Towards Better Legal Reasoning LLMs: Signal Balancing and Reward Scheduling

Anonymous ACL submission

Abstract

In the legal domain, LLMs can improve reliability and interpretability by generating explicit reasoning traces. However, training LLMs to produce high-quality reasoning traces remains challenging. Existing supervised fine-tuning methods struggle when reasoning traces are significantly longer than final answers, as the learning signal for the answer becomes diluted. Meanwhile, reinforcement learning methods such as Group Relative Policy Optimization also have several limitations, including costly reward design, performance plateaus, and reward hacking. To address these challenges, we propose a two-stage training framework. In Stage I, JurisCoT-SFT employs a length-normalized dual objective to balance learning signals between reasoning traces and final answers. In Stage II, Lifecycle-Aware Backtrackable Policy Optimization dynamically evaluates and retires auxiliary reward signals based on their impact on final evaluation objectives, enabling efficient reward utilization. Our framework substantially improves legal reasoning performance on relevant tasks selected from multiple legal benchmarks, including ChJG-Bench, LawBench, and LegalBench. Experimental results demonstrate consistent improvements in both reasoning trace quality and final answer accuracy, while achieving competitive performance compared with larger models.

1 Introduction

Recent advances in large language models (LLMs) have highlighted the importance of explicit reasoning, where intermediate thought processes are generated before arriving at final answers, to enhance reliability and interpretability across complex tasks. This form of structured reasoning is particularly crucial in the legal domain, where decisions must be grounded in legal principles and statutory provisions and logically coherent (Levi, 2022).

We observe that high-quality legal reasoning traces are generally quite long, as they must encompass both necessary legal knowledge and a logically structured analysis, as shown in Figure 1. This requires a deep understanding of legal principles, including identifying relevant statutes and legal precedents, as well as the ability to accurately interpret and apply them (Gerhardt, 2011; Li et al., 2025). At the same time, legal reasoning must adhere to several critical logical constraints, including consistency between statutes and conclusions and internal logical coherence throughout the analysis (Desai et al., 2016). Due to the high demand for domain-specific knowledge, rigorous logical reasoning, and the low tolerance for hallucinations (Dahl et al., 2024), generating reliable legal reasoning traces remains significantly more

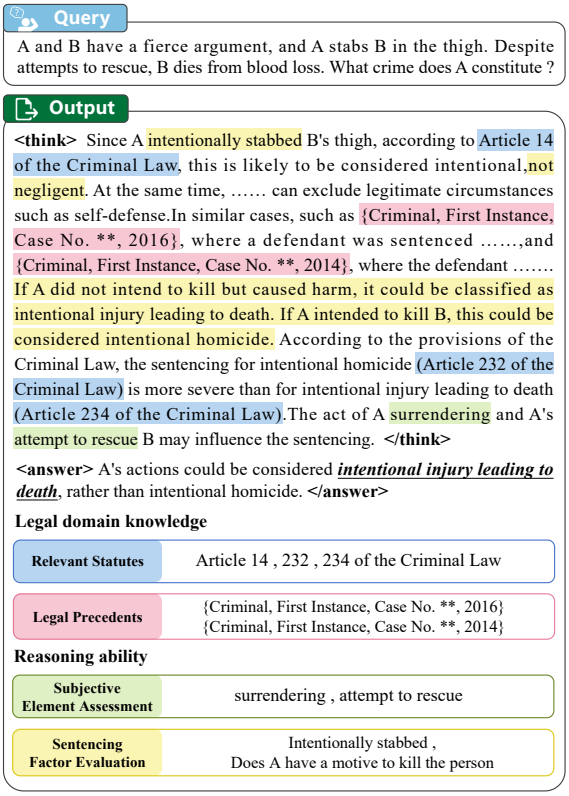

Figure 1: A diagram of a legal reasoning trace.

challenging than in general domains.

Directly post-training a foundation model on legal-domain data is insufficient to address these challenges. During Supervised Fine-Tuning (SFT), reasoning traces are often three to four times longer than the final answers, causing token-level loss to bias optimization toward the traces and weaken the learning signal for final answers. Moreover, when reasoning traces contain flaws, such as misapplied precedents, SFT cannot provide targeted feedback to correct such errors. In reinforcement learning, methods such as Group Relative Policy Optimization (GRPO) have become mainstream alignment approaches. However, GRPO relies on outcome-level rewards that evaluate only final answers and ignore the legal soundness of intermediate reasoning. A straightforward solution is to introduce multiple rewards for different aspects of reasoning (Cai et al., 2025; Wu, 2025; Wang et al., 2025). However, this requires multiple manually designed reward functions, which are costly to construct, may introduce conflicting optimization objectives, gradually lose effectiveness during training, or even lead to reward hacking (Khalaf et al., 2025; Laidlaw et al., 2025).

To address these challenges, we propose a two-stage training framework. In Stage I, we introduce **JurisCoT-SFT**, which not only injects domain knowledge into the model but also adopts a length-normalized dual objective to mitigate signal imbalance. In Stage II, we propose **LABPO** (**L**ifecycle-**A**ware **B**acktrackable **P**olicy **O**ptimization), a self-regulating reinforcement learning framework for automatically managing reward signals during policy optimization.

The central idea of LABPO is to explicitly separate reward signals into two roles: ***primary rewards***, which directly correspond to the final evaluation objectives (e.g., F1 score and accuracy), and ***auxiliary rewards***, which serve as stage-dependent training signals (e.g., output format and response length). Unlike conventional approaches that keep all reward components active throughout training, LABPO treats auxiliary rewards as adaptive optimization instruments whose usefulness evolves over the course of learning. Each auxiliary signal is retained only while it yields measurable improvements on the final evaluation objectives, and is automatically retired once its marginal benefit saturates or begins to degrade overall performance. Combined with a backtrackable optimization mechanism that restores previously stable checkpoints when ineffective signals are detected, LABPO transforms reward design from a static, manually tuned process into an automatically managed lifecycle-driven optimization process.

We evaluate LLMs on selected relevant tasks from three legal benchmarks: the statutory law benchmark **ChJGBench** (Hu et al., 2026), **LawBench** (Fei et al., 2023), and the case law benchmark **LegalBench** (Guha et al., 2023). Using Qwen3-8B as the main base model, our framework substantially improves legal reasoning performance. On ChJGBench, Qwen3-8B improves from 0.6406 to 0.7786, achieving the highest average score among all evaluated models and outperforming strong general-purpose LLMs such as Gemini-3-Pro-Preview. On LawBench, the model achieves 0.848 on *AP*, ranking second overall, and 0.666 on *CP*, ranking first. The method also generalizes well to case law reasoning in LegalBench, achieving the best result on *LRC* with an accuracy of 0.964, as well as strong performance on numerical reasoning tasks such as *Sara_N*. When applied to Llama-3.1-8B, our framework also brings substantial gains, increasing the ChJGBench average score from 0.3374 to 0.6452 and consistently improving its performance across LawBench and LegalBench. Despite having only 8B parameters, the model remains competitive with, and in some cases surpasses, much larger models. Ablation studies further confirm the necessity of both training stages.

Our contributions can be summarized as follows:

1. We propose **JurisCoT-SFT**, which balances learning signals across reasoning traces and final answers, improving the model's ability to internalize legal knowledge and produce higher-quality reasoning outputs.

2. We introduce **LABPO**, which automatically manages the lifecycle of reward signals during policy optimization by dynamically evaluating and retiring auxiliary rewards based on their impact on final evaluation objectives.

3. We train a lightweight LLM that achieves competitive performance across multiple benchmarks, establishing a new strong baseline for the legal LLM community.

2 Related Work

Traditional legal-domain LLMs typically follow a two-stage paradigm consisting of continued pre-

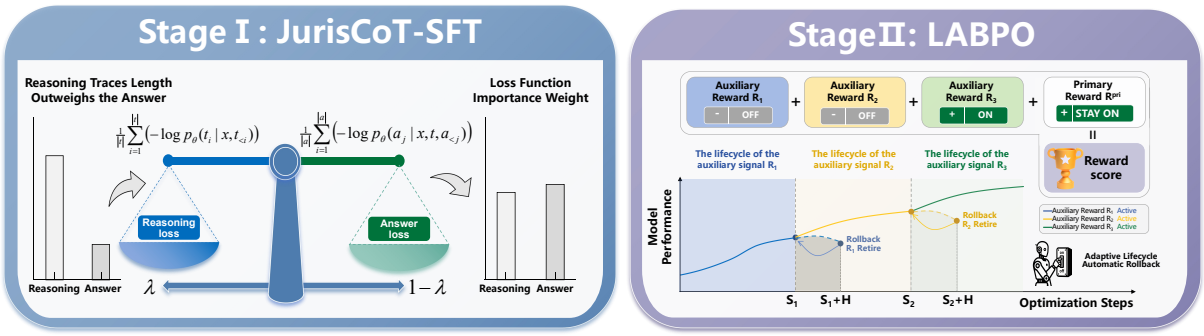

Figure 2: The framework consists of two stages: In Stage I, the coefficient λ balances learning between reasoning traces and final answers; in Stage II, auxiliary reward signals are dynamically activated and deactivated.

training and domain-specific fine-tuning to enhance legal capabilities. Representative models include LawGPT (Zhou et al., 2024), HanFei (He et al., 2023), LawLLM (Shu et al., 2024), InternLM-Law (Fei et al., 2024), and WisdomInterrogatory (Wu et al., 2026). Previous studies have shown that explicitly generating intermediate reasoning traces before producing final answers can improve performance on legal reasoning tasks (Hu et al., 2025b). Recent work has further introduced reinforcement learning to improve reasoning quality in legal LLMs. DISC-LawLLM (Yue et al., 2023) incorporates a legal syllogism framework into training data design. LexPam (Zhang et al., 2025a) focuses on legal numerical reasoning using a two-stage GRPO approach, optimizing numerical precision while incorporating legal constraints into the reward function. SyLeR (Zhang et al., 2025b) combines SFT pre-training with PPO to explore multiple reasoning paths, though it mainly relies on general metrics such as ROUGE, which may not fully capture the nuances of legal reasoning. LexPro (Chen et al., 2025) integrates SFT and GRPO but lacks a reward function specifically tailored to legal contexts. LegalΔ (Dai et al., 2026) improves reasoning through chain-of-thought guided information gain with a multidimensional reward mechanism. Despite recent advances, challenges remain in learning from long reasoning traces and in designing reward functions that effectively optimize model reasoning. To address these challenges, we propose a two-stage training framework for legal reasoning models.

3 Method

We introduce a two-stage training framework for signal rebalancing and adaptive reward scheduling, as illustrated in Figure 2.

3.1 Stage I: JurisCoT-SFT

In the legal domain, reliable LLMs must ensure both procedural legitimacy and conclusion correctness. This requires a think-then-answer paradigm. However, using standard SFT presents a critical challenge: **the severe length imbalance between the typically lengthy reasoning traces and the concise final answers**. During training, this imbalance leads to undue emphasis on the "reasoning" part, weakening the learning signal for the shorter but crucial "answering" part. Given that even minor inaccuracies in the final answer can have high-stakes consequences, addressing this training imbalance is particularly important.

To address this issue, we propose **JurisCoT-SFT**, a variant of standard SFT that introduces a length-normalized dual-objective. Formally, let a training instance be (x, t, a), where x denotes the input query, $t = (t_1, \ldots, t_{|t|})$ represents the provided reasoning sequence, and $a = (a_1, \ldots, a_{|a|})$ denotes the final answer. Given model parameters θ, we optimize the next-token cross-entropy loss, with the reasoning and answer segments weighted separately and normalized by their respective lengths:

$$\mathcal{L}_{\text{JurisCoT-SFT}}(\theta) = \lambda \cdot \frac{1}{|t|} \sum_{i=1}^{|t|} \big(-\log p_\theta(t_i \mid x, t_{<i}) \big) + (1-\lambda) \cdot \frac{1}{|a|} \sum_{j=1}^{|a|} \big(-\log p_\theta(a_j \mid x, t, a_{<j}) \big) \tag{1}$$

where λ is a balancing coefficient that trades off the importance between the reasoning traces (t) and the final answer (a). A smaller λ prioritizes answer correctness, ensuring the model learns to produce correct final answers while still being guided by the structured reasoning process.

Signal	Description
S_1	Tag-format Validation
S_2	JSON Schema Validation
S_3	Statute Citation Format Verification
S_4	Language Style Validation
S_5	Statute Authenticity Verification
S_6	Statute-conclusion Consistency Verification

Table 1: List of auxiliary signals used as rewards during training to guide legal reasoning. For details, please refer to Appendix A.4.

3.2 Stage II: Lifecycle-Aware Backtrackable Policy Optimization

Although models fine-tuned with JurisCoT-SFT demonstrate strong legal knowledge and produce structured outputs, supervised fine-tuning cannot provide fine-grained supervision over intermediate reasoning steps or deliver differentiated feedback when reasoning traces contain errors.

To address these challenges, we propose **LABPO**, a reinforcement learning framework that **automatically manages reward signals** during policy optimization. The central idea of LABPO is to decompose reward signals into two functional roles. **Primary rewards** are aligned with the final evaluation objectives and remain active throughout training. **Auxiliary rewards** serve as stage-dependent training signals that may facilitate optimization during certain phases of learning but are not necessarily beneficial throughout the entire training trajectory. Instead of treating auxiliary rewards as fixed components of the reward function, LABPO assigns each auxiliary reward an independent lifecycle: it remains active only while it produces measurable improvements on the final evaluation objectives and is automatically retired once its marginal contribution diminishes or begins to degrade overall performance.

Formally, for an input x and model output y, the training reward at stage τ is defined as:

$$R(y;\tau) = R^{\text{pri}}(y) + \sum_{i=1}^{m} \alpha_i(\tau)\, R_i^{\text{aux}}(y) \tag{2}$$

where R^{pri} denotes the primary reward, $\{R_i^{\text{aux}}\}_{i=1}^{m}$ denote auxiliary rewards, and $\alpha_i(\tau) \in \{0,1\}$ is a gating variable indicating whether the i-th auxiliary reward is active.

3.2.1 Adaptive Auxiliary Signal Scheduling

To operationalize the lifecycle management of auxiliary rewards, we propose an Adaptive Auxiliary Signal Scheduling mechanism that automatically regulates the activation and retirement of auxiliary signals during training, thereby alleviating optimization plateaus and suppressing reward hacking. The pseudo-code of the algorithm can be found in Appendix A.2.

Design Principles. Instead of keeping all auxiliary signals active throughout the entire training process, we allow for the dynamic activation of auxiliary signals. To avoid interference between multiple auxiliary signals, only one auxiliary signal is active at any given time.

Periodic Evaluation. Every n optimization steps, we evaluate the current policy π_{θ_τ} on a hold-out development set $\mathcal{D}_{\text{dev}}$, which is independent of the validation and test sets. The performance score $S(\theta_\tau)$ obtained from this evaluation serves as the sole criterion for determining the lifecycle of the auxiliary signal.

Dynamic Update of Auxiliary Signal Weights. Based on the continuous evaluation results, we update the weight α_a of the currently active auxiliary signal R_a^{aux}. The update rule is as follows:

$$\alpha_a(\tau+1) = \begin{cases} 0, & \max\limits_{k\in\{1,\dots,H\}} S(\theta_\tau) - S(\theta_{\tau-k}) < \epsilon, \\ 1, & S(\theta_\tau) - S(\theta_{\tau-1}) > 0, \\ \alpha_a(\tau), & \text{otherwise.} \end{cases} \tag{3}$$

If the latest evaluation score shows improvement over the previous one, the current auxiliary signal is considered beneficial, and its activation state is maintained. Otherwise, we monitor performance over a sliding evaluation window of the most recent H evaluations. If no significant improvement is observed within this window compared to the best historical checkpoint, we conclude that the auxiliary signal's marginal benefit has diminished or that it has induced optimization divergence.

Checkpoint Rollback and Mechanism Iteration. When the weight of the auxiliary signal is reset to 0, the policy parameters are rolled back to the checkpoint $\theta_{\tau-H}$, which was recorded $H \times n$ steps earlier. This rollback operation undoes any negative effects caused by ineffective auxiliary signals, returning the training to a more stable checkpoint. When an auxiliary signal is deactivated, the mechanism selects and activates the next auxiliary signal in a predetermined order and repeats the evaluation and dynamic weight update cycle.

3.2.2 Auxiliary Reward Signal Integration

The goal of LABPO is not to eliminate reward design, but to make reward signals more reusable and easier to deploy. Unlike conventional reinforcement learning approaches that require carefully crafted reward functions to remain consistently effective throughout training, LABPO allows researchers to explore multiple auxiliary reward signals more freely, including potentially noisy or unstable heuristic signals, without carefully predesigning, tuning, or manually combining them in advance. Through dynamic evaluation during training, LABPO automatically identifies beneficial signals and rolls back detrimental ones, thereby reducing the cost of designing and tuning multi-dimensional reward functions.

As shown in Table 1, we introduce a set of reward functions to enhance the model's legal reasoning ability. These reward functions are designed based on prior work and common evaluation criteria used in legal reasoning tasks. Since it is unclear whether the order in which different auxiliary signals are used for optimization affects the model's final performance, we conduct an auxiliary-signal ordering experiment using the six proposed auxiliary signals, as reported in Appendix A.3. The results show that the optimization order has only a limited impact on final performance under our proposed metrics.

4 Experiments

In this section, we describe the experimental setup, report the main results, analyze the training dynamics of auxiliary signal scheduling, compare reasoning performance, and conduct ablation studies.

4.1 Experimental Setup

Training Data. We prepared training data for the LLMs in both statutory law and case law domains. For **statutory law**, we construct a large-scale legal training corpus consisting of {Query, Reasoning, Answer} triplets. The query and answer are extracted from judicial documents, while the reasoning traces are generated by a strong LLM (e.g., DeepSeek-R1). To ensure generation quality, we apply a variant of rejection sampling to filter generated traces (Appendix A.1). The dataset contains 2.73M instances (Table 11). For **case law**, we construct a corresponding dataset of triplets. The same rejection sampling procedure is applied to ensure the quality of the generated reasoning traces. The dataset contains 35,965 instances (Table 12).

Evaluation Data. We evaluate LLMs on selected relevant tasks from three legal benchmarks: the statutory law benchmark ChJGBench (Hu et al., 2026), LawBench (Fei et al., 2023), and the case law benchmark LegalBench (Guha et al., 2023). Task details are summarized in Table 13, and metric definitions are provided in Appendix A.10.

ChJGBench is a real-court judicial decision benchmark comprising 3,462 cases across criminal, civil, and administrative domains. It evaluates three general dimensions shared by all cases: statutory provision application (*Law*), disputed issues identification (*ISS*), and judicial justification (*JUS*). Domain-specific tasks include criminal charge prediction (*Charge*), monetary fine recall (*Fine_R*), recidivism assessment (*REC*), mitigating circumstances identification (*MIT*), penalty duration prediction (*Pen_M*), civil claim support and payment obligation recall (*SUP*, *PAY_R*), and administrative disposition prediction (*DIS*).

LawBench is a public benchmark dataset for evaluating Chinese legal capabilities. We select three tasks: Article Prediction (*AP*), Charge Prediction (*CP*), and Sentence Prediction (*SP*).

LegalBench is a benchmark for evaluating the legal reasoning abilities of LLMs using tasks derived from real legal datasets and expert-designed evaluations. We select three representative reasoning tasks for evaluation: Legal Reasoning Causality (*LRC*), New York State Judicial Ethics (*NYSJE*), and SARA Numeric Reasoning (*Sara-N*).

Baselines. We evaluate our model against a series of strong baselines on the test set. These baselines include open-source general-purpose LLMs, such as Llama3.1-405B (Grattafiori et al., 2024), DeepSeek-R1 (Guo et al., 2025), Qwen3-235B-A22B-Instruct (Yang et al., 2025) and Grok-4 (xAI, 2025); closed-source general-purpose LLMs, including GPT-5.1, Claude-Sonnet-4.5 (Anthropic, 2025) and Gemini-3-Pro-Preview (Google DeepMind, 2026); and legal-domain-specific LLMs, including WisdomInterrogatory (Wu et al., 2026) and LawLLM (Yue et al., 2024).

Implementation Details. We adopt Llama3.1-8B and Qwen3-8B as base backbones for subsequent fine-tuning. Since Qwen3-8B achieves better overall performance, we conduct all ablation studies based on Qwen. Detailed training hyperparameters are provided in Appendix A.11, and the

Models	Avg.	Law	ISS	JUS	Criminal Case					Civil Case		Admin.
					Fine_R	MIT	Charge	Pen_M	REC	SUP	PAY_R	DIS
Specialized Legal LLMs												
WisdomInterrogatory	0.3728	0.158	0.582	0.384	0.135	0.408	0.436	0.608	0.108	0.649	0.244	0.326
LawLLM	0.4482	0.181	0.729	0.546	0.080	0.727	0.414	0.466	0.370	0.670	0.056	0.420
Open-source LLMs												
Llama-3.1-405B	0.6006	0.342	0.712	0.575	0.476	0.885	0.957	0.602	0.500	0.682	0.616	0.712
DeepSeek-R1	0.7080	0.422	0.819	0.718	0.563	0.961	0.969	0.740	0.715	0.783	0.753	0.787
Qwen3-235B-A22B-Instruct	0.7136	0.461	0.812	0.748	0.603	0.974	0.967	0.780	0.765	0.778	0.744	0.653
Grok-4	0.7166	0.499	0.818	0.728	0.501	0.973	**0.976**	0.724	0.668	0.761	0.709	0.799
Proprietary LLMs												
GPT-5.1	0.6795	0.422	0.763	0.707	0.556	0.965	0.975	0.691	0.665	0.764	0.722	0.687
Claude-Sonnet-4.5	0.7293	0.540	0.811	0.729	0.577	0.968	0.970	0.751	0.705	0.779	0.743	0.764
Gemini-3-Pro-Preview	0.7470	0.644	0.810	0.683	**0.611**	0.973	0.971	**0.786**	0.753	**0.831**	**0.757**	0.731
Our Methods												
Llama-3.1-8B	0.3374	0.099	0.592	0.357	0.102	0.507	0.221	0.513	0.318	0.575	0.117	0.254
with JurisCoT-SFT	0.6212	0.414	0.722	0.594	0.493	0.877	0.862	0.718	0.656	0.708	0.648	0.587
with JurisCoT-SFT + LABPO	0.6452	0.437	0.743	0.621	0.518	0.891	0.893	0.734	0.672	0.726	0.671	0.628
Qwen3-8B	0.6406	0.315	0.791	0.652	0.459	0.925	0.960	0.727	0.673	0.726	0.708	0.624
with JurisCoT-SFT	0.7667	0.667	0.830	0.748	0.574	**0.976**	0.975	0.770	0.747	0.801	0.717	0.802
with JurisCoT-SFT + LABPO	**0.7786**	**0.689**	**0.846**	**0.751**	0.579	0.971	0.974	0.766	**0.784**	0.822	0.724	**0.812**

Table 2: Evaluation results of different models on the ChJGBench. The best and second-best results among all models are indicated in **bold** and underline, respectively.

computational overhead of LABPO is reported in Appendix A.12.

4.2 Main Results

4.2.1 Results on Statutory Law Benchmark

We report the main experimental results on the statutory law benchmark, ChJGBench, and the LawBench in Table 2 and Table 3.

Specialized legal LLMs built on outdated base models show limited capability. Models designed specifically for the legal domain perform relatively poorly overall. On ChJGBench, WisdomInterrogatory and LawLLM achieve average scores of only 0.3728 and 0.4482, respectively, far below most modern general-purpose LLMs and our trained models. Their weaknesses are particularly evident in decision-oriented tasks such as *Fine_R* and *PAY_R*, where both models obtain substantially lower scores than recent open-source and proprietary LLMs. A similar trend is observed on LawBench: WisdomInterrogatory achieves only 0.491 on *LRC* and 0.387 on *NYSJE*, while LawLLM still exhibits a large error on *Sara_N* with 19.330. These results suggest that domain-specific legal pretraining alone is insufficient when the underlying base models lack strong general reasoning and instruction-following capabilities.

General-purpose LLMs perform reasonably well overall but still exhibit limitations in legal knowledge. Modern general-purpose LLMs achieve relatively strong overall performance across both benchmarks, indicating that large-scale pretraining provides a useful foundation for legal reasoning. On ChJGBench, Gemini-3-Pro-Preview obtains the best average score among general-purpose models with 0.7470. However, these models become less stable when tasks require precise grounding in legal knowledge. For example, on the *Law* metric of ChJGBench, Gemini-3-Pro-Preview achieves the highest score among general-purpose models, but its score is only 0.644, which remains lower than our best model's 0.689.

Our two-stage training framework consistently improves lightweight base models. Applying our proposed two-stage framework consistently improves both lightweight backbones. On ChJGBench, Llama-3.1-8B improves from 0.3374 to 0.6452 after JurisCoT-SFT and LABPO, while Qwen3-8B improves from 0.6406 to 0.7786. The final Qwen3-8B-based model achieves the best overall score among all evaluated models and obtains top performance on key metrics such as *Law* (0.689), *ISS* (0.846), *JUS* (0.751), *REC* (0.784), and *DIS* (0.812). A similar trend is observed on LawBench.

4.2.2 Results on Case Law Benchmark

We report the main experimental results on three case law tasks in LegalBench in Table 4. The

Models	AP	CP	SP
Specialized Legal LLMs			
WisdomInterrogatory	0.328	0.351	0.804
LawLLM	0.054	0.127	0.670
Open-source LLMs			
Llama-3.1-405B	0.306	0.352	0.807
DeepSeek-R1	0.831	0.606	**0.891**
Qwen3-235B-A22B-Instruct	**0.865**	0.607	0.857
Grok-4	0.844	0.553	0.885
Proprietary LLMs			
GPT-5.1	0.721	0.561	0.883
Claude-Sonnet-4.5	0.818	0.550	0.775
Gemini-3-Pro-Preview	0.779	0.637	0.822
Our Methods			
Llama-3.1-8B	0.057	0.223	0.707
with JurisCoT-SFT	0.756	0.254	0.849
with JurisCoT-SFT + LABPO	0.771	0.288	0.862
Qwen3-8B	0.792	0.537	0.837
with JurisCoT-SFT	0.822	0.613	0.843
with JurisCoT-SFT + LABPO	0.848	**0.666**	0.854

Table 3: Evaluation results of different models on LawBench. **AP**, **CP**, and **SP** denote Article Prediction, Charge Prediction, and Sentence Prediction, respectively. The best and second-best results among all models are highlighted in **bold** and underline, respectively.

Models	LRC	NYSJE	Sara_N
Specialized Legal LLMs			
WisdomInterrogatory	0.491	0.387	–
LawLLM	0.746	0.630	19.330
Open-source LLMs			
Llama-3.1-405B	0.909	0.709	7.720
DeepSeek-R1	0.927	0.771	**0.250**
Qwen3-235B-A22B-Instruct	0.891	0.791	9.618
Grok-4	0.909	0.805	5.802
Proprietary LLMs			
GPT-5.1	0.946	**0.836**	7.051
Claude-Sonnet-4.5	0.891	0.747	0.994
Gemini-3-Pro-Preview	0.873	0.764	7.677
Our Methods			
Llama-3.1-8B	0.764	0.493	21.088
with JurisCoT-SFT	0.855	0.575	6.125
with JurisCoT-SFT + LABPO	0.909	0.623	4.846
Qwen3-8B	0.873	0.596	8.093
with JurisCoT-SFT	0.927	0.627	3.949
with JurisCoT-SFT + LABPO	**0.964**	0.688	1.753

Table 4: Evaluation results of different models on LegalBench. The best and second-best results among all models are indicated in **bold** and underline, respectively.

results show that our JurisCoT-SFT and LABPO framework generalizes beyond statutory law tasks and remains effective on case law benchmarks. In particular, Qwen3-8B with JurisCoT-SFT + LABPO achieves the best performance on *LRC*, improving from 0.873 to 0.964 and outperforming all evaluated models. It also substantially reduces the numerical error on *Sara_N*, from 8.093 to 1.753. For the *NYSJE* task, performance is limited by the model size, yielding lower results compared to larger proprietary models. However, it still demonstrates a significant improvement over the base model. Overall, these results demonstrate the effectiveness and cross-domain applicability of our training framework.

4.3 Training Dynamics of Auxiliary Signal Scheduling

As shown in the Figure 3, the auxiliary signal (S_5) initially improves the development score but later reaches a plateau. When no further improvement is observed within the evaluation window ($H = 4$), the signal is retired and the policy rolls back to the best checkpoint within the window. The next auxiliary signal (S_6) is then activated, after which the model performance resumes an upward trend and eventually surpasses the previous peak obtained under S_5. This training trajectory suggests that auxiliary reward signals may only be beneficial during certain stages of optimization and can become less effective as training progresses, and even introduce negative effects. The rollback mechanism helps mitigate potential performance degradation by reverting the model to previously stable checkpoints.

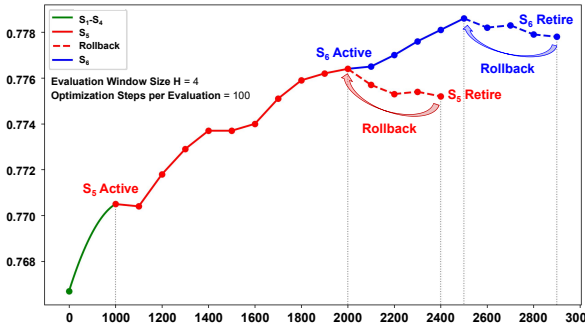

Figure 3: Training dynamics of auxiliary signal scheduling in LABPO. The early stages (S_1–S_4) are omitted for clarity, and the figure visualizes the training trajectories of the last two auxiliary signals.

4.4 Reasoning Performance Comparison

To further investigate improvements in reasoning ability, we conduct a direct evaluation of reasoning traces. We adopt a win-rate comparison framework to perform pairwise evaluations among the baseline model Qwen3-8B, our proposed method, and the strong open-source model DeepSeek-R1. We evaluate reasoning traces along two general di-

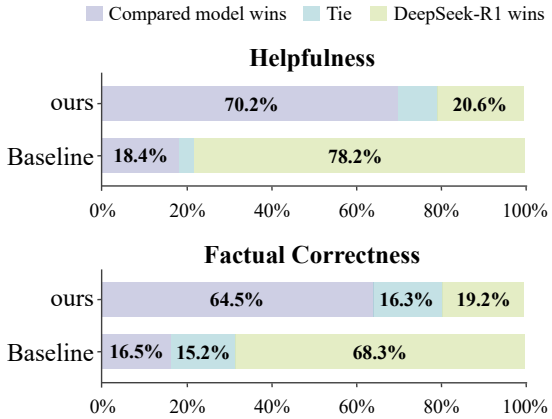

Figure 4: Comparison of reasoning traces in helpfulness and factual accuracy on the ChJGBench. Purple: loss; Blue: tie; Green: win for DeepSeek-R1.

No.	SFT		RL			Average
	Used	Variant	Used	Retire	Rollback	
1	✓	✓	✓	✓	✓	0.7786
2	✓	✓	✓	✓	×	0.7718
3	✓	✓	•	×	×	0.7671
4	✓	✓	○	×	×	0.7684
5	✓	✓	×	×	×	0.7667
6	✓	×	✓	✓	✓	0.7687
7	✓	×	×	×	×	0.7399
8	×	×	✓	✓	✓	0.6735
9	×	×	×	×	×	0.6406

Table 5: Ablation study of different training configurations. ✓indicates the component is enabled and × indicates it is disabled. Row 3 reduces to GRPO with aggregated primary and auxiliary rewards, while Row 4 reduces to GRPO using only the primary reward.

mensions: **Helpfulness** and **Factual Correctness**. Gemini-3-Pro-Preview is employed as the judge.

The details of this experiment can be found in Appendix A.5. The specifics of the alignment evaluation between LLMs and human annotators are available in Appendix A.7. The LLM evaluator shows strong alignment with aggregated human judgments, achieving an accuracy of 0.84 and a Cohen's κ of 0.756.

As shown in Figure 4, our method demonstrates notable improvements in reasoning quality compared to the baseline Qwen3-8B. In direct comparisons with the strong open-source model DeepSeek-R1, the win rate of our approach increases from 18.37% to 70.19% on Helpfulness and from 16.46% to 64.47% on Factual Correctness. Overall, the proposed training strategy enables an 8B-scale model to achieve reasoning performance on legal tasks that is competitive with, or even superior to, the 671B-parameter MoE model DeepSeek-R1.

4.5 Ablation Studies

Our ablation study on ChJGBench (Table 5) evaluates each component's contribution; the trends on LegalBench and LawBench are consistent (see Appendix A.8).

(1) LABPO Mechanisms Improve RL Training Stability. Rows 1-4 analyze the effect of the Retire and Rollback mechanisms during RL. The full configuration (Row 1) achieves the best performance of 0.7786. Removing Rollback (Row 2) leads to a consistent performance drop to 0.7718, and removing both mechanisms (Row 3) further degrades the result to 0.7671, indicating that LABPO and both of its core components improve training stability. Moreover, Row 3 performs slightly worse than Row 4, which optimizes using only the primary reward. A possible reason is that different rewards may not share fully aligned optimization objectives, and aggregating multiple rewards can introduce noise or optimization conflicts.

(2) JurisCoT-SFT Improves the Effectiveness of Supervised Training. When JurisCoT-SFT is adopted (Row 5), the model achieves a score of 0.7667, outperforming naive SFT (Row 7, 0.7399). Further ablation studies on the SFT stage and sensitivity analysis of the lambda coefficient are provided in Appendix A.6.

(3) LABPO Improves Performance Across Initialization Settings. Comparisons of Rows 1/5, 6/7, and 8/9 show that LABPO consistently improves performance under different initializations, boosting JurisCoT-SFT, naive SFT, and base models from 0.7667 to 0.7786, 0.7399 to 0.7687, and 0.6406 to 0.6735, respectively. This indicates that the gains stem not only from JurisCoT-SFT initialization but also from the proposed RL strategy.

5 Conclusion

In this paper, we introduce a two-stage training framework to enhance the legal reasoning capabilities of LLMs. We propose **JurisCoT-SFT**, which balances learning signals between reasoning traces and final answers, and **LABPO**, a dynamic reward scheduling method that optimizes auxiliary reward signals. Our approach achieves strong performance on statutory and case law benchmarks. Ablation studies further validate its effectiveness.

Limitations

Despite the strong empirical performance and methodological innovations of our approach, several limitations should be acknowledged. First, although the auxiliary signals in the LABPO framework no longer require meticulous selection thanks to the backtracking mechanism, the model still depends on a predefined reward function. Fully automating the design of reward functions (e.g., enabling LLMs to autonomously design appropriate reward signals) remains an open challenge, and we leave this direction for future work. Second, due to resource constraints, we were unable to validate the effectiveness of our proposed two-stage approach on larger-scale models. Third, compared with simpler reinforcement learning methods such as GRPO, LABPO introduces additional computational overhead. The lifecycle management of auxiliary reward signals, periodic evaluation on the development set, and potential rollback operations require extra training iterations and evaluation costs. Although these mechanisms improve training stability and final performance, they may increase the overall computational budget. Designing more efficient reward scheduling or lightweight rollback strategies to reduce this overhead is an important direction for future research.

Ethics Statement

Given the sensitive nature of the legal domain, the application of artificial intelligence in this field necessitates rigorous ethical management. To address potential ethical concerns, we implement several safeguards. In particular, to prevent the leakage of private information (e.g., personal names), we anonymize or replace sensitive data with neutral third-person references when constructing both training datasets and evaluation benchmarks. This ensures that our research upholds the principles of privacy protection and responsible AI development.

References

Anthropic. 2025. Claude Sonnet 4.5 System Card.

Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

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. *Preprint*, arXiv:2510.10072.

Haotian Chen, Yanyu Xu, Boyan Wang, Chaoyue Zhao, Xiaoyu Han, Fang Wang, Lizhen Cui, and Yonghui Xu. 2025. Lexpro-1.0 technical report. *Preprint*, arXiv:2503.06949.

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

Xin Dai, Buqiang Xu, Zhenghao Liu, Yukun Yan, Huiyuan Xie, Xiaoyuan Yi, Shuo Wang, and Ge Yu. 2026. Legalδ: Enhancing legal reasoning in llms via reinforcement learning with chain-of-thought guided information gain. *Preprint*, arXiv:2508.12281.

Saoirse Desai, Stian Reimers, and David Lagnado. 2016. Consistency and credibility in legal reasoning: A bayesian network approach. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 38.

Zhiwei Fei, Xiaoyu Shen, Dawei Zhu, Fengzhe Zhou, Zhuo Han, Songyang Zhang, Kai Chen, Zongwen Shen, and Jidong Ge. 2023. Lawbench: Benchmarking legal knowledge of large language models. *arXiv preprint arXiv:2309.16289*.

Zhiwei Fei, Songyang Zhang, Xiaoyu Shen, Dawei Zhu, Xiao Wang, Maosong Cao, Fengzhe Zhou, Yining Li, Wenwei Zhang, Dahua Lin, and 1 others. 2024. Internlm-law: An open source chinese legal large language model. *arXiv preprint arXiv:2406.14887*.

Yi Feng, Chuanyi Li, and Vincent Ng. 2022. Legal judgment prediction: A survey of the state of the art. In *IJCAI*, pages 5461–5469.

Michael J Gerhardt. 2011. *The power of precedent*. Oxford University Press.

Google DeepMind. 2026. Gemini 3 Pro Model Card. Model release: November 2025; last updated: May 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu,

Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Wanwei He, Jiabao Wen, Lei Zhang, Hao Cheng, Bowen Qin, Yunshui Li, Feng Jiang, Junying Chen, Benyou Wang, and Min Yang. 2023. Hanfei-1.0. https://github.com/siat-nlp/HanFei.

Yinghao Hu, Leilei Gan, Wenyi Xiao, Kun Kuang, and Fei Wu. 2025a. Fine-tuning large language models for improving factuality in legal question answering. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4410–4427.

Yinghao Hu, Linggang Kong, Qian Xiao, Yuting Huang, Yunlong Zhang, Yujie Zhang, Fei Duan, Yiquan Wu, Kaisong Song, Jun Lin, Kun Kuang, and Fei Wu. 2026. Chjgbench: A comprehensive benchmark for chinese legal judgment generation with diagnostic evaluation.

Yinghao Hu, Yaoyao Yu, Leilei Gan, Bin Wei, Kun Kuang, and Fei Wu. 2025b. Evaluating test-time scaling llms for legal reasoning: Openai o1, deepseek-r1, and beyond. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, page 13759–13781. Association for Computational Linguistics.

Hadi Khalaf, Claudio Mayrink Verdun, Alex Oesterling, Himabindu Lakkaraju, and Flavio du Pin Calmon. 2025. Inference-time reward hacking in large language models. *arXiv preprint arXiv:2506.19248*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Cassidy Laidlaw, Shivam Singhal, and Anca Dragan. 2025. Correlated proxies: A new definition and improved mitigation for reward hacking. *Preprint*, arXiv:2403.03185.

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

Ang Li, Yiquan Wu, Yifei Liu, Ming Cai, Lizhi Qing, Shihang Wang, Yangyang Kang, Chengyuan Liu, Fei Wu, and Kun Kuang. 2025. UniLR: Unleashing the power of LLMs on multiple legal tasks with a unified legal retriever. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11953–11967, Vienna, Austria. Association for Computational Linguistics.

Jorge Nocedal and Stephen J Wright. 2006. *Numerical optimization*. Springer.

Dong Shu, Haoran Zhao, Xukun Liu, David Demeter, Mengnan Du, and Yongfeng Zhang. 2024. Lawllm: Law large language model for the us legal system. In *Proceedings of the 33rd ACM International Conference on information and knowledge management*, pages 4882–4889.

Beining Wang, Weihang Su, Hongtao Tian, Tao Yang, Yujia Zhou, Ting Yao, Qingyao Ai, and Yiqun Liu. 2025. From< answer> to< think>: Multidimensional supervision of reasoning process for llm optimization. *arXiv preprint arXiv:2510.11457*.

Xiaobao Wu. 2025. A comprehensive survey on learning from rewards for large language models: Reward models and learning strategies. *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 17847–17875.

Yiquan Wu, Yuhang Liu, Yifei Liu, Ang Li, Siying Zhou, Kun Kuang, and Fei Wu. 2026. Wisdominterrogatory (luwen): An open-source legal large language model technical report. *Preprint*, arXiv:2604.06737.

xAI. 2025. Grok 4 Model Card. Last updated: August 20, 2025.

Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, and 1 others. 2018. Cail2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478*.

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. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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

Shengbin Yue, Shujun Liu, Yuxuan Zhou, Chenchen Shen, Siyuan Wang, Yao Xiao, Bingxuan Li, Yun Song, Xiaoyu Shen, Wei Chen, and 1 others. 2024. Lawllm: Intelligent legal system with legal reasoning and verifiable retrieval. In *International Conference on Database Systems for Advanced Applications*, pages 304–321. Springer.

Wang ZeJun. 2022. simbert-base-chinese. https://huggingface.co/WangZeJun/simbert-base-chinese.

Kepu Zhang, Guofu Xie, Weijie Yu, Mingyue Xu, Xu Tang, Yaxin Li, and Jun Xu. 2025a. Legal mathematical reasoning with llms: Procedural alignment through two-stage reinforcement learning. *Preprint*, arXiv:2504.02590.

Kepu Zhang, Weijie Yu, Zhongxiang Sun, and Jun Xu. 2025b. An explicit syllogistic legal reasoning framework for large language models. *Preprint*, arXiv:2504.04042.

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. *Preprint*, arXiv:2406.04614.

A Appendix

A.1 Attempt-Bounded Self-Verification

This section presents a method for improving the quality of training data by filtering reasoning traces, as illustrated in Figure 5. This method is called **Attempt-Bounded Self-Verification**. Specifically, given a question–answer pair (x, a), we prompt a strong LLM to generate a reasoning trace t. We then feed the pair (x, t) into the model to be trained, which produces multiple candidate answers $\hat{a}^{(k)}$. If any candidate answer matches the ground-truth answer a, we regard the corresponding training triple (x, t, a) as valid and retain it in the training set. We allow up to K attempts; if none of the generated candidates matches the reference answer, the training triple is discarded.

$$\mathcal{D}' = \Big\{(x,t,a) \in \mathcal{D} \;\Big|\; \exists\, k \le K,\ \hat{a}^{(k)} \sim p_\theta(\cdot \mid x,t),\ v(\hat{a}^{(k)}, a) = 1\Big\} \tag{4}$$

Here, $\hat{a}^{(k)}$ denotes the k-th candidate answer sampled from the model conditioned on (x, t), $p_\theta(\cdot \mid x, t)$ represents the conditional distribution defined by model parameters θ, and $v(\hat{a}^{(k)}, a) \in \{0, 1\}$ is a verification function indicating whether the candidate answer agrees with the ground-truth answer a.

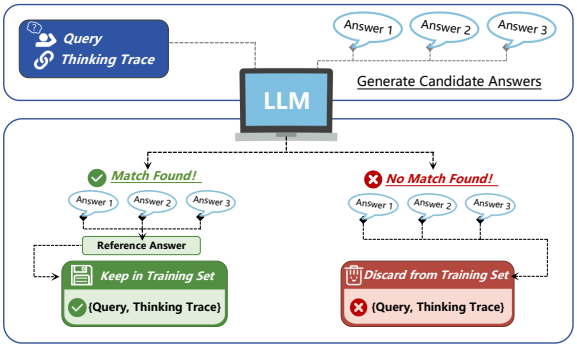

Figure 5: An overview of the Attempt-Bounded Self-Verification framework, in which a query and its reasoning trace are fed into the target model, and the pair is retained only if the model can infer the correct answer within a bounded number of attempts.

Signal Order	Average
$S_1 \to S_2 \to S_3 \to S_4 \to S_5 \to S_6$	0.6635
$S_6 \to S_5 \to S_4 \to S_3 \to S_2 \to S_1$	0.6556
$S_4 \to S_3 \to S_6 \to S_1 \to S_5 \to S_2$	0.6579
$S_5 \to S_1 \to S_2 \to S_4 \to S_3 \to S_6$	0.6584
$S_3 \to S_4 \to S_1 \to S_6 \to S_5 \to S_2$	0.6572

Table 6: Ablation study on the effect of auxiliary signal ordering using Qwen3-8B. Reported values are average performance scores computed using Eq. (24).

A.2 Pseudo-code for Adaptive Auxiliary Signal Scheduling

This section provides the pseudo-code for Lifecycle-Aware Backtrackable Policy Optimization, as shown in Algorithm 1.

A.3 Effect of Auxiliary Signal Ordering

We investigate whether the predefined ordering of auxiliary signals affects the training dynamics and final performance of LABPO. Since LABPO activates auxiliary signals sequentially through the Adaptive Auxiliary Signal Scheduling mechanism, it is important to examine whether the specific ordering of these signals plays a critical role.

To isolate the effect of signal ordering, we conduct a controlled ablation study under a standardized setting. All experiments are performed using Qwen3-8B on the Chinese Statutory Law training dataset, and evaluation is conducted on ChJGBench with the same metrics as in the main experiments. In each run, the full LABPO framework is retained to ensure consistency. Specifically, training begins with only the first auxiliary signal activated, while all remaining signals are initialized in a retired state. Each signal remains active until it is automatically retired by the Adaptive Auxiliary Signal Scheduling mechanism. Upon retirement, training rolls back to checkpoint $\theta_{\tau-H}$ and proceeds with the next auxiliary signal according to a predefined sequence.

To further control experimental variables, we use the same set of auxiliary signals as defined in Table 1. We evaluate five ordering strategies:

1. Forward order: $S_1 \to S_2 \to S_3 \to S_4 \to S_5 \to S_6$
2. Reversed order: $S_6 \to S_5 \to S_4 \to S_3 \to S_2 \to S_1$
3. Random order 1: $S_4 \to S_3 \to S_6 \to S_1 \to S_5 \to S_2$
4. Random order 2: $S_5 \to S_1 \to S_2 \to S_4 \to S_3 \to S_6$
5. Random order 3: $S_3 \to S_4 \to S_1 \to S_6 \to S_5 \to S_2$

Table 6 summarizes the final evaluation results under these different signal orderings. To assess

Algorithm 1 Adaptive Auxiliary Signal Scheduling for Legal Reasoning

Require: training dataset $\mathcal{D}_t$; policy π_θ with parameters θ; reference π_{ref} (KL); fixed held-out development set $\mathcal{D}_{\text{dev}}$; primary metric score $S(\theta)$ (score of π_θ on $\mathcal{D}_{\text{dev}}$); auxiliary rewards $\{R_i^{\text{aux}}\}_{i=1}^m$ (at most one active at a time); evaluation interval n; gain threshold ϵ; backtrack window H=3.

Select an active auxiliary index a; set $\alpha_a \leftarrow 1$ and $\alpha_{j\neq a} \leftarrow 0$; save checkpoint θ_0
for optimization step $s = 1, 2, \ldots$ **do**
  Sample $x \sim \mathcal{D}_t$, roll out $y \sim \pi_\theta(\cdot|x)$, and update π_θ by GRPO using Eq. 2
  if $s \bmod n = 0$ **then**
    $\tau \leftarrow s/n$; save checkpoint $\theta_\tau \leftarrow \theta$; compute $S(\theta_\tau)$ on $\mathcal{D}_{\text{dev}}$
    if $S(\theta_\tau) - S(\theta_{\tau-1}) > 0$ **then**
      continue ▷ keep R_a^{aux} active
    else if $\max_{k\in\{1,\ldots,\min(H,\tau)\}} \big(S(\theta_\tau) - S(\theta_{\tau-k})\big) < \epsilon$ **then**
      Retire R_a^{aux}: set $\alpha_a \leftarrow 0$; backtrack parameters $\theta \leftarrow \theta_{\tau-H}$
      Switch to a new auxiliary reward a and set $\alpha_a \leftarrow 1$
    end if
  end if
end for

sensitivity to signal ordering, we examine the variability across different configurations. The results show a mean performance of 0.6585 with a standard deviation of 0.0030, indicating low dispersion. Performance ranges from 0.6556 to 0.6635, corresponding to an absolute difference of 0.0079, which is marginal. Furthermore, the 95% confidence interval is [0.6547, 0.6623], largely covering the observed results. **These findings suggest that, for the auxiliary signals considered in this work, variations caused by different signal orderings are limited and do not lead to meaningful performance differences.** It should be noted that this experiment serves as a robustness check for the selected auxiliary signals, rather than evidence that signal ordering is generally unimportant.

A.4 Reward function for auxiliary signals

In Section 3.2.2 and Table 1, we provide a set of simple and intuitive reward functions, and the specific calculation formulas are as follows.

A.4.1 Tag-format Validation

Model outputs are expected to follow a strict structure, with the reasoning process enclosed in `<think>...</think>`. Specifically, if the tags are correct and no extraneous content exists outside them, the model receives a reward of 1. If these conditions are not met, the reward is 0. The reward function is defined as:

$$R_{\text{Fmt}}(y) = \begin{cases} 1, & \text{if the format is correct} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

A.4.2 JSON Schema Validation

Model outputs must be valid JSON and contain all required keys. The reward depends on the JSON structure's completeness and validity. If the output is valid JSON with all required keys and correct data types, the model receives a reward of 1. If the JSON is valid but some optional keys are missing, the reward is 0.5. If the output is invalid JSON, the reward is 0. The reward function is defined as:

$$R_{\text{JSON}}(y) = \begin{cases} 1, & \text{valid, all required keys} \\ 0.5, & \text{valid, missing optional keys} \\ 0, & \text{invalid JSON} \end{cases} \tag{6}$$

A.4.3 Statute Citation Format Verification

Model outputs should include statute citations in a canonical format. If all statute citations are correctly formatted, the model receives a reward of 1. If the citations are missing or incorrect, the reward is 0. The reward function is defined as:

$$R_{\text{CFmt}}(y) = \begin{cases} 1, & \text{correct citation format} \\ 0, & \text{incorrect format or missing} \end{cases} \tag{7}$$

A.4.4 Language Style Validation

The model's output needs to ensure a formal, neutral, and objective writing style. It is difficult to judge the language style using rules alone, so we use a strong LLM, such as Gemini-3-Pro-Preview, to evaluate the text's writing style. We provide the LLM with a set of instructions detailing the criteria for style judgment. Based on these instructions, the LLM outputs a score.

$$R_{\text{Style}}(y) = \text{LLM}(\text{prompt}, \text{text}) \tag{8}$$

The meanings of the scores are as follows:

- A score of 1 is assigned when the style is formal and neutral.

- A score of 0.5 is given when the style is mostly formal and objective, with minor subjective emotions that slightly affect the user's experience.

- A score of 0 is assigned when the style is informal or when the response is filled with a large amount of subjective emotions.

A.4.5 Statute Authenticity Verification

In our process, we perform a truthfulness check on all the statutes cited in the reasoning traces of the generated answer y. We require that the referenced statutes be real and not fabricated. The design of this reward score is inspired by the procedure outlined in (Hu et al., 2025a).

1. Extract the generated statutes using regular expressions or LLMs, denoted as $L_{\text{gen}} = \{S_{\text{name}}, S_{\text{number}}, S_{\text{content}}\}$, where S_{name} is the name of the statute, S_{number} is the statute number, and S_{content} is the content of the statute.

2. Embed the extracted content using the SimBERT (ZeJun, 2022) model, alongside all contents from a real statute database.

3. Calculate the semantic similarity between the generated content and the real statutes. Select the real statute with the highest similarity as the response, denoted as $L_{\text{best}} = \{S'_{\text{name}}, S'_{\text{number}}, S'_{\text{content}}\}$. The rationale behind this process is that although large models may not generate entirely hallucination-free statutes, they generally maintain high semantic consistency with real statutes. The closer the model-generated statute is to a real statute, the higher its semantic similarity.

4. Using rule-based comparisons, assess whether L_{gen} and L_{best} indicate hallucinations in the model-generated statutes. Specifically, we consider the generated statute L_{gen} to be non-hallucinated if: $S'_{\text{content}} \subseteq S_{\text{content}}, S'_{\text{number}} = S_{\text{number}}, S'_{\text{name}} \in$ valid appellations of S_{name}.

We consider the rate of non-hallucinated statutes as the reward score:

$$R_{\text{StatReal}}(y) = \frac{\text{non-hallucinated statutes}}{\text{Total number of statutes}} \tag{9}$$

where R_{StatReal} is a value between 0 and 1. A higher value of R indicates that a larger proportion of the generated statutes are accurate and do not contain hallucinations.

A.4.6 Statute-conclusion Consistency Verification

Model outputs must ensure that the cited statutes are consistent with the final conclusion. Due to the difficulty of judging the consistency between the cited statutes and the final conclusion through rules alone, we utilize a strong language model, such as Gemini-3-Pro-Preview, to assess this consistency.

If the statutes directly support the conclusion, the model receives a reward of 1. If the statutes potentially support the conclusion, the reward is 0.5. If the statutes contradict or are irrelevant to the conclusion, the reward is 0. The reward function is defined as follows:

$$R_{\text{Consistency}}(y) = \begin{cases} 1, & \text{direct support} \\ 0.5, & \text{potential support} \\ 0, & \text{contradiction/irrelevance} \end{cases} \tag{10}$$

A.5 Reasoning Performance Comparison Details

1. We use the test set mentioned in 4.1 to generate complete reasoning traces for three different models: Qwen3-8B, JurisCoT-SFT + LABPO, and DeepSeek-R1, all under the same prompt template. The generation settings are as follows: temperature = 0.7, top_p = 0.95, and max_new_tokens = 8192.

2. We then provide the input, formatted as {query, ground truth answer, golden answer reasoning traces, modelA reasoning traces, modelB reasoning traces}, to Gemini-3-Pro-Preview. The model is asked to output the win, tie, and lose results for the reasoning traces of modelA and modelB across two dimensions (Helpfulness and Factual Correctness). To ensure fairness, the temperature setting for Gemini-3-Pro-Preview is set to 0.

3. The definitions for these two evaluation dimensions are as follows: Helpfulness primarily evaluates whether the reasoning process is clear, logical, and contributes to solving the problem. Factual Correctness assesses whether the facts in the reasoning process are accurate and error-free. For Factual Correctness, the evaluator model, Gemini-3-Pro-Preview, refers to the ground truth answer and golden answer reasoning traces to compare which model's reasoning traces contain factual elements that align more closely with the

reference answers, or have smaller discrepancies.

4. For any pairwise model comparison, the win rate for a specific dimension is defined as:

$$\text{Win Rate} = \frac{\text{Number of wins in that dimension}}{\text{Total number of samples}} \quad (11)$$

A.6 Ablation Study of SFT Signal-Rebalancing Objectives

Experimental Setup. To further examine the source of the performance gains achieved by JurisCoT-SFT during the SFT stage, we compare it with the standard SFT objective and several simple signal-rebalancing alternatives, including Answer-token Upsampling and Separate Normalization. All experiments in this section are conducted without LABPO, so that the effect of the SFT objective can be isolated. We use the same base model, Qwen3-8B, training data, batch size, learning-rate schedule, maximum sequence length, and number of training steps across all variants.

Standard SFT. For each training instance, let x denote the input query, $t = (t_1, \ldots, t_{|t|})$ denote the reasoning trace, and $a = (a_1, \ldots, a_{|a|})$ denote the final answer. We first define the reasoning-token loss and the answer-token loss as follows:

$$\mathcal{L}_t = \sum_{i=1}^{|t|} -\log p_\theta(t_i \mid x, t_{<i}). \quad (12)$$

$$\mathcal{L}_a = \sum_{j=1}^{|a|} -\log p_\theta(a_j \mid x, t, a_{<j}). \quad (13)$$

The standard SFT objective optimizes the token-level negative log-likelihood over the concatenated reasoning and answer sequence:

$$\mathcal{L}_{\text{SFT}} = \frac{\mathcal{L}_t + \mathcal{L}_a}{|t| + |a|}. \quad (14)$$

Answer-token Upsampling. Since legal reasoning traces are often much longer than final answers, the standard SFT objective may over-emphasize reasoning tokens and weaken the learning signal from the final answer. We therefore introduce an Answer-token Upsampling baseline, which increases the contribution of answer tokens by a fixed factor k:

$$\mathcal{L}_{\text{AnsUp}} = \frac{\mathcal{L}_t + k\mathcal{L}_a}{|t| + k|a|}. \quad (15)$$

We set $k = 4$, motivated by the empirical observation that reasoning traces are typically about three to four times longer than final answers. This baseline serves as a direct heuristic for compensating for the diluted answer-token learning signal, although it inevitably requires choosing k based on prior experience.

Objective	Setting	Avg.
Standard SFT	–	0.7399
Answer-token Upsampling	$k = 4$	0.7428
Separate Normalization	–	0.7665
JurisCoT-SFT	$\lambda = 0.25$	0.7629
JurisCoT-SFT	$\lambda = 0.50$	0.7667
JurisCoT-SFT	$\lambda = 0.75$	0.7612

Table 7: Ablation study of different SFT objectives without LABPO. All variants are trained under the same experimental configuration using Qwen3-8B as the base model. Avg. denotes the average score on the ChJG-Bench dataset.

Separate Normalization. A more direct approach is to normalize the reasoning and answer losses separately before combining them. This prevents the lengthy reasoning trace from diminishing the relative contribution of the final answer:

$$\mathcal{L}_{\text{SepNorm}} = \frac{\mathcal{L}_t}{|t|} + \frac{\mathcal{L}_a}{|a|}. \quad (16)$$

JurisCoT-SFT. To further investigate the influence of the relative weight between the reasoning and answer components, we additionally report results for a parameterized variant with $\lambda \in \{0.25, 0.50, 0.75\}$. Smaller values of λ emphasize the final answer, while larger values emphasize the reasoning trace:

$$\mathcal{L}_{\text{JurisCoT}} = \lambda \frac{\mathcal{L}_t}{|t|} + (1 - \lambda) \frac{\mathcal{L}_a}{|a|}. \quad (17)$$

Results. Standard SFT achieves an average score of 0.7399. Answer-token Upsampling with $k = 4$ yields only a marginal improvement, increasing the average score to 0.7428. In contrast, Separate Normalization substantially improves the average score to 0.7665. This suggests that balancing the training signal between the reasoning trace and the final answer is effective. The limited performance of Answer-token Upsampling is likely due to the fact that a single global value of k cannot adapt to instance-level variations in the relative lengths of reasoning traces and final answers.

For JurisCoT-SFT, the results with $\lambda = 0.50$, $\lambda = 0.25$, and $\lambda = 0.75$ are relatively close, but $\lambda = 0.50$ performs best. This indicates that, under

our experimental data and settings, maintaining a balanced contribution from the reasoning trace and the final answer is more effective than over-emphasizing either component.

It is worth noting that Separate Normalization and JurisCoT-SFT with $\lambda = 0.50$ are essentially equivalent at the objective level. Specifically, when $\lambda = 0.50$, the JurisCoT-SFT objective differs from Separate Normalization only by a positive constant scaling factor. Since multiplying an objective by a positive constant does not change its set of minimizers, the two objectives should have the same optimum under an idealized optimization setting (Nocedal and Wright, 2006; Boyd and Vandenberghe, 2004; Kingma and Ba, 2017).

A.7 Human Evaluation and Inter-Annotator Agreement

Human Evaluation Setup. We conduct a human evaluation to assess the reliability of the proposed reasoning evaluation protocol and to analyze inter-annotator consistency. Three annotators with formal legal training independently evaluate 500 queries sampled from the test set using a stratified sampling strategy. To ensure representativeness, the queries are stratified by (i) legal domain, including criminal, civil, and administrative cases, and (ii) case complexity, measured by the length of the factual description. Each query corresponds to one pairwise comparison instance. The 500 instances are evenly divided into two comparison settings: our proposed method versus DeepSeek-R1 and our proposed method versus Qwen3-8B, ensuring a balanced number of pairs for both baselines.

Annotation Protocol. For each query, annotators perform pairwise comparisons between two anonymized reasoning traces along two dimensions: **Helpfulness** and **Factual Correctness**. The order of the two candidates is randomized for each instance to avoid positional bias. All annotations are conducted in a double-blind manner, where annotators are unaware of the model identities associated with the reasoning traces.

Aggregation Rule. For each query and each evaluation dimension, annotators assign one of three labels: $\{A \text{ wins}, \text{ Tie}, B \text{ wins}\}$. Final human labels are determined using a conservative majority rule: a model is assigned a *win* label only if at least two out of three annotators explicitly prefer its reasoning trace. If neither model receives at least two win votes, the final label is determined as *Tie*. This aggregation strategy reduces the impact of subjective or ambiguous judgments and yields more reliable human evaluation results.

α	Majority	LLM Acc.	LLM κ
0.615	96.0%	0.84	0.756

Table 8: Human inter-annotator agreement and human–LLM alignment results on 500 queries.

Agreement Metrics. We report a set of standard metrics to evaluate both human annotator consistency and human–LLM alignment.

For human inter-annotator agreement, we use Krippendorff's α and Majority Agreement. Krippendorff's α measures chance-corrected agreement among multiple annotators and is suitable for nominal labels. In our setting, the three comparison outcomes (*A wins*, *Tie*, *B wins*) are treated as nominal categories. Majority agreement measures the percentage of queries for which at least two of the three annotators assign the same label, indicating whether a clear dominant human preference exists and supporting the use of majority voting for label aggregation.

To assess the reliability of the LLM evaluator (Gemini-3-Pro-Preview), we compare its judgments with the aggregated human labels using Accuracy and Cohen's κ. Accuracy measures the proportion of exact matches between the LLM prediction and the human consensus, while Cohen's κ provides a chance-corrected estimate of agreement between the two.

Results. Table 8 summarizes both human inter-annotator agreement and human–LLM alignment results. The chance-corrected agreement among human annotators, as measured by Krippendorff's $\alpha = 0.615$, indicates a moderate level of consistency. This suggests that while there is some variability in individual annotations, a reasonable level of agreement is present. The majority agreement, at 96.0%, demonstrates that for most queries, at least two of the three annotators assigned the same label, reflecting a strong overall consensus. Moreover, the LLM evaluator shows strong alignment with aggregated human judgments, with an accuracy of 0.84 and a Cohen's κ of 0.756. These metrics indicate that the LLM evaluator closely matches human judgments, capturing the dominant expert preferences with a high degree of reliability under the same evaluation protocol. Overall, the results sug-

No.	SFT		RL			Average
	Used	Variant	Used	Retire	Rollback	
1	✓	✓	✓	✓	✓	0.7893
2	✓	✓	×	×	×	0.7593
3	✓	×	✓	✓	✓	0.7651
4	✓	×	×	×	×	0.7484
5	×	×	×	×	×	0.7220

Table 9: Ablation studies on LawBench. A ✓indicates that the component is enabled, while × indicates that it is disabled. Average denotes the mean of all metrics in the range [0, 1].

No.	SFT		RL			Average	Sara_N
	Used	Var	Used	Ret	Roll		
1	✓	✓	✓	✓	✓	0.826	1.753
2	✓	✓	×	×	×	0.777	3.949
3	✓	×	✓	✓	✓	0.787	5.906
4	✓	×	×	×	×	0.759	7.231
5	×	×	×	×	×	0.735	8.093

Table 10: Ablation studies on LegalBench. A ✓indicates that the component is enabled, and × indicates that it is disabled. Abbreviations: Var = Variant, Ret = Retire, Roll = Rollback. Average denotes the mean of all accuracy metrics. Sara_N corresponds to the Log MSE metric and is therefore reported separately.

gest that the LLM evaluator can effectively serve as an automatic judge for this task, with substantial agreement with human annotators and a high degree of accuracy in predicting the majority label.

A.8 Ablation Studies on LawBench and LegalBench

Table 9 and Table 10 report the ablation results on the LawBench and LegalBench datasets, respectively, using Qwen3-8B as the backbone model.

A.9 Training Dataset Statistics

A.9.1 Statutory Law Dataset

Table 11 provides detailed statistics of the statutory law training dataset.

A.9.2 Case Law Dataset

Table 12 provides detailed statistics of the case law training dataset.

A.10 Detailed Definitions of Evaluation Metrics

A.10.1 ChJGBench Evaluation Metrics

We list our specific evaluation tasks and their corresponding evaluation metrics (Feng et al., 2022; Xiao et al., 2018) in Table 13. Depending on the

Item	Count
Training instances ($\{Q, T, A\}$)	2,734,100
Distinct statutes (clause level)	35,010
Causes of action	>400
Criminal cases	581,579
Civil cases	1,625,584
Administrative cases	526,937
Average token count in Reasoning Traces	3922.4
Average token count in Answer	902.2

Table 11: Statistics of the Statutory Law Training Dataset, including the total number of training instances, distinct statutes at the clause level, and the distribution of legal cases. It also provides the average token count in reasoning traces and answers.

Item	Count
Training instances ($\{Q, T, A\}$)	35,965
Average token count in Reasoning Traces	2369.2
Average token count in Answer	3.5

Table 12: Statistics of the Case Law Training Dataset, including the total number of training instances and the average token counts in reasoning traces and answers.

nature of the task, we employ different evaluation criteria, which are detailed below.

For classification-based tasks, we adopt the **F1 score**, defined as the harmonic mean of precision and recall:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

For amount-related judicial decision tasks, we instead employ **NAD-recall** (Normalized Absolute Deviation Recall) to evaluate the accuracy of numerical predictions. Given a tolerance threshold τ, a prediction $\hat{y}$ is considered correct if its normalized absolute deviation from the ground-truth value y satisfies:

$$\frac{|\hat{y} - y|}{y} \leq \tau \tag{19}$$

NAD-recall is then computed as the proportion of samples whose predictions fall within this acceptable deviation range:

$$\text{NAD-recall} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(\frac{|\hat{y}_i - y_i|}{y_i} \leq \tau \right) \tag{20}$$

where N denotes the total number of samples and $\mathbb{I}(\cdot)$ is the indicator function.

In the main text, the **AVERAGE** metric reported in Tables 2 and 5 is computed as a *sample-size weighted* average rather than a simple arithmetic mean. For each metric (or metric group) indexed

by k, let n_k be the number of samples it covers and N be the total number of samples. We define the weight as

$$w_k = \frac{n_k}{N} \tag{21}$$

We further define the domain set $\mathcal{D} = \{\text{Criminal, Civil, Administrative}\}$.

For each domain $d \in \mathcal{D}$, the domain score is computed as the mean of its corresponding sub-metrics:

$$\bar{s}_d = \frac{1}{|M_d|} \sum_{m \in M_d} s_m \tag{22}$$

where M_d denotes the set of metrics belonging to domain d.

The final AVERAGE score is calculated as a weighted combination of the general metrics and the domain-level scores. The general metrics (**Law**, **Reason**, and **Focus**) are defined for all cases regardless of the litigation domain.

$$S_{\text{gen}} = \frac{\sum_{m \in \{\text{Law, Reason, Focus}\}} w_m s_m}{\sum_{m \in \{\text{Law, Reason, Focus}\}} w_m} \tag{23}$$

$$\text{AVG} = \frac{S_{\text{gen}} + \sum_{d \in \mathcal{D}} w_d \bar{s}_d}{1 + \sum_{d \in \mathcal{D}} w_d} \tag{24}$$

A.10.2 LawBench Evaluation Metrics

We select three representative Legal Judgment Prediction tasks from **LawBench** for evaluation. For classification-based tasks, namely Article Prediction and Charge Prediction, we adopt the **F1 score** as the evaluation metric, following Equation 18.

For the Sentence Prediction task, we use **Normalized Log-Distance (NLD)** as the evaluation metric, defined as:

$$\text{NLD} = 1 - \frac{\left|\log(y_{\text{pred}} + 1) - \log(y_{\text{true}} + 1)\right|}{\log(y_{\max} + 1)} \tag{25}$$

where y_{pred} denotes the predicted sentence length, y_{true} denotes the ground-truth sentence length, and $y_{\max}$ is the maximum allowable value in the task.

A.10.3 LegalBench Evaluation Metrics

For classification-based tasks in **LegalBench**, we adopt the **Accuracy (Acc)** metric to measure prediction correctness. Accuracy is defined as the proportion of correctly predicted samples among all evaluated instances:

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i) \tag{26}$$

where N denotes the total number of samples, $\hat{y}_i$ is the predicted label, y_i is the ground-truth label, and $\mathbb{I}(\cdot)$ is the indicator function that returns 1 when the prediction matches the ground truth and 0 otherwise.

For numerical reasoning tasks such as **Sara-N**, we employ **Log Mean Squared Error (Log MSE)** to evaluate prediction accuracy on continuous numerical outputs. Log MSE measures the squared difference between the logarithms of predicted and ground-truth values, which helps stabilize variance when values span multiple orders of magnitude:

$$\text{LogMSE} = \frac{1}{N} \sum_{i=1}^{N} (\log(\hat{y}_i) - \log(y_i))^2 \tag{27}$$

where N denotes the number of samples, $\hat{y}_i$ is the predicted numerical value, and y_i is the corresponding ground-truth value.

A.11 Implementation Details

We adopt a two-stage training strategy. Training is conducted on 32 NVIDIA A100 GPUs with 80 GB of memory each.

In the first stage, we perform SFT on the Qwen3-8B and Llama3.1-8B base models with a learning rate of 1×10^{-5} and a sequence length cutoff of 16,384. The per-device training batch size is set to 1, with 2 gradient accumulation steps. We use a cosine learning rate scheduler with a warmup ratio of 0.1. The balancing coefficient is set to $\lambda = 0.5$.

In the second stage, we apply reinforcement learning with a learning rate of 1×10^{-6}, a global training batch size of 256, and a micro batch size of 128. In LABPO, the threshold ϵ, the window size H, and the evaluation interval n, i.e., evaluating the current policy π_{θ_τ} every n optimization steps, are used to detect stagnation in marginal gains. Conceptually, these parameters function similarly to early-stopping-style criteria and therefore do not directly alter the gradient directions or the underlying optimization objective. Instead, they control when the rollback mechanism is triggered. In our experiments, we find that $\epsilon \geq 0.0005$, $H \geq 4$, and $n = 100$ are reasonable configurations. Notably, n and H should be considered jointly. When n is small, using a small H is generally undesirable, since reinforcement learning is inherently prone to oscillations and often requires several optimization steps before performance improvements become observable. If computational resources permit, one can reduce n while moderately increasing H. This

Task	Task Full Name	Litigation Type	Benchmark	Metric
	Statutory Law Tasks			
Law	Statutory Provision Application	General	ChJGBench	F1 score
ISS	Disputed Issues Identification	General	ChJGBench	F1 score
JUS	Judicial Justification	General	ChJGBench	F1 score
Charge	Criminal Charge Prediction	Criminal	ChJGBench	F1 score
Fine_R	Monetary Fine Recall	Criminal	ChJGBench	NAD_Recall
REC	Recidivism Assessment	Criminal	ChJGBench	F1 score
MIT	Mitigating Circumstances Identification	Criminal	ChJGBench	F1 score
Pen_M	Penalty Duration Prediction	Criminal	ChJGBench	NAD_Recall
SUP	Civil Claim Support Determination	Civil	ChJGBench	F1 score
PAY_R	Payment Obligation Recall	Civil	ChJGBench	NAD_Recall
DIS	Administrative Disposition Prediction	Administrative	ChJGBench	F1 score
AP	Article Prediction	Criminal	LawBench	F1 score
CP	Charge Prediction	Criminal	LawBench	F1 score
SP	Sentence Prediction	Criminal	LawBench	NLD
	Case Law Tasks			
LRC	Legal Reasoning Causality	-	LegalBench	Accuracy
NYSJE	New York State Judicial Ethics	-	LegalBench	Accuracy
Sara-N	SARA Numeric Reasoning	-	LegalBench	Log MSE

Table 13: Summary of evaluation tasks, including statutory law tasks and case law tasks, with their corresponding litigation types, benchmark sources, and evaluation metrics.

allows the algorithm to more reliably identify the best-performing checkpoint and roll back to it when necessary.

The development set is used strictly for model selection, such as determining reward retirement or rollback decisions, and is never involved in gradient updates or in the optimization of the policy or reward models. Although it is queried multiple times during training to guide the training trajectory, its role is limited to checkpoint selection based on performance. To reduce potential sensitivity to specific instances, we randomly sample a subset from the full dataset to construct a diverse and representative development set. Crucially, the training, development, and test sets are strictly disjoint, and the final reported results are evaluated on a completely held-out test set that is never accessed during training or model selection.

A.12 Computational Overhead Analysis

LABPO introduces additional computation compared with standard GRPO because it periodically evaluates the current policy on a held-out development set and may roll back the policy when an auxiliary signal becomes ineffective. To quantify this cost, we conduct a fixed-hyperparameter overhead analysis using Qwen3-8B initialized from the JurisCoT-SFT checkpoint. We compare standard GRPO and LABPO under the same training data, hardware environment, batch size, rollout configuration, maximum generation length, and target optimization budget. For LABPO, the scheduling hyperparameters are fixed as $n = 100$, $H = 4$, and $\epsilon = 0.0005$. Both methods are trained to the same effective target optimization step $N = 3000$.

Let t_{step} denote the average time of one optimization step, and t_{eval} denote the time required for one development-set evaluation. Since LABPO evaluates the current policy every $n = 100$ steps, each rollback discards $Hn = 400$ optimization steps. In our implementation, we use six auxiliary reward functions. Therefore, in the worst case, up to six rollbacks may occur as training proceeds, introducing up to 2400 additional optimization steps. To reach the same effective target step $N = 3000$ as GRPO, LABPO may execute up to 5400 optimization steps in total.

The computational cost of standard GRPO is:

$$C_{\text{GRPO}} = N t_{\text{step}}. \tag{28}$$

The worst-case computational cost of LABPO can be estimated as:

$$C_{\text{LABPO}} \leq (N + mHn) t_{\text{step}} + \frac{N + mHn}{n} t_{\text{eval}} + C_{\text{io}}, \tag{29}$$

where $m = 6$ denotes the number of auxiliary rewards, and C_{io} denotes the checkpoint saving/loading and state restoration cost caused by rollback. Substituting $N = 3000$, $m = 6$, $H = 4$, and $n = 100$, we obtain:

$$C_{\text{LABPO}} \leq 5400 t_{\text{step}} + 54 t_{\text{eval}} + C_{\text{io}}. \tag{30}$$

The relative overhead of LABPO over GRPO is computed as:

$$\text{Overhead} = \frac{C_{\text{LABPO}} - C_{\text{GRPO}}}{C_{\text{GRPO}}}. \tag{31}$$

With our hyperparameter values, this overhead is upper-bounded by:

$$\text{Overhead} \leq 0.8 + 0.018\frac{t_{\text{eval}}}{t_{\text{step}}} + \frac{C_{\text{io}}}{3000 t_{\text{step}}}. \tag{32}$$

Thus, under the worst-case assumption that each auxiliary reward triggers one rollback, the rollback-induced overhead is upper-bounded by 80%. When periodic development-set evaluation is further considered, the total overhead additionally depends on the relative cost of development-set evaluation compared with a single optimization step. Excluding checkpoint I/O, LABPO costs approximately $1.8\times$ standard GRPO plus the additional cost of periodic development-set evaluation. The additional cost mainly comes from periodic development-set evaluation and the re-execution of rolled-back optimization steps.

This overhead should be interpreted as the computational cost of adaptive reward scheduling. LABPO does require additional optimization and evaluation time to filter out ineffective auxiliary rewards and recover from unstable training trajectories. However, this extra computation also reduces the manual burden of reward engineering: instead of carefully pre-designing, tuning, and manually combining reward functions in advance, researchers can introduce even coarse-grained or potentially unstable auxiliary reward signals, while LABPO dynamically evaluates their utility during training and filters out detrimental ones.

B Case study

To analyze whether the model maintains high accuracy in reasoning traces and final answers when handling complex reasoning tasks, we selected representative examples from Figure 6 for qualitative analysis. This case originates from a typical criminal case of illegal business operations, involving multiple sub-tasks such as charge identification, sentencing discretion, application of legal provisions, and fine calculation. We will compare the reasoning traces of the model trained using the algorithm proposed in this paper with those of two open-source models, Qwen3-8B and DeepSeek-R1, across three dimensions: "Charge and Sentence," "Law Articles," and "Fine."

The case shows that our model accurately identified key facts, such as the defendant's voluntary surrender, low social harm, and eligibility for probation, with only a minor error of a 2-month discrepancy in the predicted prison sentence (6 months vs. the reference of 8 months). However, it missed one legal provision on the probation supervision mechanism (Article 76 of the Criminal Law). In contrast, both Qwen3-8B and DeepSeek-R1 failed to recognize the possibility of probation, directly predicting prison sentences of 12 and 18 months, respectively. Additionally, both models missed legal provisions related to probation (such as Articles 72, 73, and 76). This incomplete coverage of legal knowledge led to an incomplete sentencing reasoning chain, with the models defaulting to a prison sentence due to the absence of probation judgment. This demonstrates that the completeness of legal provision recall is essential for ensuring the reasonableness of sentencing.

Regarding fine generation, Qwen3-8B predicts a fine of 10,000 CNY and DeepSeek-R1 predicts 50,000 CNY, both deviating substantially from the reference fine of 20,000 CNY. In contrast, our model produces a more accurate amount. Notably, before generating the final answer, our model explicitly cites relevant legal provisions governing fine determination, although these provisions are not included in the reference answer. This suggests that reliable legal numerical decision-making depends not only on strong multi-step reasoning, but also on the integration of domain-specific legal knowledge or structural constraints.

In summary, this case illustrates common limitations of general-purpose LLMs in judicial reasoning, such as inadequate sensitivity to procedural and conditional legal rules and weak modeling of legal numerical logic. The proposed training method mitigates these issues, improving judgment element accuracy while maintaining interpretable reasoning traces.

Query		Defendant **, from June to September 2016, illegally sold gasoline without the required retail and hazardous chemicals licenses. He procured the gasoline, and co-conspirators sold it at 4.5 CNY per liter by truck to drivers on the roadside, totaling 83,523 CNY. In September 2016, they were caught during a sale with 2.98 tons of gasoline (worth about 18,000 CNY). ** voluntarily surrendered in June 2017, confessed to the crime, and agreed to a guilty plea. What should the judgment be, and which laws should be referenced?			
Responser		**Ground truth**	**Qwen3-8B**	**DeepSeek-R1**	**Ours**
Key Points of Reasoning Traces	**Charge and Sentence**	Illegal business operations 8 months prison 1-year probation	**Illegal business operations** ✅ **1 year prison** ❌	**Illegal business operations** ✅ **1 year 6 months prison** ❌	**Illegal business operations** ✅ **6 months prison** ⚠️ **(reasonable error)** **1-year probation** ✅
	Law articles	Articles 225, 67(1), 72, 73, and 76 of the Criminal Law.	Articles **225**, **67**, ✅ and **64** ❌ of the Criminal Law. ⚠️ **Omission of Articles 72, 73, and 76 of the Criminal Law**	The provisions for illegal business operations are in Article **225** ✅ of the Criminal Law... The provisions regarding voluntary surrender are in Article **67(1)** ✅ of the Criminal Law. ⚠️ **Omission of Articles 72, 73, and 76 of the Criminal Law**	According to Article **225** ✅ of the Criminal Law... The provisions on voluntary surrender are found in Article **67(1)** ✅ of the Criminal Law... The basis for imposing fines is Article **52** 🏆 **(Supplemented the legal provisions regarding fines)** of the Criminal Law... Probation may be applicable, so the conditions for applying probation are governed by Articles **72** ✅ and **73** ✅ of the Criminal Law. ⚠️ **Omission of Articles 76 of the Criminal Law**
	Fine	A fine of 20,000 CNY.	A fine of **50,000 CNY** ❌; recovery of illegal gains amounting to 83,523 CNY.	Illegal business operations usually involve a fine, which may be based on the business volume, such as one or multiple times that amount. However, the specific amount is uncertain and needs to be reasonably set, estimated at **10,000 CNY** ❌.	The basis for imposing fines is Article 52 of the Criminal Law of the People's Republic of China …… Therefore, the defendant is fined **20,000 CNY** ✅.

Figure 6: Case Study: Legal Reasoning Accuracy and Completeness in Illegal Business Operations Judgment.