Chain of Logic: Rule-Based Reasoning with Large Language Models

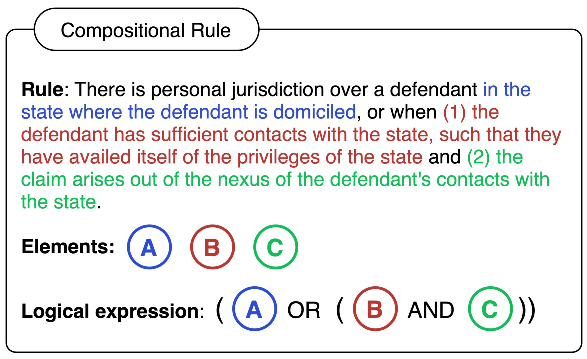
Sergio Servantez[†] **Joe Barrow**[▼‡] **Kristian Hammond**[†] **Rajiv Jain**[‡]
[†]Northwestern University [‡] Adobe Research [▼] Pattern Data
servantez@u.northwestern.edu joe.barrow@patterndataworks.com
Kristian.Hammond@northwestern.edu rajijain@adobe.com

Abstract

Rule-based reasoning, a fundamental type of legal reasoning, enables us to draw conclusions by accurately applying a rule to a set of facts. We explore causal language models as rule-based reasoners, specifically with respect to compositional rules - rules consisting of multiple elements which form a complex logical expression. Reasoning about compositional rules is challenging because it requires multiple reasoning steps, and attending to the logical relationships between elements. We introduce a new prompting method, Chain of Logic, which elicits rule-based reasoning through decomposition (solving elements as independent threads of logic), and recomposition (recombining these sub-answers to resolve the underlying logical expression). This method was inspired by the IRAC (Issue, Rule, Application, Conclusion) framework, a sequential reasoning approach used by lawyers. We evaluate chain of logic across eight rule-based reasoning tasks involving three distinct compositional rules from the LegalBench benchmark and demonstrate it consistently outperforms other prompting methods, including chain of thought and self-ask, using open-source and commercial language models.

1 Introduction

The surge in reasoning capabilities of language models (LMs) has the potential to transform the legal industry, a sector inherently reliant on sophisticated reasoning. The development of models that can assist and validate the reasoning processes of legal practitioners promises to usher in a new legal era. Such advancements would not only enhance the efficiency of legal services but also expand the capacity of professionals to service more clients, thereby broadening access to justice. At present, language models are prone to hallucinations in a legal setting (Dahl et al., 2024), and even powerful LMs like GPT-4 struggle to perform basic legal tasks (Blair-Stanek et al., 2023a).



Rule: There is personal jurisdiction over a defendant in the state where the defendant is domiciled, or when (1) the defendant has sufficient contacts with the state, such that they have availed itself of the privileges of the state and (2) the claim arises out of the nexus of the defendant's contacts with the state.

Elements: A B C

Logical expression: (A OR (B AND C))

Figure 1: Example showing compositional structure of rule for Personal Jurisdiction task from LegalBench. Color coding is used to identify rule elements and illustrate how these elements form a complex logical expression. Reasoning about compositional rules requires not only correctly applying each element to a fact pattern, but also resolving the logical expression. If the logical expression evaluates to true, it triggers a consequence (personal jurisdiction exists).

Legal tasks typically require sophisticated rule-based reasoning. These rules are written in natural language and expressed in many forms, including statutes, judicial holdings and even contract provisions. Similar to an if/then statement, a rule has an antecedent (a condition that can be evaluated to true or false) and a consequent (the outcome triggered if the antecedent is satisfied). Even the earliest recorded laws, written in Sumerian on clay tablets, have used this conditional structure (Roth, 1995).

Rule-based reasoning allows us to draw conclusions by applying a rule to a set of facts to determine if these preconditions are satisfied. For example, if parking is prohibited (consequent) between 2pm and 4pm (antecedent), and we know it is currently 3pm, then we can conclude that parking is currently prohibited. Often rules, especially complex rules, are compositional in nature meaning the antecedent consists of multiple conditions joined by *and* and *or* operators forming a complex logical expression (see Figure 1). In law, these constituent conditions are called rule elements. Recent work demonstrates that models can struggle to reason about even basic compositional rules (Blair-Stanek et al., 2023b).

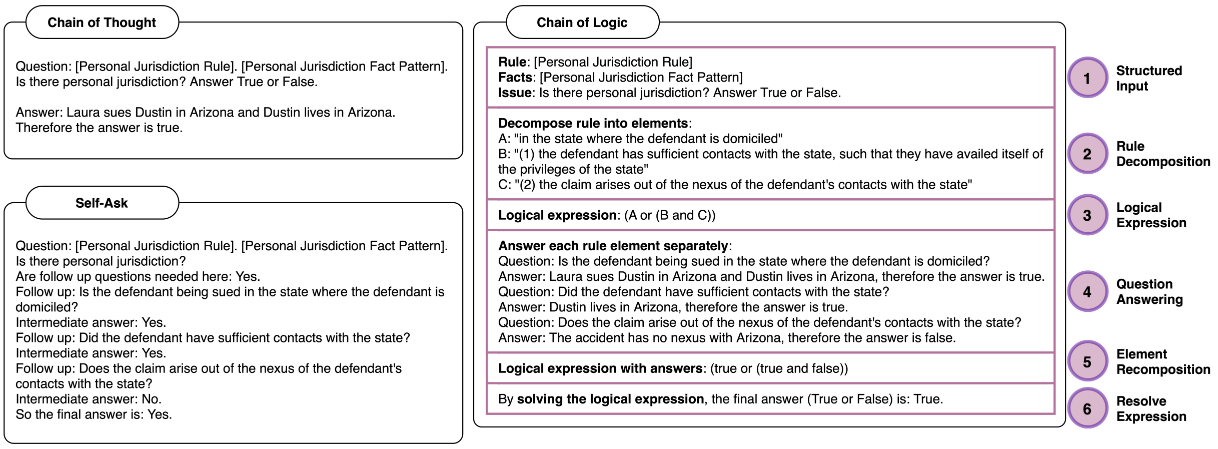
We explore large language models as rule-based reasoners, evaluating several common in-context learning methods across 8 tasks and 3 distinct rules from LegalBench, a collaboratively constructed legal reasoning benchmark (Guha et al., 2023). LegalBench tasks were designed by subject matter experts to evaluate useful legal reasoning skills. Prior work on LegalBench has focused on evaluating the capacity of models to perform rule-based reasoning given in-context demonstrations of the same rule being applied to varying fact patterns. These few-shot prompts are valuable in law where annotated data is scarce, limiting the ability to fine-tune models for specific legal tasks. Yet requiring several reasoning examples for each rule in a real world setting poses a challenge in terms of both cost and scalability. Additionally, a same-rule setting allows the model to largely mimic the reasoning path of the in-context examples. In contrast, a different-rule setting enables us to examine a model's ability to dynamically generate this reasoning path - the same way a lawyer would reason. Our work shifts focus to evaluating and improving a model's ability to perform rule-based reasoning given only a *single* in-context demonstration of a *different* rule application, thus removing the need to store examples for each rule.

In this work, we introduce **Chain of Logic**, a new prompting approach to guide LMs in reasoning about compositional rules by decomposing rule elements into a series of logical statements (evaluates to true or false), before explicitly recomposing these element answers to resolve the underlying logical expression. Understanding natural language questions requires understanding how to break down a question into a set of steps yielding an answer (Wolfson et al., 2020). The inspiration for our method is rooted in the IRAC Framework, an approach to legal reasoning where lawyers break down the reasoning process into four sequential steps: issue spotting, rule identification, rule application and conclusion. During the rule application step, rules are typically divided into their elements and addressed separately before recombining these threads to arrive at a conclusion. In devising our chain of logic approach, our interdisciplinary team combined this domain knowledge with a variety of insights from prior work on in-context learning.

Our experiments show chain of logic outperforms other prompting methods, including chain of thought (Wei et al., 2023) and self-ask (Press et al., 2023), across a variety of rule-based reasoning tasks using both open-source and commercial models. We demonstrate that given a single example of chain of logic, a model can learn to generalize this approach to a different rule and fact pattern, and thereby improve its reasoning capabilities.

2 Background

Large language models have demonstrated strong capabilities as zero-shot (Kojima et al., 2023) and few-shot (Brown et al., 2020) reasoners. Chain of thought prompting enhanced problem solving performance even further by eliciting models to produce intermediate reasoning steps (Wei et al., 2023). This approach proved effective at solving complex tasks, including arithmetic and commonsense reasoning tasks. Many other prompting methods followed with a similar aim of decomposing a complex task into a sequence of simpler subtasks. Self-ask improved multi-hop question answering performance beyond chain of thought prompting by guiding the model to explicitly pose and answer intermediate questions (Press et al., 2023). Similar to decomposing a multi-hop question into a sequence of intermediate questions, compositional rules can be decomposed into rule elements. However, reasoning over these rules requires not only answering each element correctly, but also understanding the logical relationships that exists between these elements. For example, if a rule with two elements resolves to (true, false), then the final answer depends on whether the boolean relationship between those elements is *and* or *or*. In Section x, we show existing prompting methods can resolve rule elements correctly while still getting the final answer incorrect. This motivates the need for a more robust prompting method which attends to both the element level answers and the underlying logical structure. Further motivating this work, zero-shot methods have been observed outperforming one-shot and few-shot approaches on legal reasoning tasks (Yu et al., 2022; Blair-Stanek et al., 2023b), suggesting models can struggle with in-context learning in a legal setting.


Chain of Thought

Question: [Personal Jurisdiction Rule]. [Personal Jurisdiction Fact Pattern]. Is there personal jurisdiction? Answer True or False.

Answer: Laura sues Dustin in Arizona and Dustin lives in Arizona. Therefore the answer is true.

Self-Ask

Question: [Personal Jurisdiction Rule]. [Personal Jurisdiction Fact Pattern]. Is there personal jurisdiction?
Are follow up questions needed here: Yes.
Follow up: Is the defendant being sued in the state where the defendant is domiciled?
Intermediate answer: Yes.
Follow up: Did the defendant have sufficient contacts with the state?
Intermediate answer: Yes.
Follow up: Does the claim arise out of the nexus of the defendant's contacts with the state?
Intermediate answer: No.
So the final answer is: Yes.

Chain of Logic

Rule: [Personal Jurisdiction Rule]
Facts: [Personal Jurisdiction Fact Pattern]
Issue: Is there personal jurisdiction? Answer True or False.

Decompose rule into elements:
A: "in the state where the defendant is domiciled"
B: "(1) the defendant has sufficient contacts with the state, such that they have availed itself of the privileges of the state"
C: "(2) the claim arises out of the nexus of the defendant's contacts with the state"

Logical expression: (A or (B and C))

Answer each rule element separately:
Question: Is the defendant being sued in the state where the defendant is domiciled?
Answer: Laura sues Dustin in Arizona and Dustin lives in Arizona, therefore the answer is true.
Question: Did the defendant have sufficient contacts with the state?
Answer: Dustin lives in Arizona, therefore the answer is true.
Question: Does the claim arise out of the nexus of the defendant's contacts with the state?
Answer: The accident has no nexus with Arizona, therefore the answer is false.

Logical expression with answers: (true or (true and false))

By **solving the logical expression**, the final answer (True or False) is: True.

Figure 2: Comparing one-shot examples demonstrating chain of thought, self-ask and our chain of logic method on the Personal Jurisdiction task. Through a sequence of reasoning steps, chain of logic decomposes a rule into elements which are solved independently, before recomposing sub-answers to arrive at a final conclusion. See Section 3 for a detailed discussion on the chain of logic approach.

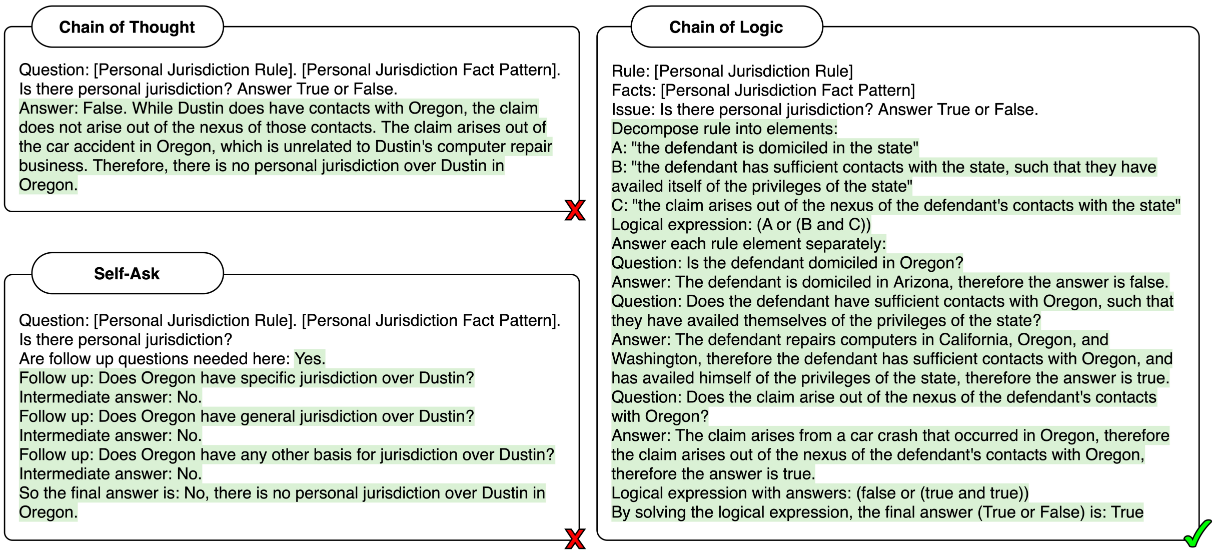
3 Chain of Logic

We introduce chain of logic, a new prompting approach to elicit rule-based reasoning in LMs through a series of instructive reasoning steps. Each step in this series helps inform the next, and enables the model to unravel the many reasoning tasks needed to arrive at the right conclusion. Our method builds on chain of thought and self-ask by not only considering problem decomposition, but also attending to relationships between subtasks that can dictate the final answer, which we later show improves performance. Our prompt begins with a single example demonstrating the chain of logic approach, appended with the inference-time problem. Figure 2 contains a one-shot example of our approach depicting the sequence of reasoning steps which we describe below:

- **Step 1:** Structured Input. Clearly label the task inputs to identify rule, facts and issue, similar to the IRAC Framework.
- **Step 2:** Rule Decomposition. Decompose the rule into elements by identifying relevant text spans and assigning elements to variables. Similar to the least-to-most approach (Zhou et al., 2023), our method first identifies the subtasks before trying to solve them.
- **Step 3:** Logical Expression. Construct a complex logical expression representing the logical relationships that exist between these element variables.
- **Step 4:** Question Answering. Iterate through each rule element, rephrasing the element as a question before providing a rationale and answer. Like self-ask, we dynamically construct this series of sub-questions, but for our approach this construction is informed by the elements identified in step 2. The model need only rephrase each element as question.
- **Step 5:** Element Recomposition. Replace element variables in the logical expression with the corresponding sub-answers from the previous step.
- **Step 6:** Resolve Expression. Resolve the logical expression populated with sub-answers to arrive at the final solution.

During inference, the model performs each step in this process automatically given only a one-shot example and the inference task inputs (formatted as described in step 1). Since the one-shot task inputs (rule, facts, issue) are different from the task inputs being tested, the model is reasoning on its own at each step, including deciding how many rule elements exist, the text span of each element and the logical relationships between them. This more closely resembles a real world setting where lawyers apply the IRAC framework to a rule and fact pattern being observed for the first time.

The intermediate steps in our approach provide a detailed accounting of each factor contributing to the final conclusion (see Figure 3 for an example of model output). This allows for transparent and explainable decision-making, which is particularly


Chain of Thought

Question: [Personal Jurisdiction Rule]. [Personal Jurisdiction Fact Pattern]. Is there personal jurisdiction? Answer True or False.
Answer: False. While Dustin does have contacts with Oregon, the claim does not arise out of the nexus of those contacts. The claim arises out of the car accident in Oregon, which is unrelated to Dustin's computer repair business. Therefore, there is no personal jurisdiction over Dustin in Oregon.

Self-Ask

Question: [Personal Jurisdiction Rule]. [Personal Jurisdiction Fact Pattern]. Is there personal jurisdiction?
Are follow up questions needed here: Yes.
Follow up: Does Oregon have specific jurisdiction over Dustin?
Intermediate answer: No.
Follow up: Does Oregon have general jurisdiction over Dustin?
Intermediate answer: No.
Follow up: Does Oregon have any other basis for jurisdiction over Dustin?
Intermediate answer: No.
So the final answer is: No, there is no personal jurisdiction over Dustin in Oregon.

Chain of Logic

Rule: [Personal Jurisdiction Rule]
Facts: [Personal Jurisdiction Fact Pattern]
Issue: Is there personal jurisdiction? Answer True or False.
Decompose rule into elements:
A: "the defendant is domiciled in the state"
B: "the defendant has sufficient contacts with the state, such that they have availed itself of the privileges of the state"
C: "the claim arises out of the nexus of the defendant's contacts with the state"
Logical expression: (A or (B and C))
Answer each rule element separately:
Question: Is the defendant domiciled in Oregon?
Answer: The defendant is domiciled in Arizona, therefore the answer is false.
Question: Does the defendant have sufficient contacts with Oregon, such that they have availed themselves of the privileges of the state?
Answer: The defendant repairs computers in California, Oregon, and Washington, therefore the defendant has sufficient contacts with Oregon, and has availed himself of the privileges of the state, therefore the answer is true.
Question: Does the claim arise out of the nexus of the defendant's contacts with Oregon?
Answer: The claim arises from a car crash that occurred in Oregon, therefore the claim arises out of the nexus of the defendant's contacts with Oregon, therefore the answer is true.
Logical expression with answers: (false or (true and true))
By solving the logical expression, the final answer (True or False) is: True

Figure 3: Comparing GPT-3.5 output for chain of thought, self-ask and our chain of logic method on the same Personal Jurisdiction task. Chain of logic prompting elicits large language models to reason about complex rules while also constructing an interpretable reasoning path. The prompts here are abridged, omitting a one-shot example for each method from the Diversity Jurisdiction task (see Section 4.1).

vital in the legal domain where the justification for an answer can be as important as the answer itself. Chain of logic also enables us to debug inaccurate conclusions by retracing the reasoning path to locate errors in rule application and logic.

4 Experiments

We conduct a series of experiments to compare our proposed chain of logic approach with existing prompting methods, and to evaluate the rule-based reasoning capabilities of LMs in a different-rule setting. While we sometimes observe an inverse relationship between prompt complexity and performance (simpler prompts perform better), we find that chain of logic consistently outperforms other prompting methods across a range of rule-based reasoning tasks.

4.1 Tasks

LegalBench (Guha et al., 2023) is a legal reasoning benchmark designed and constructed by an interdisciplinary team of computer scientists and legal professionals. The tasks in this benchmark have been designed by subject matter experts to measure legal reasoning capabilities that are both interesting and useful. We consider 8 rule-based reasoning tasks involving 3 distinct compositional rules from this benchmark:

- **Personal Jurisdiction** (1 task): This task involves determining whether a court has the authority to preside over a dispute based on where the defendant is domiciled, and whether the defendant had sufficient contacts with the forum state and the claim arose out of the nexus of those contacts. There are 50 test samples for this task.

- **Diversity Jurisdiction** (6 tasks): This task involves determining whether a federal court can preside over a lawsuit pertaining to state law based on whether complete diversity exists (no plaintiff and defendant are citizens of the same state) and the amount in controversy exceeding $75,000. There are 6 diversity jurisdiction tasks with increasingly complex fact patterns, from Diversity Jurisdiction 1 (easiest) to Diversity Jurisdiction 6 (hardest). There are 300 test samples for each diversity jurisdiction task.

- **J.Crew Blocker** (1 task): This task involves determining whether a provision from a loan agreement contains a J.Crew Blocker restrictive covenant based on whether the provision prohibits transferring IP to an unrestricted subsidiary or requires lender consent for IP transfers to a subsidiary. There are 54 test samples for this task.

Each sample from all tasks contains a rule, fact pattern and question. To correctly answer the question, the rule must be applied to the fact pattern.

	GPT-3.5	GPT-4	Llama-2	Mistral	Average
Zero-Shot	76.3	90.4	74.2	60.8	75.4
Zero-Shot-LR (Yu et al., 2022)	57.2	90.6	65.3	62.8	69.0
Zero-Shot-LS (Jiang and Yang, 2023)	75.5	90.1	58.4	52.2	69.1
Standard Prompting (Brown et al., 2020)	72.6	88.6	65.1	45.5	68.0
Chain of Thought (Wei et al., 2023)	70.1	89.1	71.8	46.4	69.4
Self-Ask (Press et al., 2023)	68.2	86.3	67.5	46.5	67.1
Chain of Logic (ours)	**87.0**	**92.3**	**74.6**	**63.1**	**79.3**

Table 1: Aggregated performance (accuracy) across all rules, including Personal Jurisdiction, Diversity Jurisdiction and J. Crew Blocker, using both open-source and commercial language models. Chain of logic consistently outperforms other prompting methods. All methods are one-shot, except zero-shot approaches.

The prompt for each task contains a one-shot example from another rule, followed by the test sample. The benchmark expects an answer of yes or no for each question. For our method we reformat the answer to true or false to more closely resemble formal logic. If the model's response is ambiguous, we use a second prompt "Therefore the answer (true or false) is", following Kojima et al. (Kojima et al., 2023).

4.2 Baseline Methods

In addition to our chain of logic approach (see Section 3), we examine six prompting methods. We first explore zero-shot prompting where the prompt includes only the test sample (rules, facts, question) with no in-context demonstrations. Blair-Stanek et al. showed that zero-shot prompting can outperform few-shot methods for some legal reasoning tasks, even with chain of thought reasoning (Blair-Stanek et al., 2023b). This suggests some models could have difficulty learning through in-context demonstrations in a legal setting. We include zero-shot prompting in our experiments to further explore this hypothesis. For completeness, we also include two zero-shot methods designed for legal reasoning tasks from prior work. Legal syllogism (LS) prompting is a zero-shot approach to legal judgement prediction where the prompt first defines legal syllogism before instructing the model to perform syllogistic reasoning (Jiang and Yang, 2023). Legal reasoning (LR) prompting (Yu et al., 2022) is also a zero-shot method where the prompt includes a simple approach description: "Approach: Issue, rule, application, conclusion".

For the remaining methods, we use a one-shot approach where a single demonstration with correct answer is included in the prompt. First, we use standard prompting (Brown et al., 2020) where the in-context demonstration includes only the sample and answer. Second, we explore chain of thought (Wei et al., 2023) prompting which includes a rational written by a legal professional explaining the relevant reasoning before the demonstration answer. Last, we include the self-ask approach (Press et al., 2023) where the one-shot example demonstrates explicitly asking and answering an intermediate question for each rule element before the final answer. See Figure 2 for an illustration of these one-shot examples. While self-ask was originally evaluated on another compositional reasoning task, multi-hop question answering, its demonstrated ability to increase reasoning capacity by decomposing complex problems makes it a compelling method to explore in this work.

4.3 Language Models

We experiment with two commercial models, GPT-3.5-Turbo and GPT-4 (OpenAI, 2023), and two leading open-source models, Llama-2-70b-chat (Touvron et al., 2023) and Mistral-7B-OpenOrca, which is a Mistral-7b (Jiang et al., 2023) model fine-tuned with Orca (Mukherjee et al., 2023). We select open-source models of disparate size to investigate the relationship between model size and rule-based reasoning abilities. For all language models we set the temperature to 0.0 to make the output more focused and reproducible, and otherwise use default settings.

5 Results and Discussion

Table 1 presents results for baseline methods and our chain of logic approach macro-averaged across all rule sets (Personal Jurisdiction, Diversity Jurisdiction and J.Crew Blocker), using commercial

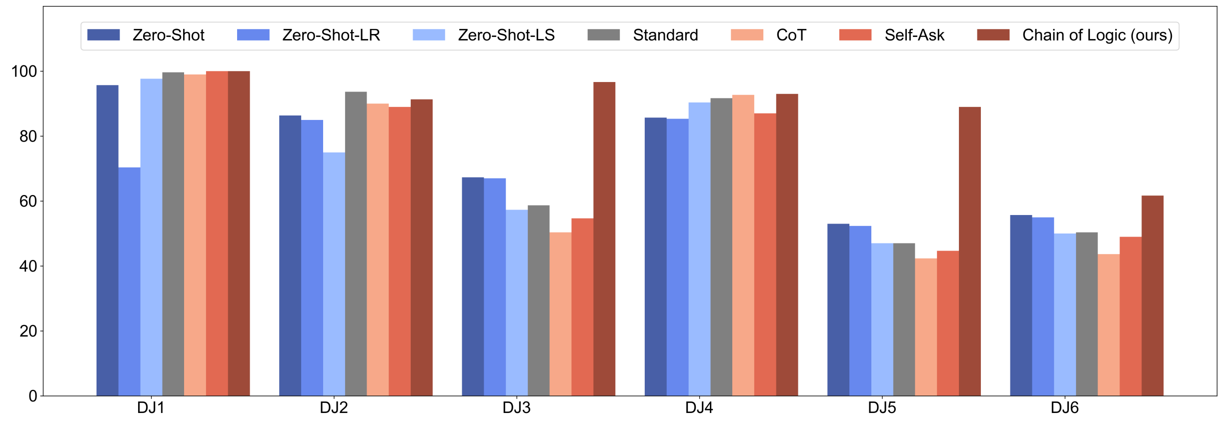


Figure 4: Accuracy (%) across all 6 Diversity Jurisdiction tasks using GPT-3.5. The fact patterns in these tasks are increasingly complex, from DJ1 (easiest) to DJ6 (hardest). Chain of logic particularly outperforms other prompting methods for tasks requiring arithmetic operations (DJ3, DJ5, DJ6). See Section 5.1 for a detailed discussion.

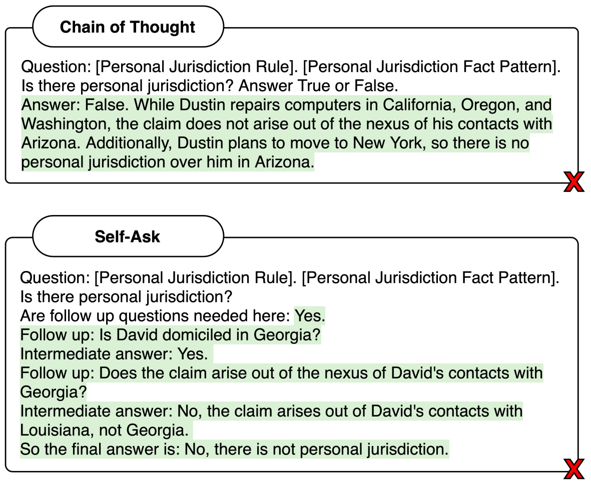
and open-source models. We report the macro-average here to provide a balanced view across rule sets, though as shown later in Figure 5.1 and the appendix, Chain of logic performs best on the Diversity Jurisdiction task, which is downweighted with this metric. See the Appendix for more detailed model performance by rule. For each model, a zero-shot method is the best performing baseline. This is consistent with prior work finding that zero-shot methods can outperform one-shot and few-shot prompting on legal reasoning tasks (Yu et al., 2022; Blair-Stanek et al., 2023b). Chain of logic significantly improves rule-based reasoning through in-context learning.

For each model, chain of logic outperforms all baseline methods, including zero-shot methods. The performance gap between chain of logic and the best performing baseline is 3.9% on average. This gap is smaller for open source models (0.3% and 0.4%) and larger for commercial models, with GPT-3.5 having the largest performance improvement of 10.7%. For this work we observe a large performance difference between the commercial and open source models, and hypothesize the performance gains may be larger for the stronger commercial models which can more easily follow the longer prompts.

This can also be seen in the performance differences between the baseline zero and one-shot methods. For GPT-4 we believe the absolute performance gain may be lower due to ceiling effects of the model. The specialized zero-shot legal baselines demonstrate comparable performance to zero-shot for some models, but are also inconsistent. LR prompting slightly outperforms zero-shot using Mistral-7b and GPT-4, but also underperforms by 19.1% for GPT-3.5.

The one-shot baselines demonstrate the challenges to in-context learning in this setting by consistently underperforming zero-shot methods. The model output depicted in Figures 3 and 5 illustrate common errors we observe in rule-based reasoning using chain of thought and self-ask. First, chain of thought often generates fluent but illogical reasoning paths. The output in Figure 5 shows the model incorrectly conclude that personal jurisdiction does not exist because Dustin plans to move to New York, even though future residency is not an element of the rule. Second, the question/answer pairs generated by the self-ask approach commonly decompose the rule incorrectly (Figure 3) or incompletely (Figure 5). The output in the latter figure shows the model only asks intermediate questions for two out of three rule elements, omitting whether the defendant had sufficient contact with the forum state. Chain of logic prevents these omissions by first restating the rule elements (step 2 in the approach), similar to how System 2 Attention (Weston and Sukhbaatar, 2023) first restates only the relevant portions of a context before attending to a task.

In the ablation study, we show that removing this step significantly decreases performance. Last, Figure 5 also demonstrates how not recognizing logical relationships can lead to an incorrect solution, even when sub-questions are answered correctly. The model correctly determines the defendant is domiciled in the forum state, which is sufficient to conclude personal jurisdiction exists, but still reaches the wrong final answer.


Chain of Thought

Question: [Personal Jurisdiction Rule]. [Personal Jurisdiction Fact Pattern]. Is there personal jurisdiction? Answer True or False.
Answer: False. While Dustin repairs computers in California, Oregon, and Washington, the claim does not arise out of the nexus of his contacts with Arizona. Additionally, Dustin plans to move to New York, so there is no personal jurisdiction over him in Arizona.

Self-Ask

Question: [Personal Jurisdiction Rule]. [Personal Jurisdiction Fact Pattern]. Is there personal jurisdiction?
Are follow up questions needed here: Yes.
Follow up: Is David domiciled in Georgia?
Intermediate answer: Yes.
Follow up: Does the claim arise out of the nexus of David's contacts with Georgia?
Intermediate answer: No, the claim arises out of David's contacts with Louisiana, not Georgia.
So the final answer is: No, there is not personal jurisdiction.

Figure 5: GPT-4 output demonstrating common errors in one-shot methods.

5.1 Diversity Jurisdiction Series

The Diversity Jurisdiction series gives us a unique opportunity to examine how performance is affected by task complexity. Recall there are 6 tasks involving the diversity jurisdiction rule. The fact patterns in these tasks increase in complexity across two dimensions: 1) the number of parties (plaintiffs and defendants), and the number of claims per plaintiff-defendant pair. The simplest task, Diversity Jurisdiction 1 (DJ1), contains one plaintiff, one defendant and one claim. The most complex task, Diversity Jurisdiction 6 (DJ6), contains two plaintiffs, two defendants, and 4 total claims.

Figure 4 shows results across all 6 Diversity Jurisdiction tasks using GPT-3.5. All methods, except zero-shot-LR, perform above 95% accuracy on DJ1, yet even the best performing method, chain of logic, only scores a 61.7% accuracy on DJ6. Chain of logic outperforms the baseline methods for 5 out of 6 tasks, with standard prompting outperforming our method by 2.3% for DJ2. Most notably, chain of logic significantly outperforms other methods on tasks requiring arithmetic reasoning to add dollar amounts from multiple claims (DJ3, DJ5, DJ6). The performance gap between chain of logic and zero-shot, the best performing baseline, is 36% for DJ5. All baselines perform near random chance for this task. For tasks involving arithmetic reasoning, the baselines also demonstrate an inverse relationship between prompt complexity and performance. That is to say, simpler prompts tend to perform better. Chain of logic is the clear exception to this trend being both the most complex prompt and the best performing model.

Reasoning Step	PJ
Step 1: Structured Input	-25.9
Step 2: Rule Decomposition	-12.9
Step 3: Logical Expression	-14.8
Step 4: Question Answering	-49.3
Step 5: Element Recomposition	-5.5
Step 6: Resolve Expression	-3.7

Table 2: **Ablations.** Reduction in performance (absolute percentage) observed when removing the specified reasoning step from the chain of logic approach using GPT-3.5 on the Personal Jurisdiction task.

5.2 Ablation Study

Our experiments show chain of logic improves in-context learning for rule-based reasoning tasks. In this section, we assess the contribution of each reasoning step to the overall performance as shown in Table 2. We isolate these effects by removing a single reasoning step at a time. We evaluate on the most complex rule, personal jurisdiction, using the model with the best comparable performance, GPT-3.5. Unsurprisingly, removing step 4 drastically reduces overall performance since there is no clear path between the logical expressions in steps 3 and 5. Notably, the removal of step 1, switching from structured to unstructured task inputs, leads to a 25.9 absolute percentage point performance decrease. We believe this structured content is useful for the rule decomposition step that follows. Similarly, the ablations show that decomposing the rule into elements (step 2: -12.9%) and generating a logical expression representing the relationships between elements (step 3: -14.8%) are both critical to the performance gains we observe for this approach.

6 Related Work

There has been extensive work on improving the problem solving capabilities of LMs by enabling a model to use more computation for more complex tasks. Graves introduces an adaptive algorithm for dynamically selecting the number of computation steps a recurrent neural network should take based on the task complexity (Graves, 2017). Ling et al. first demonstrated the use of answer rationales, natural language explanations generated before the solution, to solve algebraic word problems (Ling et al., 2017). Several approaches followed this work in exploring the use of intermediate reasoning steps (Nye et al., 2021; Wei et al., 2023; Wang et al., 2023). More similar to our work, other methods have explored decomposing compositional questions into simpler sub-questions using supervised models (Qi et al., 2019; Min et al., 2019; Khot et al., 2021) and prompting techniques (Mishra et al., 2022; Press et al., 2023). Talmor and Berant also recognized the importance of explicitly identifying relationships between sub-questions (Talmor and Berant, 2018). The authors train a supervised model to translate questions into a computation tree and answer sub-questions by querying a search engine. We extend these works with our decomposition-recomposition approach, exploring simple questions involving compositional rules where the solution depends not only on rule element answers, but also the logical relationships between elements.

Recently, agent-based systems driven by LMs have demonstrated strong compositional reasoning capabilities through task planning and program execution (Lu et al., 2023; Shen et al., 2023; Yin et al., 2024). Our method is not agent-based - instead of performing many forward passes across an assortment of specialized models and programs, our method demonstrates similar decomposition/recomposition with only a single generation pass on one model. Additionally, our approach does not require fine-tuning which is particularly important in the legal domain where the scarcity of annotated data can be prohibitive.

Prior work in natural language processing has explored a broad range of legal tasks, including clause classification (Hendrycks et al., 2021) and recommendation (Aggarwal et al., 2021), contract summarization (Manor and Li, 2019), legal judgment prediction (Chalkidis et al., 2019) and even answering bar exam questions (au2 and Katz, 2022; Zhong et al., 2019). Rule-based reasoning can involve rules from many sources. Saeidi et al. explored conversational question answering where the solution required rule-based reasoning about a collection of regulations (Saeidi et al., 2018). Servantez et al. captured rules in contract text through graph-based extraction and converted them into code (Servantez et al., 2023). Similar to our work, Holzenberger and Van Durme introduced an approach to reasoning about tax statutes by decomposing the reasoning process (Holzenberger and Van Durme, 2021). This approach first extracts key arguments (entities, dates, dollar amounts) from the statute text and fact pattern using fine-tuned BERT models, before arriving at a final answer. We do not explore this method since it requires a fine-tuned model and does not easily translate to the tasks in this work.

7 Limitations and Future Work

While LegalBench tasks were crafted by subject matter experts, the rules have been simplified to ensure answers are objectively correct. This greatly improves evaluation, but also means the scope of complexity is narrower than many real world rules. Additionally, our current approach only addresses rules where the solution is based on whether the antecedent has been triggered. Real world rules can contain complex consequences which themselves require some form of reasoning. For example, calculating tax liability after determining the applicable tax bracket. In future work, we hope to build on our current approach in several directions. First, for simplicity chain of logic performs all reasoning step in a single forward pass. However, a multiple pass approach would allow us to incorporate other reasoning tasks like rule identification which is not addressed here. This moves us toward more real world scenarios where the applicable rule is not known a priori. Second, investigating whether rule-based reasoning could be improved further by dynamically sampling multiple reasoning paths. And third, incorporating retrieval augmented generation (Lewis et al., 2021) by allowing the model to access external sources of knowledge like term definitions. This is particularly useful in the legal domain where terms or concepts have a distinct meaning based on the jurisdiction or contract.

8 Conclusion

In this work, we explore causal language models as rule-based reasoners, and show these models can have difficulty learning from in-context demonstrations using common prompting approaches. We present a new prompting method, chain of logic, to elicit rule-based reasoning in language models through decomposition and recomposition. Our experiments show chain of logic consistently outperforms other prompting methods, including chain of thought and self-ask, across a variety of rule-based reasoning tasks using both open-source and commercial language models. We show how chain of logic creates a coherent and interpretable reasoning path by unraveling the many reasoning steps required by compositional rules. These findings may also be a useful signal for future instruction tuning of language models to imbue them with reasoning. By enhancing the rule-based reasoning capabilities of LMs through in-context learning, chain of logic also reduces the need for annotated legal data, which has historically been a bottleneck for the legal domain.

Acknowledgements

This work was supported in part by Adobe Research. The authors thank Neel Guha for helpful discussions related to LegalBench.

References

Vinay Aggarwal, Aparna Garimella, Balaji Vasan Srinivasan, Anandhavelu N, and Rajiv Jain. 2021. ClauseRec: A clause recommendation framework for AI-aided contract authoring. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8770–8776, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Bommarito II au2 and Daniel Martin Katz. 2022. Gpt takes the bar exam.

Andrew Blair-Stanek, Nils Holzenberger, and Benjamin Van Durme. 2023a. Blt: Can large language models handle basic legal text?

Andrew Blair-Stanek, Nils Holzenberger, and Benjamin Van Durme. 2023b. Can gpt-3 perform statutory reasoning? In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, ICAIL '23, page 22–31, New York, NY, USA. Association for Computing Machinery.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in english.

Matthew Dahl, Varun Magesh, Mirac Suzgun, and Daniel E. Ho. 2024. Large legal fictions: Profiling legal hallucinations in large language models.

Alex Graves. 2017. Adaptive computation time for recurrent neural networks.

Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, Jessica Wu, Joe Nudell, Joel Niklaus, John Nay, Jonathan H. Choi, Kevin Tobia, Margaret Hagan, Megan Ma, Michael Livermore, Nikon Rasumov-Rahe, Nils Holzenberger, Noam Kolt, Peter Henderson, Sean Rehaag, Sharad Goel, Shang Gao, Spencer Williams, Sunny Gandhi, Tom Zur, Varun Iyer, and Zehua Li. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. Cuad: An expert-annotated nlp dataset for legal contract review.

Nils Holzenberger and Benjamin Van Durme. 2021. Factoring statutory reasoning as language understanding challenges. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2742–2758, Online. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Cong Jiang and Xiaolei Yang. 2023. Legal syllogism prompting: Teaching large language models for legal judgment prediction.

Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2021. Text modular networks: Learning to decompose tasks in the language of existing models.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation : Learning to solve and explain algebraic word problems.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2023. Chameleon: Plug-and-play compositional reasoning with large language models.

Laura Manor and Junyi Jessy Li. 2019. Plain English summarization of contracts. In *Proceedings of the Natural Legal Language Processing Workshop 2019*, pages 1–11, Minneapolis, Minnesota. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop reading comprehension through question decomposition and rescoring.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022. Reframing instructional prompts to gptk's language.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show your work: Scratchpads for intermediate computation with language models.

OpenAI. 2023. Gpt-4 technical report. *View in Article*, 2:13.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation.

Martha T Roth. 1995. *Law collections from Mesopotamia and Asia minor*, volume 6. Scholars Press.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. 2018. Interpretation of natural language rules in conversational machine reading.

Sergio Servantez, Nedim Lipka, Alexa Siu, Milan Aggarwal, Balaji Krishnamurthy, Aparna Garimella, Kristian Hammond, and Rajiv Jain. 2023. Computable contracts by extracting obligation logic graphs. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, ICAIL '23, page 267–276, New York, NY, USA. Association for Computing Machinery.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Jason Weston and Sainbayar Sukhbaatar. 2023. System 2 attention (is something you might need too).

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark.

Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. 2024. Agent lumos: Unified and modular training for open-source language agents.

Fangyi Yu, Lee Quartey, and Frank Schilder. 2022. Legal prompting: Teaching a language model to think like a lawyer.

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2019. Jec-qa: A legal-domain question answering dataset.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models.

A Appendix

A.1 LegalBench Task Examples

Personal Jurisdiction (Figure 3)

- Rule: There is personal jurisdiction over a defendant in the state where the defendant is domiciled, or when (1) the defendant has sufficient contacts with the state, such that they have availed itself of the privileges of the state and (2) the claim arises out of the nexus of the defendant's contacts with the state.
- Fact Pattern: Dustin is a repairman who lives in Arizona and repairs computers in California, Oregon, and Washington. While travelling to repair a computer in Washington, Dustin is involved in a car crash in Oregon with Laura, a citizen of Texas. After the accident, Dustin returns to Arizona. Laura sues him in Oregon.
- Issue: Is there personal jurisdiction?

Diversity Jurisdiction 1

- Rule: Diversity jurisdiction exists when there is (1) complete diversity between plaintiffs and defendants, and (2) the amount-in-controversy (AiC) is greater than $75k.
- Fact Pattern: James is from Arizona. Lucas is from Arizona. James sues Lucas for negligence for $64,000.
- Issue: Is there diversity jurisdiction?

Diversity Jurisdiction 2

- Rule: Diversity jurisdiction exists when there is (1) complete diversity between plaintiffs and defendants, and (2) the amount-in-controversy (AiC) is greater than $75k.
- Fact Pattern: Sophia is from Arkansas. Benjamin is from Hawaii. Noah is from Arkansas. Sophia sues Benjamin and Noah each for defamation for $24,000.
- Issue: Is there diversity jurisdiction?

Diversity Jurisdiction 3

- Rule: Diversity jurisdiction exists when there is (1) complete diversity between plaintiffs and defendants, and (2) the amount-in-controversy (AiC) is greater than $75k.
- Fact Pattern: William is from Montana. Theodore is from Connecticut. William sues Theodore for medical malpractice for $9,000 and negligence for $35,000.
- Issue: Is there diversity jurisdiction?

Diversity Jurisdiction 4

- Rule: Diversity jurisdiction exists when there is (1) complete diversity between plaintiffs and defendants, and (2) the amount-in-controversy (AiC) is greater than $75k.
- Fact Pattern: Emma is from New Hampshire. Mia is from Wisconsin. Evelyn is from California. Emma and Mia both sue Evelyn for copyright infringement for $5,400,000.
- Issue: Is there diversity jurisdiction?

Diversity Jurisdiction 5

- Rule: Diversity jurisdiction exists when there is (1) complete diversity between plaintiffs and defendants, and (2) the amount-in-controversy (AiC) is greater than $75k.
- Fact Pattern: Elijah is from Hawaii. Ava is from Oklahoma. Amelia is from Minnesota. Elijah and Ava both sue Amelia for defamation for $3,000 and copyright infringement for $80,000.
- Issue: Is there diversity jurisdiction?

Diversity Jurisdiction 6

- Rule: Diversity jurisdiction exists when there is (1) complete diversity between plaintiffs and defendants, and (2) the amount-in-controversy (AiC) is greater than $75k.
- Fact Pattern: Theodore is from North Dakota. Amelia is from Georgia. Benjamin is from Delaware. Mia is from Illinois. Theodore and Amelia both sue Benjamin for trademark infringement for $42,000 and copyright infringement for $71,000. Theodore and Amelia both sue Mia for securities fraud for $45,000 and medical malpractice for $57,000.
- Issue: Is there diversity jurisdiction?

J.Crew Blocker

- Rule: The JCrew Blocker is a provision that typically includes (1) a prohibition on the borrower from transferring IP to an unrestricted subsidiary, and (2) a requirement that the borrower obtains the consent of its agent/lenders before transferring IP to any subsidiary.

- Fact Pattern: Notwithstanding anything to foregoing, no Intellectual Property that is material to the Borrower and its Restricted Subsidiaries, taken as a whole (as reasonably determined by the Borrower), shall be owned by or licensed, contributed or otherwise transferred to any Unrestricted Subsidiary.
- Issue: Do the following provisions contain JCrew Blockers?

A.2 Model Performance by Rule

Model performance (accuracy) by rule for all reasoning tasks. Diversity Jurisdiction results are aggregated across all six datasets.

	Personal Jurisdiction	Diversity Jurisdiction	J.Crew Blocker
Zero-Shot	68.0	73.9	87.0
Zero-Shot-LR	60.0	69.2	42.6
Zero-Shot-LS	70.0	69.5	87.0
Standard Prompting	72.0	73.5	72.2
Chain of Thought	74.0	69.7	66.7
Self-Ask	60.0	70.7	74.0
Chain of Logic (ours)	**78.0**	**88.6**	**94.4**

Table 3: Performance by rule using GPT-3.5

	Personal Jurisdiction	Diversity Jurisdiction	J.Crew Blocker
Zero-Shot	84.0	87.1	**100**
Zero-Shot-LR	86.0	87.8	98.1
Zero-Shot-LS	78.0	94.2	98.1
Standard Prompting	76.0	93.6	96.3
Chain of Thought	82.0	89.2	96.3
Self-Ask	84.0	78.5	96.3
Chain of Logic (ours)	**90.0**	**94.3**	92.6

Table 4: Performance by rule using GPT-4

	Personal Jurisdiction	Diversity Jurisdiction	J.Crew Blocker
Zero-Shot	66.0	65.9	90.7
Zero-Shot-LR	52.0	66.0	77.8
Zero-Shot-LS	42.0	50.0	83.3
Standard Prompting	56.0	57.7	81.5
Chain of Thought	62.0	60.7	**92.6**
Self-Ask	70.0	52.8	79.6
Chain of Logic (ours)	**72.0**	**66.6**	85.2

Table 5: Performance by rule using Llama-2-70b

	Personal Jurisdiction	Diversity Jurisdiction	J.Crew Blocker
Zero-Shot	**56.0**	65.4	61.1
Zero-Shot-LR	50.0	66.1	**72.2**
Zero-Shot-LS	48.0	62.4	46.3
Standard Prompting	44.0	46.3	46.3
Chain of Thought	36.0	66.2	37.0
Self-Ask	52.0	70.8	16.7
Chain of Logic (ours)	54.0	**72.4**	63.0

Table 6: Performance by rule using MistralOrca-7b