfectly answering all 9 test queries used for evaluation. These test queries were simply Prolog translations of the natural language queries used to assess the previous approach (see Appendix A.2). DeepSeek-R1, however, produced one encoding with a syntactic error due to an unclosed parenthesis. This, along with some failed test cases in another one of its encodings, resulted in a lower accuracy rate than the other models.

4.2 Guided LLM-generated Prolog for coverages in a larger policy

The Stanford Cardinal Care health insurance policy comprises many individual "coverages". For a claim to be covered under the policy, it must be covered under one of these coverages. Thus, while the *Insurance Analyst* has an overarching "covered" rule (which should be satisfied exactly when



(a) Insurance Analyst Functionality (b) LLM Coverage Rule Generation (c) Generated Coverage Rule Testing

Figure 1: Experimental overview: (a) Functionality of the CodeX Insurance Analyst coverage rules. (b) The LLM is prompted to generate its own version of the coverage rule given the text of the coverage and documentation of the valid claims and helper rules it can call. (c) The LLM's generated coverage rule is tested by passing it test claims and determining if the correct coverage decisions were made.

a claim is covered under the Cardinal Care policy), it also contains many rules associated with specific coverages, one of which must be satisfied for the overarching one to be satisfied. The Prolog code undergirding the *Insurance Analyst* is available in its public code repository (CodeX, 2025b). We asked LLMs to encode the ART and CI coverages, testing the accuracy of these encodings through 20 test cases from the *Insurance Analyst*'s publicly available code repository.

OpenAI o1 was substantially more successful at encoding these Cardinal Care coverages than GPT-4o, OpenAI o3-mini and DeepSeek-R1. As shown in Table 3, OpenAI o1's ART encodings achieved an average accuracy of 95%, far superseding the $50-60\%$ accuracies of the encodings generated by the other models. Similarly, as shown in Table 4, OpenAI o1's encodings on CI coverages had an 87% accuracy significantly outperforms that of the other models.

Since the ART and CI coverages are longer and more logically complex than the simplified Chubb policy, they serve as better differentiators of the logical capabilities of the tested LLMs. As an example of the difference in logical correctness between the logic programs written by OpenAI o1 and GPT-4o, consider the following excerpt from the ART coverage:

> For women 39 years of age and older, ovarian responsiveness is determined by measurement of day 3 FSH obtained within the prior 6 months. For women who are less than 40 years of age, the day 3 FSH must be less than 19 mIU/mL in their most recent laboratory test to use their own eggs. For women 40 years of age and older, their unmedicated day 3 FSH must be less than 19 mIU/mL in all prior tests to use their own eggs.

Note that there are two age-based boundaries specified in this excerpt. Firstly, women who are at least 39 years of age must have had an FSH test within the prior 6 months, whereas this condition does not apply to younger women. Secondly, women who are at least 40 will have *all FSH tests* past age 40 examined, whereas younger women will only have the most recent test looked at.

GPT-4o, in its first trial, encoded the FSH criteria in the rule `validate_day_3_fsh(C)` (see Appendix A.4.1). This rule correctly checks for the strictness criterion with a boundary at age 40, but there is no sign of the recency criterion with a boundary at age 39. By contrast, consider the analogous encoding generated by OpenAI o1 in *its* first trial of the rule `meets_fsh_criteria(C)` (see Appendix A.4.2), which correctly delineates *both* age-based boundaries–at age 40 as well as 39. Unlike the encoding generated by GPT-4o, it ensures that the most recent FSH test for women who are at least 39 years of age was conducted no more than 6 months ago.

This and other examples demonstrate the significant gap in logical ability between OpenAI o1 and GPT-4o, explaining the former's significantly higher accuracy in representing insurance coverages in a logical form.

OpenAI o3-mini and DeepSeek also performed worse than OpenAI o1 on ART due to logical

Model	Accuracy ± SEM
GPT-4o	0.56 ± 0.09
OpenAI o1	0.95 ± 0.00
OpenAI o3-mini	0.58 ± 0.13
DeepSeek-R1	0.72 ± 0.16

Table 3: Evaluation of the LLM-generated Cardinal Care ART coverage logic program encodings.

Model	Accuracy ± SEM
GPT-4o	0.37 ± 0.10
OpenAI o1	0.87 ± 0.04
OpenAI o3-mini	0.72 ± 0.04
DeepSeek-R1	0.47 ± 0.18

Table 4: Evaluation of the LLM-generated Cardinal Care CI coverage logic program encodings.

errors and syntactical mistakes. One major issue was the misapplication of the premature ovarian failure (POF) exception. OpenAI o3-mini wrongly applied this exception to *all* women aged $40+$ (not just ones with POF), allowing some to qualify when they should not have. Both models also made syntax errors that prevented their encodings from running. OpenAI o3-mini referenced the nonexistent rule `day_3_fsh_tests_since_age_40` where it should have been referring to `day_3_fsh_tests_since_age_40_in_claim`, while DeepSeek introduced an unclosed parenthesis, making its Prolog code invalid and thus impossible to evaluate.

Since the CI coverage is even longer and more complex than ART, all of the models performed worse on encoding this coverage (see Table 4). However, the differences in capability persisted, with OpenAI o1 leading the pack by producing logically and syntactically superior Prolog encodings. These results show that strong reasoning LLMs such as OpenAI o1 could play a critical role in developing computable contracts that provide reliable, interpretable, and auditable coverage decisions for insurers.

5 Conclusion and Future Work

We are on the cusp of an exciting era when AI can enhance access to legal solutions by incorporating human-like thinking, such as planning and reasoning. While LLMs show promise, their probabilistic nature, lack of consistency, and potential for hallucination make their application in the legal domain risky.

We propose a neuro-symbolic approach using LLMs with logic encoding. We compared a vanilla LLM with an LLM that encodes a legal contract as logic. Our key observation is that advancements in foundational models enable the vanilla LLM to reasonably determine if a claim is covered under a contract, but it lacks full accuracy and consistency.

Next, we used an LLM to convert a legal contract into logic encoding. As expected, the quality of the LLM-generated encodings was poor, worse than the vanilla LLM. We guided the LLM to improve this by providing a structured framework of basic information a human encoder would need. Our findings suggest that a guided approach significantly improves the quality of generated encodings.

Beyond our approach of using LLMs to generate logical representations through unguided and guided methods, we propose exploring several additional approaches in future work.

Our first proposal involves using high-quality, human-generated logic encodings to fine-tune foundational models. Generating a logic encoding for a legal segment or contract resembles writing a piece of Python code. However, current foundational models have significantly more training data on high-quality Python code than on logic encodings. Fine-tuning with curated logic encodings can enhance LLMs' ability to generate accurate and structured representations.

We see an opportunity to enhance LLM-generated Prolog accuracy using agentic AI. This includes automating LLM logic encoding in a Prolog interpreter like SWISH. Our experiments found frequent syntax errors in LLM encodings, but this method allows for automatic error identification and correction, improving accuracy. Another method uses multiple LLMs: one encodes legal terms and queries, a cost-efficient model executes them, while another evaluates the outcomes. However, it's unclear if this will ensure accurate and reliable results.

Our third proposal for future work is to use reinforcement learning with synthetic data and the Prolog interpreter outputs as a post-training process to enhance LLMs' ability to generate accurate Prolog encodings.

6 Limitations

Our current approach addresses only a limited scope. It is a first step in a novel direction: combining LLM and logic programs to form a neuro-

symbolic AI for legal analysis.

The explained experiments are limited in problem space, architecture design, data sets, foundational models incorporated, logic interpreters incorporated, prompt tuning, measurements, and analysis conducted.

Our long-term ambition is to apply a neuro-symbolic approach in the legal domain in the broader sense. Currently, we only cover health insurance-related coverage questions and answers. Further application areas, like reasoning civil and corporate legal terms, have been out of scope.

The architectural design for combining LLMs with logic programs is demonstrated only through LLM-generated logic programs and their execution via logic interpreters. The paper does not address post-training fine-tuning, adapter layers, retrieval-augmented generation, knowledge injection, or reinforcement learning, which is part of future work.

Only a very narrow set of policies, questions, and answers are processed in the experiments in terms of data. In future work, a wider selection of cases should be addressed to gain better insights into the performance of the demonstrated approach.

We included only a subset of available LLMs in the analysis and covered only Prolog as a logic interpreter. Future work should address a more extensive variety of foundational models and interpreters.

Regarding prompt-tuning, we only applied an explicit *Chain-of-logic* Prolog encoding, derived from the learning on the Stanford CodeX *Insurance Analyst*. Future work must also address other kinds of planning on encodings (e.g., Self-Ask decomposition-based reasoning, iterative refinement, or reinforcement learning with thought tracing).

In the paper, we only addressed accuracy and consistency measurements and highlighted certain aspects of explainability and audibility qualitatively without measuring them. Future work should address a broader scope of metrics to give a more holistic picture of the performance gain of neuro-symbolic AI designs.

References

Aetna Life Insurance. 2023. Aetna Student Health Insurance Open Access Elect Choice EPO Medical and Outpatient Prescription Drug Plan Schedule of benefits. Accessed: Feb. 4, 2025.

Abdi Aidid. 2024. Toward an ethical human-computer division of labor in law practice. *Fordham Law Review*, 92(5):1797–1813.

James Stuart Aitken. 2002. Learning information extraction rules: An inductive logic programming approach. In *Proceedings of the 15th European Conference on Artificial Intelligence*, pages 355–359. IOS Press.

Miquel Noguer I Alonso and Foteini Samara Chatzianastasiou. 2024. Automating legal contracts using logic rules with large language models. Accessed: Feb. 4, 2025.

Bipartisan House Task Force on Artificial Intelligence. 2024. Bipartisan house task force on artificial intelligence report. Accessed: Feb. 11, 2025.

Andrew Blair-Stanek and Benjamin Van Durme. 2025. Llms provide unstable answers to legal questions. *arXiv preprint*.

Nasim Borazjanizadeh and Steven Piantadosi. 2024. Reliable reasoning beyond natural language. *arXiv preprint*.

Diego Calanzone, Stefano Teso, and Antonio Vergari. 2024. Logically consistent language models via neuro-symbolic integration. *arXiv preprint*.

Stanford CodeX. 2025a. Insurance analyst. Accessed: Feb. 2, 2025.

Stanford CodeX. 2025b. Insurance analyst stanford cardinal care encoding. Accessed: Feb. 2, 2025.

Brandon C. Colelough and William Regli. 2025. Neuro-symbolic ai in 2024: A systematic review. *arXiv preprint*.

Contributors to SWI-Prolog. 2024. SWISH – SWI-Prolog for SHaring. Accessed: Feb. 4, 2025.

J. Cummins, J. Dávila, R. Kowalski, and D. Ovenden. 2025. Computable contracts for insurance: Establishing an insurance-specific controlled natural language - insurle. Accessed: Feb. 11, 2025.

Matthew Dahl, Varun Magesh, Mirac Suzgun, and Daniel E Ho. 2024. Large legal fictions: Profiling legal hallucinations in large language models. *Journal of Legal Analysis*, 16(1):64–93.

European Union. 2016. Regulation (eu) 2016/679 of the european parliament and of the council on the protection of natural persons with regard to the processing of personal data and the free movement of such data, and repeal directive 95/46/ec (general data protection regulation). Regulation (EU) 2016/679. Articles 13–15 and 22 require transparency for automated decisions.

European Union. 2024. Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence and amending regulations (ec) no 300/2008, (eu) no 167/2013, (eu) no 168/2013, (eu) 2018/858, (eu) 2018/1139 and (eu) 2019/2144 and directives

2014/90/eu, (eu) 2016/797 and (eu) 2020/1828 (artificial intelligence act).

Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Re, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, and et al. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. In *Thirty-Seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yinghao Hu, Leilei Gan, Wenyi Xiao, Kun Kuang, and Fei Wu. 2025. Fine-tuning large language models for improving factuality in legal question answering. *arXiv preprint*.

International Association of Insurance Supervisors. 2024. Draft application paper on the supervision of artificial intelligence. Accessed: Feb. 11, 2025.

Varun Magesh, Faiz Surani, Matthew Dahl, Mirac Suzgun, Christopher D. Manning, and Daniel E. Ho. 2024. Hallucination-free? assessing the reliability of leading ai legal research tools. *arXiv preprint*.

Oliver R. Goodenough and Preston J. Carlson. 2023. Words or code first? Is the legacy document or a code statement the better starting point for complexity-reducing legal automation? Accessed: Feb. 4, 2025.

Avinash Patil. 2025. Advancing reasoning in large language models: Promising methods and approaches. *arXiv preprint*.

Lee Peoples. 2025. Artificial intelligence and legal analysis: Implications for legal education and the profession. *arXiv preprint*.

Sergio Servantez, Joe Barrow, Kristian Hammond, and Rajiv Jain. 2024. Chain of logic: Rule-based reasoning with large language models. *arXiv preprint*.

Zhongxiang Sun, Kepu Zhang, Weijie Yu, Haoyu Wang, and Jun Xu. 2024. Logic rules as explanations for legal case retrieval. *arXiv preprint*.

Xiaoyu Tan, Yongxin Deng, Xihe Qiu, Weidi Xu, Chao Qu, Wei Chu, Yinghui Xu, and Yuan Qi. 2024. Thought-like-pro: Enhancing reasoning of large language models through self-driven prolog-based chain-of-thought. *arXiv preprint*.

Priyesh Vakharia, Abigail Kufeldt, Max Meyers, Ian Lane, and Leilani H. Gilpin. 2024. *ProSLM: A Prolog Synergized Language Model for explainable Domain Specific Knowledge-Based Question Answering*, pages 291–304. Springer Nature Switzerland.

Vals.ai. 2025. LegalBench Benchmark. Accessed: Feb. 4, 2025.

Wenya Wang and Sinno Jialin Pan. 2020. Integrating deep learning with logic fusion for information extraction. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 9225–9232. AAAI Press.

Bin Wei, Yaoyao Yu, Leilei Gan, and Fei Wu. 2025. An llms-based neuro-symbolic legal judgment prediction framework for civil cases. *Artificial Intelligence and Law*.

A Appendix / supplemental material

A.1 Simplified Chubb Hospital Cash Benefit Policy

Between:
CODEX INSURANCE LIMITED (“us”)
and
_____ (“You”)

This policy is provided on the following terms and conditions:
POLICY IN EFFECT AND CONDITIONS
1.1 The payment of any benefit under this policy is conditioned on the policy being in effect at the time of the hospitalization for sickness or accidental injury on which the claim for such benefit is premised. The policy will be in effect if:

1. This agreement is signed,
2. The applicable premium for the policy period has been paid, and
3. The condition set out in Section 1.3 is still pending or has been satisfied in a timely fashion, and
4. The policy has not been canceled.

1.2 Cancelation will be deemed to have occurred if there is fraud, or any misrepresentation or material withholding of any information provided by you to the Company in connection with any communication or information relating to this policy, or if the condition set out in Section 1.3 has not been satisfied in a timely fashion. It will also be automatically canceled at midnight, US Eastern time then in effect, on the last day of the policy term described in Section 5 below.
1.3 No later than the 7th month anniversary of the effective date of this policy, you will supply us with written confirmation from the medical provider in question of a wellness visit for yourself with a qualified medical provider occurring no later than the 6th month anniversary of the effective date of this policy.
GENERAL EXCLUSIONS
2.1 Your policy will not apply to, and no benefit will be paid with respect to, any event causing sickness or accidental injury arising directly or indirectly out of:

1. Skydiving; or
2. Service in the military; or
3. Service as a fire fighter; or
4. Service in the police; or
5. If your age at the time of the hospitalization is equal to or greater than 80 years of age.

GENERAL CONDITIONS
3.1 Where does Your Policy apply?
3.1.1 Your Policy insures You twenty-four (24) hours a day anywhere in the world.
3.2 Arbitration
3.2.1 If any dispute or disagreement arises regarding any matter pertaining to or concerning this Policy, the dispute or disagreement must be referred to arbitration in accordance with the provisions of the Arbitration Act (Cap. 10) and any statutory modification or re-enactment thereof then in force, such arbitration to be commenced within three (3) months from the day such parties are unable to settle the dispute or difference. If You fail to commence arbitration in accordance with this clause, it is agreed that any cause of action and any right to make a claim that You have or may have against Us shall be extinguished completely. Where there is a dispute or disagreement, the issuance of a valid arbitration award shall also be a condition precedent to our liability under this Policy. In no case shall You seek

to recover on this Policy before the expiration of sixty (60) days after written proof of claim has been submitted to Us in accordance with the provisions of this Policy.

3.3 Laws of New York

3.3.1 Your Policy is governed by the laws of New York.

3.4 US Currency

3.4.1 All payments by You to Us and by Us to You or someone else under your policy must be in United States currency.

3.5 Premium

3.5.1 The premium described in Section 5 below shall be paid in one lump sum at the signing of the policy.

3.6 Policy Term The term of this policy will begin on the date accepted by Us as signified by our signature of the policy (the effective date) and will last for a period of one year from that date, unless previously canceled pursuant to Section 1 above.

A.2 Queries and Correct Answers for Empirical Evaluation

All queries are preceded by the disclaimer: "Assuming all other conditions are met and no other exclusions apply (where by 'other,' I mean anything not referenced in the query that follows),. . . "

Query 1: "will my policy apply if I was hospitalized by burns suffered while doing my duty as a firefighter?" **Answer:** "No."

Query 2: "will my policy apply if I am 78 years old at the time of hospitalization?" **Answer:** "Yes."

Query 3: "will my policy apply if I was hospitalized for pneumonia 5 months after the policy's effective date, and my age at the time of hospitalization is 65?" **Answer:** "Yes."

Query 4: "will my policy apply if I was hospitalized due to a fall while traveling abroad and I had given confirmation of my wellness visit 8 months after the policy's effective date?" **Answer:** "No."

Query 5: "will my policy apply if I was hospitalized for punching my own face to show off for my friends and I did not commit fraud or misrepresentation?" **Answer:** "No."

Query 6: "will my policy apply if I was hospitalized due to an injury sustained while skydiving, my age at the time of hospitalization was 79, and proof of my wellness visit was provided 6.5 months after the policy's effective date?" **Answer:** "No."

Query 7: "will my policy apply if I was hospitalized for a heart attack, proof of the wellness visit was submitted 2 months after the policy's effective date, and my age at the time of hospitalization was 75?" **Answer:** "Yes."

Query 8: "will my policy apply if I was hospitalized after being injured in a military training exercise, the hospitalization occurred within the policy term, and I did not commit fraud?" **Answer:** "No."

Query 9: "will my policy apply if I was hospitalized due to my son biting me in the ankle, proof of my wellness visit was provided 6 months after the effective date, and I was serving as a police officer at the time of hospitalization?" **Answer:** "Yes."

A.3 Prompts Provided to LLMs

A.3.1 Prompt for Vanilla LLM Approach

The following is the prompt used in the Vanilla LLM approach described in §3.1.

- Below, you are provided
    1. The full text of an insurance contract
    2. A specific question about whether a claim in the given scenario is covered under the terms of this insurance contract

- Assume that the policy agreement has been signed, and the premium has been paid on time.
- Assume that all other conditions are satisfied, and no exclusions apply unless explicitly referenced in the query.
- Your task:
  1. Evaluate whether the claim described in the question is covered under the insurance contract.
  2. Respond with **only** one of the following: “Yes”, “No”, or “I do not know”.
  3. Do not provide any explanations or reasoning.
- Insurance contract: {text_content}
- Question: {query}

A.3.2 Prompt for Unguided Prolog Generation

The following is the prompt used in §3.2 to generate Chubb insurance policy encoding.

- Given the insurance contract below, translate the document into valid Prolog rules so that I can run a Prolog query on the code regarding whether or not some claim is covered under the policy and receive the correct answer to the question.
- Please fully define all predicates and DO NOT define any facts, only rules that can be used to answer queries on this insurance contract.
- Assume that all dates/times in any query to this code (apart from the claimant’s age) will be given RELATIVE to the effective date of the policy (i.e. there will never be a need to calculate the time elapsed between two dates). Take dates RELATIVE TO the effective date into account when writing this encoding.
- Assume that the agreement has been signed and the premium has been paid (on time). There is no need to encode rules or facts for these conditions.
- Return only Prolog code in your reply. No explanation is necessary.
- Ensure that:
  1. The legal text is appropriately translated into correct Prolog rules.
  2. The output does not redefine, misuse, or conflict with any built-in Prolog predicates.
  3. If dynamic predicates are necessary, they are declared and managed correctly.
  4. All predicates used in the generated Prolog code, including those referenced in the query, are fully defined and error-free to prevent issues like “procedure does not exist.”
  5. Logical relationships, conditions, and dependencies in the text are faithfully represented in the Prolog rules to ensure accurate query results.
- Insurance contract: {text_content}

The following is the prompt used in §3.2 to generate claim encodings.

- I have given below:
  1. A question about whether or not the policy defined in a given insurance contract applies in a particular situation
  2. The text of the insurance contract
  3. A Prolog encoding of the insurance contract
- Encode the question into a Prolog query such that it can be run on the given Prolog encoding of the insurance contract, returning the correct answer to the question.

- Assume that the agreement has been signed and the premium has been paid (on time). There is no need to encode rules or facts for these conditions.
- Return only Prolog query in your reply. No explanation is necessary.
- Ensure that:
    1. The output does not redefine, misuse, or conflict with any built-in Prolog predicates.
    2. If dynamic predicates are necessary, they are declared and managed correctly.
    3. All predicates used in the generated Prolog code, including those referenced in the query, are fully defined and error-free to prevent issues like "procedure does not exist."
    4. Logical relationships, conditions, and dependencies in the text are faithfully represented in the Prolog rules to ensure accurate query results.
    5. No absolute dates/times (apart from the claimant's age) are encoded in your query. Only include dates/times RELATIVE to the effective date of the policy (again, except for age).
    6. Set any facts/rules/parameters in the code such that ALL conditions (for the policy to apply) which are UNRELATED to the above query are satisfied.
    7. Set any facts/rules/parameters in the code such that NO exclusions (which would prevent the policy from applying) which are UNRELATED to the above query are satisfied.
- Question:{query}
- Insurance contract: {text_content}
- Insurance contract Prolog encoding: {policy_encoding}

A.3.3 Prompt for Generating LLM Encodings of Insurance Analyst Coverages

- I have provided below all of the text that pertains to a coverage (or section) of a health insurance policy.
    - The text defines all conditions and exclusions that determine whether a patient's claim is covered under this coverage of the policy.
- Please encode a Prolog rule, `'covered(C)'`, which is true exactly when the patient's claim, `'C'`, is covered.
    - I have also provided below a set of claim facts that will be defined in any Prolog query to the `'covered(C)'` rule.
        - These are of the form `'claim_Fact(C, Value)'`, meaning they will be defined for the claim `'C'` by filling in a value for `Value`.
        - Under each of these facts, I have listed the valid inputs which can be filled in for `Value` (or in some cases, the type of valid input, such as number or date) in the fact.
        - Use ALL OF (AND ONLY) these claim facts to write your Prolog encoding for the policy coverage.
    - Finally, I have provided below a set of supporting (pre-defined) Prolog predicates.
        - Under each supporting predicate, I have listed the arguments to the predicate with their category (e.g. atom, number) and a description, as well as a description of what the predicate does.
        - Use ALL OF these supporting predicates to write your Prolog encoding for the policy coverage.
- The Prolog rule that you encode, `'covered(C)'`, must incorporate all relevant criteria from the policy coverage.
- Return only Prolog code in your reply. NO EXPLANATION is necessary.

A.4 LLM Encodings of FSH Criteria in ART Coverage

A.4.1 GPT-4o Encoding of FSH Criteria in ART Coverage

```
validate_day_3_fsh(C) :-
    % Get patient age at time of service
    patient_age_at_time_of_service(C, Age),
    (Age < 40 -> % For patients under 40
        most_recent_day_3_fsh_test(C, fsh_test(FSHLevel, _, _)),
        FSHLevel < 19
    ; Age >= 40 -> % For patients 40 and above
        day_3_fsh_tests_since_age_40_in_claim(C, FSHTests),
        \+ exists_test_with_level_at_least_19(FSHTests)).
```

A.4.2 OpenAI o1 Encoding of FSH Criteria in ART Coverage

```
meets_fsh_criteria(C) :-
    patient_age_at_time_of_service(C, Age),
    (
        ( Age < 39 ) ->
            most_recent_day_3_fsh_test(C, fsh_test(Level, _, _)),
            Level < 19
        ;
        ( Age = 39 ) ->
            claim_date_service_received(C, ServiceDate),
            most_recent_day_3_fsh_test(C, fsh_test(Level, TestDate, _)),
            leq_n_months_diff(TestDate, ServiceDate, 6),
            Level < 19
        ;
        ( Age >= 40 ) ->
            claim_date_service_received(C, ServiceDate),
            (
                claim_patient_has_premature_ovarian_failure(C, yes) ->
                    most_recent_day_3_fsh_test(C, fsh_test(Level, TestDate, _)),
                    leq_n_months_diff(TestDate, ServiceDate, 6),
                    Level < 19
                ;
                    day_3_fsh_tests_since_age_40_in_claim(C, Tests),
                    \+ exists_test_with_level_at_least_19(Tests),
                    most_recent_day_3_fsh_test(C, fsh_test(_, TestDate, _)),
                    leq_n_months_diff(TestDate, ServiceDate, 6)
            )
    ).
```