R^2: A Novel Recall & Ranking Framework for Legal Judgment Prediction

Yuquan Le, Zhe Quan, Jiawei Wang, Da Cao, and Kenli Li, *Senior Member, IEEE*

***Abstract*—The legal judgment prediction (LJP) task is to automatically decide appropriate law articles, charges, and term of penalty for giving the fact description of a law case. It considerably influences many real legal applications and has thus attracted the attention of legal practitioners and AI researchers in recent years. In real scenarios, many confusing charges are encountered, which makes LJP challenging. Intuitively, for a controversial legal case, legal practitioners usually first obtain various possible judgment results as candidates based on the fact description of the case; then these candidates generally need to be carefully considered based on the facts and the rationality of the candidates. Inspired by this observation, this paper presents a novel Recall & Ranking framework, dubbed as R^2, which attempts to formalize LJP as a two-stage problem. The recall stage is designed to collect high-likelihood judgment results for a given case; these results are regarded as candidates for the ranking stage. The ranking stage introduces a verification technique to learn the relationships between the fact description and the candidates. It treats the partially correct candidates as semi-negative samples, and thus has a certain ability to distinguish confusing candidates. Moreover, we devise a comprehensive judgment strategy to refine the final judgment results by comprehensively considering the rationality of multiple probable candidates. We carry out numerous experiments on two widely used benchmark datasets. The experimental results demonstrate our proposed approach's effectiveness compared to the other competitive baselines.**

***Index Terms*—Natural language processing, legal artificial intelligence, legal judgment prediction, verification, ranking.**

I. Introduction

Legal artificial intelligence (LegalAI) [1], [2] aims to exploit deep learning techniques to aid legal tasks [3], [4], [5], [6]. Legal judgment prediction is a fundamental and crucial task in LegalAI. It aims to automatically determine the appropriate judgment results (e.g., law articles, charges, and term of penalty) for a given fact description. LJP considerably influences many legal applications, such as legal assistance systems. legal assistance systems are beneficial for legal practitioners because they can provide references to lawyers or judges. They also offer affordable and convenient consultations for nonlegal practitioners. Therefore, LJP has attracted much attention from legal practitioners and AI researchers in recent years.

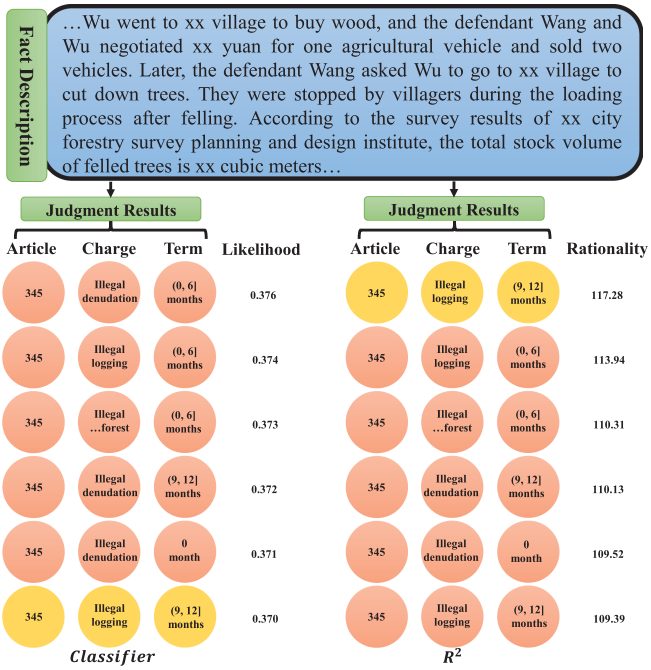


Fig. 1. Illustration of the LJP process in a civil law system. Generally, a judge conducts a professional analysis of the fact description, then determines the judgment results, including the applicable law articles, charges, and term of penalty. In this example, the *classifier* predicts the correct law article, but incorrectly predicts the charge and term. R^2 correctly predicts all the judgment results.

Most existing works [7], [8], [9], [10], [11], [12] treat LJP as a multi-task text classification problem. Fig. 1 illustrates an instance collected from the China Judgments Online.[1] This law case includes a fact description stating that the defendant cut a tree illegally. Generally, legal practitioners first determine the applicable law articles based on the context of fact descriptions, and then decide on charges according to facts and law articles. After, the term of penalty needs to be decided according to these information. The existing state-of-the-art works [7], [8], [12] follow this characteristic and focus on exploiting the topological dependency relationships among different

Manuscript received 27 June 2022; revised 21 November 2023 and 25 January 2024; accepted 29 January 2024. Date of publication 19 February 2024; date of current version 29 February 2024. This work was supported in part by 173 Program under Grant 2020-JCJQ-ZD-029, in part by the National Natural Science Foundation of China under Grant 61802121, and in part by the Natural Science Foundation of Hunan Province, China under Grant 2022JJ30159 and Grant 2023JJ20013. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Suma Bhat. *Yuquan Le and Jiawei Wang contributed equally to this work and are co-first authors. (Corresponding author: Zhe Quan.)*

The authors are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (e-mail: leyuquan@hnu.edu.cn; quanzhe@hnu.edu.cn; wangjiawei0531@gmail.com; caoda0721@gmail.com; lkl@hnu.edu.cn).

Digital Object Identifier 10.1109/TASLP.2024.3365389

[1] [Online]. Available: https://wenshu.court.gov.cn/

subtasks. Although they have achieved attractive progress in LJP, distinguishing confusing charges remains challenging. Many confusing charges are involved in real scenarios, and they are difficult to differentiate because the definitions of these charges are highly similar [9]. As revealed in Fig. 1, the *classifier* mispredicts the *crime of illegal logging* as the *crime of illegal denudation*, possibly because these two crimes are related to illegal tree felling. The subtle difference between the two charges is whether the perpetrator has ownership of the fallen trees (or the right to harvest them). To alleviate this issue, the literature [13] has manually defined discriminative attributes for charges, and designed attribute prediction auxiliary tasks. However, this work relies on manual labor performed by legal experts and are thus difficult to scale. Intuitively, for a legal case, especially for a controversial case, legal practitioners first obtain various possible judgment results as candidates based on the fact description of the case; then these candidates generally need to be carefully considered based on the facts and the rationality of the candidates. Therefore, we argue that exploiting the relationship between facts and candidates is beneficial for distinguishing confusing candidates. Unfortunately, it is impractical to consider the entire combination space of judgment results as candidates because the combination space is large.[2]

To address the above limitations, this paper presents a novel **R**ecall & **R**anking framework, dubbed as $\boldsymbol{R}^2$, which attempts to formalize LJP as a two-stage problem. The recall stage is used to obtain high-likelihood judgment results for a given case. Specifically, it consists of a *classifier* and a *high-likelihood sampling strategy* (HSS). The classifier obtains the likelihood distribution of labels, which are sampled via the HSS as candidates. Each candidate is automatically labeled based on the ground truth. Then, the ranking stage formalizes LJP as a ranking problem over the candidates, in which partially correct candidates are treated as semi-negative samples. It is composed of a *verification* and a *comprehensive judgment strategy* (CJS). For a given case, we follow the idea of verification techniques [14], [15] and implement a transformer-based [16] ranking module to judge the correctness of the candidates. The CJS is devised to refine the judgment results via multiple possible candidates for a law case during the test process. In a nutshell, the $\boldsymbol{R}^2$ has a certain ability to distinguish confusing judgment results, since it utilizes the relationships between fact descriptions and high-likelihood judgment results. $\boldsymbol{R}^2$ can also be used as a general framework, allowing a developer to replace the *classifier* with other on-shelf LJP methods. We list our major contributions here:

- We present a novel **R**ecall & **R**anking framework, which can effectively exploit the relationships between fact descriptions and possible candidates for legal judgment prediction.
- We adopt verification techniques to distinguish confusing candidates. A CJS is devised to refine the final judgment results via multiple possible candidates.
- Numerous experiments are carried out on two public benchmark datasets. The experimental results illustrate the effectiveness of $\boldsymbol{R}^2$ compared to the other competitive baselines.

[2]Take the CAIL-small dataset as an example. The CAIL-small dataset includes 103 charges, 119 law articles, and 11 non-overlapping sentence intervals, which can form a total of 103×119×11=134827 judgment result combinations.

TABLE I
MAIN MATHEMATICAL NOTATION

Notation	Description
$\boldsymbol{x} = \{w_1, \ldots, w_{n_f}\}$	Word sequence of a fact description
$\mathbf{Y}_\ell = \{\mathbf{y}_\ell^1, \ldots, \mathbf{y}_\ell^{n_\ell}\}$	Set of law articles categorical vector
$\mathcal{L} = \{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{n_\ell}\}$	Set of law articles description
$\mathbf{Y}_c = \{\mathbf{y}_c^1, \ldots, \mathbf{y}_c^{n_c}\}$	Set of charges categorical vector
$\mathcal{C} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{n_c}\}$	Set of charges description
$\mathbf{Y}_t = \{\mathbf{y}_t^1, \ldots, \mathbf{y}_t^{n_t}\}$	Set of term of penalty categorical vector
$\mathcal{T} = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_{n_t}\}$	Set of term of penalty description
$\mathbf{Y}_v = \{(0,0,0), \ldots, (1,1,1)\}$	Set of labels for ranking stage
$\mathcal{X}_{cand}$	Set of candidates
$\mathcal{Y}_{cand}$	Set of candidate labels

II. MODEL

Herein, the problem formulation is first presented in Section II-A. Then, we describe the architecture of the $\boldsymbol{R}^2$ model, which is illustrated in Fig. 2, including its recall stage and ranking stage. The recall stage is used to sample high-likelihood candidates as the input of the ranking stage (Section II-B). The ranking stage is tailor-made to distinguish confusing candidates for a given law case (Section II-C).

A. Problem Formulation

$\boldsymbol{R}^2$ formulates LJP as a two-stage problem. The details are as follows:

Recall stage: Considering a fact description $\boldsymbol{x}$, the recall stage aims to recall high-likelihood candidates $\mathcal{X}_{\text{cand}} = \{(\boldsymbol{\ell}_i, \boldsymbol{c}_j, \boldsymbol{t}_k)\}$ $(i \in \text{top}(\hat{\mathbf{y}}_\ell, m_\ell), j \in \text{top}(\hat{\mathbf{y}}_c, m_c), k \in \text{top}(\hat{\mathbf{y}}_t, m_t))$. These can be obtained via a function $\mathcal{X}_{\text{cand}} = \mathbf{\Gamma}_{\text{recall}}(\boldsymbol{\theta}_{\text{recall}}, \boldsymbol{x}, \mathcal{L}, \mathcal{C}, \mathcal{T})$. $\hat{\mathbf{y}}_\ell$, $\hat{\mathbf{y}}_c$ and $\hat{\mathbf{y}}_t$ represent the predicted probabilities of the judgment results (e.g., law articles, charges and term of penalty) produced by the classifier. The $\text{top}(\hat{\boldsymbol{y}}_\ell, m_\ell)$ function obtains the m_ℓ maximum probabilities from the $\hat{\mathbf{y}}_\ell$ and returns their corresponding subscripts. $\mathcal{L}$, $\mathcal{C}$, and $\mathcal{T}$ represent the text descriptions of law articles, charges, and term of penalty, respectively. The corresponding label of each candidate (e.g., $\mathbf{y}_v^{ijk} \in \mathcal{Y}_{\text{cand}}$) indicates the rationality of the judgment results[3], which takes values from the label set $\mathbf{Y}_v$. $\boldsymbol{\theta}_{\text{recall}}$ denotes the model parameters of the function $\mathbf{\Gamma}_{\text{recall}}$.

Ranking stage: The ranking stage is tailor-made design to distinguish confusing candidates for a given case. To this end, the ranking stage is to learn a function $\hat{\mathcal{Y}}_{\text{cand}} = \mathbf{\Gamma}_{\text{rank}}(\boldsymbol{\theta}_{\text{rank}}, \boldsymbol{x}, \mathcal{X}_{\text{cand}})$. $\boldsymbol{\theta}_{\text{rank}}$ denotes the model parameters of function $\mathbf{\Gamma}_{\text{rank}}$. During the testing process, $\hat{\mathcal{Y}}_{\text{cand}}$ is fed into the CJS, which comprehensively considers multiple candidates to produce the final judgment results (e.g., $\hat{\mathbf{y}}_\ell^v$, $\hat{\mathbf{y}}_c^v$, and $\hat{\mathbf{y}}_t^v$). The corresponding notations are shown in Table I.

[3]Please note that in this paper, the "judgment results" and "candidates" have the same meaning in most cases: the combination of the subtask labels. The main difference is that the judgment results focus on representing labels. The candidates represent textual descriptions of the corresponding labels.

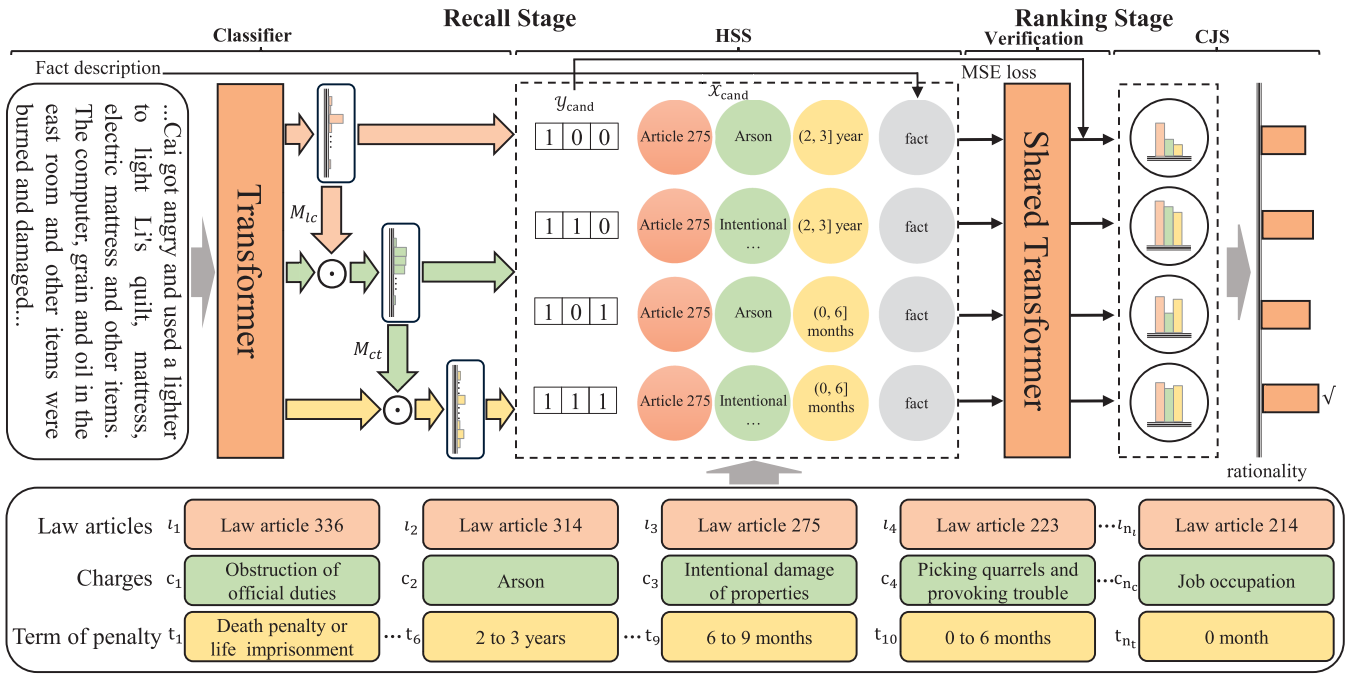


Fig. 2. Overview of the R^2 framework. The recall stage is composed of a classifier and a high-likelihood sampling strategy. The ranking stage consists of a verification and a comprehensive judgment strategy. In this figure, the hyperparameters m_ℓ, m_c, and m_t are set to 1, 2, and 2, respectively.

B. Recall Stage

The recall stage is composed of a classifier and a high-likelihood sampling strategy. The purpose of the classifier is to obtain the likelihood distribution of the judgment results. The HSS aims to construct effective candidates, which are then served ranking stage.

1) Classifier: The classifier consists of a fact encoder and a forward constraint (FC) operation. Given a fact description $\boldsymbol{x} = \{w_1, w_2, \ldots, w_{n_f}\}$, $w_i \in \boldsymbol{V}$, we utilize the fact encoder to encode the fact description into dense and continuous vectors, and represent law case semantic information.

$$\mathbf{h}_f = f_{\text{fact\_encoder}}(\boldsymbol{x}), \tag{1}$$

where $\mathbf{h}_f \in \mathbb{R}^d$. d is the embedding dimensionality of the fact representation. n_f is the maximum facts length. $\boldsymbol{V}$ indicates the word vocabulary. We choose the famous BERT model [17] as the encoder and the CLS vector as the fact representation.[4]

The classifier aims to provide the HSS with the likelihood distribution of different subtasks. The HSS recalls high-likelihood judgment results based on the corresponding likelihood distribution for further distinction during the ranking stage. For a certain subtask, the failure of the ground truth retrieval process results in failure of the ranking stage.[5] Fortunately, previous researchers [10] have found that the labels are entangled among different subtasks. Specifically, once the classifier knows the ground truth of the law article subtask, it can reduce the label space of the charge prediction subtask based on the relationships between law articles and charges, because law articles are usually related only to certain charges. As long as the selection value of m_c in the HSS stage is larger than the number of these certain charges, it is guaranteed that the candidates can recall the ground truth of the charge subtask. This property also applies to the subtask of the term of penalty. To this end, we design a forward constraint operation. The FC operation constrains the outcome probability by building the relationship matrix among subtasks. $\mathbf{M}_{\ell c} \in \mathbb{R}^{n_\ell \times n_c}$ represents the law-charge matrix, and $\mathbf{M}_{ct} \in \mathbb{R}^{n_c \times c_t}$ denotes the charge-term matrix. n_ℓ, n_c, and n_t denotes the number of labels for law article, charge and term, respectively. Here we take the matrix $\mathbf{M}_{\ell c}$ as an example to illustrate its construction process as follows:

$$\mathbf{M}_{\ell c}(i,j) = \begin{cases} 1, & \text{when}(\ell_i, c_j) \text{ is co-occurrence;} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Then, the relationship matrices are incorporated into the classifier.[6] The associated formula is as follows:

$$\begin{aligned} \hat{\mathbf{y}}_\ell &= \text{softmax}(\mathbf{W}_l \mathbf{h}_f), \\ \hat{\mathbf{y}}'_c &= \text{softmax}(\mathbf{W}_c \mathbf{h}_f), \\ \hat{\mathbf{y}}_c &= \text{norm}(\hat{\mathbf{y}}'_c \odot (\mathbf{M}_{lc}^T \hat{\mathbf{y}}_\ell)), \\ \hat{\mathbf{y}}'_t &= \text{softmax}(\mathbf{W}_t \mathbf{h}_f), \end{aligned}$$

[4]Please note that any traditional neural networks (e.g., CNN [18], GRU [19], and LSTM [20]) can also be used as the fact encoder without loss of generality.

[5]This problem can be alleviated by increasing the value of the hyperparameters (e.g., m_ℓ, m_c, and m_t) in the HSS. As the value of each hyperparameter increases, the amount of calculation in the ranking stage increases accordingly. Therefore, the recall stage need ensure not only that the values of the hyperparameters (e.g., m_ℓ, m_c, and m_t) are not too large but also that the candidates recalled by the HSS can cover the ground truth of different subtasks to the greatest extent possible.

[6]Please note that this design cannot constrain the label space predicted by the law article subtask, which we discuss in the section on the limitations and future directions.

Algorithm 1: High-likelihood Sampling Strategy.

input : A fact description: $\mathbf{x}$;
Likelihood distribution of the subtasks: $\hat{\mathbf{y}}_\ell$, $\hat{\mathbf{y}}_c$, $\hat{\mathbf{y}}_t$;
Set of law articles: $\mathcal{L} = \{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_{n_\ell}\}$;
Set of charges: $\mathcal{C} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{n_c}\}$;
Set of term of penalty: $\mathcal{T} = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_{n_t}\}$;
output: candidates dataset: $\text{Data}_{\text{cand}} = [\mathcal{X}_{\text{cand}}, \mathcal{Y}_{\text{cand}}]$;

$\mathcal{X}_{\text{cand}} \leftarrow [\,]$, $\mathcal{Y}_{\text{cand}} \leftarrow [\,]$;
for $i \leftarrow \text{top}(\hat{\mathbf{y}}_\ell, m_\ell)$ **do**
  for $j \leftarrow \text{top}(\hat{\mathbf{y}}_c, m_c)$ **do**
    for $k \leftarrow \text{top}(\hat{\mathbf{y}}_t, m_t)$ **do**
      $\mathbf{y}_v^{ijk} \leftarrow ()$;
      if is_true($\hat{\mathbf{y}}_\ell$, i) **then** $\mathbf{y}_v^{ijk}$.add(1);
      else $\mathbf{y}_v^{ijk}$.add(0);
      if is_true($\hat{\mathbf{y}}_c$, j) **then** $\mathbf{y}_v^{ijk}$.add(1);
      else $\mathbf{y}_v^{ijk}$.add(0);
      if is_true($\hat{\mathbf{y}}_t$, k) **then** $\mathbf{y}_v^{ijk}$.add(1);
      else $\mathbf{y}_v^{ijk}$.add(0);
      $\mathcal{X}_{\text{cand}} \leftarrow (\boldsymbol{\ell}_i, \boldsymbol{c}_j, \boldsymbol{t}_k)$;
      $\mathcal{Y}_{\text{cand}} \leftarrow \mathbf{y}_v^{ijk}$;
$\text{Data}_{\text{cand}} \leftarrow [\mathcal{X}_{\text{cand}}, \mathcal{Y}_{\text{cand}}]$;

$$\hat{\mathbf{y}}_t = \text{norm}(\hat{\mathbf{y}}'_t \odot (\mathbf{M}_{ct}^T \hat{\mathbf{y}}_c)), \tag{3}$$

where $\hat{\mathbf{y}}_l, \hat{\mathbf{y}}'_c, \hat{\mathbf{y}}'_t$ are the prediction results of the law article, charge, and term of penalty, respectively. $\hat{\mathbf{y}}_c$ is the charge prediction result of the $\hat{\mathbf{y}}'_c$ after it is constrainted by the law-charge matrix. $\hat{\mathbf{y}}_t$ is the term of penalty prediction result of the $\hat{\mathbf{y}}'_t$ after it is constrainted by the charge-term matrix. $\odot$ indicates the Hadamard product. $\mathbf{W}_\ell$, $\mathbf{W}_c$ and $\mathbf{W}_t$ are weight matrices. $\text{norm}(\cdot)$ is the normalization function. The classifier involves multiple optimization objectives. Thus, we adopt hyperparameters to control the strengths of the optimization for different objectives. The formula of the cost function is as follows:

$$\begin{aligned}\mathcal{L}_{\text{recall}} = {} & \boldsymbol{\lambda}_\ell \mathcal{L}_{\text{ce}}(\hat{\mathbf{y}}_\ell, \mathbf{y}_\ell) + \boldsymbol{\lambda}_c \mathcal{L}_{\text{ce}}(\hat{\mathbf{y}}'_c, \mathbf{y}_c) + \boldsymbol{\lambda}_t \mathcal{L}_{\text{ce}}(\hat{\mathbf{y}}'_t, \mathbf{y}_t) \\ & + \boldsymbol{\beta}_c \mathcal{L}_{\text{ce}}(\hat{\mathbf{y}}_c, \mathbf{y}_c) + \boldsymbol{\beta}_t \mathcal{L}_{\text{ce}}(\hat{\mathbf{y}}_t, \mathbf{y}_t),\end{aligned} \tag{4}$$

where $\boldsymbol{\lambda}_\ell$, $\boldsymbol{\lambda}_c$, $\boldsymbol{\lambda}_t$, $\boldsymbol{\beta}_c$ and $\boldsymbol{\beta}_t$ are the weight factors. $\mathcal{L}_{\text{ce}}$ indicates the cross-entropy loss function.

2) High-Likelihood Sampling Strategy: Here, we describe the candidate construction process, whose outputs serve the ranking stage. Generally, one original law case can be used to construct one positive sample and $n_\ell * n_c * n_t - 1$ negative or semi-negative samples, which may cause the ranking stage to suffer from the disadvantage of a large negative sample space.[2] Previous studies [21], [22] have observed that effective negative samples are critical to the success of ranking techniques. Towards this end, we develop a high-likelihood sampling strategy, inspired by beam search technique [23]. The beam search algorithm is a constrained breadth-first search method and is widely used in various NLP generation tasks [23], [24]. It retains the m sequences with the highest likelihood scores at each token and then uses these m sequences to generate the next token. Motivated by these observations, the HSS regards the labels of different subtasks as tokens according to their topological order.

TABLE II
CATEGORIES OF CONSTRUCTED SAMPLE FOR VERIFICATION (1 MEANS THAT THE CORRESPONDING LABEL IS CORRECT; 0 MEANS THAT THE LABEL IS INCORRECT)

Types of Samples	Articles	Charges	Term
Positive	1	1	1
Semi-negative	1	1	0
Semi-negative	1	0	1
Semi-negative	1	0	0
Semi-negative	0	1	1
Semi-negative	0	1	0
Semi-negative	0	0	1
Negative	0	0	0

Algorithm 1 shows the pseudo-codes of the HSS. Its inputs are fact description, the likelihood distribution of different subtasks, and the text descriptions of labels. The output consists of the candidates, serving for the ranking stage. Unlike the beam search, the HSS retains the m largest probabilities at each token. The purpose of this approach is to keep different types of labels in the candidates balanced for a given law case. Therefore, if there are m maximum probabilities at each token, a law case can generate m^3 samples. These samples can be divided into eight categories (see Table II), including positive, semi-negative, and negative samples, according to the correctness of the judgment results.

C. Ranking Stage

The ranking stage consists of a verification and a comprehensive judgment strategy. The verification aims to distinguish confusing candidates. The CJS is devised to refine the judgment results based on multiple high-likelihood candidates.

1) Verification: The verification techniques [14], [15], which originated from the field of math word problem [25], [26], are designed to judge the correctness of the high-likelihood solutions generated from the first-stage generative pre-training language model. Motivated by this observation, we follow this insight and implement a transformer-based [16] ranking module to capture the relationships between the fact description and possible candidates. Given the fact description and the candidates, the goal of verification is to judge the correctness of each possible judgment result.

Here we take a training sample $(\boldsymbol{x}, \boldsymbol{\ell}, \boldsymbol{c}, \boldsymbol{t}, \mathbf{y}_v)$ as an example. We utilize the fact encoder to extract the semantics of $\boldsymbol{x}$ and generate fact representation $\mathbf{h}_f \in \mathbb{R}^d$. Previous studies [27], [28] point out that leveraging the linguistic knowledge of labels can benefit legal tasks. Most existing LJP research treats labels as meaningless one-hot vectors and ignores this valuable information in LJP. Thus, we convert $\boldsymbol{\ell}$, $\boldsymbol{c}$ and $\boldsymbol{t}$ to dense vectors (e.g., $\mathbf{v}_\ell \in \mathbb{R}^d$, $\mathbf{v}_c \in \mathbb{R}^d$ and $\mathbf{v}_t \in \mathbb{R}^d$), and learn meaningful label representations as training progresses.[7]

Intuitively, when judging the rationality of a candidate, people may combine several kinds of information: (1) the matching

[7]In our experiments, the label embeddings are randomly initialized for simplicity. Please note that we can also get the initial label vector via the fact encoder.

degree of the fact description and multiple possible judgment results and (2) the rationality of the combinations of labels among different subtasks. To achieve this goal, we utilize the transformer [16] to learn their relationships due to its advantage in capturing arbitrary pairwise relationships in sequences. Specifically, we add the token embeddings (i.e.,$\mathbf{h}_f$, $\mathbf{v}_\ell$, $\mathbf{v}_c$ and $\mathbf{v}_t$) and its segment embeddings $\mathbf{E} \in \mathbb{R}^{4\times d}$ to obtain input of verification. Segment embeddings represent different types (i.e., facts, law articles, charges, and term). The formula is as follows:

$$\mathbf{I} = \text{concat}(\mathbf{h}_f, \mathbf{v}_\ell, \mathbf{v}_c, \mathbf{v}_t) + \mathbf{E}, \tag{5}$$

where $\text{concat}(\cdot)$ is concatenate function. Then, the input $\mathbf{I}$ is fed into the self-attention module, which can learn arbitrary pairwise relationships. The formula is as follows:

$$\mathbf{A}^{l-1} = \text{softmax}\left(\frac{\mathbf{I}^{l-1}\mathbf{W}^q(\mathbf{I}^{l-1}\mathbf{W}^k)^T}{\sqrt{d}}\right),$$
$$\mathbf{I}^l = \mathbf{I}^{l-1} + \text{FFN}(\mathbf{A}^{l-1}\mathbf{I}^{l-1}\mathbf{W}^v), \tag{6}$$

where $\mathbf{W}^q$, $\mathbf{W}^k$, and $\mathbf{W}^v$ are projection parameter matrices. $\text{FFN}(\cdot)$ represents a feed-forward module. Transformer encoding layers can be combined with multiple stack layers to fully exploit the relationships among arbitrary pairwise input. We take the last layer of representation $\mathbf{I}^l \in \mathbb{R}^{4\times d}$ and send it to the fully connected layer. The formula is as follows:

$$\hat{\mathbf{y}}_{v_i} = \mathbf{I}^l_{i+1}\mathbf{W} + \mathbf{b}, i \in [0, 1, 2], \tag{7}$$

where $\hat{\mathbf{y}}_{v_i}$ denotes the predicted probability of corresponding subtask. $\mathbf{W} \in \mathbb{R}^d$ indicates the weight matrix and $\mathbf{b}$ represents the bias. For the verification training process, we optimize the following loss function:

$$\mathcal{L}_{\text{rank}} = \mathcal{L}_{\text{mse}}(\hat{\mathbf{y}}_v, \mathbf{y}_v), \tag{8}$$

where $\hat{\mathbf{y}}_v$ is the prediction probability, which denotes the rationality of a specific judgment results. $\mathcal{L}_{\text{mse}}$ is the mean squared error function.

2) Comprehensive Judgment Strategy: Intuitively, people usually rethink a given case by considering the rationality of multiple possible candidates. Following this idea, we devise a comprehensive judgment strategy to simulate this process. Specifically, given a fact description and a possible candidate as inputs, the verification produces a three-element probability vector. Each probability value reflects the rationality of the corresponding label under this combination. A refined judgment can be made by probabilistically fusing the likelihoods of multiple high-likelihood candidates. Algorithm 2 shows the pseudo-codes of the CJS. First, the rationality of each label is the sum of the rationality values that appears under different combinations (lines 3–16). Then, the rationality of the candidates is refined (lines 17–18). Finally, we select the combination with the largest probabilities sum among the refined candidates as the comprehensive judgment result (line 19). The $\text{get\_top\_candidates}(\hat{\mathcal{Y}}_{\text{cand}}, m_{\text{cjs}})$ function obtains m_{cjs} maximum probabilities of candidates from $\hat{\mathcal{Y}}_{\text{cand}}$, and returns their corresponding subscripts. The $\text{get\_candidates\_prob}(\hat{\mathcal{Y}}_{\text{cand}}, (i, j, k))$ function gets the predicted probabilities of the corresponding candidate. The $\text{get\_max\_rationality}(\cdot)$ function selects the candidate with the largest rationality value as the prediction result.

Algorithm 2: Comprehensive Judgment Strategy.

input : Candidate probabilities for case via verification: $\hat{\mathcal{Y}}_{\text{cand}} = \{(p_{\ell_i}, p_{c_j}, p_{t_k})\}$, where $i \in \text{top}(\hat{\mathbf{y}}_\ell, m_\ell)$, $j \in \text{top}(\hat{\mathbf{y}}_c, m_c)$, $k \in \text{top}(\hat{\mathbf{y}}_t, m_t)$;

output: The prediction results after CJS: $\hat{\mathbf{y}}^v_\ell$, $\hat{\mathbf{y}}^v_c$, $\hat{\mathbf{y}}^v_t$;

1 rationality $\leftarrow$ [ ];
2 $\text{rationality}_\ell \leftarrow \{\}$, $\text{rationality}_\text{c} \leftarrow \{\}$, $\text{rationality}_\text{t} \leftarrow \{\}$;
3 **for** $(i, j, k) \leftarrow \text{get\_top\_candidates}(\hat{\mathcal{Y}}_{\text{cand}}, m_{\text{cjs}})$ **do**
4 $(p_{\ell_i}, p_{c_j}, p_{t_k}) \leftarrow \text{get\_candidates\_prob}(\hat{\mathcal{Y}}_{\text{cand}}, (\text{i}, \text{j}, \text{k}))$;
5 **if** $\boldsymbol{\ell}_i$ *in* rationality_ℓ **then**
6 $\text{rationality}_\ell[\boldsymbol{\ell}_i]$ += p_{ℓ_i};
7 **else**
8 $\text{rationality}_\ell[\boldsymbol{\ell}_i] = p_{\ell_i}$;
9 **if** $\boldsymbol{c}_j$ *in* $\text{rationality}_\text{c}$ **then**
10 $\text{rationality}_\text{c}[\boldsymbol{c}_j]$ += p_{c_j};
11 **else**
12 $\text{rationality}_\text{c}[\boldsymbol{c}_j] = p_{c_j}$;
13 **if** $\boldsymbol{t}_k$ *in* $\text{rationality}_\text{t}$ **then**
14 $\text{rationality}_\text{t}[\boldsymbol{t}_k]$ += p_{t_k};
15 **else**
16 $\text{rationality}_\text{t}[\boldsymbol{t}_k] = p_{t_k}$;
17 **for** $(i, j, k) \leftarrow \text{get\_top\_candidates}(\hat{\mathcal{Y}}_{\text{cand}}, m_{\text{cjs}})$ **do**
18 $\text{rationality} \leftarrow \text{sum}(\text{rationality}_\ell[\boldsymbol{\ell}_\text{i}], \text{rationality}_\text{c}[\text{c}_\text{j}], \text{rationality}_\text{t}[\text{t}_\text{k}])$
19 $\hat{\mathbf{y}}^v_\ell$, $\hat{\mathbf{y}}^v_c$, $\hat{\mathbf{y}}^v_t \leftarrow \text{get\_max\_rationality}(\text{rationality})$

III. Experiments

In this section, we carry out numerous experiments on two widely used datasets to examine the effectiveness of $\boldsymbol{R}^2$. Specifically, we design extensive experiments to investigate several research questions as follows.

- **RQ1:** Can $\boldsymbol{R}^2$ outperform other competitive baselines, including state-of-the-art models?
- **RQ2:** Whether the different parts of $\boldsymbol{R}^2$ are beneficial for its capabilities?
- **RQ3:** How does main hyperparameters (e.g., the hyperparameters in the HSS and, the number of candidates in the CJS.) affect $\boldsymbol{R}^2$?
- **RQ4:** How does the recall stage reinforce the performance of the classifier?
- **RQ5:** Can we replace the classifier with other existing LJP models?

A. Experimental Setup

1) Datasets: This section introduces the two widely-used benchmark datasets[8], called CAIL-small and CAIL-big, used in our experiments. These datasets contain numerous criminal cases, which are collected from China Judgments Online[9], and are published by the Chinese AI and Law challenge (CAIL2018) [29]. Each instance contains a fact description and judgment results (i.e., law articles, charges,

[8][Online]. Available: https://cail.oss-cn-qingdao.aliyuncs.com/CAIL2018_ALL_DATA.zip

[9][Online]. Available: https://wenshu.court.gov.cn/

TABLE III
STATISTICS OF CAIL-SMALL AND CAIL-BIG

Dataset	CAIL-small	CAIL-big
Train Datasets	101,685	1,588,894
Dev Datasets	13,787	-
Test Datasets	26,766	185,228
Law Articles	103	118
Charges	119	130
Term of Penalty	11	11

and term of penalty). Following the settings of most existing studies [9], [10], we perform the same preprocessing pipeline, whose code was published in LADAN.[10] Preprocessing involves the following steps. First, meaningless or incomplete data whose law documents contain fewer than ten words are removed. Second, we filter out the cases with multiple law articles or charges. Third, the long-tailed labels (those with fewer than 100 law articles or crimes) are filtered. Fourth, the sentence is divided into 11 non-overlapping intervals. Table III contains the statistical information of the CAIL-small and CAIL-big datasets after the above preprocessing.

2) Evaluation Metrics: In this section, we introduce the evaluation metrics in brief. Following the mainstream research [7], [9], [10], [12], we use four widely-used metrics in multi-class classification to evaluate the performance, including macro precision (MP), macro recall (MR), macro-F1 (F1) and accuracy (Acc.). Note that MP, MR, and macro-F1 are computed as the macro average of the above metrics correspondingly by taking all classes equally important.

3) Baselines: To evaluate the effectiveness of $\boldsymbol{R}^2$, we compare $\boldsymbol{R}^2$ against a variety of competing methods as follows.

- *HAN [30]:* This is a popular document classification model, which is designed to capture the hierarchic architecture of texts. It contains a word attention mechanism and a sentence attention mechanism, which are adopted in the word and sentence levels, respectively.
- *FLA [4]:* This model considers the interaction between law articles and fact descriptions via an attention-based neural network. The authors argue that corresponding law article information is beneficial for charge prediction.
- *Attribute-Attentive [13]:* This method alleviates the confusing charges problem by developing an attribute-attentive model. They manually summarize ten discriminative attributes to offer signals for distinguishing confusing charges, and design the attributes prediction auxiliary task.
- *TOPJUDGE [7]:* This model presents a topological multi-task learning framework. It aims to explore the dependencies between different subtasks. Specifically, they first treat the dependencies between different subtasks as the directed acyclic graph, and then incorporate this characteristic into LJP.
- *MPBFN-WCA [8]:* This method aims to explore the dependencies of subtasks results. It exploits interactions among subtasks according to topological order.
- *LADAN [9]:* This method is an end-to-end neural network for addressing the confusing charge problem. They utilize graph neural networks to extract subtle discrepancies among similar law articles, and obtain discriminative features from legal text via subtle differences.
- *NeurJudge [12]:* NeurJudge utilizes the vector rejection operation [31] to predict verdicts or sentences based on different circumstances of crime. This model is one of the SOTA method publicly released to the best of our knowledge.

4) Implementation Details: We implement the $\boldsymbol{R}^2$ framework with PyTorch.[11] The popular pre-trained bert-base-chinese model[12] is utilized to encode fact description. We freeze the parameters of the first six layers of the bert-base-chinese model to accelerate the training process since the training sets contain numerous cases. The relationship matrices (the law-charge matrix and the charge-term matrix) are constructed based on the statistical training set, and the co-occurrence threshold is set to 1. The number of classifier epoch is set as {5, 1} on the CAIL-small and CAIL-big datasets, respectively, since the purpose of classifier is to recall the correct label among the top-k prediction results. The number of verification epoch is set to 50 for the CAIL-small dataset and seven for the CAIL-big dataset to balance the training speed and performance of the $\boldsymbol{R}^2$. Table IV shows the settings and hyperparameters in detail. The $\boldsymbol{R}^2$ framework obtains high performance within a comparatively small hyperparameter search process. The experiments are run on the GeForce GTX 3090 GPU (24 GB). To eliminate randomness bias, multiple trials are conducted and the average results are reported.

Additionally, the settings of the other baselines are described here. For NeurJudge,[13] we use word2vec[14] to train word embeddings for initialization and the other hyperparameters remain unchanged from those provided in the officially released code.[15] The processing of the datasets is completely consistent with the work in the literature [9], [10], and the results of TOPJUDGE and MPBFN-WCA are collected from [9] to conduct a fair comparison. Similarly, the performances of the other baselines (e.g., FLA, HARNN, Attribute-Attentive, and LADAN) are collected from [10].

B. Main Performance Comparison (RQ1)

In this part, to demonstrate the superiority of the $\boldsymbol{R}^2$ framework, we compare the performance of our proposed method with several competitive methods, which contain state-of-the-art work. Tables V and VI show the results of the performance comparison on CAIL-small and CAIL-big datasets. From these tables, the following observations can be found.

[10] [Online]. Available: https://github.com/prometheusXN/LADAN

[11] [Online]. Available: https://pytorch.org/

[12] [Online]. Available: https://huggingface.co/bert-base-chinese

[13] Please note that we follow the dataset preprocessing approach used in most of the existing studies. However, this preprocessing method is different from that used in the NeurJudge paper, resulting in inconsistent data volumes. Therefore, we rerun the experiments using the official code of the NeurJudge paper for a fair comparison.

[14] [Online]. Available: https://github.com/tmikolov/word2vec

[15] [Online]. Available: https://github.com/bigdata-ustc/NeurJudge/tree/main/neurjudge

TABLE IV
IMPLEMENTED DETAILS OF $\boldsymbol{R}^2$

Recall stage		**Ranking stage**	
Parameter	**Setting**	**Parameter**	**Setting**
Epochs	{1, 5}	Epochs	{7, 50}
Batch size	24	Batch size	24
Learning rate	1e-5	Learning rate	1e-5
Optimizer	AdamW	Optimizer	AdamW
Cost function	CE	Cost function	MSE
The dimension of fact embedding	768	Layer of transformer	3
Length of sequence	512	Embedding size of articles, charges and term	768
Weight factor $\boldsymbol{\lambda}_\ell$, $\boldsymbol{\lambda}_c$, $\boldsymbol{\lambda}_t$, $\boldsymbol{\beta}_c$ and $\boldsymbol{\beta}_t$	1	The number of candidates in verifier	275
Hyperparameters of m_ℓ, m_c, and m_t	{5, 5, 11}	The number of candidates (m_{cjs}) in CJS	120

TABLE V
OVERALL PERFORMANCE COMPARISON ON CAIL-SMALL DATASET

Models	Law Article				Charge				Term of Penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
HAN	79.79	75.26	76.79	74.90	83.80	82.44	**82.78**	82.12	36.17	34.66	31.26	31.40
FLA	77.74	75.32	74.36	72.93	80.90	79.25	77.61	76.94	36.48	30.94	28.40	28.00
Attribute-Attentive	79.30	77.80	**77.59**	76.09	83.65	80.84	82.01	81.55	36.52	35.07	26.88	27.14
TOPJUDGE	79.88	**79.77**	73.67	73.60	82.10	**83.60**	78.42	79.05	36.29	34.73	**32.73**	29.43
MPBFN-WCA	79.12	76.30	76.02	74.78	82.14	82.28	80.72	80.72	36.02	31.94	28.60	29.85
LADAN	**81.20**	78.24	77.38	**76.47**	**85.07**	83.42	82.52	**82.74**	**38.29**	**36.16**	32.49	**32.65**
NeurJudge	79.95	76.25	77.09	75.29	83.08	82.14	81.74	81.44	36.23	34.24	31.47	32.43
$\boldsymbol{R}^2$	**83.85***	**83.11***	**83.48***	**82.04***	**89.31***	**87.48***	**88.22***	**87.61***	**41.62***	**40.89***	**37.01***	**38.32***
%Improvement	2.65%	3.34%	5.89%	5.57%	4.24%	3.88%	5.44%	4.87%	3.33%	4.73%	4.28%	5.67%

The numbers marked with stars (*) highlight the best results, and the bold number highlights the strongest baseline results.

TABLE VI
OVERALL PERFORMANCE COMPARISON ON CAIL-BIG DATASET

Models	Law Article				Charge				Term of Penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
HAN	95.63	81.48	74.57	77.13	95.58	85.59	79.55	81.88	57.38	43.50	40.79	42.00
FLA	93.23	72.78	64.30	66.56	92.76	76.35	68.48	70.74	57.63	48.93	45.00	46.54
Attribute-Attentive	96.12	85.43	80.07	81.49	96.04	88.30	80.46	83.88	57.84	47.27	42.55	43.44
TOPJUDGE	95.85	84.84	74.53	77.50	95.78	86.46	78.51	81.33	57.34	47.32	42.77	44.05
MPBFN-WCA	96.06	85.25	74.82	78.36	95.98	**89.16**	79.73	83.20	58.14	45.86	39.07	41.39
LADAN	**96.57**	**86.22**	**80.78**	**82.36**	**96.45**	88.51	**83.73**	**85.35**	**59.66**	**51.78**	**45.34**	**46.93**
NeurJudge	95.65	83.55	76.13	78.18	94.79	83.56	76.14	78.02	55.28	44.79	39.02	38.77
$\boldsymbol{R}^2$	**97.24***	**87.44***	**82.89***	**84.46***	**97.23***	**90.77***	**86.67***	**88.23***	**61.27***	**54.32***	**47.99***	**50.10***
%Improvement	0.67%	1.22%	2.11%	2.10%	0.78%	1.61%	2.94%	2.88%	1.61%	2.54%	2.65%	3.17%

The numbers marked with stars (*) highlight the best results, and the bold number highlights the strongest baseline results.

- Compared to FLA, the TOPJUDGE, MPBFN-WCA, and NeurJudge models perform better in most cases. This shows that it is beneficial for LJP tasks to utilize the topological information between subtasks. Moreover, MPBFN-WCA generally outperforms the TOPJUDGE in most cases. This improvement might be attributed to the utilization of interactive information among different subtasks.
- Attribute-Attentive and LADAN outperform FLA in most cases, especially for law articles and charges. A possible reason for this finding is that these two methods are specially designed to address confusing charges. The evidence also shows that the confusing charge problem is one of the main issues of LJP.
- $\boldsymbol{R}^2$ consistently yields the best performance on the CAIL-small and CAIL-big datasets in terms of all the metrics. Specifically, on the CAIL-small dataset, $\boldsymbol{R}^2$ outperforms the best methods *w.r.t.* the F1 score by 5.57%, 4.87% and 5.67% in the law article, charge, and term subtasks, respectively. On the CAIL-big dataset, $\boldsymbol{R}^2$ surpasses the most competitive models *w.r.t.* the F1 score by 2.10%, 2.88% and 3.17% in the law article, charge, and term

TABLE VII
ABLATION STUDY OF $\boldsymbol{R}^2$ ON CAIL-SMALL DATASET

Models	Law Article				Charge				Term of Penalty			
	Acc.	MP	MR	F1	Acc.	MP	MR	F1	Acc.	MP	MR	F1
$\boldsymbol{R}^2$	83.85	**83.11**	**83.48**	**82.04**	**89.31**	**87.48**	**88.22**	**87.61**	**41.62**	**40.89**	**37.01**	**38.32**
$\boldsymbol{R}^2$-CJS	**83.92**	81.76	82.80	81.18	89.02	86.86	87.90	87.12	41.37	40.29	36.72	37.89
$\boldsymbol{R}^2$-CJS-verification	82.61	79.71	80.24	78.78	87.89	86.35	85.55	85.34	40.36	35.89	34.25	33.09
$\boldsymbol{R}^2$-CJS-verification-FC	82.38	81.65	79.45	78.53	87.69	86.32	84.41	84.40	39.88	36.39	32.02	32.45

TABLE VIII
CO-OCCURRENCE STATISTICAL OF CHARGES AND LAW ARTICLES ON THE CAIL-SMALL DATASET

Charges	Law Articles	
Crime of arson	Article 114 (95.2%)	Article 115 (3.8%)
Crime of contract fraud	Article 224 (88.6%)	Article 266 (10.7%)
Crime of bombing	Article 125 (49.6%)	Article 114 (47.0%)

subtasks, respectively. This is a very encouraging result. The possible reason is that $\boldsymbol{R}^2$ can effectively learn the relationships between fact description and the possible candidates. By transforming LJP into a ranking problem over high-likelihood candidates, $\boldsymbol{R}^2$ has the ability to distinguish confusing judgment results. Moreover, $\boldsymbol{R}^2$ devises a comprehensive judgment strategy to mimic how people usually revisit a case based on the rationality of multiple high-likelihood candidates.

C. Study of the Different Components in R^2 (RQ2)

As mentioned before, $\boldsymbol{R}^2$ framework involves three major design ideas: a forward constraint operation, a verification, and a comprehensive judgment strategy. In this part, we first investigate the motivation behind the forward constraint operation. Then, ablation experiments are carried out to examine the effectiveness of these design ideas.

1) Study of Statistical Relation: In the real world, the labels of different subtasks are only partially correlated. This section explores the statistical relationships between the charges and law articles in the CAIL-small training set. Table VIII shows three charges, their corresponding top 2 high-frequency law articles and their corresponding proportions. The co-occurrences of the three crimes and the top 2 laws account for as much as 96% of the occurrences of various crimes, which shows that each charge is logically related to some fixed law articles. This inspires us to design the forward constraint operation.

2) Ablation Study: This section conducts an ablation study to further study the effect of different components on $\boldsymbol{R}^2$. Specifically, $\boldsymbol{R}^2$-CJS variant denotes that the CJS is removed. $\boldsymbol{R}^2$-CJS-verification variant denotes the classifier with the forward constraint operation. $\boldsymbol{R}^2$-CJS-verification-FC variant denotes the classifier without the forward constraint operation. Table VII reports the experimental results on the CAIL-small dataset. The following observations can be drawn from this table.

- When $\boldsymbol{R}^2$ removes the CJS, verification, and FC step by step, the performance shows a downward trend in terms of almost all metrics. This evidence indicates that these modules are beneficial for $\boldsymbol{R}^2$.

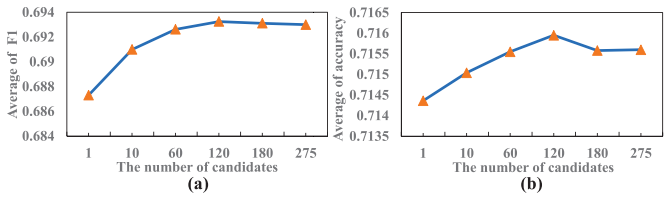


Fig. 3. Average accuracy and macro-F1 variations exhibited on CAIL-small as the number of candidates changes.

- The largest performance decrement is achieved by removing the verification. This phenomenon shows that performance improvement is mostly brought by verification. This is because the verification can capture the relationships between fact description and possible judgment results.

D. Study of Hyperparameters (RQ3)

Here, we study the important hyperparameters of $\boldsymbol{R}^2$. First, we study how the number of candidates in the CJS affects the resulting performance. Second, we study the effect of the number of high-likelihood labels used in the HSS.

1) Effect of the Number of Candidates in the CJS: To study whether the CJS is beneficial for $\boldsymbol{R}^2$, we vary the number of candidates in the CJS. Specifically, we search the numbers of candidates of CJS in the range of {1, 10, 60, 120, 180, 275}. Fig. 3 shows the experimental results on CAIL-small, wherein Fig. 3(a) is the effect of the number of candidates on the average F1 score, and the Fig. 3(b) is the effect of the number of candidates on the average accuracy. The abscissa is the number of candidates in CJS for a given law case, and the ordinate denotes the average accuracy and average F1 score of different subtasks. We can obtain the following conclusions from the above figures.

- As the number of candidates selected for a case increases, the model performance gradually increases. This shows that for a given case, every combination of high-likelihood candidates is beneficial to the final decision.
- As the number of candidates increases, the speed at which the performance improves gradually slows down and decreases when the number of candidates exceeds 120. This is because the candidates are sorted by their likelihood, and later candidates have lower likelihood values; adding them may negatively affect the ability of $\boldsymbol{R}^2$. To balance performance and calculation amount, 120 is selected by default in our experiments.

2) Effect of the Number of Labels in the HSS: To study how many high-likelihood candidates are appropriate in the HSS,

TABLE IX
ACCURACY (%) OF *CLASSIFIER* ON CAIL-SMALL

	Top-1	Top-3	Top-5	Top-7	Top-9	Top-11
Article	82.61	90.49	91.82	93.47	95.27	96.46
Charge	87.89	97.07	98.08	98.52	98.87	99.03
Term	40.36	78.11	93.82	98.15	99.45	100

TABLE X
STATISTICS OF ERROR CATEGORIES ON THE SMALL DATASET

Model	CC	PC	CW
classifier	9154	15526	2086
$\boldsymbol{R^2}$-CJS	9676	15168	1922
$\boldsymbol{R^2}$	9704	15176	1886

we consider the top-k accuracies achieved for different subtasks conducted on the CAIL-small dataset. The purpose of the HSS is to ensure the classifier covers more correct samples while generating as few samples as possible. In particular, we calculate the top-k accuracies achieved for different subtasks, with k in the range of {1, 3, 5, 7, 9, 11}. Table IX reports the results of the experiment. The following findings can be drawn from this table.

- The top-1 accuracies of different subtasks are unsatisfactory. This means that the classifier has difficulty in distinguishing confusing subtask labels.
- When the hyperparameters (e.g., m_ℓ and m_c) are set to 5, the top-k accuracies of the charge and law article subtasks exceed 91%. This evidence shows that for difficult law cases, the correct predicted charges or articles tend to be among the top 5 high-probability outcomes. When increasing hyperparameters, it generates more negative samples. To balance the computational efficiency and accuracy of the model, the hyperparameters m_ℓ and m_c are set to {5, 5} in this paper.
- For the term subtask, the top-1 accuracy is poor, indicating that this is a very difficult subtask. To better distinguish the terms, the hyperparameter m_t is set to the total number of labels in the term subtask (11) since the label space of this subtask is relatively small.

E. Study of the Ranking Stage (RQ4)

The ranking stage has a certain ability to distinguish confusing candidates. We study how the ranking stage is beneficial for the classifier in this section. To this end, we first analyze the error categories of the judgment results. Second, we conduct the case study to provide some intuitionistic examples.

1) Analysis of the Error Categories: We analyze the error categories of the judgment results to verify whether $\boldsymbol{R^2}$ can facilitate the classifier in this part. To this end, we divide the prediction results into three types: (1) completely correct (CC) results, in which the articles, charges, and terms are all predicted correctly; (2) partially correct (PC) results, in which the articles, charges, and terms are partially predicted correctly; and (3) completely wrong (CW) results, in which the articles, charges, and terms are all incorrectly predicted. Table X shows the statistical results produced by the $\boldsymbol{R^2}$, $\boldsymbol{R^2}$-CJS and classifier models on the CAIL-small dataset. From this table, we can draw the following conclusions.

- $\boldsymbol{R^2}$-CJS has fewer PCs or CWs than the classifier, and it has more CCs than the classifier. This evidence shows that our model can correct the prediction error of the classifier to some extent. The reason for this is that the verification can effectively capture the relationships between fact descriptions and label combinations, the relationships among intra-combinations, and the relationships among inter-combinations.
- $\boldsymbol{R^2}$ has fewer CWs than $\boldsymbol{R^2}$-CJS and more CCs than $\boldsymbol{R^2}$-CJS. This demonstrates that the CJS can further improve the performance of the model by considering the rationality of multiple high-likelihood candidates.

2) Case Study: We perform the case study to provide two intuitionistic examples from the CAIL-small testing dataset in this part. For each example, we show the top 8 likelihood label combinations of the predictions and highlight the completely correct judgment results with red text.

In the example shown in Fig. 4, the classifier fails to completely produce the correct prediction, while $\boldsymbol{R^2}$ completely predicts the correct results. The ground truths of the labels in this case are *article 275*, *crime of intentional damage of properties*, and *0 to 6 months*. This law case describes the fact that the defendant deliberately burned the victim's quilts, mattresses, computers and other items. The classifier misjudges the *crime of intentional damage of properties* as the *crime of arson*, which may be caused by the fact that the fact description contains arson-related content, which may further affect the judgment of the term of penalty. One of the differences between these charges is that the objects of the violations are different. The *crime of arson* usually violates public safety, while the *crime of intentional damage of properties* violates public and private property. Our framework treats the partially correct judgment results (e.g., (*article 275*, *arson*, *2 to 3 years*) and (*article 275*, *intentional damage of properties*, *2 to 3 years*)) produced by the *classifier* as semi-negative samples via the HSS. Then, the ranking stage is utilized to capture the relationships between the law document and the possible judgment results, and the confusing judgment results are optimized as negative or semi-negative samples. Therefore, $\boldsymbol{R^2}$ can obtain the correct judgment results.

As illustrated in Fig. 5, $\boldsymbol{R^2}$-CJS is partially predicted correctly, while $\boldsymbol{R^2}$ was predicted correctly. The ground truths of the labels in this case are *article 232*, *crime of intentional homicide*, and *death penalty or life imprisonment*. This legal example states the fact that the defendant caused the victim's death. $\boldsymbol{R^2}$-CJS misjudges the *crime of intentional homicide* as the *crime of negligent homicide* and misjudges *death penalty or life imprisonment* as *1 to 2 years*. A possible reason for this is that the consequences of these two charges are often a victim's death, so they are difficult to distinguish. The key difference is whether the action of the defendant is intentional. From the multiple possible prediction results of $\boldsymbol{R^2}$-CJS, we can observe that (1) the charges are mainly the *crime of negligent homicide* and the *crime of intentional homicide* and that (2) the sentence of *death penalty or life imprisonment* appears more frequently. This evidence shows that these multiple possible outcomes may benefit the final judgment results. $\boldsymbol{R^2}$ arrives at the correct

事实描述： 2015年6月20日11时许，被告人蔡某来到xx市xx区xx村李某家找李某的儿子李某甲说事，李某甲未在家。蔡某气急后用打火机将李某的被子、褥子、电褥子等物品给点着，东屋电脑、粮油等物品被烧毁坏。经xx市xx区涉案物品价格鉴定中心鉴定，被损毁物品价值9337元。

法条： 刑法275条
罪名： 故意毁坏财物罪
刑期： 0至6个月

Fact Description: At about 11 o'clock on June 20, 2015, the defendant Cai came to Li's house in xx Village, xx District, xx City to talk to Li's son Li A, but Li A was not at home. Cai got angry and used a lighter to light Li's quilt, mattress, electric mattress and other items. The computer, grain and oil in the east room and other items were burned and damaged. After appraisal by the Price Appraisal Center of the items involved in the case in xx District, xx City, the value of the damaged items was 9,337 yuan.

Law article: Article 275
Charge: Crime of intentional damage of properties
Term of penalty: 0 to 6 months

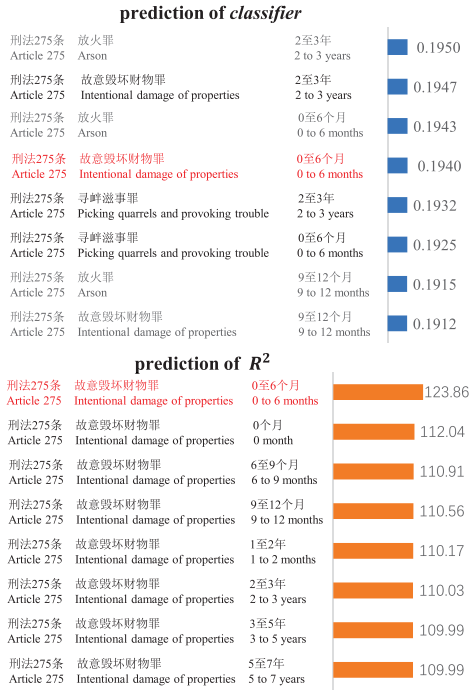


Fig. 4. Example from CAIL-small testing dataset: the *classifier* is partially predicted correctly, while the R^2 predicts correctly.

事实描述： ……被告人李某怕事情暴露，用右手从裤袋拿出随身带的一把尖刀向汤某捅去，直到把汤某捅到在地，被告人李某把正好压在汤某身子下面的一个黑色女式挎包拿出，发现里面有钥匙、现金、手机，于是李某用钥匙开门后把黑色女士挎包拿走逃离现场。……案发后不久，被害人汤某的亲属王某报案，被害人汤某已经死亡……

法条： 刑法232条
罪名： 故意杀人罪
刑期： 死刑或无期徒刑

Fact Description: ...Defendant Li was afraid that the matter would be exposed, so he took out a sharp knife from his trousers pocket with his right hand and stabbed Tang until he stabbed Tang to the ground. A black women's satchel was taken out and found that there were keys, cash and mobile phones in it, so Li opened the door with the key, took the black women's satchel and fled the scene. ...Shortly after the incident, Wang Mou, a relative of the victim Tang Mou, reported the case, and the victim Tang Mou had died...

Law article: Article 232
Charge: Crime of intentional homicide
Term of penalty: death penalty or life imprisonment

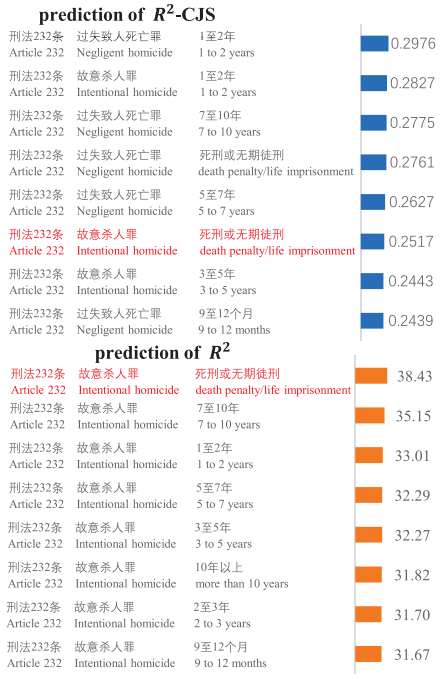
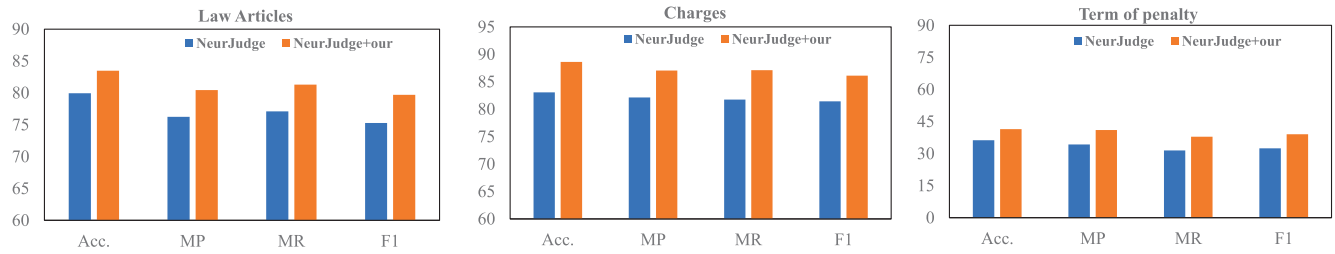


Fig. 5. Example from CAIL-small testing dataset: the R^2-CJS is partially predicted correctly, while R^2 predicts correctly.

judgment results. A possible reason for this phenomenon is that R^2 refines the final judgment results based on the rationality of multiple high-likelihood judgment results.

F. Study of the Universality of R^2 (RQ5)

In this section, we study the universality of the classifier of R^2. To figure out whether the R^2 framework also works for other LJP models. We replace the classifier with NeurJudge and keep the other modules unchanged, and we implement a variant of our method called *NeurJudge+our*. We experiment on the CAIL-small dataset. Fig. 6 shows NeurJudge and its R^2-enhanced models. From the figure, we can find that the *NeurJudge+our* outperforms NeurJudge in terms of all the metrics. This finding demonstrates that R^2 can be viewed as a general framework to some extent, allowing developers to replace the classifier using other on-shelf models.



a) Visual results of NeurJudge and NeurJudge+our on law article sub-task.

b) Visual results of NeurJudge and NeurJudge+our on charge sub-task.

c) Visual results of NeurJudge and NeurJudge+our on term of penalty sub-task.

Fig. 6. Visual results of NeurJudge and NeurJudge + our on CAIL-small dataset.

IV. Limitations and Future Directions

This section discusses the shortcomings of our model and the future directions of LJP.

- *The recall stage affects the ranking stage:* R^2 is a two-stage framework. The performance of R^2 is highly dependent on its recall stage. Although we propose a forward constraint operation to alleviate this problem, this design cannot constrain the label space of the law article subtask. This is one of the limitations of the proposed method.
- *Long-tailed crimes or law articles:* In real-world scenarios, numerous rare crimes and law articles are encountered, and the data of these crimes and law articles are difficult to collect. In the original CAIL-small training dataset, the 20 highest-frequency charges (or articles) account for more than 46.47% (or 55.05%) of the total, and the 20 lowest-frequency charges (or articles) account for 0.29% (or 0.35%) of the total. This shows that this task faces an imbalance problem, which includes numerous long-tailed labels. Most existing studies [7], [8], [10], [12] have focused only on high-frequency crimes or law articles, filtering labels with frequencies below 100. This paper follows this setting and does not yet specifically consider rare crimes or law articles. Therefore, how to improve the effect achieved for long-tailed charges or law articles is still a challenge.
- *Multiple law articles or charges:* In actual scenarios, the perpetrator may violate multiple law articles, resulting in several crimes. Taking the original CAIL-small training dataset as a statistical example, the samples that violate multiple law articles account for more than 22% of the total, and the samples that commit several crimes account for more than 24% of the total. To the best of our knowledge, the vast majority of current studies [7], [8], [10], [12] fail to address this topic. They only filter samples related to multiple law articles or crimes. This paper follows this setting and has not yet designed a mechanism for this situation. Therefore, how to design models for multiple law articles or multiple charges is still an open problem.
- *Confusing charges:* There are many confusing charges, such as the *crime of contractual fraud* and the *crime of fraud*, as well as the *crime of gambling* and the *crime of opening a casino*. It is not easy to distinguish them, and this issue may be caused by their highly similar law articles. Our model can alleviate this problem via a certain procedure. However, the challenge remains.
- *The performance of term of penalty:* Existing researches [7], [8], [10], [12] perform poorly on term of penalty. Even though our framework achieves SOTA capability, the performance is still unsatisfactory. Existing literature divides sentences into 11 non-overlapping intervals and formalizes the term of penalty subtask as a classification problem. We only follow this setting and do not delve into possibly more appropriate forms (e.g., regression problem). Therefore, how to explore the term of penalty subtask and improve performance remains challenging.

V. Ethical Concerns

Over these years, LJP has become a valuable research direction and, thus attracted the attention of many scholars and legal experts. Due to the sensitivity of the research topic, some ethical issues need to be discussed. The first thing to note is that the LJP's research is not intended to replace legal experts. Its research significance includes but is not limited to the following. (1) It can assist legal experts as an intelligent system. It can provide some references for legal practitioners, and improve their work effectively. (2) Provide cheap legal consultation to those who are financially disadvantaged. We should be aware that the judgment results predicted by LJP may be errors, may casuing by the complicacy of legal documents. For example, numbers can affect sentence judgment, but it is difficult for algorithms to distinguish. Similar laws or charges may misjudge each other. New crimes or law articles will appear as society develops. Therefore, we should be alert to the usage scenarios of this technology and update the technology in a timely manner.

VI. Related Works

A. Legal Judgment Prediction Methods

This paper divides the LJP into two categories according to whether they solve several tasks simultaneously.

Single-Task Learning Approaches: In the branch of single-task learning, researchers have mainly focus on charge prediction [32]. Early works involve statistical methods [33], [34], [35], [36] or machine learning techniques [37], [38], [39], [40], [41]. Recently, charge prediction methods have focused mainly on solving confusing charges and low-frequency crimes

by meanings of powerful neural networks. For example, dynamic pairwise attention [42] explores pairwise attention for charge prediction. Hu et al. [13] manually design ten charge attributes as discriminative features to alleviate low-frequency and confusing charges. SECaps [43] automatically captures high-level generalized attribute features by devising a seq-caps layer. SAttCaps [44] devises a self-attentive dynamic routing mechanism to relieve long-tailed charges. Unlike the above works, our work aims to simultaneously solve the law article, charge, and sentence subtasks.

Multi-task Learning Models: In recent years, the approaches belonging to this branch have usually utilize the task dependencies among law articles, charges, and terms. Several studies jointly model charge prediction and law article prediction, aiming to improve crime prediction performance. FLA [4] jointly optimizes law articles and charge prediction by designing a matching task between the law articles and crimes. HMN [45] utilizes a hierarchical structure for articles and crimes to perform charge prediction. Other works aim to simultaneously solve the law article, charge, and term subtasks. LADAN [9] utilizes a graph neural network to capture the subtle discrepancies among similar law articles, which is beneficial for LJP. R-former [10] utilizes the dependencies of labels and formalizes LJP as a node classification task. Besides, it is more relevant to our work for researchers to explore the interaction between the labels of different subtasks. For example, TopJudge [7] introduces the topology of multiple legal subtasks into a multi-task learning framework. MPBFN-WCA [8] explores the dependencies among the prediction results obtained for various subtasks. Ma et al. [6] mine the interactions between the court and the plaintiff's claims in an encyclopedic manner. NeurJudge [12] predicts verdicts or sentencing based on different criminal circumstances. Although extensive studies have been carried out on LJP, there is rarely a way to comprehensively consider multiple possible judgment results before making a final judgment. Different from the above works, our proposed $\boldsymbol{R}^2$ is a novel two-stage framework. It utilizes verification techniques to distinguish confusing judgment results, and devises a CJS to consider the rationality of multiple possible candidates for a case.

B. Other Related Techniques

Our work is also related to other techniques, such as verification and label embedding.

Verification: Verification techniques [14], [15] are proposed in the word math task. It is designed to address the issue that originated from mathematical reasoning. When generating a solution, errors in the former can affect the latter. To resolve this issue, the verifier is presented to be trained on model-generated candidate solutions. The candidate solutions are automatically labeled according to the ground truths during the training process. The representative works belonging to this branch include [14], [15]. Shen et al. [14] jointly model the process of generating solutions and ranking completions into a unified framework. Cobbe et al. [15] first train a generated model to generated completions. Then, they design a verification model to determine the correctness of the model-generated solutions.

Label Embedding: Existing studies [27], [28] show that the rich linguistic knowledge of legal labels is beneficial for LJP. Charges or law articles are usually well-defined. Several studies have attempted to combine this information into models. LEAM [46] calculates the cosine similarity between words and labels and subsequently applies a convolutional layer to measure the attention scores of words. In literature [9], TF-IDF [47] is used to encode law articles, and a graph neural network is utilized to learn subtly different representations. NeurJudge [12] utilizes the vector rejection strategy to capture the similar or dissimilar features between labels. MPBFN-WCA [8] utilizes three different latent state matrices to represent law articles, charges, and sentences, respectively. Unlike the above works, we use label description as the model inputs to explore the relationships between fact description and possible judgment results through the transformer.

VII. Conclusion and Future Work

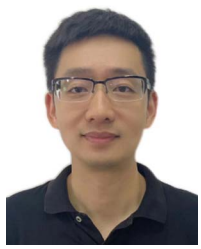
We attempt to formalize the LJP as a two stage problem and propose a novel two stage framework, called $\boldsymbol{R}^2$. The recall stage aims to sample multiple high-likelihood label combinations for a given law case. The ranking stage treats LJP as a rank problem over high-likelihood judgment results as candidates and introduces a verification method to distinguish confusing candidates. Moreover, the $\boldsymbol{R}^2$ can be used as a general framework, allowing the developers to replace the classifier with other on-shelf LJP methods. Experimental results show that $\boldsymbol{R}^2$ outperforms the competitive LJP models. We argue that $\boldsymbol{R}^2$ provides some insights for LJP. In the future, the areas that can be further optimized include but are not limited to the following directions. (1) A more efficient sample construction method can be designed since the sample construction process is the key to the ranking algorithm. (2) More powerful ranking models can be explored.

References

[1] J. Cui, X. Shen, and S. Wen, "A survey on legal judgment prediction: Datasets, metrics, models and challenges," *IEEE Access*, pp. 102050–102071, 2023.
[2] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun, "How does NLP benefit legal system: A summary of legal artificial intelligence," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 5218–5230.
[3] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun, "JEC-QA: A legal-domain question answering dataset," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 9701–9708.
[4] B. Luo, Y. Feng, J. Xu, X. Zhang, and D. Zhao, "Learning to predict charges for criminal cases with legal basis," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2727–2736.
[5] J. Ge, Y. Huang, X. Shen, C. Li, and W. Hu, "Learning fine-grained fact-article correspondence in legal cases," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 3694–3706, 2021.
[6] L. Ma et al., "Legal judgment prediction with multi-stage case representation learning in the real court setting," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 993–1002.
[7] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, and M. Sun, "Legal judgment prediction via topological learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3540–3549.
[8] W. Yang, W. Jia, X. Zhou, and Y. Luo, "Legal judgment prediction via multi-perspective BI-feedback network," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 4085–4091.
[9] N. Xu, P. Wang, L. Chen, L. Pan, X. Wang, and J. Zhao, "Distinguish confusing law articles for legal judgment prediction," in *Proc. 58th Annu. Meeting Assoc. Computat. Linguistics*, 2020, pp. 3086–3095.

[10] Q. Dong and S. Niu, "Legal judgment prediction via relational learning," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 983–992.
[11] Y. Feng, C. Li, and V. Ng, "Legal judgment prediction via event extraction with constraints," in *Proc. 60th Annu. Meeting Assoc. Computat. Linguistics*, 2022, pp. 648–664.
[12] L. Yue et al., "Neurjudge: A circumstance-aware neural framework for legal judgment prediction," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 973–982.
[13] Z. Hu, X. Li, C. Tu, Z. Liu, and M. Sun, "Few-shot charge prediction with discriminative legal attributes," in *Proc. 27th Int. Conf. Computat. Linguistics*, 2018, pp. 487–498.
[14] J. Shen et al., "Generate & rank: A multi-task framework for math word problems," in *Proc. Findings Assoc. Computat. Linguistics*, 2021, pp. 2269–2279.
[15] K. Cobbe et al., "Training verifiers to solve math word problems," 2021, *arXiv:2110.14168*.
[16] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Adv. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Computat. Linguistics*, 2019, pp. 4171–4186.
[18] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1746–1751.
[19] K. Cho et al., "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.* 2014, pp. 1724–1734.
[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[21] H. Valizadegan, R. Jin, R. Zhang, and J. Mao, "Learning to rank by optimizing NDCG measure," in *Proc. 22nd Int. Adv. Conf. Neural Inf. Process. Syst.*, 2009, pp. 1883–1891.
[22] J. Wang et al., "Learning fine-grained image similarity with deep ranking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1386–1393.
[23] A. Rush, Y.-W. Chang, and M. Collins, "Optimal beam search for machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 210–221.
[24] S. Wiseman and A. M. Rush, "Sequence-to-sequence learning as beam-search optimization," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1296–1306.
[25] Y. Wang, X. Liu, and S. Shi, "Deep neural solver for math word problems," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 845–854.
[26] D. Huang, J. Liu, C.-Y. Lin, and J. Yin, "Neural math word problem solver with reinforcement learning," in *Proc. 27th Int. Conf. Computat. Linguistics*, 2018, pp. 213–223.
[27] L. Xiao, X. Huang, B. Chen, and L. Jing, "Label-specific document representation for multi-label text classification," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2019, pp. 466–475.
[28] Z. Liu, C. Tu, and M. Sun, "Legal cause prediction with inner descriptions and outer hierarchies," in *China Nat. Conf. Chin. Computat. Linguistics*, 2019, pp. 573–586.
[29] C. Xiao et al., "CAIL2018: A large-scale legal dataset for judgment prediction," 2018, *arXiv:1807.02478*.
[30] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. Conf. North Amer. Chapter Assoc. Computat. Linguistics: Hum. Lang. Technol.*, 2016, pp. 1480–1489.
[31] Z. Wang, H. Mi, and A. Ittycheriah, "Sentence similarity learning by lexical decomposition and composition," in *Proc. 23rd Int. Conf. Computat. Linguistics*, 2016, pp. 1340–1349.
[32] Y.-H. Liu, Y.-L. Chen, and W.-L. Ho, "Predicting associated statutes for legal problems," *Inf. Process. Manage.*, vol. 51, no. 1, pp. 194–211, 2015.
[33] F. Kort, "Predicting supreme court decisions mathematically: A quantitative analysis of the "right to counsel" cases.," *Amer. Political Sci. Rev.*, vol. 51, no. 1, pp. 1–12, 1957.
[34] E. Mackaay and P. Robillard, "Predicting judicial decisions: The nearest neighbour rule and visual representation of case patterns," *Datenverarbeitung im Recht*, vol. 3, no. 34, pp. 302–331, 1974.
[35] S. S. Nagel, "Applying correlation analysis to case prediction," *Texas Law Rev.*, vol. 42, 1963, Art. no. 1006.
[36] R. Keown, "Mathematical models for legal prediction," *Computer/LJ*, vol. 2, 1980, Art. no. 829.
[37] C.-L. Liu, C.-T. Chang, and J.-H. Ho, "Case instance generation and refinement for case-based criminal summary judgments in chinese," *J. Inf. Sci. Eng.*, vol. 20, no. 4, pp. 783–800, 2004.
[38] C.-L. Liu and C.-D. Hsieh, "Exploring phrase-based classification of judicial documents for criminal charges in chinese," in *Proc. Int. Symp. Methodologies Intell. Syst.*. Springer, 2006, pp. 681–690.
[39] W.-C. Lin, T.-T. Kuo, T.-J. Chang, C.-A. Yen, C.-J. Chen, and S.-d. Lin, "Exploiting machine learning models for Chinese legal documents labeling, case classification, and sentencing prediction," *Int. J. Computat. Linguistics Chin. Lang. Process.*, vol. 17, pp. 49–68, 2012.
[40] D. M. Katz, M. J. Bommarito II, and J. Blackman, "A general approach for predicting the behavior of the supreme court of the United States," *PLoS One*, vol. 12, no. 4, 2017, Art. no. e0174698.
[41] O.-M. Sulea, M. Zampieri, S. Malmasi, M. Vela, L. P. Dinu, and J. van Genabith, "Exploring the use of text classification in the legal domain," in *Proc. Automat. Semantic Anal. Inf. Legal Texts (ASAIL) Workshop*, 2017, pp. 1–5.
[42] P. Wang, Z. Yang, S. Niu, Y. Zhang, L. Zhang, and S. Niu, "Modeling dynamic pairwise attention for crime classification over legal articles," in *41st Int. ACM SIGIR Conf. Res. & Develop. Inf. Retrieval*, 2018, pp. 485–494.
[43] C. He, L. Peng, Y. Le, J. He, and X. Zhu, "Secaps: A sequence enhanced capsule model for charge prediction," in *Int. Conf. Artif. Neural Networks*, 2019, pp. 227–239.
[44] Y. Le, C. He, M. Chen, Y. Wu, X. He, and B. Zhou, "Learning to predict charges for legal judgment via self-attentive capsule network," in *Proc. Eur. Conf. Artif. Intell.*, 2020, pp. 1802–1809.
[45] P. Wang, Y. Fan, S. Niu, Z. Yang, Y. Zhang, and J. Guo, "Hierarchical matching network for crime classification," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 325–334.
[46] G. Wang et al., "Joint embedding of words and labels for text classification," in *Proc. 56th Annu. Meeting Assoc. Computat. Linguistics*, 2018, pp. 2321–2331.
[47] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Inf. Process. Manage.*, vol. 39, no. 1, pp. 45–65, 2003.

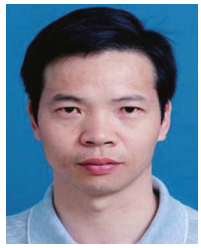
Yuquan Le received the B.S. degree from Nanchang University, Nanchang, China, in 2016, and the M.S. degree in 2019 from Hunan University, Changsha, China, where he is currently working toward the Ph.D. degree. He has authored or coauthored research papers in venues such as IJCAI, ECAI, IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, CIKM, ICASSP, and AACL. His research interests include natural language processing and legal artificial intelligence.



Zhe Quan received the Ph.D. degree in computer science from the University de Picardie Jules Verne, France, in 2010. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. He was with the National University of Defense Technology, Changsha, China, and was also a Postdoctoral Research Fellow with Berkeley and Livermore Lab, University of California, Berkeley, CA, USA. He has authored or coauthored a set of research papers in venues such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, AAAI, IJCAI, ICSC, BIBM. His research interests mainly include machine learning, artificial intelligence, and parallel and high-performance computing.

Jiawei Wang received the B.S. and M.S. degrees from the Guangdong University of Technology, Guangzhou, China, in 2017 and 2020, respectively. He is currently working toward the Ph.D. degree with Hunan University, Changsha, China. He has authored or coauthored academic papers in the most influential conferences such as *ACM Multimedia*, *NeurIPS*, and EMNLP. His research interests include causal inference, multimodal learning, and natural language processing.

Da Cao received the M.S and Ph.D. degrees from Xiamen University, Xiamen, China, in 2013 and 2017, respectively. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. Various parts of his work have been published in top forums including SIGIR, TOIS, and INS. His research interests include recommender systems, multimedia information retrieval, and natural language processing. He was a Reviewers for various journals and conferences including SIGIR, SIGKDD, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, CVPR, *ACM Multimedia*, *ACM Transactions on Knowledge Discovery from Data*, INS, *Knowledge-Based Systems*.



Kenli Li (Senior Member, IEEE) received the Ph.D. degree in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2003. From 2004 to 2005, he was a Visiting Scholar with the University of Illinois at Urbana-Champaign, Champaign, IL, USA. He is currently a Full Professor of computer science and electronic engineering with Hunan University, Changsha, China. He is also the Deputy Director with the National Supercomputing Center, Changsha. He was on the Editorial Boards of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON SERVICES COMPUTING, and IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS. His research interests include parallel computing, cloud computing, Big Data computing, and neural computing.