LegalDrill: Diagnosis-Driven Synthesis for Legal Reasoning in Small Language Models

Tianchun Li[1], Haochen Liu[2], Vishwa Pardeshi[2], Xingchen Wang[1], Tianci Liu[1], Huijun Zhao[2], Wei Fan[2], Jing Gao[1]

[1]Purdue University, West Lafayette, IN, USA
[2]Fidelity Investments, Boston, MA, USA

{li2657, wang2930, liu3351, jinggao}@purdue.edu
{haochen.liu, vishwa.pardeshi, huijun.zhao, wei.fan}@fmr.com

Abstract

Small language models (SLMs) are promising for real-world deployment due to their efficiency and low operational cost. However, their limited capacity struggles with high-stakes legal reasoning tasks that require coherent statute interpretation and logically consistent deduction. Furthermore, training SLMs for such tasks demands high-quality, concise reasoning trajectories, which are prohibitively expensive to manually collect and difficult to curate via standard rejection sampling, which lacks granularity beyond final verdicts. To address these challenges, we propose LegalDrill, a diagnosis-driven synthesis framework that extracts and iteratively refines reasoning trajectories from a capable teacher via fine-grained prompting, then a self-reflective verification is employed to adaptively select the most effective data for the SLM student. The resulting data empower SLM training through supervised fine-tuning and direct preference optimization. Extensive experiments on several legal benchmarks demonstrate that LegalDrill significantly bolsters the legal reasoning capabilities of representative SLMs while bypassing the need for scarce expert annotations, paving a scalable path toward practical legal reasoning systems.

1 Introduction

Large Language Models (LLMs) (Vaswani et al., 2017; Radford et al., 2019; Brown et al., 2020; Comanici et al., 2025) have demonstrated remarkable capabilities to understand, reason, and generate text (Gu et al., 2024; Nam et al., 2024; Guo et al., 2025), pushing new boundaries across a wide range of domains. As one of the most critical applications, there has been a growing trend to leverage LLMs for legal domains in legal document retrieval (Siino et al., 2025), judgment prediction (Wu et al., 2023), and complex legal question answering (Guha et al., 2023).

Despite their powerful capabilities, legal-domain frequently handles sensitive personal and confidential information. Routing such data to external APIs (e.g., GPT or Gemini) or cloud-based retrieval-augmented generation (RAG) pipelines introduces unacceptable privacy and security risks (Siino et al., 2025). Therefore, it is strictly required that models be deployed within secure, local environments. The larger variants of open-sourced LLMs, such as Qwen3-32B (Yang et al., 2025a) or Llama3-70B (Grattafiori et al., 2024), can be deployed locally but require expensive computational resources. Consequently, Small Language Models (SLMs)[1] have emerged as the pragmatic choice for secure, on-device deployment for legal applications (Lu et al., 2024).

However, the utility of SLMs remains limited in legal domains due to their weak reasoning ability (Fei et al., 2024; Yu et al., 2025). Legal reasoning is not merely retrieving relevant clauses, but coherently interpreting statutes under case-specific (often ambiguous) contexts and carrying out logically valid deductions (Levi, 2022; Fan et al., 2025; Ioannou et al., 2025). In practice, current SLMs often generate fluent, lawyer-like narratives, yet their reasoning is often fragile. They frequently make subtle errors—such as misreading statutory terms or making logical leaps—that undermine their final arguments. This gap is fundamentally tied to limited model size: constrained by their parameter scale, SLMs struggle to represent and execute the multi-step, dependency-heavy reasoning required for statute interpretation and logically consistent deduction.

Ideally, fine-tuning SLMs for high-stakes legal reasoning requires *high-quality and concise* reasoning trajectories (Li et al., 2025b; Wang et al., 2026). Since manually collecting such legal reasoning data at scale is prohibitively expensive, a

[1]In this work, we refer to SLMs as models with a parameter size of less than 3B.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 1755–1773
July 2-7, 2026

pragmatic alternative is to leverage stronger LLMs to generate reasoning traces via rejection sampling, e.g., selecting samples that match the final verdict or satisfy basic formatting constraints. However, this approach faces a fundamental behavioral and learnability mismatch between LLMs and SLMs (Ranaldi and Freitas, 2024; Guo et al., 2025; Yeo et al., 2025). Reinforcement Learning (RL)-aligned LLMs typically rely on verbose, self-corrective deliberations—exhaustively exploring alternatives and revisiting premises—to ensure correctness (Jaech et al., 2024; Guo et al., 2025). In contrast, recent findings suggest that SLMs, constrained by their limited parameter scale, cannot effectively internalize or benefit from learning and imitating such ultra-long reasoning chains (Liu et al., 2025c; Li et al., 2025b; Yang et al., 2025b). This discrepancy renders standard rejection sampling insufficient for the legal domain: it fails to curate legal reasoning paths that are concise enough for SLMs to learn.

To bridge this behavioral gap and enable SLMs to *learn effectively*, we propose LegalDrill, an iterative framework designed to synthesize high-quality, concise reasoning trajectories tailored to the capacity of the student model. Instead of generating verbose, exploration-heavy chains that overwhelm smaller models from stronger models, we employ a diagnosis-driven mechanism. An audit Agent first scrutinizes the SLM's errors to pinpoint root causes, such as statutory misinterpretation or logical leaps. Guided by this diagnosis, a stronger model synthesizes a preference pair: a *rejected* response mimicking the specific error pattern, and a *chosen* response that rectifies it through a *concise*, logically tight deduction. This process effectively converts the implicit knowledge of LLMs into compact, focused reasoning paths that are explicitly aligned with the SLM's learning behavior.

Furthermore, to ensure that the SLM learns efficiently, we introduce a self-reflective verification mechanism to filter out trivial samples. The preference pair generated by a stronger model might be objectively high-quality but subjectively trivial if the SLM is already capable of recognizing the correct reasoning. To address this mismatch, we leverage the SLM's own probability distribution to calculate a Difficulty Score that quantifies the model's confusion level, which is then used to filter the synthesized pairs. Specifically, this score identifies instances where the SLM incorrectly assigns higher confidence to the wrong reasoning (rejected response) over the corrected one (chosen response). By retaining these *confused* samples, we curate a highly targeted training set that focuses exclusively on the model's actual blind spots. Finally, the SLM is optimized on these verified pairs via Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO) (Rafailov et al., 2023).

The contribution of our work is summarized as follows: First, we propose a diagnosis-driven synthesis framework that translates the implicit knowledge of strong LLMs into concise, error-correcting reasoning traces specifically tailored to the SLM's capacity. Second, we introduce a self-reflective verification mechanism that filters training data based on the student model's intrinsic confusion, ensuring that the model focuses on its actual blind spots. Finally, our framework demonstrates effectiveness not only on open benchmarks but also in a *real-world industrial setting* with experiments on proprietary datasets, comprising complex legal documents and financial contracts.

2 Preliminaries

2.1 Legal Task Formulation

Legal reasoning requires analytical explanation and concept interpretation. For a given dataset $\mathcal{D}$, we denote the input as $x = (c, q)$, where c is the legal context and q is the query. Given x, the model generates an output sequence y that contains the reasoning and, when applicable, the final answer. Depending on the task, y may include an explicit verdict/prediction (e.g., judgment prediction) or consist purely of explanation (e.g., concept interpretation). Unless stated otherwise, we treat y as a single sequence and omit the verdict a.

2.2 Iterative Preference Optimization for Reasoning

Direct Preference Optimization (DPO) (Rafailov et al., 2023) and its variants (Azar et al., 2024; Melnyk et al., 2024; D'Oosterlinck et al., 2025; Jung et al., 2025), instead of relying on an explicit reward model, directly uses pair-wise preference data to optimize the policy model with an equivalent optimization objective. Specifically, for a preference triple $(x, y^+, y^-) \sim \mathcal{D}$, DPO minimizes the following objective

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}\Big[\log \sigma\big(\beta(\log \tfrac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \log \tfrac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)})\big)\Big],$$

where π_θ is the policy model, π_{ref} is the fixed reference model, $\sigma(\cdot)$ is the sigmoid function, and β is a parameter controlling the deviation from the reference model.

Recent works (Pang et al., 2024; Deng and Mineiro, 2024; Guo et al., 2024; Tu et al., 2025; Xu et al., 2025) suggest that the reasoning abilities of language models could also be improved through an iterative online DPO. The training process will take T rounds. At each iteration, the current model π_{θ_t} interacts with the environment (or a stronger teacher model) to generate a preference dataset $\mathcal{D}^t$. The model is then updated to $\pi_{\theta_{t+1}}$ by minimizing the $\mathcal{L}_{\text{DPO}}$ on $\mathcal{D}^t$.

3 Methodology

As illustrated in Fig. 1, we propose an iterative framework designed to progressively refine the legal reasoning capabilities of SLMs. In the following sections, we refer to the stronger LLM as the ***teacher model*** and the target SLM as the ***student model***. At each iteration t, we first synthesize reasoning trajectories that are concise and explicitly aligned with the student's behavior (Section 3.1). We then employ a verification mechanism to filter these trajectories based on the student's current capabilities, ensuring only non-trivial samples are retained (Section 3.2). Finally, in Section 3.3, we leverage this curated dataset to update the student model's parameters.

3.1 Agent-Driven Instruction Refinement and Preference Data Generation

To bridge the gap between LLM capabilities and SLM constraints, we propose an iterative data synthesis framework. The core intuition is to diagnose the SLM's specific error patterns and synthesize concise, corrective reasoning traces that directly address these weaknesses. We structure this process into three stages: *Exploration*, *Diagnosis*, and *Targeted Generation*.

Exploration: Given a portion of the training data consisting of N legal queries, the *Exploration* stage prompts the current student model π_{θ_t} to generate a preliminary response $\hat{y}_i \sim \pi_{\theta_t}(\cdot \mid x_i)$ for each query x_i. To further encourage detailed reasoning traces, we utilize the Chain-of-Thought (CoT) system prompt, forcing the model to explain its internal logic and thereby making latent reasoning errors observable.

Diagnosis: For *Diagnosis*, we employ a specialized *Audit Agent* (π_{audit}) to scrutinize each student response $\hat{y}_i$. The Agent identifies root causes of reasoning errors (e.g., misinterpretation of statutes or logical leaps) and abstracts them into generalized *error simulation instructions*. To ensure that the diagnosis aligns with industry standards, we inject a taxonomy of common legal mistakes into the Agent's prompt. Formally, for each sample in the batch, we let the Agent to generate an instruction:

$$\mathcal{I}^{(i)} \sim \pi_{\text{audit}}(\cdot \mid \textit{Agent Prompt}(x_i, \hat{y}_i)).$$

Crucially, we constrain the Agent to generate *context-agnostic* instructions (e.g., "Ignore the time frame when calculating the limitation period"). This decoupling allows us to compile a reusable *Error Instruction Bank*, denoted as $\Phi_{\text{err}} = \{\mathcal{I}^{(1)}, \ldots, \mathcal{I}^{(N)}\}$, where instructions are no longer bound to their original source contexts.

Targeted Generation: Finally, we leverage the *Error Instruction Bank* Φ_{err} to synthesize preference data at scale. The key motivation is twofold. First, because each instruction $\mathcal{I}$ is *context-agnostic*, the same reasoning failure can be reproduced under many different legal contexts. This acts as a regularizer against shortcut learning: the contrast between the chosen and rejected responses is driven by reasoning rigor rather than superficial cues (e.g., response length or lexical patterns). Second, decoupling *error types* from *contexts* enables combinatorial expansion: we can generate arbitrarily many training pairs by recombining contexts with diverse error instructions, instead of being constrained by the size of the original dataset.

Concretely, for each training sample, we sample K error instructions $\{\mathcal{I}_k\}_{k=1}^{K}$ from Φ_{err}, where K is a hyperparameter controlling the expansion ratio. For each instruction $\mathcal{I}_k$, a stronger *teacher model* (π_{teach}) synthesizes one preference pair by a two-step procedure.

For each instruction $\mathcal{I}_k$, the teacher model first generates a *rejected response* by intentionally following the specified erroneous logic,

$$y_-^{(k)} \sim \pi_{\text{teach}}(\cdot \mid x, \mathcal{I}_k).$$

To generate a high-quality *chosen* response in a more targeted way, we further provide the teacher model with this paired *rejected* response. The generation process is given by

$$y_+^{(k)} \sim \pi_{\text{teach}}(\cdot \mid x, \mathcal{I}_k, y_-^{(k)}).$$

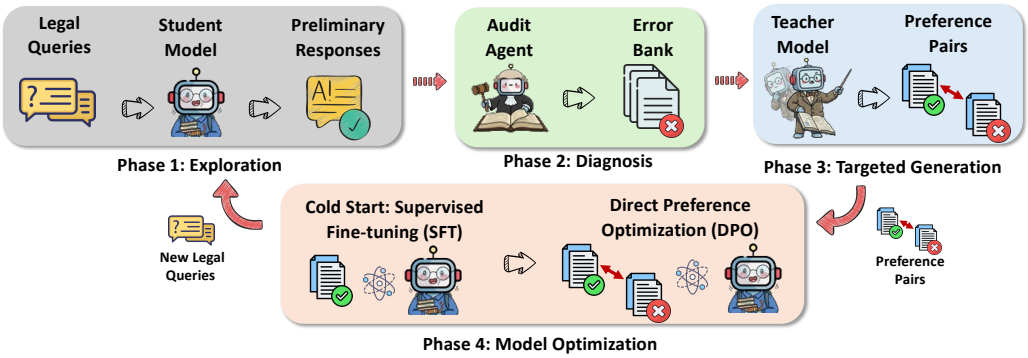

Figure 1: The overview of LegalDrill.

Overall, each original sample x yields K preference pairs, leading to a synthesized dataset of size $K \cdot |\mathcal{D}|$:

$$\mathcal{D}^t_{\text{syn}} = \{(x, y^{(k)}_+, y^{(k)}_-) \mid x \in \mathcal{D},\ k = 1, \ldots, K\}.$$

3.2 Self-Reflective Quality Verification

The synthetic pairs $\mathcal{D}^t_{\text{syn}}$ generated in the previous stage may contain samples that are trivial if the student model π_{θ_t} can already determine the correct reasoning. To focus optimization on the model's actual blind spots, we introduce a *Self-Reflective Verification* mechanism. This process filters the dataset to retain only those pairs where the student genuinely struggles to differentiate the correct reasoning trajectory from the error.

We propose the *Difficulty Score* (DS) to quantify the alignment-or-conflict between the student model's belief and the teacher-synthesized preferences. Importantly, we do not estimate this belief via the likelihood $\pi_{\theta_t}(y \mid x)$ over the entire sequence, which could be sensitive to response length and surface form. Instead, we leverage the student's instruction-following capability to perform a forced-choice prediction task.

Concretely, given a legal context c, query q, and a candidate response y, we construct a structured verification prompt $\mathcal{P}_{\text{ver}}(c, q, y)$ that mirrors the formulation of the legal task described in Sec. 2. We strictly constrain the output words to a binary set $\mathcal{V} = \{\texttt{correct}, \texttt{incorrect}\}$ in the prompt $\mathcal{P}_{\text{ver}}(c, q, y)$. Rather than relying on raw probabilities, we normalize the prediction confidence over these two words to isolate the model's prediction. We define the normalized correctness score as:

$$s_{\theta_t}(y \mid x) = \frac{\pi_{\theta_t}(\texttt{correct} \mid \mathcal{P}_{\text{ver}})}{\pi_{\theta_t}(\texttt{correct} \mid \mathcal{P}_{\text{ver}}) + \pi_{\theta_t}(\texttt{incorrect} \mid \mathcal{P}_{\text{ver}})}$$

This normalization ensures that even if the absolute probability of generating the specific token "correct" fluctuates during the iterative training, the *relative* preference remains a valid signal of the model's internal belief. For each preference pair $(x, y^{(k)}_+, y^{(k)}_-)$, we simply compute the Difficulty Score as the margin between the student's endorsement of the error and the correction:

$$\text{DS}(x, y^{(k)}_+, y^{(k)}_-) = s_{\theta_t}(y^{(k)}_- \mid x) - s_{\theta_t}(y^{(k)}_+ \mid x).$$

The DS measures how strongly the student is *misled* by the rejected reasoning relative to the chosen one. If DS < 0, the student identifies the correct reasoning $y^{(k)}_+$, making the pair relatively easy. If DS > 0, the student explicitly assigns higher confidence to the flawed reasoning $y^{(k)}_-$, revealing a genuine blind spot (or "confusion"). Based on this metric, we construct the final training set $\mathcal{D}^t_{\text{train}}$ by strictly filtering for high-value samples. We apply a thresholding strategy where only pairs satisfying DS $> \tau$ are retained. This filtering focuses the optimization budget exclusively on logical fallacies where the student model is most vulnerable.

3.3 Optimization Objectives: SFT and DPO

After targeted generation and self-reflective filtering, we obtain a training set of preference triples $(x, y^{(k)}_+, y^{(k)}_-) \in \mathcal{D}^t_{\text{train}}$. We optimize the student model in two stages. First, when $t = 0$ we perform supervised fine-tuning on the teacher-preferred responses to provide a stable cold start for preference optimization. Then, we apply DPO on the filtered preference pairs.

Supervised fine-tuning (cold start). When $t = 0$, we initialize the student by maximizing the likelihood of the chosen responses:

$$\mathcal{L}_{\text{SFT}}(\theta_0) = -\mathbb{E}_{(x, y_+) \sim \mathcal{D}^0_{\text{train}}} \big[\log \pi_{\theta_0}(y_+ \mid x)\big].$$

This warm-up stage anchors the model on high-quality reasoning traces before applying the pairwise preference objective.

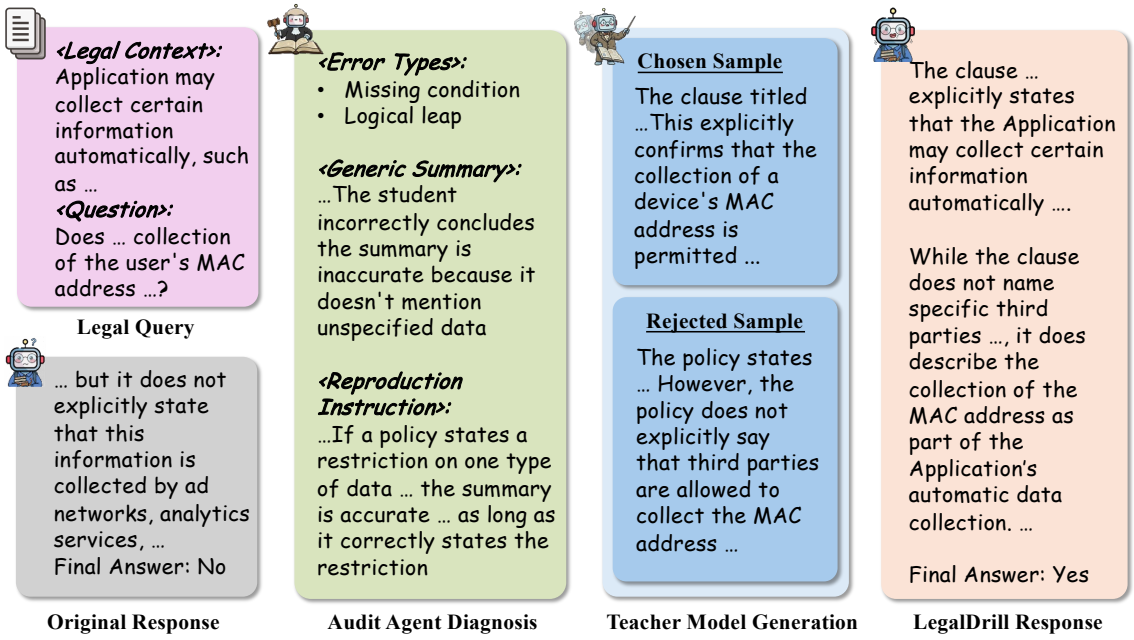

Figure 2: Illustrative examples of legal reasoning refinement of LegalDrill. After the optimization, the student model is asked to generate a response under the same legal query, which is marked as LegalDrill response

Direct preference optimization. Following the SFT warm-up (or for subsequent iterations $t \geq 0$), we refine the model by minimizing the preference objective $\mathcal{L}_{\text{DPO}}$. Specifically, we update the policy from π_{θ_t} to $\pi_{\theta_{t+1}}$ using the filtered dataset $\mathcal{D}^t_{\text{train}}$. During the iterative process, we set the reference model π_{ref} to be the current policy π_{θ_t}.

4 Experiment

4.1 Experimental Setup.

Backbone models, teacher models, and agent models We train our student models with two SLM backbones: Qwen3-0.6B and Qwen3-1.7B. For both the teacher model and the audit agent, we consistently use the same stronger model, choosing between the open-sourced Qwen3-30B-A3B-Instruct and the closed-sourced GPT-4o.

Baselines. We evaluate performance using zero-shot CoT prompting across two categories: (1) **General LLMs**, including the teacher models (GPT-4o, Qwen3-30B-Instruct) and the student base models (Qwen3-0.6B, Qwen3-1.7B); and (2) **Legal-Specific LLMs**, namely Law-LLM-13B (Cheng et al., 2023), DeepSeek ESFT-16B (Wang et al., 2024), and DiscLaw-13B (Yue et al., 2023).

Datasets We evaluate our method on six datasets: four public legal-reasoning benchmarks from LegalBench (Guha et al., 2023): *Consumer QA*, *Contracts QA*, *Sara Entailment*, and *Privacy Policy Entailment*, as well as two proprietary datasets from real-world financial-industry legal document review scenarios, namely *Real-World power of attorney (POA)* and *Real-World Trust*. All datasets consist of document-grounded, binary yes/no question–answer pairs. Detailed descriptions of both the public benchmarks and the proprietary datasets are provided in appendix B.

Metrics. We report accuracy and F1 to evaluate each model's judgment performance on the binary QA tasks. In addition, we introduce a "judge accuracy" metric to assess the quality of the generated reasoning, using an LLM as a Judge. Implementation details are provided in appendix A.1 and D.

4.2 Quantitative Results

Table 1 reports the main results on four public LegalBench benchmarks as well as two in-domain document QA benchmarks from Real-World. We compare against both (i) zero-shot prompting of the teacher LLMs and (ii) undistilled student backbones. Overall, LegalDrill consistently improves both accuracy and F1 over the Qwen3-0.6B/1.7B baselines across most datasets, indicating that error-driven preference data effectively transfers the teachers' decision boundary and reasoning behaviors to substantially smaller models.

On LegalBench, LegalDrill notably boosts the student models on contract QA tasks (Consumer

Table 1: Overall performances on public datasets.

	Cos. QA			Con. QA			Sara Ent.			Priv. Ent.		
	Acc	*F1*	*Judge*	*Acc*	*F1*	*Judge*	*Acc*	*F1*	*Judge*	*Acc*	*F1*	*Judge*
Law Model Zero Shot												
DeepSeek ESFT-16B	0.81	0.80	0.74	0.88	0.87	0.79	0.46	0.44	0.27	0.75	0.64	0.73
Law-LLM-13B	0.49	0.38	0.36	0.88	0.56	0.77	0.51	0.38	0.18	0.64	0.38	0.29
DiscLaw-13B	0.24	0.18	0.11	0.42	0.33	0.38	0.10	0.07	0.05	0.01	0.01	0.01
General SLMs and LLMs Zero Shot												
Qwen3-0.6B	0.69	0.34	0.66	0.83	0.83	0.75	0.59	0.27	0.18	0.30	0.29	0.24
Qwen3-1.7B	0.79	0.58	0.70	0.87	0.85	0.81	0.66	0.38	0.37	0.47	0.45	0.30
Qwen3-30B-A3B-Instruct	0.98	0.97	0.92	0.96	0.95	0.93	0.86	0.43	0.51	0.83	0.75	0.65
GPT-4o	0.98	0.98	0.81	0.92	0.92	0.92	0.83	0.81	0.62	0.67	0.30	0.60
Ours: Distill from Qwen3-30B-A3B-Instruct												
LegalDrill-0.6B	0.84	0.83	0.77	0.91	0.88	0.85	0.74	0.45	0.44	0.81	0.80	0.58
LegalDrill-1.7B	0.96	0.96	0.89	0.93	0.92	0.83	0.73	0.39	0.42	0.85	0.80	0.59
Ours: Distill from GPT-4o												
LegalDrill-0.6B	0.86	0.84	0.75	0.95	0.84	0.91	0.75	0.42	0.41	0.59	0.32	0.52
LegalDrill-1.7B	0.94	0.94	0.88	0.97	0.94	0.94	0.75	0.46	0.43	0.60	0.31	0.52

Table 2: Performances on Real-World datasets.

	Real-World POA			Real-World Trust		
	Acc	*F1*	*Judge*	*Acc*	*F1*	*Judge*
General SLMs and LLMs						
Qwen3-0.6B	0.76	0.51	0.27	0.74	0.49	0.44
Qwen3-1.7B	0.78	0.53	0.50	0.79	0.54	0.51
GPT-4o	0.91	0.90	0.67	0.89	0.60	0.73
Ours: Distill from GPT-4o						
LegalDrill 0.6B	0.87	0.58	0.72	0.86	0.58	0.69
LegalDrill 1.7B	0.92	0.91	0.73	0.90	0.60	0.70

QA, Contracts QA) and improves entailment tasks in both accuracy and F1, suggesting more reliable binary decisions under class imbalance. Importantly, distillation narrows the teacher–student gap: with Qwen3-30B-A3B-Instruct as teacher, the 1.7B student reaches near-teacher performance on multiple benchmarks. On Real-World datasets, distilling from GPT-4o yields student performance on par with the teacher, enabling cost-effective deployment for automated compliance workflows in the industry. Beyond final yes/no correctness, the "judge accuracy" further improves over the Qwen3 backbones on most datasets, indicating fewer judge-detected errors and more reliable legal rationales.

We observe a consistent scaling trend: across zero-shot and distilled settings, the 1.7B backbone outperforms the 0.6B backbone, with distillation amplifying this advantage. In several cases (e.g., contract QA), the 1.7B LegalDrill model becomes comparable to the Qwen3-30B-A3B-Instruct teacher while being substantially lighter. This highligts that our method yields even greater gains on small models with moderately larger size, enabling them to approach the performance of large models.

4.3 Qualitative Results

We further present qualitative results to demonstrate how LegalDrill refines the legal reasoning and corrects the student's weak spots. After optimization, the student model is asked to generate a response for the same legal query. Fig. 2 illustrates the effect of LegalDrill on a randomly selected example. The full, detailed responses are included in App. A.4. Given the legal query, the student responds with reasoning and a conclusion that appear logical on the surface. However, the response inherently contains an error because the final yes/no conclusion is incorrect. Next, the Audit Agent analyzes potential errors in the reasoning and categorizes them into two error types. Furthermore, the Audit Agent generates instructions on how to reproduce the error under *any* legal context by deliberately avoiding context-specific details, only mentioning terms like "policy" or "one type of data". Then, based on the analysis, the teacher model is instructed to generate a targeted pair, which reproduces and corrects the error. In the example, the teacher model pinpoints the key step in the reasoning and flips the logic between the chosen and rejected responses. After optimization, to demonstrate the effect of LegalDrill, the student model is asked to generate a response for the same

legal context. The response from the student model demonstrates that the SLM clearly resolves the previous error and does not simply repeat the chosen response generated by the teacher model.

4.4 Ablation Study

We further validate the effectiveness of the DPO technique via an ablation study. As shown in Figure 5, across all settings the model trained with DPO consistently achieves higher accuracy than the SFT-only counterpart, which indicates that leveraging both chosen and rejected responses provides strong contrastive signals. Notably, the DPO almost universally improves the student model's reasoning robustness.

We refer more experiment results in App. A with the effects of number of iterations, K, etc.

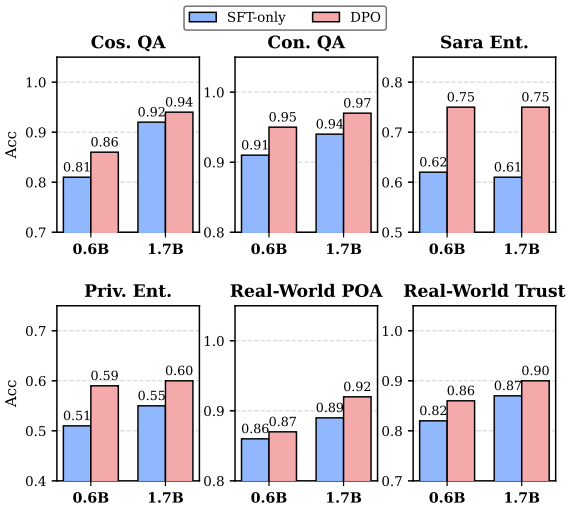

Figure 3: The ablation study on DPO.

5 Related Work

LLMs for Legal Reasoning Recent works (Cui et al., 2023; Wang et al., 2024; Zhou et al., 2024; Dai et al., 2025; Shi et al., 2025; Cai et al., 2025; Li et al., 2025a) have explored adapting LLMs to the legal domain. Most recent works adopt supervised fine-tuning and reinforcement learning strategies. Unlike rewards in general domains (Zheng et al., 2025; Gunjal et al., 2025; Liu et al., 2025b; Xu et al., 2026), the legal domain requires domain-specific rewards to enable models to accurately handle legal queries. To name a few, Shi et al. incorporate progressiveness and potential, which describe the step-wise reward for intermediate legal reasoning steps. Cai et al. design a legal validity reward to encourage legal accuracy. Dai et al. formulate the legal reward from an information theory perspective. However, even with Parameter-Efficient Fine-Tuning (PEFT) methods (Hu et al., 2022; Zhang et al., 2023; Wu et al., 2024; Liu et al., 2025a) and quantization or pruning strategies, LLMs still consume a considerable amount of resources to train and deploy.

Knowledge Distillation and Reasoning Distillation Knowledge distillation (Hinton et al., 2015; Phuong and Lampert, 2019; Gou et al., 2021; Tan et al., 2023) focuses on transferring knowledge from a larger teacher model to a smaller student model by aligning their output logits or intermediate representation features. With the advancement of language models, the focus of distillation has shifted towards distilling complex, multi-step reasoning capabilities from LLMs into SLMs (Shridhar et al., 2023; Kang et al., 2023; Feng et al., 2024; Zhao et al., 2025; Yang et al., 2025a; Zhang et al., 2025; Kim et al., 2025). However, standard methods struggle with a fundamental behavioral mismatch: constrained by their parameter scale, SLMs cannot effectively internalize the verbose, self-corrective reasoning chains typical of strong LLMs. While recent studies explore trajectory refinement, methods such as reasoning compression (Zhao et al., 2025; Zhang et al., 2025) primarily prune long CoT traces to bridge the reasoning gap between SLMs and LLMs. In contrast, LegalDrill does not merely compress reasoning chains; it generates trajectories specifically targeting the SLM's unique blind spots. Furthermore, unlike frameworks such as SMART (Kim et al., 2025) that rely on external LLMs during inference, LegalDrill uses curated trajectories during training to resolve the model's logical blind spots, enabling the SLM to reason correctly without external inference-time intervention.

6 Conclusion

While LLMs demonstrate strength in legal reasoning, real-world legal deployment often requires privacy-preserving and cost-efficient solutions. SLMs are promising for real-world deployment to resource constrained devices due to their efficiency and low operational cost, therefore providing privacy guaranties for the legal domain. We proposed LegalDrill, a diagnosis-driven synthesis framework that translates the implicit knowledge of strong LLMs into concise, corrective reasoning traces explicitly tailored to the student model's capacity. Experiments on LegalBench and real-world datasets show consistent gains over baseline SLMs.

Limitations

While LegalDrill effectively enhances SLMs, the framework employs the DPO algorithm. Inside DPO, there are some hyperparameters to tune, e.g., learning rate, weight decay, epochs, etc. This will require some additional tuning. However, we find that typically, $1-3$ epochs are enough with a learning rate of 1×10^{-4}. We include the range of each parameter in the App.

Acknowledgment

This work is supported in part by the US National Science Foundation under grant NSF IIS-2141037. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Hua Cai, Shuang Zhao, Liang Zhang, Xuli Shen, Qing Xu, Weilin Shen, Zihao Wen, and Tianke Ban. 2025. Unilaw-r1: A large language model for legal reasoning with reinforcement learning and iterative inference. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 18128–18142.

Daixuan Cheng, Shaohan Huang, and Furu Wei. 2023. Adapting large language models to domains via reading comprehension. *arXiv preprint arXiv:2309.09530*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Jiaxi Cui, Munan Ning, Zongjian Li, Bohua Chen, Yang Yan, Hao Li, Bin Ling, Yonghong Tian, and Li Yuan. 2023. Chatlaw: A multi-agent collaborative legal assistant with knowledge graph enhanced mixture-of-experts large language model. *arXiv preprint arXiv:2306.16092*.

Xin Dai, Buqiang Xu, Zhenghao Liu, Yukun Yan, Huiyuan Xie, Xiaoyuan Yi, Shuo Wang, and Ge Yu. 2025. Legal Δ: Enhancing legal reasoning in llms via reinforcement learning with chain-of-thought guided information gain. *arXiv preprint arXiv:2508.12281*.

Yihe Deng and Paul Mineiro. 2024. Flow-dpo: Improving llm mathematical reasoning through online multi-agent learning. *arXiv preprint arXiv:2410.22304*.

Karel D'Oosterlinck, Winnie Xu, Chris Develder, Thomas Demeester, Amanpreet Singh, Christopher Potts, Douwe Kiela, and Shikib Mehri. 2025. Anchored preference optimization and contrastive revisions: Addressing underspecification in alignment. *Transactions of the Association for Computational Linguistics*, 13:442–460.

Yu Fan, Jingwei Ni, Jakob Merane, Yang Tian, Yoan Hermstrüwer, Yinya Huang, Mubashara Akhtar, Etienne Salimbeni, Florian Geering, Oliver Dreyer, and 1 others. 2025. Lexam: Benchmarking legal reasoning on 340 law exams. *arXiv preprint arXiv:2505.12864*.

Zhiwei Fei, Xiaoyu Shen, Dawei Zhu, Fengzhe Zhou, Zhuo Han, Alan Huang, Songyang Zhang, Kai Chen, Zhixin Yin, Zongwen Shen, and 1 others. 2024. Lawbench: Benchmarking legal knowledge of large language models. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 7933–7962.

Tao Feng, Yicheng Li, Li Chenglin, Hao Chen, Fei Yu, and Yin Zhang. 2024. Teaching small language models reasoning through counterfactual distillation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5831–5842.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *The Innovation*.

Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters,

Brandon Waldon, Daniel Rockmore, Diego Zambrano, and 1 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in neural information processing systems*, 36:44123–44279.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, and 1 others. 2024. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3.

Antreas Ioannou, Andreas Shiamishis, Nora Hollenstein, and Nezihe Merve Gürel. 2025. Evaluating the limits of large language models in multilingual legal reasoning. *arXiv preprint arXiv:2509.22472*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Seungjae Jung, Gunsoo Han, Daniel Wontae Nam, and Kyoung-Woon On. 2025. Binary classifier optimization for large language model alignment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1858–1872.

Minki Kang, Seanie Lee, Jinheon Baek, Kenji Kawaguchi, and Sung Ju Hwang. 2023. Knowledge-augmented reasoning distillation for small language models in knowledge-intensive tasks. *Advances in Neural Information Processing Systems*, 36:48573–48602.

Yujin Kim, Euiin Yi, Minu Kim, Se-Young Yun, and Taehyeon Kim. 2025. Guiding reasoning in small language models with llm assistance. *arXiv preprint arXiv:2504.09923*.

Edward H Levi. 2022. *An introduction to legal reasoning*. University of Chicago Press.

Ang Li, Yiquan Wu, Ming Cai, Adam Jatowt, Xiang Zhou, Weiming Lu, Changlong Sun, Fei Wu, and Kun Kuang. 2025a. Legal judgment prediction based on knowledge-enhanced multi-task and multi-label text classification. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6957–6970.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. 2025b. Small models struggle to learn from strong reasoners. *arXiv preprint arXiv:2502.12143*.

Tianci Liu, Haoxiang Jiang, Tianze Wang, Ran Xu, Yue Yu, Linjun Zhang, Tuo Zhao, and Haoyu Wang. 2025a. Roserag: Robust retrieval-augmented generation with small-scale llms via margin-aware preference optimization. *arXiv preprint arXiv:2502.10993*.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. 2025b. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment. *arXiv preprint arXiv:2510.07743*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025c. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*.

Igor Melnyk, Youssef Mroueh, Brian Belgodere, Mattia Rigotti, Apoorva Nitsure, Mikhail Yurochkin, Kristjan Greenewald, Jiri Navratil, and Jarret Ross. 2024. Distributional preference alignment of llms via optimal transport. *Advances in Neural Information Processing Systems*, 37:104412–104442.

Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. 2024. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *Advances in Neural Information Processing Systems*, 37:116617–116637.

Mary Phuong and Christoph Lampert. 2019. Towards understanding knowledge distillation. In *International conference on machine learning*, pages 5142–5151. PMLR.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Leonardo Ranaldi and Andre Freitas. 2024. Aligning large and small language models via chain-of-thought reasoning. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1812–1827, St. Julian's, Malta. Association for Computational Linguistics.

Weijie Shi, Han Zhu, Jiaming Ji, Mengze Li, Jipeng Zhang, Ruiyuan Zhang, Jia Zhu, Jiajie Xu, Sirui Han, and Yike Guo. 2025. Legalreasoner: Step-wised verification-correction for legal judgment reasoning. *arXiv preprint arXiv:2506.07443*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073.

Marco Siino, Mariana Falco, Daniele Croce, and Paolo Rosso. 2025. Exploring llms applications in law: A literature review on current legal nlp approaches. *IEEE Access*.

Shicheng Tan, Weng Lam Tam, Yuanchun Wang, Wenwen Gong, Shu Zhao, Peng Zhang, and Jie Tang. 2023. Gkd: A general knowledge distillation framework for large-scale pre-trained language model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 134–148.

Songjun Tu, Jiahao Lin, Xiangyu Tian, Qichao Zhang, Linjing Li, Yuqian Fu, Nan Xu, Wei He, Xiangyuan Lan, Dongmei Jiang, and 1 others. 2025. Enhancing llm reasoning with iterative dpo: A comprehensive empirical investigation. *arXiv preprint arXiv:2503.12854*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Li Wang, Changhao Zhang, Zengqi Xiu, Kai Lu, Xin Yu, Kui Zhang, and Wenjun Wu. 2026. Decoupling understanding from reasoning via problem space mapping for small-scale model reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33575–33583.

Zihan Wang, Deli Chen, Damai Dai, Runxin Xu, Zhuoshu Li, and Yu Wu. 2024. Let the expert stick to his last: Expert-specialized fine-tuning for sparse architectural large language models. *arXiv preprint arXiv:2407.01906*.

Yiquan Wu, Siying Zhou, Yifei Liu, Weiming Lu, Xiaozhong Liu, Yating Zhang, Changlong Sun, Fei Wu, and Kun Kuang. 2023. Precedent-enhanced legal judgment prediction with llm and domain-model collaboration. *arXiv preprint arXiv:2310.09241*.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. 2024. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962.

Huimin Xu, Xinnian Mao, Feng-Lin Li, Xiaobao Wu, Wang Chen, Wei Zhang, and Luu Anh Tuan. 2025. Full-step-dpo: Self-supervised preference optimization with step-wise rewards for mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24343–24356.

Ran Xu, Tianci Liu, Zihan Dong, Tony Yu, Ilgee Hong, Carl Yang, Linjun Zhang, Tao Zhao, and Haoyu Wang. 2026. Alternating reinforcement learning for rubric-based reward modeling in non-verifiable llm post-training. *arXiv preprint arXiv:2602.01511*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Shu Yang, Junchao Wu, Xin Chen, Yunze Xiao, Xinyi Yang, Derek F Wong, and Di Wang. 2025b. Understanding aha moments: from external observations to internal mechanisms. *arXiv preprint arXiv:2504.02956*.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*.

Wenhan Yu, Xinbo Lin, Lanxin Ni, Jinhua Cheng, and Lei Sha. 2025. Benchmarking multi-step legal reasoning and analyzing chain-of-thought effects in large language models. *arXiv preprint arXiv:2511.07979*.

Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Xuanjing Huang, and 1 others. 2023. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *arXiv preprint arXiv:2309.11325*.

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. 2025. Lightthinker: Thinking step-by-step compression. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13318–13339.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng,

Weizhu Chen, and Tuo Zhao. 2023. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*.

Shangziqi Zhao, Jiahao Yuan, Jinyang Wu, Zhenglin Wang, Guisong Yang, and Usman Naseem. 2025. Can pruning improve reasoning? revisiting long-cot compression with capability in mind for better reasoning. *arXiv preprint arXiv:2505.14582*.

Congming Zheng, Jiachen Zhu, Zhuoying Ou, Yuxiang Chen, Kangning Zhang, Rong Shan, Zeyu Zheng, Mengyue Yang, Jianghao Lin, Yong Yu, and 1 others. 2025. A survey of process reward models: From outcome signals to process supervisions for large language models. *arXiv preprint arXiv:2510.08049*.

Zhi Zhou, Jiang-Xin Shi, Peng-Xiao Song, Xiao-Wen Yang, Yi-Xuan Jin, Lan-Zhe Guo, and Yu-Feng Li. 2024. Lawgpt: A chinese legal knowledge-enhanced large language model. *arXiv preprint arXiv:2406.04614*.

A More Experiment and Experimental details

A.1 Judge Accuracy

Judge accuracy is computed by employing another LLM as a judge: given each model's generated legal reasoning, the judge checks whether the reasoning contains any potential error; if so, the entire reasoning is marked as incorrect. We then report the resulting accuracy as a proxy for reasoning quality. Specifically, we use the Qwen3-8B as the judge.

A.2 The impact of number of iterations

We study how the number of self-improvement iterations affects downstream performance. In general, we find that only a small number of iterations is needed: most of the gains are achieved within the first one to two iterations, after which additional iterations yield only marginal improvements. To validate this trend, we run our framework for four iterations using Qwen3-30B-A3B-Instruct as the teacher model on two open benchmarks.

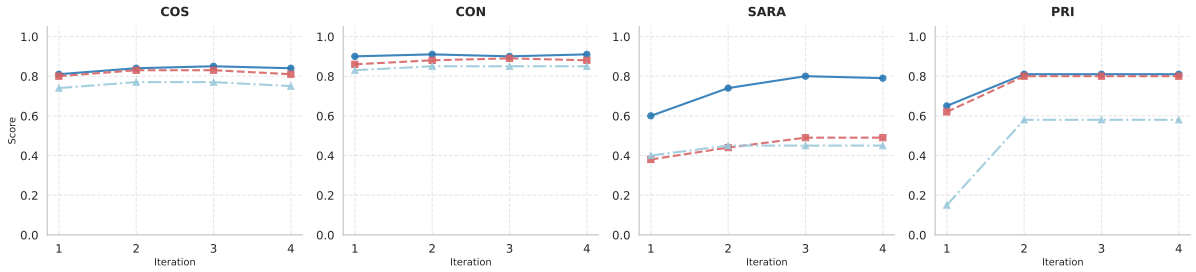

Figure 4: The ablation study on the number of iterations.

From the above results, we can see that the largest performance gain, typically the Judge score, happens around the second iteration. This actually confirms that the reasoning ability can be greatly improved after two iterations but will not keep improving.

A.3 The impact over K preference pairs

We investigate how the number of preference pairs K used to construct the final training set affects performance. Overall, we observe that the distilled training set does not need to be large to be effective: in practice, a final training dataset of roughly 1,000–2,000 samples typically achieves the best results, while further increasing the data size brings only marginal gains. As a consequence, the appropriate K depends on the size of the original legal dataset (and thus how many preference pairs can be harvested). Using Consumer QA (*Cos. QA*) as an example, we sweep $K \in \{2, 4, 6, 8, 10, 12, 16\}$; when $K = 16$, the resulting training set contains ∼3,000+ samples, and performance largely saturates within this range.

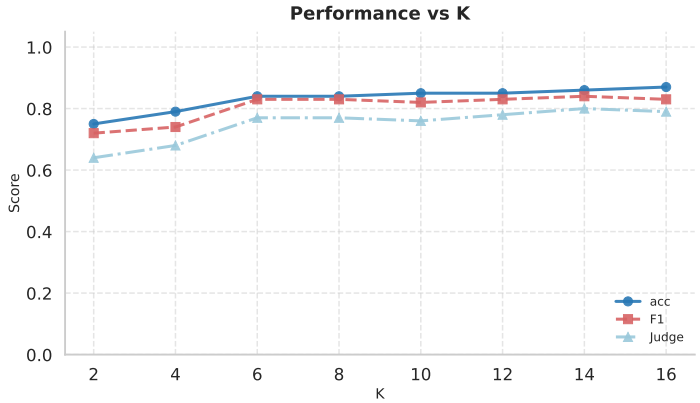

Figure 5: The ablation study on the number of K.

A.4 LegalDrill Generation Demonstration

Response Generated by Qwen3 1.7B

Step 1: Identify the relevant clause in the contract. The contract mentions the "Automatically Collected Information" section, which states: "In addition, the Application may collect certain information automatically, such as: the type of mobile device you use, mobile device identifiers, such as the resettable advertising identifier assigned to your device, the IP address of your mobile device, the location of your device, your mobile operating system, Wi-Fi mac address in your area, the type of mobile Internet browsers you use, and information about the way you use the Application. See 'Automatic Data Collection and Advertising' section for examples." This section clearly outlines the types of information that may be collected automatically, including the Wi-Fi mac address in your area. Step 2: Determine if the MAC address is collected by ad networks, analytics services, or other third parties. The contract explicitly mentions that the Wi-Fi mac address in your area is included in the list of automatically collected information. It does not specify that this information is collected by ad networks, analytics services, or other third parties. While the contract does not explicitly state that the MAC address is collected by third parties, it does not deny that it may be collected by such entities. The clause is neutral in this regard and does not make a claim about the source of the data. Step 3: Evaluate whether the description accurately summarizes the clause. The question asks whether the policy accurately describes the clause. The clause does mention that the Wi-Fi mac address in your area is collected, but it does not explicitly state that this information is collected by ad networks, analytics services, or other third parties. Therefore, the description does not fully capture the intent of the clause, as it lacks a definitive statement about the source of the information. Final Answer: No

The above is an example of the SLM's response. Note that while the step-by-step reasoning appears logical on the surface, it inherently contains an error because the final yes/no conclusion is wrong. In the next step, we will feed this response to the Audit Agent to conduct the error analysis.

Audit Agent Instruction to Teacher Model

Status:
`INCORRECT_ANSWER` Error Types:

- Missing condition
- Logical leap

Generic Summary:
The student correctly identifies that the clause permits collection of unspecified location data but fails to recognize that the policy description explicitly excludes only precise location data, thereby implying it may still collect unspecified data. The student incorrectly concludes the summary is inaccurate because it doesn't mention unspecified data, when in fact the policy's statement is accurate as written—because it only claims a restriction on precise data, which matches the clause.

Reproduction Instruction:
When evaluating a policy summary, ensure that the summary's claim does not contradict the contract's explicit language. If a policy states a restriction on one type of data (e.g., precise location), and the contract allows another (e.g., unspecified location), the summary is accurate even if it does not mention the permitted data, as long as it correctly states the restriction.

The above is an example of the Audit Agent's response. Note that only the Reproduction Instruction will later be used to generate a chosen and rejected sample. Notably, by properly instructing the agent through prompting, the agent is able to transform the error analysis into a generalizable reproduction instruction. This crucial step allows us to, firstly construct an error bank, and, secondly, to reproduce the SLM's behavior easily.

Then, we demonstrate the generated chosen and rejected sample from the Teacher model under a completely different legal context from the error bank.

Chosen Sample Generated by Teacher

The clause titled “Automatically Collected Information” states that the Application may collect certain information automatically, including “Wi-Fi mac address in your area.” This explicitly confirms that the collection of a device’s MAC address is permitted as part of the automatic data collection process. The clause further clarifies that such data may be collected by third parties, including ad networks and analytics services, through the “Automatic Data Collection and Advertising” section, which is referenced in the text.
While the policy does not specify the exact context or method of collection by third parties, it does not restrict or prohibit the collection of MAC addresses by these entities. The absence of any explicit prohibition, combined with the clear mention of Wi-Fi MAC address as an automatically collected data point, supports the conclusion that such collection is permitted. The policy’s description that “the user’s MAC address is collected by ad networks, analytics services, or other third parties” aligns directly with this provision, as it reflects the actual scope of collection described in the clause.
There is no logical inconsistency or factual misrepresentation in the description, and no requirement to mention additional data types (such as unspecified location data) for the summary to be accurate. The clause only restricts precise location data, which is not relevant to the claim about MAC address collection. Therefore, the policy description is accurate and consistent with the terms of the clause.

Final Answer: Yes

Rejected Sample Generated by Teacher

The policy states that the Application collects certain information automatically, including the “Wi-Fi MAC address in your area.” This means that third parties like ad networks and analytics services can access the MAC address when you are connected to Wi-Fi. However, the policy does not explicitly say that these third parties are allowed to collect the MAC address—only that the Application collects it.
Since the clause does not clearly state that third parties are permitted to access or use this data, it’s a logical leap to assume that the policy allows such collection by external entities. Moreover, the policy only mentions that the Application collects the MAC address, not that it shares it with ad networks or analytics services. Therefore, the summary overreaches by implying that third parties are involved in collecting the MAC address, which is not confirmed by the clause. Because the policy does not specify that third parties collect the MAC address, the description is misleading and inaccurate.

Final Answer: No

From above we can see that, without any legal expertise and any prior knowledge, it would be hard to clearly see which one is the chosen response. This highlights the complexity of legal reasoning. The rejected sample constructs a cohesive and syntactically fluent argument that appears entirely plausible on the surface. By accurately citing specific clauses (e.g., "Wi-Fi MAC address") while subtly distorting the interpretative scope regarding third-party permissions, it creates a deceptive facade of validity. The error here is not a simple factual hallucination, but a sophisticated "logical leap"—a failure to properly deduce implied permissions from the text’s structure. Unlike obvious errors, this type of flawed reasoning adheres to the form of legal analysis while violating its substance. Consequently, distinguishing the correct interpretation requires not just textual alignment, but a rigorous verification of logical consistency between the claim and the contract terms.

The example below presents a response generated by LegalDrill 1.7B, which was distilled from the teacher model Qwen3-30B-A3B-Instruct. Despite its compact parameter size, the model successfully replicates the rigorous reasoning structure of the teacher, accurately identifying the trigger (material breach) and procedural requirements (cure period) within the clause. It moves beyond simple extraction to synthesize a coherent legal argument regarding the cessation of obligations. This demonstrates that our distillation framework effectively transfers not just linguistic fluency, but the underlying logic required for contract analysis.

Response Generated by LegalDrill 1.7B (distilled from Qwen3-30B-A3B-Instruct)

The clause titled "Automatically Collected Information" explicitly states that the Application may collect certain information automatically, including "Wi-Fi mac address in your area." This language confirms that the collection of the user's MAC address is a permitted and described activity under the policy. The question asks whether the policy describes the collection of the user's MAC address by ad networks, analytics services, or other third parties. While the clause does not name specific third parties such as ad networks or analytics services, it does describe the collection of the MAC address as part of the Application's automatic data collection. The phrase "information about the way you use the Application" further supports that data collection is ongoing and includes technical identifiers like the MAC address. The policy does not restrict or prohibit such collection, nor does it limit it to first-party use only. Furthermore, the clause refers to "Automatic Data Collection and Advertising" as a reference point for examples, implying that such collection includes third-party data sharing. Given that the MAC address is listed as one of the types of data collected, and that automatic collection often involves third-party services for analytics or advertising, the inclusion of "Wi-Fi mac address" in the list of collected data logically extends to its use by ad networks or analytics services, which are commonly engaged in such collection. Therefore, the policy does describe the collection of the user's MAC address by third parties such as ad networks and analytics services, as the collection is both permitted and implicitly supported by the broader context of automatic data collection and advertising practices. Final Answer: Yes

B Datasets

B.1 Public Benchmarks

In the experiments, we use four publicly available datasets from LegalBench (Guha et al., 2023):

- **Consumer QA** is a benchmark that assesses whether models can interpret consumer contracts by answering yes/no questions about the legal implications of terms of service from popular online platforms. The dataset contains 400 annotated question–answer pairs.

- **Contracts QA** is a dataset consisting of contract clauses and a yes/no question, typically regarding whether the clause belongs to a certain provision category. The dataset contains 88 samples.

- **Sara Entailment** dataset evaluates a model's ability to perform statutory reasoning by determining whether a legal rule applies to a fact pattern, using simplified tax-law sections paired with an entailment question that require a binary yes/no answer. The dataset contains 276 examples.

- **Privacy Policy Entailment** dataset evaluates whether a model can identify privacy practices by determining if a given practice description entails a policy clause, formulated as a binary classification task. The dataset contains 4,343 examples.

All benchmarks are formatted as query–answer pairs with binary verdicts (yes/no or equivalent) to match our experimental setting.

B.2 Real-World Datasets

We include two proprietary real-world datasets collected from authentic business scenarios within the financial industry, to assess practical utility under realistic document lengths and compliance-oriented legal document review settings.

- **Real-World POA** dataset is a document QA benchmark for reviewing power of attorney (POA) documents, where legal experts answer compliance questions with yes/no verdicts grounded in the underlying text. The dataset uses a fixed set of 13 questions and contains 780 examples in total. Example questions include "Does the document allow the agent to perform banking transactions?", "Does the document allow the agent to trade securities?", etc. Because POA documents tend to be long, we retrieve the top-5 most similar snippets as the document context.
- **Real-World Trust** dataset is a document QA benchmark for reviewing trust-related documents, consisting of 12 fixed compliance questions with yes/no labels. Example questions include "Does the Trust authorize a trustee to charge fees?", "Is there more than one individuals named as trustee at the time of this trust creation?", etc. This dataset includes 468 examples. Similar to the POA setting, we retrieve the top-5 most relevant snippets as context due to the length of the source documents.

C Implementation Details

Our experimental framework is built upon the Hugging Face `transformers`, `TRL`, and `vLLM` libraries, with all models trained on NVIDIA A6000 GPUs. Due to the manageable scale of our 0.6B and 1.7B models, we performed full-parameter fine-tuning for both SFT and DPO stages. To address the risks of overfitting and model collapse associated with full-parameter updates on smaller architectures, we applied rigorous regularization. Specifically for DPO, we selected weight decay values in the range of $[1 \times 10^{-5}, 1 \times 10^{-3}]$ and learning rates between $[1 \times 10^{-6}, 1 \times 10^{-4}]$. Furthermore, we restricted DPO training to 1–3 epochs to maintain training stability and prevent reward hacking.

D Prompts

D.1 Audit Agent Prompt

Agent System Prompt (Core Logic)

Role:
You are a rigorous AI teaching assistant specializing in legal contract analysis.

Core Objectives:
1. **Diagnose:** Evaluate the correctness and reasoning of the student's answer against the ground truth.
2. **Instruction Generation:** If an error exists, provide a specific, abstract instruction on how to *reproduce* this logic error in a completely different legal context.

Internal Evaluation Process:
1. Verify if the student's final answer matches the ground truth.
2. Check reasoning for logical soundness (e.g., missed conditions, hallucinations).
3. Classify any flaws using the provided *Error Taxonomy*.
4. Draft a `reproduction_instruction` for the Teacher AI.

Reproduction Instruction Guidelines:
When writing the instruction, you must be **Context-Agnostic** and **Actionable**. Do NOT mention specific entities or clauses.
- *Example (Good):* "Identify a condition in the text that limits a right, and generate a response that treats the right as absolute by deliberately ignoring that condition."
- *Example (Bad):* "Ignore the 5-day notice period in Clause 4."

Output Format:
Respond with a strict JSON object containing the evaluation status, error types, generic summary, and the reproduction instruction.

The user prompt provides the agent with the specific context required for each evaluation instance. It is structured as a template that sequentially inputs the Contract text, the associated Question, and the Ground Truth (correct answer) to establish the evaluation standard. These are followed by the Student Answer that needs to be assessed. The prompt concludes with a directive triggering the agent to generate the output in the strict JSON format defined in the system prompt.

D.2 Teacher Model Prompt

Teacher Model System Prompt for Generating Rejected Sample

Role & Objective:
You are an AI tutor acting as a student to create a flawed educational example. Your task is to generate an **incorrect** student answer.

Core Mechanism:
1. **Input Processing:** You will receive the *Correct Answer* and specific *Error Summaries*.
2. **Flaw Embodiment:** You must generate a step-by-step reasoning process that naturally embodies the specified error (e.g., ignoring a condition) without explicitly stating "I am making an error."
3. **Target Outcome:** Your reasoning must plausibly lead to the **opposite** of the Ground Truth.

Constraints:
Do NOT mention the error summary or the ground truth in your output. Act entirely within the persona of the confused student.

The user prompt constructs the simulation context by providing the Contract, the Question, and the

Correct Answer (Ground Truth). Crucially, it injects the specific error parameters derived from the evaluation phase: the target Error Types, a Generic Error Description, and the specific Reproduction Instruction. These inputs serve as the blueprint for the model to construct the flawed reasoning path.

Teacher Model User Prompt for Generating Rejected Sample (Core Constraint)

Generation Rules:

1. **Immediate Reasoning:** Start the response immediately with the flawed step-by-step reasoning. Do NOT include any preamble or repetition of instructions.
2. **Twist the Logic:** Plausibly embody the flaws listed in `Error Types`. Follow the `Reproduction Instruction` to manipulate the logic (e.g., if instructed to ignore a condition, simply fail to mention it).
3. **Opposite Conclusion:** The reasoning must naturally lead to a final answer that is the **opposite** of the Correct Answer.
4. **Formatting:** Strict adherence to the provided output structure is required.
5. **Final Answer:** Conclude strictly with "Final Answer: Yes" or "Final Answer: No".

When generating the chosen sample, the prompt is similar except highlighting that the model needs to generate correct reaosning with an addtional input, which the rejected sample that the teacher model just generated.

D.3 Student Model Prompt

Student Model Prompt

System Role:
You are an AI assistant specializing in legal contract analysis. Analyze contracts carefully, reference specific clauses, and provide step-by-step reasoning.

Input Context:
- **Contract:** `{contract}`
- **Question:** `{question}`

Task Instructions:

1. **Reference:** Explicitly identify and cite each clause or term relevant to the question.
2. **Chain-of-Thought:** Provide a detailed, step-by-step explanation. For each step:
   - Quote/Summarize the specific contract part (e.g., "Clause 2.1 states...").
   - Explain its logical impact on the answer.
3. **Final Answer:** Conclude clearly on a new line with "Final Answer: Yes" or "Final Answer: No".

D.4 Judge Model Prompt

Judge System Prompt

You are a strict legal reasoning judge. Field definitions: question = the claim/question to evaluate; contract = the governing legal text/context only; ground_truth = the gold final answer for the question under the contract. Given question, contract, ground_truth, and a model response, decide whether the model response contains ANY legal error (factual, logical, interpretive, or conclusion mismatch). Hard rule: if the model's final answer does not match ground_truth, the judgment must be incorrect. Output exactly one word: correct or incorrect.

Judge User Prompt Template

Question:
{question}
Legal Context:
{contract}
Ground Truth Answer:
{ground_truth}
Model Response To Judge:
{model_generation}
Instruction:
- Meaning reminder:
question = what needs to be judged; contract = legal basis; ground_truth = gold final answer
- First extract the model's final answer from the model response
- If the extracted final answer does NOT match ground_truth, output: incorrect (mandatory)
- If there is any legal error anywhere in the model response, output: incorrect
- Otherwise output: correct
- Output one word only.