Improving Legal Judgment Prediction via Quantitative Reasoning

ANONOMYOUS SUBMISSION

Legal Judgment Prediction (LJP) focuses on predicting judgment results based on the facts of cases. While state-of-the-art (SOTA) methods have shown impressive performance in law article prediction and charge prediction, they still exhibit weaknesses in prison term prediction. One major reason is that existing models fail to mimic human legal quantitative reasoning to understand monetary features in case facts. Consequently, they do not rigorously quantify the severity of the crime, which is essential for prison term prediction. In this paper, we explore and explain how to leverage monetary features to improve LJP via quantitative reasoning. Specifically, we propose QR-LJP, a quantitative reasoning-based LJP model, to integrate legal reasoning knowledge into the prediction process. QR-LJP first employs a curated LLM to extract monetary values from case facts and uses legal quantitative reasoning logic to determine the total crime amount, serving as the quantitative measure of the crime's severity. This measure is subsequently used to make judgment predictions. We evaluate our model on the real-world dataset CAIL-2018. Experimental results demonstrate that our model outperforms current SOTAs, highlighting the effectiveness of legal quantitative reasoning. Moreover, applying our quantitative reasoning strategy to existing SOTA methods yields significant improvements, especially in macro-F1 scores.

CCS Concepts: • **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Additional Key Words and Phrases: Legal Judgment Prediction, AI for Law, Large Language Models

ACM Reference Format:
Anonomyous Submission. 2018. Improving Legal Judgment Prediction via Quantitative Reasoning. *J. ACM* 37, 4, Article 111 (August 2018), 41 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Legal **J**udgment **P**rediction (LJP) focuses on predicting judgment outcomes based on the factual descriptions of cases. This area has attracted considerable research interest because of its practical value in assisting legal professionals in case analysis and providing consultation services to the public, while reducing legal costs and enhancing access to justice. The scope of LJP subtasks varies significantly across different jurisdictions. In jurisdictions that adhere to the common law system, such as the United States and India, LJP primarily focuses on predicting court decisions, i.e., determining whether claims will be accepted or rejected [24, 39]. In jurisdictions rooted in civil law systems, LJP involves aligning factual descriptions with statutory law. For instance, LJP in the European Union mainly targets at predicting applicable law articles [6, 8, 44]. Similarly, Chinese LJP aims at predicting charges, relevant law articles, and prison terms [10, 15, 61].

Author's Contact Information: Anonomyous Submission.

ACM 1557-735X/2018/8-ART111
https://doi.org/XXXXXXX.XXXXXXX

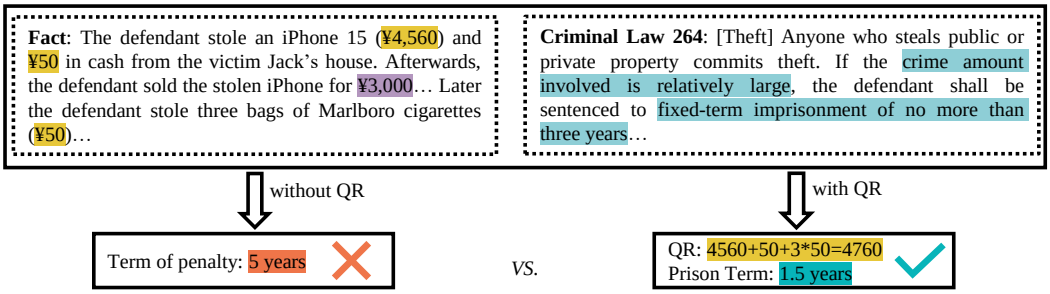


Fig. 1. An example of Quantitative Reasoning (QR) in prison term prediction. The highlighted spans in the fact description are monetary values while the highlighted span in the law article outlines the prison terms and their corresponding definitions of crime severity.

While LJP subtasks vary across different legal jurisdictions, current state-of-the-art (SOTA) approaches are all learning-based. These approaches generally frame LJP subtasks as either classification or generation tasks. The prediction targets are cast as labels in the classification paradigm whereas treated as textual outputs from the models in the generation paradigm. Existing learning-based LJP works can be divided into two categories. The first relies on feature engineering to train classifiers using algorithms such as Support Vector Machines [1, 40, 48, 49] or Random Forests [24]. The second category employs neural network-based methods that either utilize novel model structures such as BERT [7], Lawformer [59], and LLMs [54] to enhance semantics extraction from case facts, or leverage legal knowledge, such as law article text and task dependencies, to boost LJP performance [12, 36]. These methods have shown impressive results in binary classification tasks like court decision prediction and multi-class tasks like law article and charge prediction. However, they still struggle with poor performance in predicting prison terms [14].

Compared to law articles and charges, predicting prison terms involves extensive legal quantitative reasoning, especially in cases involving numerical calculations (such as thefts and robberies). Figure 1 shows an example of how legal quantitative reasoning is applied in predicting prison terms, using a theft case as an illustration. In the real world, legal professionals first analyze fact descriptions and understand key semantics involved, especially monetary features (such as the *¥4,560* measuring the price of the stolen item, the iPhone 15). Then they determine the severity of the crime based on the monetary features and make a final judgment following legal quantitative reasoning. Here, the total amount of the crime after legal quantitative reasoning (i.e., *4560+50+3*50=4760*) is *relatively large* according to the *Criminal Law 264*. And the final predicted prison term is *1.5 years*. This process is non-trivial as it involves a sophisticated sequence of quantitative reasoning. First, it is essential to analyze and extract monetary features, as not all monetary values have an impact on the total crime amount even if they are mentioned in the fact. Determining which values are relevant to the total crime amount requires reasoning along with specific legal knowledge. As shown in Figure 1, the price of the iPhone 15, i.e., *¥4,560*, is included in the total crime amount while the corresponding resale price, i.e., *¥3,000* is excluded. Here, the assessed value of the stolen property is its original value, not its resale value according to specific legal regulations. Second, after obtaining individual monetary values, determining the total crime amount not only involves arithmetic reasoning but also semantic reasoning. In Figure 1, three bags of cigarettes are stolen, with each bag valued at *¥50*. We need to convert semantics (*three*) into numerical values (*3*) and performing calculations (*3*50*). The total crime amount results from complex arithmetic reasoning: *4560+50+3*50=4760*. Finally, once the total crime amount is determined, assessing the

severity of the crime and making judgments involve aligning the total amount with the specific reasoning logic outlined in legal statutes. In Figure 1, the total amount *¥4,760* aligns with the *relatively large* stipulated in Criminal Article 264, and *no more than three years* should be referenced when predicting prison terms. Note the total amount *¥4,760* is regarded as a *relatively large* sum according to specific legal regulations. However, existing LJP works overlook the importance of legal quantitative reasoning illustrated above. These works mainly rely on the semantics of facts, neglecting to identify monetary features that correlate with the severity of crimes. As a result, they fail to evaluate the severity of the crime, hindering their ability to establish a robust correlation between the crime's severity and the corresponding prison term.

In this paper, we seek to enhance the legal quantitative reasoning capabilities of LJP models by leveraging monetary features, thereby improving LJP performance. Specifically, We aim to solve the three challenges mentioned before to inject legal quantitative reasoning into LJP models: (1) extracting monetary values that have impacts on the total crime amount, (2) employing these values to perform both semantic and arithmetic reasoning to calculate the total crime amount, and (3) aligning the total crime amount with specific legal knowledge to improve LJP. To tackle these challenges, we present a novel LJP framework that automatically extracts monetary values, quantifies the severity of crimes via legal quantitative reasoning, and consequently improves LJP performance. Specifically, we employ **Large Language Models** (LLMs) [5, 9, 46, 52] to extract monetary values and calculate the total crime amount. Calculating the total crime amount can be viewed as an arithmetic word problem. Given the complexity of the issue, the advanced language processing and reasoning abilities of LLMs make them an ideal fit for determining the total crime amount. We employ LLMs to perform quantitative reasoning, allowing them to replicate the expertise of legal professionals. Then, the calculated total crime amount is integrated into our model and utilized to align with the relevant law article texts. These texts then provide extra legal knowledge for predicting prison terms. We mainly focus on Chinese LJP.[1]

In summary, our contributions are three-fold:

- First, we identify and tackle the challenge of legal quantitative reasoning in LJP, especially in cases that involve monetary features (such as thefts and robberies).
- Second, we introduce a novel legal quantitative reasoning method to mimic human reasoning in LJP, where we improve LJP performance by utilizing monetary features to assess the severity of crimes. This method integrates quantitative reasoning into the model and aligns the total crime amount with legal texts to incorporate legal knowledge.
- Finally, experimental results demonstrate that our method outperforms SOTA methods, especially in prison term prediction. In terms of Macro-F1 scores, our method obtains a 4.17% improvement on a real-world dataset. Additionally, applying our legal quantitative reasoning strategy to SOTA methods results in further improvements. These results underscore the effectiveness of incorporating legal quantitative reasoning into LJP.

2 Related Work

In this section, we conduct a review of the relevant literature on two key areas: legal judgment prediction (LJP) and arithmetic word problem (AWP) solving.

[1]Jurisdictions are distinct from each other, leading to variations in legal quantitative reasoning methods. This paper focuses on Chinese LJP and aims to explore how to integrate legal quantitative reasoning into models and to potentially inspire solutions in other jurisdictions. Note that quantitative reasoning is significant across different legal systems.

2.1 Legal Judgment Prediction

Legal judgment prediction focuses on automatically predicting judgment results of legal cases given the fact descriptions of cases, which have been an interest of research for decades. In the early stages, researchers employ rule-based approaches to predict the judges' votes [27, 43, 47]. However, during this period, the model does not make predictions based on fact descriptions. Instead, it relies on non-textual information such as the nature and severity of the crime, as well as the judges' preferred political positions. Subsequently, researchers have adopted a feature-based learning approach to LJP [1, 24, 40, 48, 49]. In this approach, off-the-shelf algorithms such as Support Vector Machines [1, 40, 48, 49] or Random Forests [24] are employed to train classifiers. The training instances include textual inputs represented by lexical features, specifically the Bag of Word (BoW) model and n-grams extracted from factual descriptions [1, 40, 48, 49], and/or external features derived through feature engineering [24].

In recent years, approaches based on neural networks have become popular for addressing subtasks of LJP [31, 56]. These methods typically utilize pre-trained word embeddings such as Word2Vec [41] and GloVe [45] to represent textual words, which capture semantic relationships between natural language tokens more effectively than traditional BoW or n-gram models. Furthermore, some methods exploit the inherent dependencies among LJP subtasks to enhance performance [15, 61, 63, 65]. For example, there are sequential dependencies among subtasks such as charge prediction, law article prediction, and prison term prediction. A specific law article might dictate certain prison terms in accordance with the varying gravity of a crime when violated. These dependencies mirror the realistic judicial process and act as constraints, thereby improving the performance of models on LJP subtasks. Another effective practice involves integrating label semantics into the model, rather than treating them as mere atomic identifiers. In LJP subtasks, labels typically contain textual content from charge definitions and descriptions of law articles. Exploiting these semantics to match with fact descriptions can significantly enhance results. Noteworthy contributions to this endeavor include Yue et al. [63] and Ge et al. [20].

Another branch of neural network-based approaches involves utilizing pre-trained language models (PLMs). Given the significant differences in writing style and vocabulary between legal texts and general texts, directly adapting general domain PLMs like BERT to legal domain tasks often proves suboptimal [7]. To address this, domain-specific PLMs have been developed. Legal-BERT [7], for instance, is a BERT-based language model pre-trained on English legal texts, while Lawformer [59], a Longformer-based model, is pre-trained on Chinese legal texts. These tailored models are designed to better handle the nuances of legal language and can be fine-tuned for specific LJP subtasks.

The aforementioned approaches have achieved impressive results across various LJP benchmarks. However, they struggle to distinguish monetary features, which have significant impacts on prediction outcomes, from other aspects of fact descriptions. Addressing this, Gan et al. [17] introduces an approach that utilizes total crime amounts for prison term prediction, where they cast the calculation of crime amounts as a Named Entity Recognition (NER) task. A BERT-CRF model is trained to detect and compute relevant monetary values in fact descriptions. While it does enhance the accuracy of prison term predictions to a degree, the proposed method still faces several drawbacks. Specifically, it focuses exclusively on addition operations, ignoring other arithmetic operations. Moreover, the resulting crime amounts are simply concatenated to the input of the prison term classifier as an additional feature. In contrast, our method (1) encompasses a broader spectrum of quantitative reasoning, and (2) aligns total crime amounts with relevant law article texts to inject necessary legal knowledge. Our method significantly advances LJP performance via legal quantitative reasoning.

2.2 Arithmetic Word Problem Solving

The problem of calculating the total crime amount of a case based on the fact description can be regarded as an arithmetic word problem. An **Arithmetic Word Problem** (AWP) presents a mathematical challenge conveyed through written text, requiring individuals to apply mathematical skills to derive a solution. The development of an automatic AWP solver has been a focal point of research for several decades. Early works [3, 4, 16] depended on manually crafted rules and predefined schemas for pattern matching. These methods were heavily reliant on human intervention and could only solve problems within a narrow range of scenarios. Subsequently, as deep learning techniques became more popular, researchers began developing AWP solvers using data-driven approaches that require minimal human intervention. Notable contributions, such as DNS [57] and Seq2SeqET [55], have demonstrated significant success in resolving AWPs. In recent years, large language models have demonstrated impressive abilities in language comprehension and semantic parsing, both critical attributes for solving AWPs. This has inspired a new line of research focused on adapting LLMs specifically to address AWPs [25, 33, 42]. In this paradigm, a common practice is to fine-tune the backbone LLMs [11, 18, 29, 35] using general domain AWP datasets [26, 28, 53]. However, when the fine-tuning dataset comprises solely general domain data, the resulting models may not always provide optimal solutions for problems phrased within a specific domain. To address this issue, some researchers suggest enhancing the domain-specific reasoning capabilities of LLMs by fine-tuning them on datasets tailored to the specific field. TAGOP [66] and MT2Net [64] are such AWP-solving LLMs specializing in finance. The legal domain, which underpins the challenge of calculating total crime amounts, is also highly specialized. The terminology used in legal texts is often unique to the field, and legal knowledge is not typically encompassed within general domain data. By tailoring an LLM with legal-themed data, we have curated a model that specializes in solving legal AWPs, particularly in calculating the total crime amount.

Beyond generic AWP solvers, recent work proposes tool/program-augmented LLM pipelines that compute intermediate arithmetic steps via code execution or structured reasoning (e.g., program-aided reasoning and verifier–editor solvers such as PAL [19], MathPrompter [23], and CoMAT [30]). These systems are attractive candidates for computing the total crime amount *independently* of LJP. However, they are optimized for *general* quantitative reasoning rather than statute-specific aggregation: legal TCA requires handling exclusions (e.g., resale/returned sums), unit conversions, offsets/refunds, and exception rules tied to the realistic legal rules. Hence, there is a demanding need for a domain-calibrated total crime amount calculation solution.

3 Methods

In this section, we offer a detailed explanation of our method. We begin by defining LJP, followed by a detailed discussion of our model, QR-LJP. QR-LJP utilizes monetary values to conduct legal quantitative reasoning, thereby enhancing LJP performance.

3.1 Problem Definition

In this section, we demonstrate the mathematical notations in our method and formulate the LJP tasks.

Table 1 lists the notations and definitions of the input of our method. Given a case's fact description D^f and relevant legal knowledge (i.e., the textual descriptions of charges and law articles), LJP aims at predicting the judgment results including the charge, the law article, and the prison term, denoted as $\hat{c}$, $\hat{l}$, and $\hat{t}$ respectively. We cast LJP as a classification task, and three classifiers are

Table 1. The mathematical notations and corresponding definitions of the input of our method.

Notation	Definition
$D^f = \{w_1^f, \ldots, w_{fn}^f\}$	the word sequence of a fact description
$D^c = \{w_1^c, \ldots, w_{cn}^c\}$	the word sequence of a charge description
$D^l = \{w_1^l, \ldots, w_{ln}^l\}$	the word sequence of a law description
$C = \{c_1, \ldots, c_p\}$	the set of charge labels
$L = \{l_1, \ldots, l_q\}$	the set of law article labels
$T = \{t_1, \ldots, t_r\}$	the set of prison term labels
$\mathcal{K}$	legal knowledge

constructed:

$$\hat{c} = \xi_c(D^f, \mathcal{K}) \tag{1}$$

$$\hat{l} = \xi_l(D^f, \mathcal{K}) \tag{2}$$

$$\hat{t} = \xi_t(D^f, \mathcal{K}). \tag{3}$$

In the above equations, ξ_c, ξ_l, and ξ_t represent the classifiers for charge prediction, law article prediction, and prison term prediction. The symbol $\mathcal{K}$ represents legal domain knowledge. In the context of the method in this paper, $\mathcal{K}$ specifically represents the textual descriptions of charges, D^c, and law articles, D^l.

3.2 Approach Overview

In the realistic judging process, legal professionals make judgments based on cases' fact descriptions and the severity of the crime. This severity is determined via quantitative reasoning over monetary factors, especially the total crime amount involved in the case. The **Total Crime Amount** (TCA) refers to the total monetary value involved in the crime committed by the criminal in the case. To simulate the realistic judging process within an LJP framework, three major steps are essential: (1) identifying and extracting monetary values that contribute to the TCA, (2) utilizing these values to compute the TCA following relevant reasoning rules, and (3) aligning the TCA with judgment logic outlined in legal knowledge. To address these requirements, we propose the QR-LJP framework. An illustration of the QR-LJP framework is shown in Figure 2a. Our framework consists of three key components: the TCA calculation module, the pre-trained number encoder, and the judgment prediction module. First, the TCA calculation module is used to extract relevant monetary values for the TCA and to calculate the TCA via quantitative reasoning. Then, we employ the pre-trained number encoder to encode the TCA into embeddings that effectively represent its numerical features. Lastly, we align the TCA with specific legal knowledge to determine the final judgment results in the judgment prediction module. It is worth noting that we intentionally decouple these three subproblems because this design allows each component to focus on a distinct aspect of the problem, i.e., quantitative computation, numeric representation, and hierarchical prediction, making the framework more interpretable, extensible, and effective in capturing both legal logic and numeric nuances.

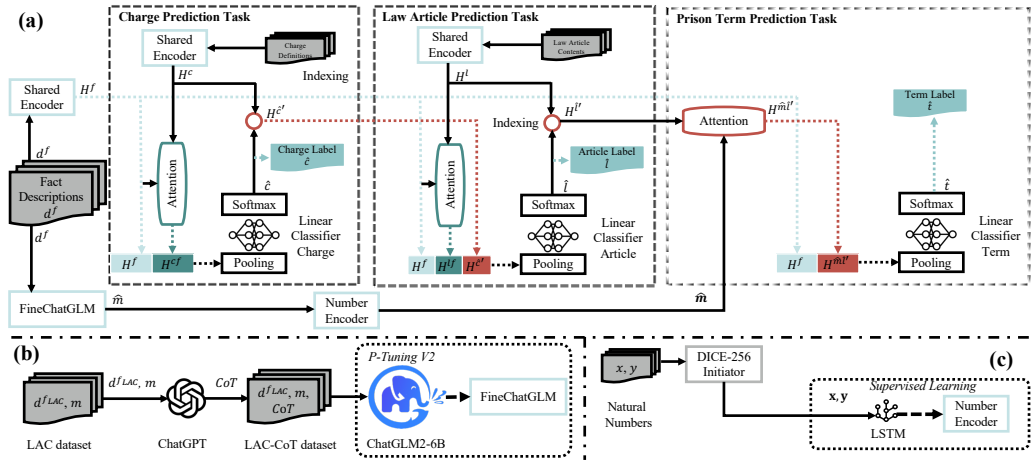


Fig. 2. (a) The overview of integrating cases' TCAs into sequentially task-dependent LJP module. (b) The prompt-tuning process of the crime amount calculation module, FineChatGLM. (c) The pre-training of our number encoder.

3.3 Total Crime Amount Calculation

We treat TCA calculation as an arithmetic word problem, where we inject detailed reasoning rules and critical legal knowledge. Specifically, amounts that are returned or compensated should not be deducted from the total value and amounts from resale should not be included. As illustrated in Figure 1, the resale amount of *¥3,000* is therefore not added to the final TCA. This necessitates that our TCA calculation module effectively identifies which monetary values in the fact description are relevant and which are not. These stipulations, along with the need for advanced language and reasoning abilities to accurately interpret textual descriptions and perform the correct calculation to arrive at the final answer, highlight the task's complexity. Given these challenges, we employ LLMs for their robust language processing and reasoning capabilities, which are crucial for navigating the complexities of this scenario. However, directly using off-the-shelf LLMs poses significant challenges. The calculation of TCA is a complex task within the legal field, characterized by specialized terminologies and specific knowledge that are often absent from the training data of general LLMs [7]. Furthermore, directly employing pre-trained LLMs might lead to their inability to effectively recognize and understand the nuances specific to the TCA calculation task [2]. Additionally, using LLMs that provide API services like ChatGPT could expose sensitive data, which conflicts with the private and discrete nature of judicial data, such as personal information and evidence details.

To overcome these challenges, we propose adapting an open-source LLM specifically for the TCA calculation task, offering a tailored solution for the TCA calculation module. To prepare the dataset needed for the adaptation, we adopt the LAC [3] dataset which provides monetary value-related judicial cases with both the fact description and the TCA. An example of one data item in the LAC dataset is shown in Figure 3. However, the LAC dataset only provides the ground truth TCAs, with neither reasoning details on the process of identifying and extracting appropriate monetary values

[2]Experimental results are shown in Table 8.
[3]https://data.court.gov.cn/pages/laic2021.html

事实描述：…李XX把被害人装钱的篮子偷走…得手后…将盗窃所得款420元分给黄XX 266元，自己分得154元。… 确定盗窃目标后，李XX把被害人装钱的小胶桶偷走…盗窃所得款1018元分给黄XX440元，自己分的578元。李XX…被公安民警抓获，并缴获赃款人民币共1018元…公安民警已将缴获的1018元发还给被害人叶XX。
Fact Description: First, defendant Li and Huang, after identifying their theft target, Li stole the victim's money basket, splitting the stolen 420 yuan between Huang (266 yuan) and himself (154 yuan). Second, in another case, Li stole the victim's money-filled plastic bucket, dividing the 1018 yuan stolen, with Huang receiving 440 yuan and Li keeping 578 yuan. Li was arrested by the police, who recovered and returned the stolen amount of 1018 yuan to the victim, Ye.

犯罪总额：文中涉及的总金额为1438元。
Total Crime Amount: The total amount mentioned in the text is 1438 yuan.

Fig. 3. An example of the data item of the LAC dataset.

for the TCA nor the determination of the calculation logic. To address this issue, we augment the LAC in the form of the Chain-of-Thought (CoT) data [58]. CoT is a series of intermediate reasoning steps that can significantly improve the ability of LLMs to perform complex reasoning. This augmentation clarifies the reasoning processes that lead to the accurate determination of the ground truth TCA. Specifically, for a data item in the LAC dataset, we query ChatGPT [4] using its fact description $D^{f_{\text{LAC}}}$ and ground truth TCA m. We explicitly require the inclusion of analyses on the monetary values used, the calculation logic, and the formula that leads to the ground truth answer in the prompt. Additional details regarding the design and selection of the prompt used are described in Appendix A. Then, with the resulting data detailing the calculation of the TCA for each instance in the LAC dataset, we construct the LAC-CoT dataset. An example of the prompt tuning dataset, namely LAC-CoT, is shown in Figure 4a. The backbone LLM we have selected is the open-source ChatGLM2-6B model [13]. Additionally, to reduce the substantial training costs associated with fine-tuning an LLM, we employ a parameter-efficient approach, P-Tuning V2 [34], to adapt the ChatGLM2-6B model using the LAC-CoT dataset. P-Tuning V2 is a prompt-tuning method, which matches fine-tuning in terms of performance while having only 0.1% - 3% tuned parameters. Finally, after prompt tuning with P-Tuning V2, we have developed our TCA calculation model, **FineChatGLM**. It is important to note that as a locally deployed model, FineChatGLM effectively avoids the exposure of sensitive data [5][6].

FineChatGLM is able to calculate the TCAs of monetary value-related cases. However, not all case descriptions involve monetary features. To identify cases featuring monetary aspects, we employ a preciously designed regular expression, denoted as r, to examine the presence of monetary terminologies within each fact description D^f. For cases that lack monetary terminologies within their descriptions, the TCA, denoted as $\hat{m}$, is assigned the special token <NEG>. Conversely, for cases where monetary terms are present, $\hat{m}$ is determined by the output of FineChatGLM. Hence, the calculation of the TCA $\hat{m}$ can be represented as:

$$\hat{m} = \begin{cases} \textbf{FineChatGLM}(D^f) & \textbf{if } D^f \textbf{ matches } r, \\ \text{<NEG>} & \textbf{otherwise}. \end{cases} \tag{4}$$

[4]All instances of ChatGPT usage in this paper refer to the GPT-3.5-turbo-16k model.
[5]In our framework, we treat legal data for training and inference in our LJP model as sensitive, reflecting real-world application scenarios. Conversely, the small sample of legal data used to query ChatGPT for prompt-tuning enhancement is considered public, sourced from an open dataset, with no further ChatGPT queries needed post-tuning.
[6]For a focused comparison between our CoT generation protocol and representative math-CoT/solver alternatives (e.g., prompt-tuned CoT and schema-based CoT), including their TCA accuracies on LAIC, please see Appendix J.

Model Inputs

…李XX把被害人装钱的篮子偷走…得手后…将盗窃所得款420元分给黄XX 266元，自己分得154元。 … 确定盗窃目标后，李XX把被害人装钱的小胶桶偷走…盗窃所得款1018元分给黄XX440元，自己分的578元。李XX…被公安民警抓获，并缴获赃款人民币共1018元…公安民警已将缴获的1018元发还给被害人叶XX。
First, defendant Li and Huang, after identifying their theft target, Li stole the victim's money basket, splitting the stolen 420 yuan between Huang (266 yuan) and himself (154 yuan). Second, in another case, Li stole the victim's money-filled plastic bucket, dividing the 1018 yuan stolen, with Huang receiving 440 yuan and Li keeping 578 yuan. Li was arrested by the police, who recovered and returned the stolen amount of 1018 yuan to the victim, Ye.

Model Outputs

犯罪总额的计算逻辑是将每次犯罪行为涉及的金额相加，即第一次盗窃所得的420元加上第二次盗窃所得的1018元。这里不需要减去返还的金额，也不需要加入转卖的金额，只需要将每次犯罪行为涉及的金额相加即可。计算式为：420元（第一次盗窃所得）+ 1018元（第二次盗窃所得）= 1438元。本案的犯罪总额为1438元。文中涉及的总金额为1438元。
The total amount is calculated by summing the amounts from each theft, without subtracting returned money or including resale profits. Specifically, 420 yuan from the first theft plus 1018 yuan from the second theft equals 1438 yuan, making the case's total amount 1438 yuan. The calculation is as follows: 420 yuan (from the first theft) + 1018 yuan (from the second theft) = 1438 yuan. The total amount mentioned in the text is 1438 yuan.

(a) An example of LAC-CoT dataset with CoT data highlighted

Model Inputs

…李XX把被害人装钱的篮子偷走…得手后…将盗窃所得款420元分给黄XX 266元，自己分得154元。 … 确定盗窃目标后，李XX把被害人装钱的小胶桶偷走…盗窃所得款1018元分给黄XX440元，自己分的578元。李XX…被公安民警抓获，并缴获赃款人民币共1018元…公安民警已将缴获的1018元发还给被害人叶XX。
First, defendant Li and Huang, after identifying their theft target, Li stole the victim's money basket, splitting the stolen 420 yuan between Huang (266 yuan) and himself (154 yuan). Second, in another case, Li stole the victim's money-filled plastic bucket, dividing the 1018 yuan stolen, with Huang receiving 440 yuan and Li keeping 578 yuan. Li was arrested by the police, who recovered and returned the stolen amount of 1018 yuan to the victim, Ye.

Model Outputs

文中涉及的总金额为1438元。
The total amount mentioned in the text is 1438 yuan.

(b) An example of data items used in prompt tuning the ChatGLM2-6B model, which doesn't have CoT content.

Fig. 4. Examples of data items used in prompt tuning the ChatGLM2-6B model.

3.4 The Number Encoder

To integrate the calculated TCA into the LJP prediction model, we must obtain embeddings of the TCA values. Since the TCA acts as an indicator of crime severity within our framework, it is crucial that these embeddings retain their numerical characteristics, particularly their magnitude, which represents the numbers' actual values. DICE [50] is an algorithm designed to produce embeddings for numbers to reflect their numerical properties. For a pair of numbers, DICE is able to encode them in a way that the cosine distance of the embeddings reflects their actual distance on the number line. For the training of our number encoder, we first generate the initial number embeddings using DICE [7]. Then, for the pair of numbers (x, y) and their respective embeddings $\boldsymbol{x}$ and $\boldsymbol{y}$, our bidirectional LSTM number encoder is trained over:

$$\mathcal{L}_{\text{num}} = \left\| 2\frac{|x-y|}{|x|+|y|} - d_{\cos}(\boldsymbol{x}, \boldsymbol{y}) \right\|_2 \tag{5}$$

[7]After multiple attempts, DICE-256 gives the best performance.

where

$$d_{\cos}(\boldsymbol{x}, \boldsymbol{y}) = 1 - \frac{\boldsymbol{x}^T \boldsymbol{y}}{\|\boldsymbol{x}\|_2 \|\boldsymbol{y}\|_2} \tag{6}$$

The resulting number encoder's parameters will be frozen during the subsequent training of LJP subtasks.

3.5 Legal Judgment Prediction with Total Crime Amounts

3.5.1 Encoding Texts and Numerical Values. Given the fact description D^f, all the textual descriptions of charges and law articles, and the TCA $\hat{m}$, first, we encode all these raw texts and numbers into vector representations. Specifically, for the word sequence of fact description D^f, we adopt Lawformer [60] as the legal text encoder to encode D^f, whose outputs are the contextual representations for each token. The process can be described as:

$$\boldsymbol{H}^f = \text{Lawformer}(D^f) \tag{7}$$

where $\boldsymbol{H}^f = \left\{\boldsymbol{h}_1^f, \ldots, \boldsymbol{h}_{d_f}^f\right\} \in \mathbb{R}^{d_f \times d_h}$, d_f is the sentence length of the fact description and d_h is the hidden state size. Similarly, for every charge $c_i \in C$ and law article $l_i \in L$, we obtain the hidden representation of their textual description $\boldsymbol{H}^{c_i} \in \mathbb{R}^{d_c \times d_h}$ and $\boldsymbol{H}^{l_i} \in \mathbb{R}^{d_l \times d_h}$, where d_c and d_l represent the sentence length of charge and law article descriptions, respectively.

For the extracted TCA $\hat{m}$, we obtain its embedding $\hat{\boldsymbol{m}}$ following:

$$\hat{\boldsymbol{m}} = \begin{cases} \boldsymbol{0} & \textbf{if } \hat{m} \textbf{ is } \text{<NEG>}, \\ \text{NumberEncoder}(\hat{m}) & \textbf{otherwise}, \end{cases} \tag{8}$$

where $\hat{\boldsymbol{m}} \in \mathbb{R}^{d_n}$, d_n is the dimension of the number embedding.

3.5.2 LJP exploiting cross-task dependencies and label semantics. As briefly discussed in Section 2.1, the legal judgment process follows a topological order. Judges initially identify the convicted charge by matching the criminal behavior outlined in the fact description with the definitions of the charges. Subsequently, leveraging both the fact description and the identified charge, the applicable law article is ascertained. The content of these law articles prescribes prison sentences corresponding to the varying degrees of criminal behavior severity. Judges assess the crime's severity, relying primarily on the fact description and employing quantitative reasoning. Finally, they assign prison sentences in accordance with the applicable law article determined earlier. To model this process, we employ the cross-task dependencies in a sequential manner to make judicial predictions, enriched with label semantics and quantitative reasoning using TCAs.

Charge Prediction. Specifically, for the first subtask topologically, the charge prediction, we first apply mean-pooling at sentence level on all charges' hidden representations $\boldsymbol{H}^C = \left\{\boldsymbol{H}^{c_1}, \ldots, \boldsymbol{H}^{c_{|C|}}\right\} \in \mathbb{R}^{|C| \times d_c \times d_h}$ and obtain $\boldsymbol{H}^c$:

$$\boldsymbol{H}^c = \frac{1}{d_c} \sum_{i=1}^{d_c} \boldsymbol{H}^C_{:,i,:} \in \mathbb{R}^{|C| \times d_h} \tag{9}$$

Then, we calculate the affinity matrix which contains affinity scores corresponding to all charge descriptions $\boldsymbol{H}^c$ and case fact descriptions $\boldsymbol{H}^f$:

$$A^{cf} = \boldsymbol{H}^f (\boldsymbol{H}^c)^T \in \mathbb{R}^{d_f \times |C|} \tag{10}$$

and subsequently, we perform max-pooling on A^{cf} and obtain the attention vector $\boldsymbol{a}^{cf}$:

$$\boldsymbol{a}^{cf} = \text{softmax}(\max(A^{cf}, \text{axis} = 1)) \in \mathbb{R}^{d_f}. \tag{11}$$

Next, we employ the attention vector $\boldsymbol{a}^{cf}$ to compute the weighted representation of the fact descriptions, denoted as $\boldsymbol{H}^{cf}$. This representation is subsequently concatenated with $\boldsymbol{H}^{f}$. After concatenation, a per-dimension mean-pooling operation is applied to $\boldsymbol{H}^{f;cf}$ to aggregate the information and construct the final input representation $\boldsymbol{h}^{f;cf} \in \mathbb{R}^{2d_h}$ for the charge classifier. The process can be represented as:

$$\boldsymbol{H}^{cf} = \text{broadcast}(\boldsymbol{a}^{cf})\boldsymbol{H}^{f} \in \mathbb{R}^{d_f \times d_h}, \tag{12}$$

$$\boldsymbol{H}^{f;cf} = [\boldsymbol{H}^{f}; \boldsymbol{H}^{cf}] \in \mathbb{R}^{d_f \times 2d_h}, \tag{13}$$

$$\boldsymbol{h}^{f;cf} = \sum_{z=1}^{d_f} \boldsymbol{h}_z^{f;cf}/d_f. \tag{14}$$

Finally, we adopt an affine transformation followed by softmax to produce the final prediction for the most relevant charge $\hat{c}$ from the set of charge labels L_c:

$$\hat{\boldsymbol{l}}_c = \text{softmax}(\boldsymbol{W}_c \boldsymbol{h}^{f;cf} + \boldsymbol{b}_c), \tag{15}$$

$$\hat{c} = \underset{i=1,\ldots,|C|}{\text{argmax}}\ \hat{l}_{c_i}, \tag{16}$$

where $\boldsymbol{W}_c \in \mathbb{R}^{|C| \times 2d_h}$ and $\boldsymbol{b}_c \in \mathbb{R}^{j}$ are the trainable weight and bias matrices, and $\hat{l}_{c_i} \in \hat{\boldsymbol{l}}_c, i = 1, \ldots, |C|$.

Law Article Prediction. Similar to the process described in Equations 9-12, we obtain the weighted representation of the fact descriptions in the light of law articles $\boldsymbol{H}^{lf} \in \mathbb{R}^{d_f \times d_h}$. Specifically:

$$\boldsymbol{H}^{l} = \frac{1}{d_l} \sum_{i=1}^{d_l} \boldsymbol{H}^{L}_{:,i,:} \in \mathbb{R}^{|L| \times d_h}, \tag{17}$$

$$A^{lf} = \boldsymbol{H}^{f}(\boldsymbol{H}^{l})^{T} \in \mathbb{R}^{d_f \times |L|}, \tag{18}$$

$$\boldsymbol{a}^{lf} = \text{softmax}(\max(A^{lf}, \text{axis} = 1)) \in \mathbb{R}^{d_f}, \tag{19}$$

$$\boldsymbol{H}^{lf} = \text{broadcast}(\boldsymbol{a}^{lf})\boldsymbol{H}^{f} \in \mathbb{R}^{d_f \times d_h}. \tag{20}$$

Then, using the previous charge prediction $\hat{c}$ from Equation 16, we obtain its hidden representation $\boldsymbol{H}^{\hat{c}} \in \mathbb{R}^{d_c \times d_h}$. Next, we get the final input representation $\boldsymbol{h}^{f;lf;\hat{c}} \in \mathbb{R}^{3d_h}$ for the law article classifier by applying first a concatenation among $\boldsymbol{H}^{f}$, $\boldsymbol{H}^{lf}$, and $\boldsymbol{H}^{\hat{c}}$ and subsequently a per-dimension mean-pooling. The process can be represented as:

$$\boldsymbol{I}^{d_f \times d_c} = \text{diag}(\underbrace{1, \ldots, 1}_{\min(d_f, d_c)\ \text{times}}) \in \mathbb{R}^{d_f \times d_c}, \tag{21}$$

$$\boldsymbol{H}^{\hat{c}'} = (\boldsymbol{I}^{d_f \times d_c})^{T} \boldsymbol{H}^{\hat{c}} \in \mathbb{R}^{d_f \times d_h}, \tag{22}$$

$$\boldsymbol{H}^{f;lf;\hat{c}} = [\boldsymbol{H}^{f}; \boldsymbol{H}^{lf}; \boldsymbol{H}^{\hat{c}'}] \in \mathbb{R}^{d_f \times 3d_h}, \tag{23}$$

$$\boldsymbol{h}^{f;lf;\hat{c}} = \sum_{z=1}^{d_f} \boldsymbol{h}_z^{f;lf;\hat{c}}/d_f. \tag{24}$$

Finally, we employ a linear transformation to predict the most relevant law article $\hat{l}$:

$$\hat{\boldsymbol{l}}_l = \text{softmax}(\boldsymbol{W}_l \boldsymbol{h}^{f;lf;\hat{c}} + \boldsymbol{b}_l), \tag{25}$$

$$\hat{l} = \underset{i=1,\ldots,|L|}{\text{argmax}}\ \hat{l}_{l_i}, \tag{26}$$

where $\hat{l}_{l_i} \in \hat{\boldsymbol{l}}_l$. The trainable weight and bias are denoted as $\boldsymbol{W}_l \in \mathbb{R}^{|L|\times 3d_h}$ and $\boldsymbol{b}_l \in \mathbb{R}^{|L|}$, respectively.

Prison Term Prediction. Prison term prediction as the third and final task in the LJP subtask sequence takes the fact description $\boldsymbol{H}^f$, the previously predicted law article $\hat{l}$ along with its textual description $\boldsymbol{H}^{\hat{l}} \in \mathbb{R}^{d_l \times d_h}$, and the TCA $\hat{\boldsymbol{m}} \in \mathbb{R}^{d_n}$ from Equation 8 as input.

We first compute the attention context of $\boldsymbol{H}^{\hat{l}}$ in the light of the calculated TCA $\hat{\boldsymbol{m}}$:

$$(\boldsymbol{a}^{\hat{\boldsymbol{m}}\hat{l}})^T = \text{softmax}(\hat{\boldsymbol{m}}^T (\boldsymbol{H}^{\hat{l}})^T) \in \mathbb{R}^{1\times l_n}, \tag{27}$$

$$\boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}} = \text{broadcast}(\boldsymbol{a}^{\hat{\boldsymbol{m}}\hat{l}}) \boldsymbol{H}^{\hat{l}} \in \mathbb{R}^{d_l \times d_h}. \tag{28}$$

The result $\boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}}$ represents the attended representation of the textual description of the predicted law article $\hat{l}$ from Equation 26. Through the above process, $\boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}}$ captures and highlights the most relevant part of the representation of textual description of a law article given the TCA $\hat{\boldsymbol{m}}$, effectively incorporating the TCA information into the prediction process. It is important to note that in scenarios where $\hat{\mathbf{m}} = \mathbf{0}$, indicating that the case description does not involve any TCA, the results of Equations 27 and 28, denoted as $\boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{\boldsymbol{l}}}$, remain unchanged in comparison to $\boldsymbol{H}^{\hat{l}}$. This is because the attention weights $\boldsymbol{a}^{\hat{\boldsymbol{m}}\hat{\boldsymbol{l}}}$ are equally distributed across all elements.

Next, we combine the fact description $\boldsymbol{H}^f$ and $\boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}}$ through concatenation and subsequently apply per-dimension mean-pooling to obtain the final representation of the input of the prison term classifier:

$$\boldsymbol{I}^{d_f \times d_l} = \text{diag}(\underbrace{1, \ldots, 1}_{\min(d_f, d_l)\text{ times}}) \in \mathbb{R}^{d_f \times d_l}, \tag{29}$$

$$\boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}'} = (\boldsymbol{I}^{d_f \times d_l}) \boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}} \in \mathbb{R}^{d_f \times d_h}, \tag{30}$$

$$\boldsymbol{H}^{f;\hat{\boldsymbol{m}}\hat{l}} = [\boldsymbol{H}^f; \boldsymbol{H}^{\hat{\boldsymbol{m}}\hat{l}'}] \in \mathbb{R}^{d_f \times 2d_h}, \tag{31}$$

$$\boldsymbol{h}^{f;\hat{\boldsymbol{m}}\hat{l}} = \sum_{z=1}^{d_f} \boldsymbol{h}_z^{f;\hat{\boldsymbol{m}}\hat{l}} / d_f \in \mathbb{R}^{2d_h}. \tag{32}$$

Eventually, we obtain the final prison time prediction through a linear function:

$$\hat{\boldsymbol{l}}_t = \text{softmax}(\boldsymbol{W}_t \boldsymbol{h}^{f;\hat{\boldsymbol{m}}\hat{l}} + \boldsymbol{b}_t), \tag{33}$$

$$\hat{t} = \underset{i=1,\ldots,|T|}{\text{argmax}}\ \hat{l}_{t_i}, \hat{l}_{t_i} \in \hat{\boldsymbol{l}}_t. \tag{34}$$

where $\boldsymbol{W}_t \in \mathbb{R}^{|T|\times 2d_h}$ and $\boldsymbol{b}_t \in \mathbb{R}^{|T|}$ are the parameters to learn.

Table 2. The statistics of datasets.

Dataset	CAIL-small	CAIL-big	LAIC-2021	LAC-CoT
Training set cases	109,291	1,569,029	19,070	1844
Testing set cases	25,073	183,408	2,119	204
Charges	125	140	43	-
Law articles	109	127	70	-
Prison Term	11	11	10	-

3.5.3 Training Objective. For the training of this model, we employ the cross-entropy loss for each subtask, specifically:

$$\mathcal{L}_c = -\sum_{i=1}^{|C|} c_i \log(\hat{c}_i), \tag{35}$$

$$\mathcal{L}_l = -\sum_{i=1}^{|L|} l_i \log(\hat{l}_i), \tag{36}$$

$$\mathcal{L}_t = -\sum_{i=1}^{|T|} t_i \log(\hat{t}_i), \tag{37}$$

and the overall loss is calculated as the average sum of all three subtask losses:

$$\mathcal{L}_{\text{total}} = \frac{1}{3}(\mathcal{L}_c + \mathcal{L}_l + \mathcal{L}_t). \tag{38}$$

4 Experimental Setup

4.1 Dataset

We evaluate the QR-LJP model on the CAIL-2018[8] and LAIC-2021[9] datasets. Both datasets provide judicial cases' fact descriptions along with corresponding meta-information, including sentenced charges, relevant law articles, and prison term predictions. The cases in both datasets vary in type, with some involving monetary values and others not. To process the data, we follow the methodology outlined by Zhong et al. [65], where infrequent charges and law articles are filtered out, retaining only those that appear more than 100 times. Additionally, we divide the terms into non-overlapping intervals to standardize the data. Although cases with multiple charges or criminals exist, for consistency with SOTA baselines, we selected samples with a single criminal and a single charge to reduce complexity. After the above processes, the statistics of our CAIL-2018 dataset (both CAIL-2018-small and CAIL-2018-big) and LAIC-2021 dataset are shown in Table 2. As for the training of FineChatGLM, we use the dataset LAC-CoT as described in Section 3.3. The statistics of LAC-CoT are also listed in Table 2.

As additional evidence of generalizability, we test our method on a small-scale dataset of cases sourced from Sherloc[10]. Further details and results are reported in the Appendix F.

[8]http://cail.cipsc.org.cn
[9]https://data.court.gov.cn/pages/laic2021.html
[10]https://sherloc.unodc.org/

4.2 Training Settings

For the training of FineChatGLM, we used the P-Tuning V2 [34] method to adapt the ChatGLM2-6B [13] model to perform TCA calculations. For the pre-trained number encoder as well as our LJP model QR-LJP, the Adam optimizer is selected for the training of both. Hyperparameters involved in all three training processes are detailed in Table 4. All of the models are trained on four V100-32GB GPUs. The parameter size of our model is listed in Table 3.

Table 3. Parameter size of QR-LJP components. "Frozen" indicates parameters not updated during training; "Trainable" indicates parameters updated in our experiments. Parameter counts are rounded to the nearest thousand.

Module	Frozen Params	Trainable Params	Total Params
Lawformer legal text encoder	104,431k	0	104,431k
BiLSTM number encoder (DICE-init h=256)	0	1,052,672	1,052,672
Task-specific classifiers[11]	0	145,430	145,430
Total	104,431k	1,198,102	105,629,206

4.3 Evaluation Metrics

For the charge prediction task, law article prediction task, and prison term prediction task, LJP results are reported in terms of Accuracy (Acc), Macro-Precision (MaP), Macro-Recall (MaR), and Macro-F1 (MaF).

4.4 Baselines

To evaluate the efficacy of our method on LJP subtasks (i.e. charge prediction, law article prediction, and prison term prediction), we have selected the following baselines:

- **ChatGLM2**: we adapted ChatGLM2 using P-Tuning V2 to perform LJP tasks [12].
- **LexiLaw** [13]: we prompt LexiLaw as the legal domain counterpart to the general domain ChatGLM2 to perform LJP tasks [14].
- **LLama3+LoRA** [52]: we adapted LLama3 using LoRA to perform LJP tasks.
- **LLama3+ICL**: we prompted LLama3 to perform LJP tasks via in-context learning.
- **Qwen2.5+LoRA** [2]: we adapted Qwen2.5 using LoRA to perform LJP [15]
- **Qwen2.5+ICL**: we prompted Qwen2.5 to perform LJP tasks via in-context learning [16]
- **TopJudge** [65]: an LJP model that utilizes a directed acyclic graph to capture interdependencies among subtasks.
- **NeurJudge** [63]: an LJP model that leverages intermediate subtasks outcomes to partition the case description into scenarios aiding the prediction of other subtasks.
- **CEEN** [38]: a reinforcement learning-based LJP framework that distinguishes ambiguous fact descriptions as well as confusing law articles to accurately locate elements and enhance LJP.
- **NumSCL** [17]: an LJP model that exploits contrastive learning and crime amounts to assist with confusing legal judgment predictions. Following the experiments in the original paper, we adopt the NeurJudge w/NumSCL line as our baseline.

[12]Details about the construction of the prompt-tuning dataset are shown in Appendix B.1
[13]https://github.com/CSHaitao/LexiLaw
[14]The prompt used along with its English Translation is shown in Appendix B.2
[15]Details about the LoRA hyper-parameters, for both LLama and Qwen are detailed in Appendix C.
[16]Details about the In-Context Learning are detailed in Appendix D.

Table 4. Hyperparameters during training.

Model	Parameter	Value
FineChatGLM	prompt embedding length	128
	max source text length	600
	max target text length	600
Number Encoder	learning rate	1e-3
	LSTM hidden size	256
	batch size	4096
	epochs	100
	lower bound	0
	upper bound	1e+8
	training data size	1e+6
QR-LJP	learning rate	1e-3
	hidden size	256
	epochs	early stopping
	patience	5
	batch size	512
	fact sentence length	512
	charge sentence length	128
	article sentence length	128
	attention window size	256

- **GJudge** [51]: a graph boosting with constraints framework that integrates fact descriptions and graphs depicting label similarity relationships, thereby enhancing robustness and improving LJP performance.
- **ML-LJP** [37]: an LJP framework that fuses constrastive statute embeddings, graph-based inter-article reasoning, and explicit numerical-fact cues to yield more consistent charge, article, and sentence predictions across complex cases.

5 Experimental Results and Analyses

5.1 Main Results

The results on CAIL-2018-small, CAIL-2018-big, and LAIC-2021 (Tables 5, 6, 7) show a clear and stable trend: **incorporating quantitative reasoning (QR-LJP) consistently lifts prison-term performance** across datasets. The gains are robust to data scale and label distributions and are accompanied by the highest or near-highest accuracies, indicating that QR-LJP not only predicts the correct direction of sentence severity but also calibrates its magnitude more reliably than non-QR counterparts.

Crucially, these improvements on sentencing do *not* come at the expense of the other subtasks. On charge prediction, QR-LJP maintains the best or competitive results in both Accuracy and Macro-F1 across all datasets. On law article prediction, it remains on par with strong baselines overall, often achieving the top accuracies at larger scales, thereby preserving statute recognition while focusing its additional capacity on the numerically sensitive sentencing head.

Finally, generic LLM variants under LoRA or in-context prompting remain substantially behind on the prison-term metric in all three tables, reinforcing that explicit quantitative cues (via TCA and

Table 5. LJP results of the baselines and our method on CAIL-2018-small.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	ChatGLM2	80.04	75.36	75.12	74.46	76.39	60.69	60.31	59.37	28.88	22.46	23.39	22.25
2	LexiLaw	29.67	41.97	18.17	20.54	2.34	14.34	1.01	1.55	14.94	28.17	10.14	5.66
3	LLama3+LoRA	18.94	18.77	12.99	11.33	5.35	4.68	3.91	3.22	18.73	17.67	9.23	8.03
4	Qwen2.5+LoRA	32.33	16.38	18.37	16.33	34.32	17.27	19.66	17.58	26.70	7.10	6.36	6.29
5	LLama3+ICL	7.98	5.20	4.45	3.66	2.70	1.81	1.67	1.21	14.38	2.34	2.34	1.99
6	Qwen2.5+ICL	11.57	6.56	5.28	4.91	13.45	7.56	6.57	6.38	19.06	2.46	1.62	1.76
7	TopJudge	80.77	63.57	79.87	77.40	77.22	71.54	74.78	70.75	38.27	33.79	34.38	32.15
8	NeurJudge	79.62	70.77	78.68	76.86	78.27	75.83	75.17	74.06	37.71	34.90	36.45	35.54
9	CEEN	80.34	83.01	80.95	79.60	82.32	84.41	82.59	81.91	38.33	35.29	33.26	31.65
10	NumSCL	83.97	82.43	83.49	82.53	80.22	79.59	81.50	79.20	38.49	34.85	33.38	33.05
11	GJudge	81.23	78.53	80.22	78.94	82.05	83.47	79.03	80.35	37.00	34.52	33.79	34.74
12	ML-LJP	67.81	70.53	62.24	61.38	71.95	71.05	65.87	64.98	35.14	35.16	30.66	29.84
13	QR-LJP	86.69	84.30	85.87	84.04	80.47	78.87	78.92	76.36	39.61	37.49	36.27	36.29
14	QR-LJP$_{w/o\text{TCA}}$	84.11	83.37	83.95	82.48	80.19	77.54	77.19	75.94	38.62	35.65	33.42	33.57

Table 6. LJP results of the baselines and our method on CAIL-2018-big.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	ChatGLM2	84.82	45.24	38.47	37.41	85.78	46.53	40.16	39.35	40.47	21.48	20.86	20.82
2	LexiLaw	65.33	50.65	24.37	27.43	5.88	25.11	1.20	2.01	12.88	39.53	10.24	4.80
3	LLama3+LoRA	72.24	11.69	7.86	7.33	43.49	4.49	3.79	3.18	42.18	15.99	9.80	9.78
4	Qwen2.5+LoRA	83.59	13.87	13.70	12.80	84.40	14.59	14.93	13.70	46.98	7.86	5.97	6.04
5	LLama3+ICL	63.04	7.03	4.86	4.72	42.06	3.12	2.77	2.40	40.04	3.18	2.49	2.04
6	Qwen2.5+ICL	76.81	11.63	10.84	10.45	80.81	12.35	12.73	11.89	36.07	4.74	9.45	4.54
7	TopJudge	95.09	59.84	85.37	75.30	95.31	68.91	83.41	71.23	56.20	39.69	45.76	40.82
8	NeurJudge	95.06	65.50	85.87	76.66	95.94	73.64	84.59	77.11	58.13	43.74	46.00	44.04
9	CEEN	95.86	83.05	69.46	72.54	95.64	87.07	71.63	75.10	57.77	48.47	40.30	41.61
10	NumSCL	95.80	84.42	76.66	79.21	95.93	80.88	74.46	76.42	59.24	50.01	45.63	46.75
11	GJudge	95.31	78.16	83.58	81.72	96.04	86.24	76.96	81.22	57.47	43.40	45.02	43.89
12	ML-LJP	89.22	86.41	75.46	74.23	90.03	81.28	77.07	78.94	43.88	42.15	40.19	40.67
13	QR-LJP	96.47	81.01	89.23	83.86	96.55	78.46	87.46	80.62	61.08	49.57	53.14	50.92
14	QR-LJP$_{w/o\text{TCA}}$	95.79	76.08	87.40	79.64	95.93	73.05	87.38	76.74	58.83	44.03	49.43	45.53

alignment) are necessary for faithful statute-aware sentencing. Overall, the expanded comparisons confirm that integrating QR yields consistent, cross-dataset gains on sentencing without degrading charge and law-article performance.

5.2 Further Analyses

5.2.1 The Efficacy of Baselines augmented with QR-LJP . We further explore whether incorporating the quantitative reasoning process, as demonstrated in QR-LJP, can enhance the performance of existing SOTA models such as TopJudge, NeurJudge, CEEN, and GJudge. To do this, we utilize FineChatGLM alongside the pre-trained DICE number encoder for each SOTA model to obtain TCAs and their subsequent encodings. We then calculate the attention context of each model's predicted law article based on the TCA, which is subsequently concatenated to the input of the original model's prison term classifier. The outcomes of these experiments, as well as the performances of the original SOTA models, are depicted in Figure 5 and evaluated using the CAIL-small dataset.

Table 7. LJP results of the baselines and our method on LAIC-2021.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	ChatGLM2	72.43	66.37	38.52	47.02	69.28	23.17	36.28	37.02	38.23	22.23	31.57	27.86
2	LexiLaw	24.47	35.54	18.95	29.40	15.51	10.12	8.41	7.72	21.23	13.81	18.42	18.38
3	LLama3+LoRA	72.79	33.00	31.25	27.28	63.45	15.33	13.24	12.15	28.20	28.86	24.31	23.95
4	Qwen2.5+LoRA	92.82	70.68	72.37	70.56	92.91	62.68	63.51	62.29	38.86	35.87	35.74	35.70
5	LLama3+ICL	19.26	6.67	3.47	3.03	21.64	3.42	2.49	2.66	19.35	14.73	14.45	13.68
6	Qwen2.5+ICL	14.37	10.09	6.01	5.22	15.13	7.48	5.32	4.83	18.95	19.45	14.41	13.88
7	TopJudge	91.56	63.02	66.77	63.54	91.29	54.42	59.88	55.54	37.16	31.07	32.51	28.75
8	NeurJudge	91.56	50.19	75.18	70.09	93.18	68.90	75.22	70.09	35.46	33.56	34.82	33.47
9	CEEN	93.19	87.04	79.15	80.51	93.36	86.08	78.60	79.82	34.77	35.70	29.29	26.01
10	NumSCL	90.82	86.90	80.23	79.77	92.57	75.71	79.30	78.97	34.78	31.09	29.97	29.46
11	GJudge	91.46	83.27	84.22	80.69	89.62	79.67	76.17	76.99	36.24	34,21	38.82	34.02
12	ML-LJP	94.53	82.96	82.15	82.39	94.01	81.63	80.55	80.88	42.19	42.75	35.86	36.43
13	QR-LJP	96.93	90.06	94.90	92.08	96.60	89.47	94.36	91.32	44.79	40.06	42.78	36.80
14	QR-LJP$_{w/o\text{TCA}}$	95.01	84.77	89.03	85.27	94.33	87.68	88.09	86.56	40.72	37.42	36.94	33.17

Our findings indicate that integrating quantitative reasoning consistently improves the Macro-F1 scores of prison term predictions across all SOTA models. Furthermore, this integration maintains comparable performance levels in charge prediction and law article prediction tasks, suggesting that the application of QR-LJP's quantitative reasoning to other baseline LJP models could yield enhanced results. Moreover, this indicates that the efficacy of QR can be universally applied[17].

5.2.2 The Maximum Capacity of Quantitative Reasoning in LJP. We further conducted an *oracle* experiment to evaluate the maximum capacity of LJP with quantitative reasoning. For this, we randomly selected 1,000 cases from the CAIL-small test dataset used in the main experiments and manually annotated the ground truth TCAs (details about the annotators are provided in Appendix H) to create the *oracle* test dataset. Then, we evaluated the QR-LJP model from Table 5 Line 8 on this dataset and compared the test results on the *oracle* dataset with those on a dataset containing the same cases but with TCAs calculated by FineChatGLM. The results, shown in Figure 6, demonstrate that LJP performance can be substantially improved by enhancing TCA calculation. Moreover, we conduct experiments on other SOTA baselines (i.e., TopJudge, NeurJudge, and CEEN) using the same oracle dataset. Details of these can be found in Appendix I.

5.2.3 The Efficacy of FineChatGLM. We designed experiments to evaluate the efficacy of our TCA calculation module, FineChatGLM.

Dataset. The dataset we used was the LAC-CoT dataset as in Section 3.3.

Baselines. To evaluate the performance of FineChatGLM, we've selected the following baselines:

- Regular Expressions-based: we used regular expressions that extract all of the monetary values within a case description, and the largest number is considered the total crime amount.
- ChatGPT: we prompted ChatGPT to calculate TCAs via API service.
- LexiLaw: we prompted LexiLaw, a legal domain large language model with 6B parameters. LexiLaw is a Chinese-oriented model fine-tuned based on the ChatGLM2-6B [13] model.
- NumSCL [17]: we use NumSCL, an adapted BERT-CRF model whose crime amount calculation dataset was annotated through the use of a 0-1 knapsack algorithm.

[17] More detailed experiments investigating existing math solvers combining with SOTA LJP models are described in Appendix G.

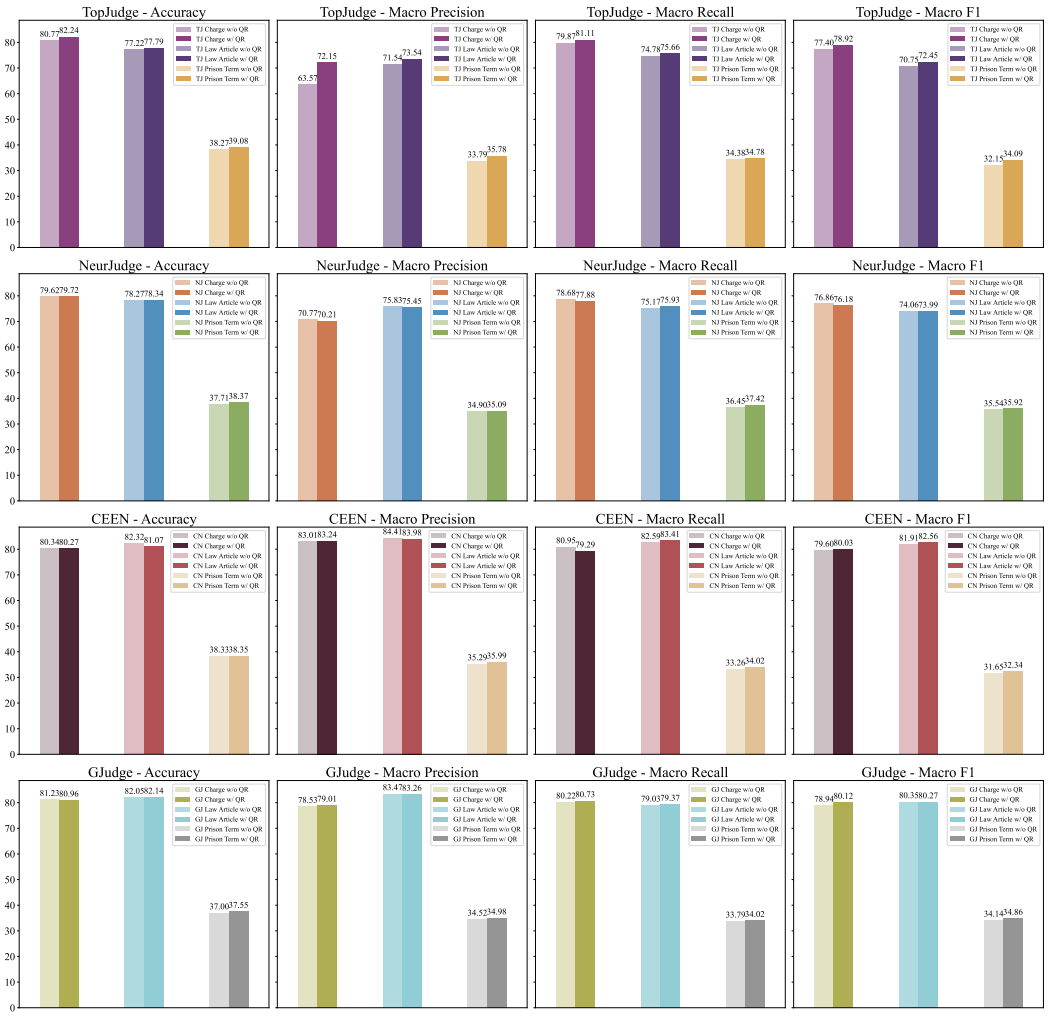


Fig. 5. Experimental results of TopJudge, NeurJudge, CEEN, and GJudge on CAIL-small dataset. Lighter shades represent the original SOTA models and darker shades represent SOTA models with quantitative reasoning.

- MathPrompter [23]: a prompting-based math solver that abstracts the problem into an algebraic template, generates both algebraic expressions and Python snippets, enforces randomized consistency checks, and applies self-consistency voting to produce a numeric answer.
- CoMAT [30]: Chain of Mathematically Annotated Thought that separates symbolic reasoning from execution—first producing a variable/equation plan, then executing/verifying a program to compute the result.

Metrics. The prediction accuracy on the testing set is chosen to evaluate the performance of each setting.

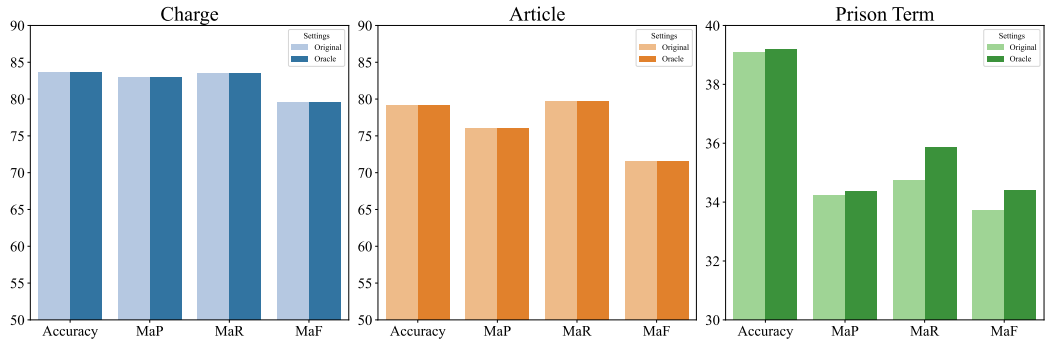


Fig. 6. Experimental results of the oracle experiment. Darker colors represent the result of data with the ground truth TCA and lighter colors represent the result of data with TCAs calculated by FineChatGLM.

Table 8. Ablation results of the crime amount calculation module.

Approach	Accuracy (%)
Regular Expressions	39.22
LexiLaw	42.16
ChatGPT	73.04
NumSCL	66.67
FineChatGLM	**80.88**
FineChatGLM$_{w/oCoT}$	77.45
MathPrompter	77.94
CoMAT	67.65
FineChatGLM$_{rep}$	80.39

Results. We conducted the experiments over the baselines above and the performances are shown in Table 8. From the results we can conclude that FineChatGLM has the highest accuracy score, outperforming all baselines[18].

5.2.4 Single-LLM SFT Alternatives. We additionally assess whether a single large language model can learn the entire QR-LJP pipeline via supervised fine-tuning (SFT). Concretely, we adapt ChatGLM2-6B in two variants:

(1) **ChatGLM2+Inline-Reason**: We adapt ChatGLM2-6B model to output total crime amount and LJP predictions in a single response. During inference, it must first state the TCA and then the judgment.
(2) **ChatGLM2+Two-Stage-Adaptation**: We adapt ChatGLM2-6B model in a sequential manner. Firstly, the model is solely tuned to calculate TCA. Next in stage 2, the model is tuned to predict LJP results.
(3) **ChatGLM2**: As a baseline, we directly adapt ChatGLM2-6B model to perform LJP tasks.

The dataset we used consists of 2,000 cases randomly selected from the CAIL-2018-small dataset. These selected cases are annotated with their ground truth labels of charge, relevant law articles,

[18]Further LJP experiments of different existing math sovlers combined with existing LJP models are shown in Appendix.

Table 9. LJP results of ChatGLM2 baseline, ChatGLM2+Inline-Reasoning (ChatGLM+IlR) and ChatGLM2+Two-Stage-Adaptation (ChatGLM+TSA).

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	ChatGLM	80.04	75.36	75.12	74.46	76.39	60.69	60.31	59.37	28.88	22.46	23.39	22.25
2	ChatGLM+IlR	71.15	69.84	66.03	65.27	69.55	57.66	56.74	55.98	24.65	19.98	19.36	19.62
3	ChatGLM+TSA	79.55	74.17	74.82	73.90	75.90	60.25	59.78	58.91	28.55	22.07	22.79	22.03
4	QR-LJP	92.30	79.82	87.10	81.36	95.95	77.94	85.67	79.88	34.65	39.74	41.02	40.18

and penalty terms, the same as the CAIL-2018 dataset. More details of the annotation process is in Appendix E.

Finally, we use precisely crafted regex patterns o model outputs to extract labels in the answers. Then, we report Accuracy (Acc), Macro Precision (MaP), Macro Recall (MaR), and Macro F1 Score (MaF) as the results.

The results are shown in Table 9. It is evident that the fully-modular QR-LJP model decisively outperforms both single-model baselines. The first baseline, ChatGLM2+Inline-Reasoning, is trained once on a conposite instruction that forces the model to state the TCA and the sentence in the same answer. In practice, the arithmetic and legal objectives fight for the same parameters and the same decoding path. The model must simultaneously extract extract scattered numbers, perform addition, map the results to statute tiers, and weigh aggravating or mitigating cues, all while predicting the final sentence token by token. Optimisation is there fore unstable and sample-ifficient, which means that small numeric hallucinations immediately shift the predicted severity tier, and that the network receives no intermediate loss signal to correct the arithmetic step. As a consequence, inline-reasoning lands about 20.58% below QR-LJP on the Macro F1 score on the prison term prediction task.

The second baseline, ChatGLM2+Two-Stage-Adaptation, trains the same network twice: step 1 on TCA targets only, step 2 on sentencing targets only. This sequential recipe introduces a new pathology: **catastrophic overwrite**. Parameters that had converged toward recognizing quantitative cues are pulled toward a different optium once the second loss is applied, so the network forgets much of what it learned in the first round. The end result is that, despite having seen explicit amount supervision, the two-stage model finishes with essentially the same overall performance as the directly fine-tuned ChatGLM2 baseline. This means the extra training pass fails to translate into a measurable gain, confirming that simply stacking objectives on a single set of weights is not enough to match the modular QR-LJP design.

5.3 Ablation Study

5.3.1 The Efficacy of Quantitative Reasoning in QR-LJP. To evaluate the effectiveness of quantitative reasoning in QR-LJP's prediction of prison terms, we conducted ablative experiments. Specifically, we excluded the training and inferencing of FineChatGLM, thereby eliminating the utilization of TCAs in LJP, and recorded the evaluation results of the remaining vanilla model. This model exploits cross-task dependencies and label semantics by using subtask predictions and corresponding legal texts (i.e., charge definitions and law article contents) as auxiliary features. More specifically, the representation of the predicted law article is directly concatenated to fact representations and then fed into the prison term classifier. The results on the CAIL-small and CAIL-big datasets, denoted as QR-LJP$_{w/o\text{TCA}}$, are shown in Line 9 of Table 5 and Table 6, respectively. It is evident that by incorporating quantitative reasoning, the performance of QR-LJP on the prison term prediction

Table 10. LJP results of QR-LJP, with varing initial encoding strategies of the number encoder, including DICE, Random initialization, Lawformer, and Roberta.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	QR-LJP+DICE	86.69	84.30	85.87	84.04	80.47	78.87	78.92	76.36	39.61	37.49	36.27	36.29
2	QR-LJP+Rdm	86.73	84.11	84.47	83.89	81.01	80.25	79.32	77.00	35.92	34.13	33.06	31.67
3	QR-LJP+Lfm	86.95	84.63	86.21	84.37	80.12	79.08	78.64	75.98	38.45	36.72	35.40	34.65
4	QR-LJP+RBT	86.41	83.92	85.22	83.65	80.88	79.56	79.20	76.72	37.88	36.05	34.71	33.62

task has been significantly enhanced in terms of the absolute F1 score, increasing by 2.72% on the CAIL-small dataset and 5.29% on the CAIL-big dataset. This demonstrates the effectiveness of QR in LJP.

5.3.2 The Efficacy of CoT. To further explore the effects of the inclusion of CoT data during the prompt-tuning of FineChatGLM, we conduct additional ablative experiments where CoT is absent in the training data. A detailed example of one data item is shown in Figure 4b. Then, we evaluated the accuracy of TCA calculation of this setting, and as Table 8, FineChatGLM suffers a 3.43% accuracy loss when CoT is omitted during prompt-tuning. This demonstrates the effectiveness of CoT data during the prompt-tuning of FineChatGLM.

5.3.3 The Efficacy of the DICE embedding. To assess whether a magnitude-preserving number encoder benefits QR-LJP, we ablate the initialization of the number embedding and compare **DICE** with two representative alternatives. *(i) Random initialization*: number embeddings are randomly initialized. *(ii) PLM-based embeddings*: numbers are tokenized as text strings and encoded by a pre-trained language model, followed by a linear projection to match the number-embedding dimension. We evaluate two variants: a legal-domain PLM (*Lawformer*) and a general-domain PLM (*Chinese RoBERTa-wwm-ext*).

All three variants were trained under the same QR-LJP pipeline. The results on CAIL-small (see Table 10) show that the DICE line achieves the highest performance in terms of Macro-F1 score in prison term, confirming that the DICE initial embeddings offer a clear advantage in our quantitative reasoning scenario.

5.3.4 The Efficacy of Legal Knowledge Alignment Module. Our QR-LJP adopts a *TCA-conditioned legal knowledge alignment* module. Concretely, the TCA is first embedded with a magnitude-preserving number encoder to obtain a severity vector, which is then used as a query over statute representations encoded by the law-text encoder. The cross-attention weights highlight provisions whose monetary thresholds are most compatible with the case severity, producing an attended law context. This context is subsequently fused with the fact representation via attention and concatenation, together with the TCA embedding, to form a severity-aware feature that feeds the prison-term head.

To further investigate the contribution of the legal knowledge alignment module, we run a controlled ablation that varies only this component on CAIL-2018-small. We compare three configurations: (i) Full Alignment (our default), where a TCA-derived query attends to statute representations and the attended law context is fused with facts before sentencing; (ii) No Alignment, which removes the attention mechanism and simply concatenates the TCA embedding with the fact representation, with statute text never conditioned on TCA; and (iii) Text-Only+Law, which removes the TCA branch and aligns facts with statute text without quantitative conditioning.

Table 11. LJP results on different legal knowledge alignment strategies.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	**Full Alignment**	86.69	84.30	85.87	84.04	80.47	78.87	78.92	76.36	39.61	37.49	36.27	36.29
2	**No Alignment**	82.90	81.14	83.37	81.18	78.50	76.13	78.75	75.17	38.41	33.18	36.62	33.77
3	**Text Only+Law**	86.23	83.05	85.51	83.02	80.14	78.02	77.81	75.78	40.49	35.23	37.68	35.48

We report Accuracy, Macro-Precision, Macro-Recall, and Macro-F1 for charge, law-article, and prison-term predictions.

As shown in Table 11, Full Alignment consistently outperforms the two alternatives across all three subtasks, with the largest gains on prison-term prediction where statute thresholds interact most directly with TCA. Removing alignment entirely (No Alignment) leads to notable drops, indicating that naïve concatenation cannot capture the structured relationships among facts, numeric severity, and statute provisions. The Text-Only+Law variant, benefiting from statute evidence but lacking quantitative conditioning, performs better than No Alignment yet remains below Full Alignment. These results demonstrate that conditioning legal knowledge on TCA is crucial for faithfully modeling statute-specific sentencing logic.

5.4 Case Study

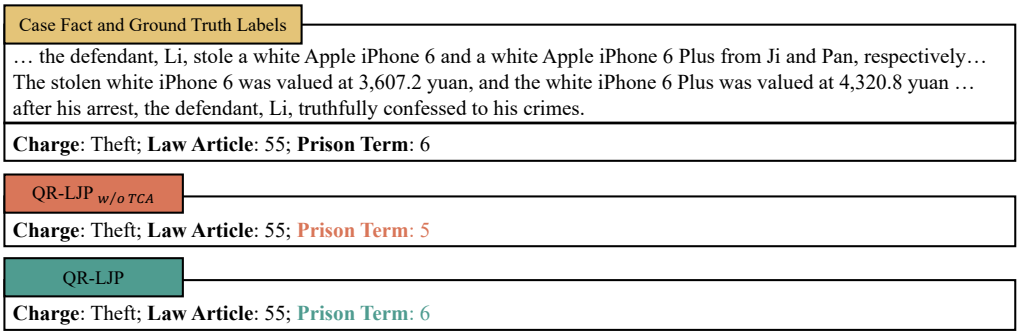
Figure 7 shows two cases we've selected from the CAIL-small test dataset to demonstrate the effect of the proposed QR-LJP model.

5.4.1 Case with numerical features. The first selected case involves explicit monetary features within its fact description. In this case, the QR-LJP$_{w/o\text{TCA}}$ model correctly predicts the charge and law article label, but the prison term prediction is the incorrect label 5. However, this error is corrected with the QR-LJP model utilizing quantitative reasoning when predicting prison terms. From this case, we can conclude that the QR-LJP model correctly predicts the prison term as label 6, meaning a sentence of fixed-term imprisonment of more than 9 months but less than 12 months.

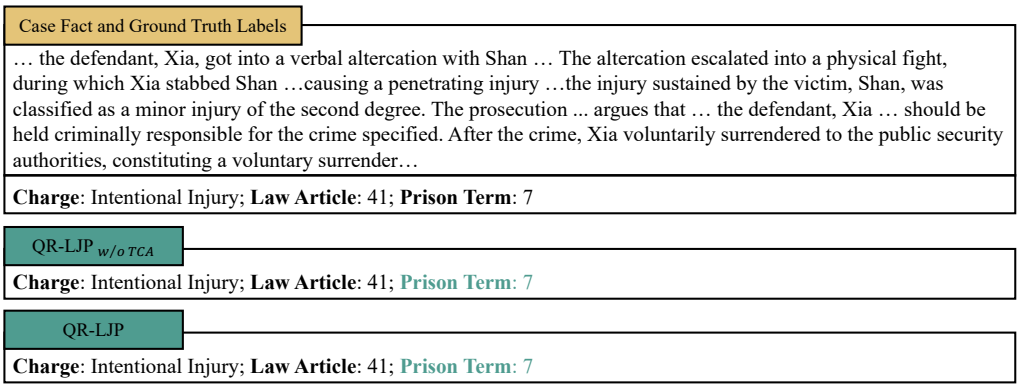
5.4.2 Case without numerical features. The second selected case does not involve monetary values in the case description, and both models correctly predict all three subtask labels. From this case, we can conclude that for cases without explicit quantitative features, the quantitative reasoning mechanism in QR-LJP doesn't hinder the performance on LJP subtasks.

5.5 Error Analysis

From Table 5 and Table 6, we observe that while our QR-LJP model achieved the best Macro-F1 scores for the tasks of charge prediction and prison term prediction, it did not yield the best results for the law article prediction task. We attribute this discrepancy to *error propagation*. Our model, designed to leverage sequential cross-task dependencies, has the prediction of the law article partially dependent on the results of charge prediction, and similarly, the prediction of the prison term partially reliant on the law article prediction. Consequently, errors in earlier predictions can adversely affect subsequent outcomes, leading to a cascade of inaccuracies. This issue is an inherent flaw in the paradigm of exploiting cross-task dependencies in LJP. Although QR-LJP still surpasses other baselines in the prison term prediction task, we believe that mitigating this error propagation could lead to further improvements. Addressing this issue is a priority for the future development of our methodology.



(a) A case where quantitative reasoning is involved during LJP.



(b) A case where quantitative reasoning is not involved during LJP.

Fig. 7. Qualitative examples to demonstrate the effect of the proposed framework.

5.6 Reproducability

In our main experiments, a very important step is to query ChatGPT to generate Chain-of-Thought data of the calculation process of the total crime amount. However, ChatGPT is not guaranteed to generate the same content with a given input in multiple attempts. One way to generate reproducible outputs is to freeze the seed and temperature of ChatGPT, which contradicts our motivation to utilize ChatGPT or other commercial LLMs alike as tools to assist with smaller models specifically tailored for a certain problem in a dynamic fashion. Hence we conducted further experiments to show the reproducibility of our method. We repeated the processes in Section 4.2 to finetune ChatGLM2 verbatim and ended up with another version of FineChatGLM, denoted as $\text{FineChatGLM}_{\text{rep}}$. The only differences between FineChatGLM and $\text{FineChatGLM}_{\text{rep}}$ originate from different CoT contents ChatGPT generated in the two attempts. The comparison between the results of FineChatGLM and $\text{FineChatGLM}_{\text{rep}}$ is shown in Table 8. From the results, we can see that though the CoT data from ChatGPT is different, the crime amount calculation accuracy remain rather consistent. We credit this to the consistent quality of answers from ChatGPT and view this as proof of the reproducibility of our method.

6 Conclusion

In this work, we proposed to improve LJP by integrating quantitative reasoning into the sequentially task-dependent LJP model paradigm. The framework is capable of determining the total crime amount of a case, and subsequently incorporating this information into the LJP process by combining it with article semantics. Results show that our method outperforms other SOTA models in terms of prison term prediction without compromising the performance on other subtasks i.e. charge prediction and law article prediction. Further experiments explored the maximum capacity of the idea of incorporating crime amounts to predict prison term sentences, validated the effectiveness of FineChatGLM, and visualized how crime amounts impacted the prediction process.

Limitations

Our QR-LJP model operates on the foundational principle of cross-task dependencies among legal subtasks. However, this approach is not without its challenges. One inherent flaw is the potential for error propagation throughout the model's topological structures, which may impact the overall accuracy. We recognize the need for more advanced strategies to effectively mitigate this issue in future developments of our model.

Currently, our model is specifically tailored to handle legal documents from the People's Republic of China and has been trained and tested across three LJP subtasks. Despite this regional focus, we believe that the core methodology—integrating quantitative reasoning into the judgment prediction process—holds significant potential for application in diverse legal systems worldwide. Adapting our model to suit different jurisdictional frameworks could substantially enhance its applicability and effectiveness. This opportunity for global adaptation and refinement will be a focus of our future research.

Ethical Considerations

Automatic legal judgment prediction is a sensitive research area. Neither the proposed QR-LJP model in this paper nor the other SOTA approaches are ready to be productized.

The LLM used for calculating total crime amounts is deployed locally to avoid sensitive data exposure. In our framework, we treat legal data for training and inference in our LJP model as sensitive, reflecting real-world application scenarios. Conversely, the small sample of legal data used to query ChatGPT for prompt-tuning enhancement is considered public, sourced from an open dataset, with no further ChatGPT queries needed post-tuning.

The original datasets used in this paper are all from publicly available resources and personal information (e.g. name, location, etc.) is properly anonymized.

Acknowledgments

References

[1] Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preoţiuc-Pietro, and Vasileios Lampos. 2016. Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective. *PeerJ computer science* 2 (2016), e93.

[2] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen Technical Report. *CoRR* abs/2309.16609 (2023). https://doi.org/10.48550/ARXIV.2309.16609 arXiv:2309.16609

[3] Yefim Bakman. 2007. Robust understanding of word problems with extraneous information. *CoRR* math/0701393 (2007). arXiv:math/0701393 https://arxiv.org/abs/math/0701393

[4] Daniel Bobrow et al. 1964. *Natural language input for a computer problem solving system.* Technical Report. MIT.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[6] Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural Legal Judgment Prediction in English. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, Italy, 4317–4323. https://doi.org/10.18653/V1/P19-1424

[7] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: "Preparing the Muppets for Court'". In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020 (Findings of ACL, Vol. EMNLP 2020)*. Association for Computational Linguistics, Online Event, 2898–2904.

[8] Ilias Chalkidis, Nicolas Garneau, Catalina Goanta, Daniel Martin Katz, and Anders Søgaard. 2023. LeXFiles and LegalLAMA: Facilitating English Multinational Legal Language Model Development. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, Toronto, Canada, 15513–15535.

[9] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. PaLM: Scaling Language Modeling with Pathways. *CoRR* abs/2204.02311 (2022). https://doi.org/10.48550/ARXIV.2204.02311 arXiv:2204.02311

[10] Wentao Deng, Jiahuan Pei, Keyi Kong, Zhe Chen, Furu Wei, Yujun Li, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. 2023. Syllogistic Reasoning for Legal Judgment Analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, Singapore, 13997–14009.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[12] Qian Dong and Shuzi Niu. 2021. Legal judgment prediction via relational learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, Virtual Event, 983–992.

[13] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 320–335.

[14] Yi Feng, Chuanyi Li, and Vincent Ng. 2022. Legal Judgment Prediction: A Survey of the State of the Art. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, Luc De Raedt (Ed.). ijcai.org, Vienna, Austria, 5461–5469. https://doi.org/10.24963/IJCAI.2022/765

[15] Yi Feng, Chuanyi Li, and Vincent Ng. 2022. Legal Judgment Prediction via Event Extraction with Constraints. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 648–664.

[16] Charles R Fletcher. 1985. Understanding and solving arithmetic word problems: A computer simulation. *Behavior Research Methods, Instruments, & Computers* 17, 5 (1985), 565–571.

[17] Leilei Gan, Baokui Li, Kun Kuang, Yating Zhang, Lei Wang, Anh Luu, Yi Yang, and Fei Wu. 2023. Exploiting Contrastive Learning and Numerical Evidence for Confusing Legal Judgment Prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 12174–12185. https://doi.org/10.18653/v1/2023.findings-emnlp.814

[18] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *CoRR* abs/2101.00027 (2021). arXiv:2101.00027 https://arxiv.org/abs/2101.00027

[19] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided Language Models. In *International Conference on Machine Learning, ICML 2023, 23-29 July*

2023, Honolulu, Hawaii, USA (Proceedings of Machine Learning Research, Vol. 202), Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 10764–10799. https://proceedings.mlr.press/v202/gao23f.html

[20] Jidong Ge, Yunyun Huang, Xiaoyu Shen, Chuanyi Li, and Wei Hu. 2021. Learning fine-grained fact-article correspondence in legal cases. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), 3694–3706.

[21] Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793* (2024).

[22] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR* 1, 2 (2022), 3.

[23] Shima Imani, Liang Du, and Harsh Shrivastava. 2023. MathPrompter: Mathematical Reasoning using Large Language Models. In *Proceedings of the The 61st Annual Meeting of the Association for Computational Linguistics: Industry Track, ACL 2023, Toronto, Canada, July 9-14, 2023*, Sunayana Sitaram, Beata Beigman Klebanov, and Jason D. Williams (Eds.). Association for Computational Linguistics, 37–42. https://doi.org/10.18653/V1/2023.ACL-INDUSTRY.4

[24] Daniel Martin Katz, Michael J Bommarito, and Josh Blackman. 2017. A general approach for predicting the behavior of the Supreme Court of the United States. *PloS one* 12, 4 (2017), e0174698.

[25] Bugeun Kim, Kyung Seo Ki, Donggeon Lee, and Gahgene Gweon. 2020. Point to the expression: Solving algebraic word problems using the expression-pointer transformer model. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online Event, 3768–3779.

[26] Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*. The Association for Computational Linguistics, San Diego California, USA, 1152–1157.

[27] Fred Kort. 1957. Predicting Supreme Court decisions mathematically: A quantitative analysis of the "right to counsel" cases. *American Political Science Review* 51, 1 (1957), 1–12.

[28] Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. The Association for Computer Linguistics, Baltimore, MD, USA, 271–281.

[29] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *CoRR* abs/1909.11942 (2019). arXiv:1909.11942 http://arxiv.org/abs/1909.11942

[30] Joshua Ong Jun Leang, Aryo Pradipta Gema, and Shay B Cohen. 2024. Comat: Chain of mathematically annotated thought improves mathematical reasoning. *arXiv preprint arXiv:2410.10336* (2024).

[31] Chuanyi Li, Jingjing Ye, Jidong Ge, Li Kong, Haiyang Hu, and Bin Luo. 2018. A novel convolutional neural network for statutes recommendation. In *PRICAI 2018: Trends in Artificial Intelligence: 15th Pacific Rim International Conference on Artificial Intelligence, Nanjing, China, August 28–31, 2018, Proceedings, Part I 15*. Springer, Springer, Nanjing, China, 851–863.

[32] Haitao Li, Qingyao Ai, Qian Dong, and Yiqun Liu. 2024. Lexilaw: A Scalable Legal Language Model for Comprehensive Legal Understanding. https://github.com/CSHaitao/LexiLaw

[33] Zhenwen Liang, Jipeng Zhang, Lei Wang, Wei Qin, Yunshi Lan, Jie Shao, and Xiangliang Zhang. 2022. MWP-BERT: Numeracy-Augmented Pre-training for Math Word Problem Solving. In *Findings of the Association for Computational Linguistics: NAACL 2022*. Association for Computational Linguistics, Seattle, United States, 997–1009. https://doi.org/10.18653/v1/2022.findings-naacl.74

[34] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Dublin, Ireland, 61–68.

[35] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[36] Yifei Liu, Yiquan Wu, Yating Zhang, Changlong Sun, Weiming Lu, Fei Wu, and Kun Kuang. 2023. ML-LJP: Multi-Law Aware Legal Judgment Prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, Taipei, Taiwan, 1023–1034.

[37] Yifei Liu, Yiquan Wu, Yating Zhang, Changlong Sun, Weiming Lu, Fei Wu, and Kun Kuang. 2023. Ml-ljp: Multi-law aware legal judgment prediction. In *Proceedings of the 46th international ACM SIGIR conference on research and development in information retrieval*. 1023–1034.

[38] Yougang Lyu, Zihan Wang, Zhaochun Ren, Pengjie Ren, Zhumin Chen, Xiaozhong Liu, Yujun Li, Hongsong Li, and Hongye Song. 2022. Improving legal judgment prediction through reinforced criminal element extraction. *Information Processing & Management* 59, 1 (2022), 102780.

[39] Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripabandhu Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. 2021. ILDC for CJPE: Indian Legal Documents Corpus for Court Judgment Prediction and Explanation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, Virtual Event, 4046–4062.

[40] Masha Medvedeva, Michel Vols, and Martijn Wieling. 2018. Judicial decisions of the European Court of Human Rights: Looking into the crystal ball. In *Proceedings of the conference on empirical legal studies*. Society for Empirical Legal Studies, Ann Arbor, MI, USA, 24.

[41] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

[42] Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Oyvind Tafjord, Ashish Sabharwal, Peter Clark, and Ashwin Kalyan. 2022. LILA: A Unified Benchmark for Mathematical Reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 5807–5832. https://aclanthology.org/2022.emnlp-main.392

[43] Stuart S Nagel. 1963. Applying correlation analysis to case prediction. *Tex. L. Rev.* 42 (1963), 1006.

[44] Joel Niklaus, Veton Matoshi, Pooja Rani, Andrea Galassi, Matthias Stürmer, and Ilias Chalkidis. 2023. LEXTREME: A Multi-Lingual and Multi-Task Benchmark for the Legal Domain. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, Singapore, 3016–3054.

[45] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543.

[46] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[47] Jeffrey A Segal. 1984. Predicting Supreme Court cases probabilistically: The search and seizure cases, 1962-1981. *American Political Science Review* 78, 4 (1984), 891–900.

[48] Octavia-Maria Sulea, Marcos Zampieri, Shervin Malmasi, Mihaela Vela, Liviu P. Dinu, and Josef van Genabith. 2017. Exploring the Use of Text Classification in the Legal Domain. In *Proceedings of the Second Workshop on Automated Semantic Analysis of Information in Legal Texts co-located with the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017), London, UK, June 16, 2017 (CEUR Workshop Proceedings, Vol. 2143)*, Kevin D. Ashley, Katie Atkinson, Luther Karl Branting, Enrico Francesconi, Matthias Grabmair, Marc Lauritsen, Vern R. Walker, and Adam Zachary Wyner (Eds.). CEUR-WS.org, London, UK. https://ceur-ws.org/Vol-2143/paper5.pdf

[49] Octavia-Maria Sulea, Marcos Zampieri, Mihaela Vela, and Josef van Genabith. 2017. Predicting the Law Area and Decisions of French Supreme Court Cases. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017, Varna, Bulgaria, September 2 - 8, 2017*, Ruslan Mitkov and Galia Angelova (Eds.). INCOMA Ltd., Varna, Bulgaria, 716–722. https://doi.org/10.26615/978-954-452-049-6_092

[50] Dhanasekar Sundararaman, Shijing Si, Vivek Subramanian, Guoyin Wang, Devamanyu Hazarika, and Lawrence Carin. 2020. Methods for numeracy-preserving word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online Event, 4742–4753.

[51] Suxin Tong, Jingling Yuan, Peiliang Zhang, and Lin Li. 2024. Legal Judgment Prediction via graph boosting with constraints. *Information Processing & Management* 61, 3 (2024), 103663.

[52] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *CoRR* abs/2302.13971 (2023). https://doi.org/10.48550/ARXIV.2302.13971 arXiv:2302.13971

[53] Shyam Upadhyay and Ming-Wei Chang. 2017. Annotating Derivations: A New Evaluation Strategy and Dataset for Algebra Word Problems. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, Mirella Lapata, Phil Blunsom, and Alexander Koller (Eds.). Association for Computational Linguistics, Valencia, Spain, 494–504. https://doi.org/10.18653/V1/E17-1047

[54] Shaurya Vats, Atharva Zope, Somsubhra De, Anurag Sharma, Upal Bhattacharya, Shubham Kumar Nigam, Shouvik Kumar Guha, Koustav Rudra, and Kripabandhu Ghosh. 2023. LLMs - the Good, the Bad or the Indispensable?: A Use Case on Legal Statute Prediction and Legal Judgment Prediction on Indian Court Cases. In *Findings of the Association for*

Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023. Association for Computational Linguistics, Singapore, 12451–12474.

[55] Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018. Translating a Math Word Problem to an Expression Tree. *CoRR* abs/1811.05632 (2018). arXiv:1811.05632 http://arxiv.org/abs/1811.05632

[56] Pengfei Wang, Yu Fan, Shuzi Niu, Ze Yang, Yongfeng Zhang, and Jiafeng Guo. 2019. Hierarchical Matching Network for Crime Classification. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer (Eds.). ACM, Paris, France, 325–334. https://doi.org/10.1145/3331184.3331223

[57] Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep Neural Solver for Math Word Problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, Copenhagen, Denmark, 845–854. https://doi.org/10.18653/V1/D17-1088

[58] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems* 35 (2022), 24824–24837.

[59] Chaojun Xiao, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. Lawformer: A pre-trained language model for Chinese legal long documents. *AI Open* 2 (2021), 79–84.

[60] Chaojun Xiao, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. Lawformer: A pre-trained language model for chinese legal long documents. *AI Open* 2 (2021), 79–84.

[61] Nuo Xu, Pinghui Wang, Long Chen, Li Pan, Xiaoyan Wang, and Junzhou Zhao. 2020. Distinguish Confusing Law Articles for Legal Judgment Prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 3086–3095. https://doi.org/10.18653/v1/2020.acl-main.280

[62] Linan Yue, Qi Liu, Binbin Jin, Han Wu, and Yanqing An. 2024. A circumstance-aware neural framework for explainable legal judgment prediction. *IEEE Transactions on Knowledge and Data Engineering* 36, 11 (2024), 5453–5467.

[63] Linan Yue, Qi Liu, Binbin Jin, Han Wu, Kai Zhang, Yanqing An, Mingyue Cheng, Biao Yin, and Dayong Wu. 2021. NeurJudge: A Circumstance-aware Neural Framework for Legal Judgment Prediction. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, Virtual Event, 973–982. https://doi.org/10.1145/3404835.3462826

[64] Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. 2022. MultiHiertt: Numerical Reasoning over Multi Hierarchical Tabular and Textual Data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 6588–6600. https://doi.org/10.18653/v1/2022.acl-long.454

[65] Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. 2018. Legal Judgment Prediction via Topological Learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 3540–3549. https://doi.org/10.18653/V1/D18-1390

[66] Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. 2021. TAT-QA: A Question Answering Benchmark on a Hybrid of Tabular and Textual Content in Finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 3277–3287. https://doi.org/10.18653/v1/2021.acl-long.254

A Prompt Design for the Acquisition of Chain-of-Thought data

A.1 The Design of the Prompts

We manually designed 3 prompts that query ChatGPT to produce Chain-of-Thought data according to each data item from the LAC dataset. We randomly selected 50 out of a total of 2,048 data items and evaluated the quality of the results of different prompts. The information about the prompts is detailed in Table 12 and the specific contents of the prompts along with the English translations are in Table 13.

Table 12. Information of our manually designed prompt candidates.

	Definition of Crime Amount	Ground Truth Answer	Requiring Formula
P_1	True	False	False
P_2	True	True	False
P_3	True	True	True

A.2 The Evaluation of the Prompts

A.2.1 Annotator Background. We hired two postgraduate law students in criminal law studies (neither as an author of this paper) to carry out this evaluation. Both completely grasps how crime amounts should be calculated.

A.2.2 Evaluation Guidelines. We conduct the evaluation on a 50-item subset whose data items are randomly selected from the entire dataset. The evaluation follows a three-dimension rubric for the CoT obtained, including answer accuracy, reasoning quality, and language quality.

Answer Accuracy. To calculate the answer accuracy of the CoT of the total crime amounts, we extract the final answer from the obtained CoT data using precisely crafted regular expressions and compare it to the ground truth value. The metric is the overall accuracy of the 50-item evaluation subset. **Due to the fact that final answers are produced deterministically by regex extraction, no human evaluation agreement is required.**

Reasoning Quality. To assess reasoning quality, each annotator independently rates all 50 items in the subset, assigning a score of 0, 1, or 2 to every case on the scoring sheet. The assignment of scores should follow the following protocols:

(1) A score of 0 represents that the calculation process is fundamentally flawed, including errors such as misidentifying or omitting key monetary values, missing or fail to produce logically sound intermediate steps, which would make the resulting crime amount diverge from the ground-truth value.
(2) A score of 1 denotes partical correctness, which means that the CoT captures the main key amounts and shows reasonable arithmetic, yet contains secondary errors such as overlooking a smaller item or mishanlding unit conversions.
(3) A score of 2 signifies full correctness, which mean every monetary element is accurately extracted, all conversions and aggregations are performed without mistake, and the concluding total matches the ground truth exactly, acccompanied by a clear, coherent explanation.

Language Quality. To assess the overall clarity and conciseness of each answer, the same two annotators independently rate all 50 items, again using a 0-2 scale:

Table 13. Prompts for ChatGPT and their translations

	Prompt Content
P_1	以下是一个涉及金额的刑事案件。请你根据案件描述分析犯罪总额的计算逻辑，并给出详细分析，用中文回答。犯罪总额是指案件中犯罪人实际犯罪涉及金额的总价值，如果有返还、转卖等行为，不需从总价值中减去返还数额或者赔偿数额、不需加入转卖数额，不要重复计算。计算过程中只涉及加法或乘法，作答时先分析犯罪总额的计算逻辑，最终作答时请以“本案的犯罪总额为xxx元。”结尾。 The following is a criminal case involving monetary amounts. Please analyze the logic and calculate the total crime amount based on the case description. Provide a detailed analysis and the calculation formula that yields the given correct total crime amount. Your answer should be in Chinese. The total crime amount refers to the overall value of money involved in the crime committed by the perpetrator(s) in the case. If actions such as returns or resales are involved, there is no need to subtract the returned amount or compensation amount, nor to add the resale amount to the total value. Avoid redundant calculations. The calculation process should only involve addition or multiplication. Begin by analyzing the calculation logic for the total crime amount and conclude your answer with "The total crime amount in this case is XXX yuan."
P_2	以下是一个涉及金额的刑事案件，该案犯罪总额为xxx元。请你根据案件描述结合提供的犯罪总额的正确答案，分析犯罪总额的计算逻辑，并给出详细分析，用中文回答。犯罪总额是指案件中犯罪人实际犯罪涉及金额的总价值，如果有返还、转卖等行为，不需从总价值中减去返还数额或者赔偿数额、不需加入转卖数额，不要重复计算。计算过程中只涉及加法或乘法，作答时先分析犯罪总额的计算逻辑，最终作答时请以“本案的犯罪总额为xxx元。”结尾。 The following is a criminal case involving a monetary amount. The total crime amount of this case is XXX yuan. Please, based on the case description and the provided correct total amount of the crime, analyze the logic behind the calculation of the total crime amount. Provide a detailed analysis and answer in Chinese. The total crime amount refers to the total value of the money involved in the crime committed by the criminal(s) in the case. If there are any returns, resales, etc., do not subtract the returned amount or compensation amount, nor add the resale amount from the total value. Avoid double counting. The calculation process only involves addition or multiplication. First analyze the logic behind the calculation of the total crime amount, then conclude your answer with "The total crime amount in this case is XXX yuan."
P_3	以下是一个涉及金额的刑事案件，该案犯罪总额为xxx元。请你根据案件描述结合提供的犯罪总额的正确答案，分析犯罪总额的计算逻辑，并给出详细分析以及获得给定正确犯罪总额的计算式，用中文回答。犯罪总额是指案件中犯罪人实际犯罪涉及金额的总价值，如果有返还、转卖等行为，不需从总价值中减去返还数额或者赔偿数额、不需加入转卖数额，不要重复计算。计算过程中只涉及加法或乘法，作答时先分析犯罪总额的计算逻辑，再给出得到犯罪总额正确答案的计算式，再最终作答时请以“本案的犯罪总额为xxx元。”结尾。 The following is a criminal case involving a monetary amount. The total crime amount of this case is XXX yuan. Please, based on the case description and the provided correct total amount of the crime, analyze the logic behind the calculation of the total crime amount. Provide a detailed analysis as well as the calculation formula used to arrive at the given correct total amount of the crime. Answer in Chinese. The total crime amount refers to the total value of the money involved in the crime committed by the criminal(s) in the case. If there are any returns, resales, etc., do not subtract the returned amount or compensation amount, nor add the resale amount from the total value. Avoid double counting. The calculation process only involves addition or multiplication. First analyze the logic behind the calculation of the total crime amount, then provide the calculation formula to arrive at the correct total crime amount. Finally, conclude your answer with "The total crime amount in this case is XXX yuan."

(1) A score of 0 represents poor quality. The text is confusing or overly verbose, key points are buried, and the answer is hard to understand without rereading.
(2) A score of 1 represents mediocre quality. The main idea is intelligible but phrasing is either somewhat redundant or vague in parts; some sentences may be missing or unclear.

Table 14. Evaluation results of different prompts.

	Answer Accuracy	Reasoning Quality	Language Quality
P_1	54.00%	62.5	96.0
P_2	72.00%	77.0	95.5
P_3	82.00%	89.5	97.0

(3) A score of 2 represents good quality. The response is succinct, well-structured, and unambiguously communicates the decision and its rationale.

To combine the 50×2 ratings into a single score, we proceed in three steps. First, for each item we compare the two annotators' scores. If they differ by at most one point, we simply average them, allowing half-point values such as 0.5 or 1.5. Second, if the two ratings are opposite (one annotator assigns 0 while the other assigns 2), the case is flagged as a conflict and the annotators begin a discussion until a final consensus is determined (0, 1, or 2). Third, after every item has a resolved score, we sum all 50 scores of the 50 items, which constitutes to a 0-100 spectrum.

The evaluation results of the three prompt candidates are presented in Table 14. Based on the results, we adopt P_3 in our main experiment.

A.3 Error analyses of obtained CoT

In this section, we analyze the errors occurred in ChatGPT's generated CoT contents (i.e., the errors represented by the 82.00% answer accuracy in Table 14) and conducted further experiments to analyze how such error rate impact the model trained on this data.

Error Analyses. We manually checked the tested 18% erroneous CoT items (i.e., 9 instances) and revealed three types of failure, which are shown in Table 15. The prompt we used to elicit CoTs is P_3 in Table 13. Even though the instructions explicitly caution the model against every error type discussed above, repeated runs reveal that these mistakes continue to appear at roughly the same frequency, simply migrating from one set of cases to another. This could suggest a chain-length effect: when the reasoning spans many hops or requires non-local back-references, small early extraction slips propagate through the chain and rarely get revised, yielding a stable error rate that simply shifts across instances. In short, the bottleneck could be the model's difficulty with maintaining long, self-consistent reasoning chains.

Table 15. Erroneous CoT types.

Error Type	Count	Typical Pattern
False Aggregation	2	Cases in which the calculation chain either omits key numbers or adds certain amounts more than once.
Contextual Misinterpretation	2	The CoT treats mistakenly aggregated key values that logically should not be added (e.g., resale values).
Hallucination	5	Spurious extra figures or arithmetic slips with otherwise correct inputs.

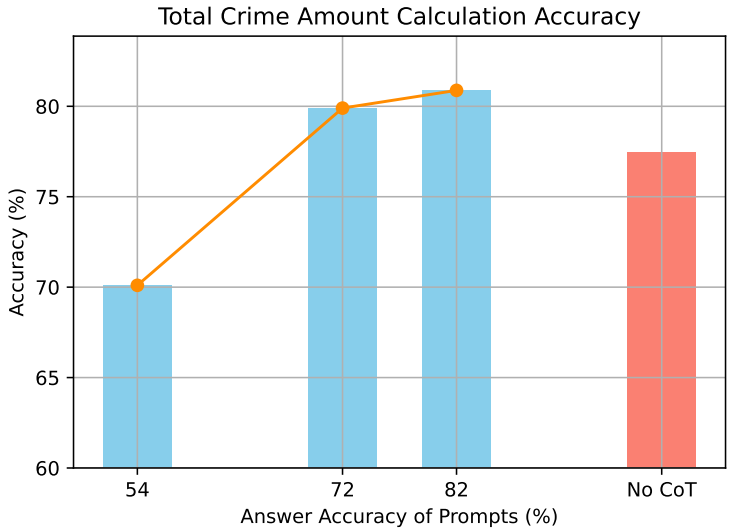


Fig. 8. Total crime amount calculation accuracy of FineChatGLM trained with LAC-CoT$_{54}$, LAC-CoT$_{72}$, and LAC-CoT$_{82}$, as well as the base setting that doesn't involve CoT in training data. Red bar represents the result of the base setting.

Impact on FineChatGLM. We explore the CoT's error rate and its relationships with the level of accuracy of the total crime amount calculation module, FineChatGLM. We constructed three alternative training sets, each derived from the CoTs produced with 3 different prompts, whose subsequent resulting CoTs have varied answer accuracy, reasoning quality, and language quality. These 3 training sets are denoted by LAC-CoT$_{54}$, LAC-CoT$_{72}$, and LAC-CoT$_{82}$, i.e. by the answer accuracy listed in Table 14.

Using the above described three training sets, we then adapted the same FineChatGLM backbone using prompt tuning on each set under identical hyper-parameters and evaluated it on the same held-out LAIC test split. We also include the baseline setting which does not include any CoT data in training as baseline comparison. Results are shown in Figure 8.

From the figure, we can conclude that the improvement curve is clearly sub-linear. When answer accuracy of the training data moves from 54% to 72%, the resulting prediction accuracy yields a gain of 9.80% (from 70.10% to 79.90%), whereas the jump from 72% to 82% delivers only a gain of 0.98% (from 79.90% to 80.88%). This pattern supports our decision to retain the 82.00% dataset in its current form, which means investing in exhaustive manual correction to reach near-perfect labels would have minimal impact on downstream performance while incurring substantial annotation cost.

B Prompts and Prompt Tuning Data Used in LJP Experiments

B.1 Prompt Tuning Data for ChatGLM2-6B

The training, validation, and testing datasets are adapted from those of the CAIL-2018 dataset accordingly, whose statistics are shown in Table 2. A data example is shown in Figure 9.

Model Inputs

凌海市人民检察院指控，… 被告人丛某某在…因琐事与鲁某某发生争执，后二人厮打，丛某某用拳头将鲁某某面部打伤。经凌海市人民医院法医司法鉴定所鉴定，鲁某某眼部损伤评定为人体轻伤二级。
The Linghai City People's Procuratorate charged that the defendant, Cong, had a dispute with Lu over trivial matters, which led to a physical altercation. Cong punched Lu in the face, causing injury. According to the forensic identification by Linghai City People's Hospital, Lu's eye injury was assessed as a Grade 2 minor injury.

Model Outputs

77;234;7

Fig. 9. A data example of prompt tuning dataset for ChatGLM2-6B on LJP tasks.

B.2 Prompts for LexiLaw

We adapted the test dataset from CAIL-2018 into prompts to query LJP results from LexiLaw. A data example is shown in Table 16.

Table 16. Prompts used to prompt LexiLaw and their translations

	Prompt Content
P_{charge}	根据下列事实预测判决罪名。只需给出判决的罪名,请将罪名放在"判决罪名:"之后, 如"判决罪名:抢劫"。公诉机关指控:...被告人李某甲在...网吧...见被害人李某乙睡着在沙发上,其装有人民币3700元的钱包在裤子口袋内,李某甲便伸手去掏李某乙的钱包,在扯出一半时,被李某乙发现并喝止,李某甲迅速逃离现场。同年...李某甲再次到上述网吧睡觉时,被民警抓获。 Predict the accusation based on the following facts. Only provide the name of the crime after "Charge:". For example, "Charge: Robbery". On ... defendant Li A attempted to steal a wallet containing 3700 yuan from victim Li B, who was asleep at ... Cafe... Li A was caught mid-action by Li B and fled. On... Li A was arrested by police while sleeping at the same cafe.
$P_{article}$	根据下列事实预测判决法条。只需给出判决的法条号,请将法条号放在"判决法条:"之后, 如"判决罪名:刑法第234条"。公诉机关指控:...被告人李某甲在...网吧...见被害人李某乙睡着在沙发上,其装有人民币3700元的钱包在裤子口袋内,李某甲便伸手去掏李某乙的钱包,在扯出一半时,被李某乙发现并喝止,李某甲迅速逃离现场。同年...李某甲再次到上述网吧睡觉时,被民警抓获。 Predict the relevant law article based on the following facts. Only provide the code of the article after "Article:". For example, "Article: No. 234 of the Criminal Code". On ... defendant Li A attempted to steal a wallet containing 3700 yuan from victim Li B, who was asleep at ... Cafe... Li A was caught mid-action by Li B and fled. On... Li A was arrested by police while sleeping at the same cafe.
P_{time}	根据下列事实预测判决的刑期。只需给出判决的刑期月份数。无期徒刑或者死刑的直接回答。请将预测的刑期月份数或者无期徒刑或者死刑放在"判决刑期:"之后作答,如"判决刑期:18个月"。公诉机关指控:...被告人李某甲在...网吧...见被害人李某乙睡着在沙发上,其装有人民币3700元的钱包在裤子口袋内,李某甲便伸手去掏李某乙的钱包,在扯出一半时,被李某乙发现并喝止,李某甲迅速逃离现场。同年...李某甲再次到上述网吧睡觉时,被民警抓获。 Predict the sentence length based on the following facts. Provide the number of months for the sentence. For life imprisonment or the death penalty, answer directly. Place the predicted number of months, life imprisonment, or death penalty after "Sentence Length:" e.g., "Sentence Length: 18 months." On ... defendant Li A attempted to steal a wallet containing 3700 yuan from victim Li B, who was asleep at ... Cafe... Li A was caught mid-action by Li B and fled. On... Li A was arrested by police while sleeping at the same cafe.

Table 17. An example of data format for the training and inferencing of new baselines.

Prompt
Based on the facts below, determine the defendant's charge, the applicable statute, and the sentence. Select exactly one label for each category from those that appear in the training data, and reply in the following format: Charge: <charge label>; Statute: <article label>; Term: <term label>.
Query
Upon investigation and trial, it was established that at approximately 3:00 p.m. on 26 December 2013, police officers arrested the defendant Gao, outside the Daolong Internet Café on San'ao-ling Road in the XX Development Zone of Huizhou City. A bag containing a white crystalline substance was seized from his person; forensic analysis determined that the substance was methamphetamine with a net weight of 32.85 grams.
Response
Charge: <38>; Statute: <59>; Term: <6>.

Table 18. Key LoRA parameters for LLama3 and Qwen2.5

Category	**Hyper-parameter**	**Value (shared)**
Backbone	Precision	FP16
	Flash Attention	True
LoRA	Rank r	8
	Alpha α	32
	Target Modules	q_proj, v_proj
Optimizer	LR	1e-4
	Batch/GPU	8
	Max Steps	450

C LoRA Details for LJP Experiments

We adopt *LoRA* [22], a parameter-efficient fine-tuning scheme that inserts tiny rank-decomposition adapters into frozen weight matrices, allowing us to adapt 7-8B parameters backbones with less than 1% extra parameters and a fraction of the GPU memory and training time required for full fine-tuning. Hypter-parameters of our experiments are summarized in Table 18.

As for the data format, each training record is stored as **alpaca format** with three fields: `"instruction"`, `"input"`, and `"output"`. The `instruction` is the fixed prompt *Based on the facts below, determine the defendant's charge...* shown in Table 17; the `input` contains the case-fact paragraph; the `output` is a *single* line following the schema

Charge: <charge label>; Statute: <statute article number>; Sentence: <sentence label>|

where all three labels are drawn from the closed sets observed in the training data.

D In-Context Learning Details for LJP Experiments

In addition to LoRA, we evaluate a retrieval-free *in-context learning* (ICL) setup for both LLaMA-3-8B and Qwen-2.5-7B. The ICL template consists of a single instruction—identical to the prompt in Table 17, shortened to fit context—followed by one demonstrations and then the test fact; the demonstration uses the same constrained output schema, `Charge: <label>; Statute: <label>; Term: <label>` (as shown in Table 19). Decoding is deterministic and labels are extracted post-hoc with the same constrained regex used in fine-tuning. To ensure fairness, the prompt format and closed label set are kept identical across models.

Table 19. An example of *in-context learning* (ICL) prompt for inference (k=1). The last block is the test query, which is the same case as in Table 17.

Instruction
Based on the facts below, determine the defendant's charge, the applicable statute, and the sentence. Select exactly one label for each category from those that appear in the training data, and reply in the following format: `Charge: <charge label>; Statute: <article label>; Term: <term label>`. Use deterministic outputs and do not add any extra text.
Demonstration 1
Fact: The defendant entered a retail store at midnight and took electronics valued at about 18,500 RMB without payment. He was apprehended by security at the exit. *Answer:* `Charge: <07>; Statute: <23>; Term: <3>.`
Test Query (same case as Table 17)
Fact: Upon investigation and trial, it was established that at approximately 3:00 p.m. on 26 December 2013, police officers arrested the defendant Gao, outside the Daolong Internet Café on San'ao-ling Road in the XX Development Zone of Huizhou City. A bag containing a white crystalline substance was seized from his person; forensic analysis determined that the substance was methamphetamine with a net weight of 32.85 grams. **Answer:**

E Details of data used on different LLM SFT LJP solutions

The dataset we used consists of 2,000 cases randomly selected from the CAIL-2018-small dataset. These selected cases are annotated with their ground truth labels of charge, relevant law articles, and penalty terms, the same as the CAIL-2018 dataset.

To facilitate this line of experiments, we further annotate the total crime amounts for each of these cases by hand. Specifically, we hired 3 annotators for this process. All 3 of them are law students specializing in Chinese Criminal Law, and fully grasp how TCAs should be calculated. Two annotators label the TCAs of all 2,000 cases, and the third annotator determines the final TCA answer for cases where conflicts between the first two annotator occur. It is worth noting that when a case does not involve a total crime amount, the annotators are instructed to label ***N/A*** for such cases.

The well-annotated cases are then transformed into two SFT-formatted datasets, each corresponding to the requirement for its baseline model. Data examples of the two SFT-formatted datasets are shown in Table 20.

Finally, we use precisely crafted regex patterns on model outputs to extract labels in the answers. Then, we report Accuracy (Acc), Macro Precision (MaP), Macro Recall (MaR), and Macro F1 Score (MaF) as the results.

F Details of QR in other jurisdictions

F.1 Dataset

Since there is no ready-made, ground-truth benchmark outside China that jointly standardizes facts, applicable provisions, and sentence terms at scale, we construct a small, test-only set from UNODC SHERLOC case pages, retaining only the fields we can rely on across jurisdictions: a fact description and the penalty term reported on the page. Because the portal's volume is limited (e.g., 400+ cases from USA, which is already the most abundant jurisdiction in the database), far less than what are typically used to train traditional neural-based models (e.g., approximately 100k cases for CAIL-small), we therefore use these cases solely for held-out inference on LLM baselines, without training or validation on these data. Statistics of the cases are shown in Table 21.

Table 20. Data examples for the ChatGLM2 baseline, ChatGLM2+Inline-Reason and ChatGLM2+Two-Stage-Adaptation.

ChatGLM2+Inline-Reason w/o TCA
Instruction: The given text is the fact description of a criminal case. Please first calculate the total crime amount of the case. If no monetary amount is involved, reply with the exact string "No crime amounts involved.". Then, with reference to the total crime amount and based on the fact description, predict the charge, law article, and penalty term for this case. Please present LJP predictions in the following format: <charge>charge prediction</charge><article>article prediction</article><term>term prediction</term>.
Prompt: ... the defendant Wei quarreled with the victim Liu, over a trivial matter in a dormitory at ... During the altercation Wei struck Liu on the head with a beer bottle, causing an injury that was later certified as a minor injury under the official forensic classification.
Response: No crime amounts involved. <charge>intentional injury</charge><article>234</article><term>12</term>.

ChatGLM2+Inline-Reason w. TCA
Instruction: The given text is the fact description of a criminal case. Please first calculate the total crime amount of the case. If no monetary amount is involved, reply with the exact string "No crime amounts involved.". Then, with reference to the total crime amount and based on the fact description, predict the charge, law article, and penalty term for this case. Please present LJP predictions in the following format: <charge>charge prediction</charge><article>article prediction</article><term>term prediction</term>.
Prompt: ... the defendant stole 206 yuan in cash and an iPhone 5 (appraised value at 1,575 yuan) from the victim Tang, in ... the defendant voluntarily surrendered to the public-security authorities.
Response: The total crime amount is 1,781 yuan. <charge>theft</charge><article>264</article><term>3</term>.

ChatGLM2+Two-Stage-Adaptation w/o TCA Stage 1
Instruction: The given text is the fact description of a criminal case. Please calculate the total crime amount of the case. If no monetary amount is involved, reply with the exact string "No crime amounts involved".
Prompt: ... the defendant Wei quarreled with the victim Liu, over a trivial matter in a dormitory at ... During the altercation Wei struck Liu on the head with a beer bottle, causing an injury that was later certified as a minor injury under the official forensic classification.
Response: No crime amounts involved.

ChatGLM2+Two-Stage-Adaptation w. TCA Stage 1
Instruction: The given text is the fact description of a criminal case. Please calculate the total crime amount of the case. If no monetary amount is involved, reply with the exact string "No crime amounts involved".
Prompt: ... the defendant stole 206 yuan in cash and an iPhone 5 (appraised value at 1,575 yuan) from the victim Tang, in ... the defendant voluntarily surrendered to the public-security authorities.
Response: The total amount mentioned in the text is 1,781 yuan.

ChatGLM2+Two-Stage-Adaptation Stage 2
Instruction: The given text is the fact description of a criminal case. Based on the fact description, predict the charge, law article, and penalty term for this case. Please present LJP predictions in the following format: <charge>charge prediction</charge><article>article prediction</article><term>term prediction</term>.
Prompt: ... the defendant stole 206 yuan in cash and an iPhone 5 (appraised value at 1,575 yuan) from the victim Tang, in ... the defendant voluntarily surrendered to the public-security authorities.
Response: <charge>theft</charge><article>264</article><term>3</term>.

ChatGLM2
Instruction: The given text is the fact description of a criminal case. Based on the fact description, predict the charge, law article, and penalty term for this case. Please present LJP predictions in the following format: <charge>charge prediction</charge><article>article prediction</article><term>term prediction</term>.
Prompt: ... the defendant stole 206 yuan in cash and an iPhone 5 (appraised value at 1,575 yuan) from the victim Tang, in ... the defendant voluntarily surrendered to the public-security authorities.
Response: <charge>theft</charge><article>264</article><term>3</term>.

Table 21. Statistics of Sherloc dataset used.

Jurisdiction	#. Cases
Australia (AUS)	231
Canada (CAN)	171
USA	436

Table 22. Prison term prediction results of cases from different jurisdictions.

		AUS				CAN				USA			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
4	Qwen+QR	19.48	16.27	12.74	13.81	16.96	14.92	11.02	12.14	12.84	10.34	8.47	8.91
5	Qwen+NQR	14.29	12.38	9.07	9.95	10.53	9.31	6.78	7.46	9.40	8.17	5.62	6.21
6	LLaMA+QR	23.38	19.58	15.83	16.92	20.47	17.26	13.87	14.63	14.91	12.71	9.83	10.28
7	LLaMA+NQR	18.18	14.06	11.64	12.31	15.20	12.97	10.12	10.78	11.70	10.03	7.54	8.07

F.2 Models

We evaluate two LLMs, namely LLama3-8B and Qwen2.5-7B via **Ollama** in inference-only mode. Considering the fact that judicial generalization is atypically challenging because sentencing depends on jurisdiction-specific norms and thresholded quantities, small models would require careful prompt/feature crafting rather than generic prompting. Moreover, building a ground-truth benchmark demands expert annotation; given the limited volume of the SHERLOC slice, we treat it as *test-only* and do not train neural-based methods, focusing instead on how minimal prompting impacts zero-shot performance. Specifically, for each model, we run two prompting variants:

(1) **Base (No-RQ)**: directly predict the sentence term from the fact.
(2) **QR-Presence**: use *inline reasoning* to first identify whether a TCA is involved in a case and then instruct the model to issue a sentence while explicitly considering that presence/absence signal.

F.3 Metrics

We report the Accuracy, Macro Precision, Macro Recall, and Macro F1 Score on the penalty term prediction task, the same way as in our main experiments.

F.4 Results and Analyses

From Table 22 we can conclude that, on the SHERLOC test-only set, generic LLMs in a zero-shot setting perform noticeably below traditional supervised LJP systems on the sentencing task. Adding the QR-Presence cue consistently improves over the base prompt (No-RQ) across all three jurisdictions, indicating that even minimal, inline identification of quantitative factors stabilizes predictions.

G Combinging Math-Solver-Based TCA with Existing LJP

G.1 Dataset

We conduct this experiment on the CAIL-2018-small dataset, using the same training, validation, and test splits as in our main QR-LJP evaluation. Each case contains a fact description, charge label, law article label, and prison term. Since the dataset does not provide gold-standard TCA values, we

first select two advanced mathematical solvers, namely MathPrompter [23] and CoMAT [30], as alternative TCA computation modules. These solvers are responsible for calculating the total crime amount according to the aggregation rules defined in the legal domain.

To enable the solvers to operate effectively, we preprocess each fact description using precisely designed regex as extractor that identifies relevant monetary amounts. The extracted values are then fed into the chosen math solver, which outputs a computed TCA for the case. The resulting TCA values are then integrated into the subsequent LJP models.

G.2 Model

We evaluate four different LJP models with the solver-generated TCA values: two traditional neural-based models, i.e., TopJudge [65] and NeurJudge [62], and two large models, ChatGLM2 [21] and LexiLaw [32]. For the neural-based models, the TCA feature is integrated using the attention-based fusion mechanism described in our main QR-LJP design, where the magnitude-aware embedding of TCA is used to attend to law article representations and then concatenated with the textual representation from the Lawformer encoder before feeding into the classification and regression heads. For the large models, the TCA is incorporated by augmenting the case input in the response field with an inline reasoning statement explicitly stating the computed TCA and the intermediate reasoning steps, enabling the model to condition its legal reasoning on the quantitative information.

Each of the four LJP models is evaluated with TCA values computed by MathPrompter and CoMAT, resulting in eight distinct experimental settings. In all settings, the downstream architecture, hyperparameters, and training configuration remain unchanged from the model's original setup, ensuring that any performance differences are attributable solely to the choice of TCA calculation method.

G.3 Metrics

We report Accuracy, Macro Precision, Macro Recall, and Macro F1 Score for all three LJP subtasks, i.e., charge, law article, and prison term predictions, as the same in our QR-LJP main experiment.

G.4 Results and Analyses

The results are shown in Table 23. The results show that replacing the QR-LJP's dedicated TCA calculation module with existing off-the-shelf maths solvers leads to a consistent performance drop across all four LJP models, for all three tasks. While the two solvers are capable of performing general quantitative reasoning, they are not optimized for the statute-specific rules and exception handling required in legal TCA computation. In particular, we observed that solver-generated values occasionally deviated from the legally valid total by omitting context-dependent monetary amounts, misinterpreting unit conversions, or double-counting monetary values because of inability to exclude resale amounts, which in turn limits its utility in downstream judgment prediction.

The gap between off-the-shelf solver-based TCA and our tailored TCA module obtained via prompt tuning highlights the benefit of a domain-specific, modular design that encodes legal aggregation rules explicitly. Unlike general-purpose solvers, our TCA module is tailored to identify important key values, understand legal TCA logic, and calculate them in a deterministic and legally consistent manner. This specialization ensures that the quantitative information fed to the LJP stage is both accurate and aligned with judicial reasoning practices, which appears critical for achieving optimal performance.

H Details of the annotation of the oracle test dataset

For annotating the ground truth TCA for all items in the oracle test dataset, we employed two postgraduate students. Both annotators are proficient in calculating TCAs for cases. Each annotator

Table 23. LJP results of math solvers combined with LJP models.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	MP+ChatGLM2	78.10	35.88	31.51	34.83	77.40	36.70	33.81	32.94	31.02	17.17	16.22	16.09
2	MP+LexiLaw	58.73	43.94	18.93	21.60	5.17	21.98	0.91	1.70	10.92	33.24	8.44	4.11
3	MP+TopJudge	88.79	51.11	78.41	68.90	76.12	61.08	76.28	74.86	38.59	34.10	36.56	34.18
4	MP+NeurJudge	78.40	69.37	79.12	73.30	81.41	76.23	77.61	77.89	35.96	37.94	31.27	35.00
5	CoMAT+ChatGLM2	79.57	46.93	52.38	41.73	78.82	38.24	34.97	34.12	32.88	18.23	17.24	16.92
6	CoMAT+LexiLaw	61.74	47.18	21.62	24.35	9.71	24.03	2.63	3.76	13.42	31.18	11.03	5.21
7	CoMAT+TopJudge	79.86	62.18	79.41	69.93	71.23	72.37	76.87	76.12	37.76	33.44	35.57	34.24
8	CoMAT+NeurJudge	80.58	72.12	77.13	72.08	82.12	77.78	78.63	75.06	38.83	38.58	32.29	34.12
9	QR-LJP	86.69	84.30	85.87	84.04	80.47	78.87	78.92	76.36	39.61	37.49	36.27	36.29

Table 24. LJP results on oracle test dataset.

		Charges (%)				Law Articles (%)				Prison Term (%)			
		Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
1	TopJudge-oracle	80.30	62.18	78.70	76.40	77.80	70.92	74.33	69.95	39.10	34.35	35.02	32.48
2	TopJudge-original	80.30	62.18	78.70	76.40	77.80	70.92	74.33	69.95	36.40	33.21	33.94	31.57
3	NeurJudge-oracle	79.80	70.22	78.11	76.52	78.10	75.44	74.69	73.83	38.90	33.80	34.92	34.34
4	NeurJudge-original	79.80	70.22	78.11	76.52	78.10	75.44	74.69	73.83	36.90	33.50	35.20	33.35
5	CEEN-oracle	80.90	82.21	80.38	79.24	83.10	85.14	83.09	82.37	34.00	35.42	33.19	31.60
6	CEEN-original	80.90	82.21	80.38	79.24	83.10	85.14	83.09	82.37	33.10	34.98	32.77	31.21
7	GJudge-oracle	80.40	76.80	78.53	77.46	80.30	82.17	77.95	79.64	37.20	34.62	33.91	34.28
8	GJudge-original	80.40	76.80	78.53	77.46	80.30	82.17	77.95	79.64	36.60	34.21	33.62	33.18
9	QR-LJP-oracle	83.60	82.93	83.53	79.59	79.10	75.96	79.67	71.52	39.20	34.38	35.87	34.40
10	QR-LJP-original	83.60	82.93	83.53	79.59	79.10	75.96	79.67	71.52	39.10	34.22	34.74	33.73

reviewed the entire oracle dataset independently. In instances of disagreement, the annotators discussed the specific data item in question and reached a consensus. The final annotations used in our experiment represent the outcomes unanimously agreed upon by both annotators.

I Oracle experiments on existing SOTA LJP models

In this line of experiment, the dataset used is the oracle dataset used in Section 5.2.2. And we report Accuracy, Macro Precision, Macro Recall, and Macro F1 Score on four SOTA LJP models: TopJudge, NeurJudge, CEEN, and GJudge.

The results are shown in Table 24. The oracle-TCA setting yields consistent improvements prison term prediction. The gains are observed in prison term Macro-F1, reflecting the direct influence of precise TCA values on determining the correct sentencing interval, particularly for cases where statutory boundaries are tightly coupled with monetary tiers.

Nevertheless, the improvements are incremental rather than dramatic. Even with gold-standard TCA values, the rise in macro-level metrics is moderate, and a substantial number of misclassifications remain. This shows that while perfect quantitative information benefits the sentencing stage, it does not remove challenges in interpreting non-monetary evidence, understanding complex fact patterns, or reasoning over the interplay of aggravating and mitigating circumstances. In summary, higher TCA accuracy is helpful, especially for sentencing. But transformative improvements in LJP

performance will likely require parallel advances in both quantitative computation and broader legal reasoning capabilities.

J Comparison between existing CoT generation approaches of math solvers and our approach

Many existing methods for generating Chain-of-Thought (CoT) solutions focus primarily on improving the generation of CoT itself, such as enhancing prompt design, tokenization, or the reasoning steps involved. In contrast, our approach emphasizes downstream effectiveness, particularly enhancing the accuracy of TCA calculation. Our goal is to provide actionable and law-relevant reasoning steps that directly improve the quantitative aspects of crime severity calculation, which is essential for downstream tasks like sentencing prediction. Moreover, existing methods often focus on producing generic CoT or reasoning chains that are designed to solve complex problems more broadly, but they are not always designed with the practical requirements of TCA calculation in mind. Our method is tailored to ensure that the generated CoT is directly usable by downstream tasks, particularly TCA calculation, by conditioning on relevant legal factors and ensuring that the reasoning produced aligns with quantitative legal rules.

Rather than focusing on perfecting every step of reasoning, we prioritize ensuring that the reasoning is sufficient for downstream tasks, and that it provides the contextual legal insights required for accurate TCA computation.

J.1 Dataset

We use the LAIC dataset as in our main experiment, which contains ground truth total crime amount of each case.

J.2 Model

We applied two existing CoT generation methods and compared their results with our approach. In this experiment, we used:

(1) **Prompt-Tuned CoT generation** [23]: A method that generates CoT by tuning the prompt for general-purpose reasoning tasks, typically focused on mathematical or broad problem-solving contexts.
(2) **Schema-Based CoT generation** [30]: a schema-guided, slot-filling CoT that performs amount selection, unit normalization, and then aggregation to produce TCA.

J.3 Metrics

The accuracy of TCA prediction is used as the primary evaluation metric.

Table 25. TCA calculation accuracy of different CoT generation strategies.

Approach	Accuracy (%)
Prompt-Tuned CoT generation	77.94
Schema-Based CoT generations	67.65
Ours	80.88

J.4 Results and Analyses

As shown in Table 25, **FineChatGLM** attains the strongest TCA accuracy, followed by **Prompt-Tuned CoT**, with **Schema-Based CoT** trailing. The gap stems from different error profiles: Schema-Based CoT's slot-filling tends to *miss-count* in noisy narratives (e.g., phrasing ambiguity/imcomplete cues leads to under-count, repeated mentions leads to over-cound), whereas Prompt-Tuned CoT is more flexible but sometimes *over-includes* ineligible quantities or misses refunds/offsets. FineChat-GLM reduces both inclusion/exclusion and unit-normalization errors, producing more stable totals across long, multi-amount cases, due to the inclusion of legal knowledge presented during CoT generation.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009