Knowledge-Based Systems 301 (2024) 112232

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Unlocking authentic judicial reasoning: A Template-Based Legal Information Generation framework for judicial views

Xiang Zhou [a,1], Yudong Wu [a,1], Ang Li [b,c,*], Ming Cai [b], Yiquan Wu [b,c], Kun Kuang [b]

[a] *Guanghua Law School, Zhejiang University, Hangzhou, 310008, Zhejiang, China*
[b] *College of Computer Science and Technology, Zhejiang University, Hangzhou, 310008, Zhejiang, China*
[c] *Law and AI Lab, Zhejiang University, Hangzhou, 310008, Zhejiang, China*

ARTICLE INFO

Keywords:
Legal AI
Judicial view generation
Judgment information on the merits
Multiple subtasks

ABSTRACT

In the field of legal AI, the research on the task of generating judicial views is relatively weak. Judicial views typically include judgment information on the merits such as charges, law articles, and sentencing circumstances. There are several issues with using universal algorithms to generate text, such as a lack of professional expression required for legal documents, and missing or incorrect judgment information on the merits. Judicial views often involve judgment information on the merits, rather than summarizing and excerpting known information. This could be why the application of conventional text generation algorithms may not produce satisfactory results. In actual judicial scenarios, judges generate their views step by step. Firstly, identify multiple judgment information on the merits in the judicial views, and then connect them with legal templates to achieve the generation of judicial views. This article proposes a **T**emplate **B**ased **L**egal **I**nformation **G**eneration (TBLIG) framework that generates higher quality judicial views based on the authentic judicial reasoning methods used by judges. The method consists of two steps: The first step involves determining the judgment information on the merits in the judicial view, while the second step involves synchronously generating a legal template based on the case. We have carried out a large number of experiments on the dataset of indictments and non-indictments. Through comparison with other models and ablation experiments, we have proved the effectiveness of our proposed method.

1. Introduction

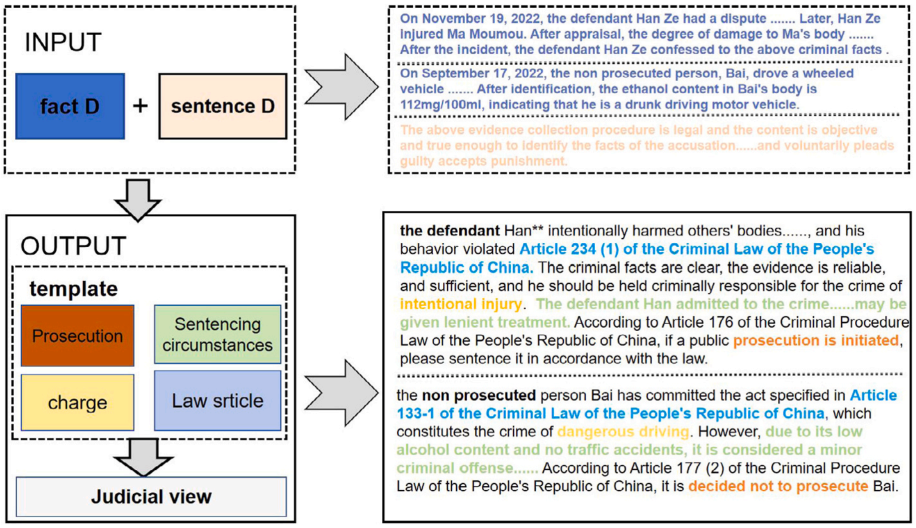
The application of deep learning technologies, particularly natural language processing, has benefited the field of law through the development of legal assistance systems. In the past, legal assistance primarily involved predicting outcomes related to legal judgments, such as charges, law articles, and sentencing circumstances. We refer to them as judgment information on the merits in the judicial views. According to statistics, these types of tasks account for over 60% of literature in the legal artificial intelligence field [1]. At the same time, real-life judicial scenarios place great emphasis on the reasoning behind the judicial process. To make machine-predicted judicial decisions trustworthy for users in the judicial field, sufficient reasoning must be provided. These are two different types of natural language processing tasks — the former focuses on classification algorithms while the latter belongs to generating algorithms. Current legal AI places more emphasis on the former, neglecting the fact that judicial scenarios also emphasize the explanation of judicial reasoning.

In the task of generating judicial views, we face issues with the text generated by general-purpose algorithms that lack professional expressions of legal terms, missing or incorrect judgment information on the merits, and other problems. Judicial views in a legitimate context often involve judgment information on the merits, rather than just summarizing and excerpting known information. This could be why the application of conventional text generation algorithms may not produce satisfactory results. Moreover, complex logical reasoning required for facts and relevant legal articles makes it challenging to generate judicial views, and there are few technical solutions available for this topic [1]. In actual judicial scenarios, judges generate their views step by step. Firstly, they complete multiple judgments information on the merits of the case. Secondly, they prepare the unique phrase template for each type of case, and then judgment information on the merits is connected using legal terminology with the template to construct the judicial view.

In order to simulate the process of judges constructing judicial views in the real world, we propose a **T**emplate **B**ased **L**egal **I**nformation

[*] Corresponding author.
E-mail addresses: zhouxiang6624@163.com (X. Zhou), 22202077@zju.edu.cn (Y. Wu), 22151344@zju.edu.cn (A. Li).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.knosys.2024.112232
Received 5 September 2023; Received in revised form 16 June 2024; Accepted 7 July 2024
Available online 22 July 2024



On November 19, 2022, the defendant Han Ze had a dispute .... Later, Han Ze injured Ma Moumou. After appraisal, the degree of damage to Ma's body .... After the incident, the defendant Han Ze confessed to the above criminal facts .

On September 17, 2022, the non prosecuted person, Bai, drove a wheeled vehicle .... After identification, the ethanol content in Bai's body is 112mg/100ml, indicating that he is a drunk driving motor vehicle.

The above evidence collection procedure is legal and the content is objective and true enough to identify the facts of the accusation......and voluntarily pleads guilty accepts punishment.

the defendant Han** intentionally harmed others' bodies......, and his behavior violated **Article 234 (1) of the Criminal Law of the People's Republic of China.** The criminal facts are clear, the evidence is reliable, and sufficient, and he should be held criminally responsible for the crime of **intentional injury**. **The defendant Han admitted to the crime......may be given lenient treatment.** According to Article 176 of the Criminal Procedure Law of the People's Republic of China, if a public **prosecution is initiated**, please sentence it in accordance with the law.

the **non prosecuted** person Bai has committed the act specified in **Article 133-1 of the Criminal Law of the People's Republic of China**, which constitutes the crime of **dangerous driving**. However, **due to its low alcohol content and no traffic accidents, it is considered a minor criminal offense......** According to Article 177 (2) of the Criminal Procedure Law of the People's Republic of China, it is **decided not to prosecute** Bai.

Fig. 1. An example of fact description and the judicial view from a legal document in a case.

Generation (TBLIG) framework to generates higher quality judicial views. The method consists of two steps: The first step involves determining the judgment information on the merits in the judicial view, while the second step involves synchronously generating a legal template based on the type of case. These two steps are then combined to form the final judicial view. Specifically, as shown in Fig. 1, it is composed of judgment information on the merits and a template. judgment information on the merits refers to information points that need to be judged by the judge, similar to the tasks of predicting charges and sentencing in traditional Legal Judgment Prediction (LJP) models. Beyond judgment information on the merits, judicial views also include a brief summary of the basic information about the case. Different judicial views have both individual case characteristics and similarities. The advantage of the proposed framework is that it fully considers the judgment information on the merits judgments in legal view text, which can significantly improve the prediction accuracy of the judgment information on the merits in a legal view text, and adjust the template appropriately according to different types of cases. We validate the above methods in a real-world dataset of indictment documents. A large number of experiments show that our TBLIG framework dramatically improves the quality of judicial view generation tasks.

In summary, we make three main contributions as follows:

- We redefine the text generation task in the legal scene, which is composed of judgment information on the merits and a template. It is a combination of a multi-classification subtask for judgment information on the merits and a task for generating templates. We identify 'judgment information on the merits' as a classification subtask, and generate 'template' as a generation task, thereby constructing the text paragraphs of judicial views with a 'cloze the blanks' style.
- We propose an enhanced subtask dependency prediction model that incorporates a more complex design. Our approach introduces a topological structure that captures the relationships between subtasks, effectively distinguishing between explicit and implicit subtasks. Additionally, we leverage task generation to provide valuable information for task classification, thereby enriching the information content of the topological structure.
- We conduct extensive experiments using a dataset comprising indictments and non-indictments. Through comparison with other models and ablation experiments, we demonstrate the effectiveness of our proposed method. While this paper specifically focuses on judicial reasoning during the review and prosecution of criminal cases, it is important to note that other judicial contexts often share similar structural characteristics. Therefore, the methods proposed in this paper possess the potential for generalization across various judicial scenarios.

2. Related work

2.1. Judicial view generation

Among the procedural tasks of judicial adjudication, the generation of judicial views holds significant importance within the legal scene. This task can be attributed to the task of generating legitimate texts. This task involves generating legally valid texts, which earlier approaches failed to adequately emphasize, neglecting the reasoning aspects inherent in judicial views [2], and there is no good mechanism to explain the results. The limitation of using the machine learning method is that its interpretation mechanism is not advanced enough, and there has been research on the key points of interpretation in the development of artificial intelligence [3]. To enhance the effectiveness of models generating judicial views, two primary approaches have been explored. One is to improve the input content of the model. For example, Yue et al. classify the case that was completely input in the past into ADC (ADjudging Circumstances) and facts related to SEC (sentencing Circumstances) [4]; Wu et al. stressed the significance of the plaintiff's claim for facts to the generation of judicial views [5]; The other is to refine the model's output mode. For example, Ye et al. employed a classification model to separately address the accusation conclusion within the reasoning process [6].

The commonness of these improvement methods is to increase the input and output information of the model or the structure of the model itself from the perspective of the law of the legal scene. On this basis, the generation model of judicial views still has significant room for improvement. In this paper, the text generation of the judiciary reasoning section is understood as a combination of multiple judgment information on the merits' subtasks and template generation. As long as the general template of judicial reasoning can be summed up, then the judgment information on the merits in it can be predicted using a better and more mature classification model, and then the generation model can be used to predict similar reasoning techniques, and finally, the prediction results can be combined.

2.2. Legal judgment prediction

The field of judicial scenario that is concerned by the current natural language processing field is mainly the task related to the prediction of legal judgment LJP. The typical LJP includes the prediction of a series of judgment results such as charges, articles of law, and sentencing. According to statistics, such documents account for nearly 60% of the legal tasks in the field of natural language processing [1]. This kind of task belongs to the relatively mature field of the current development of legal AI. The field of AI and law has been developed for 45 years, and machine learning and NLP methods are not only applied to predict legal judgments and tasks similar to judgments but also involve multi-dimensional prediction of cases [7]. Incorporating only specific pieces of information into the fact description to predict judicial views has certain limitations. Although some scholars have proposed a supervised contrastive framework that simulates the human judge's decision-making process by comparing different law articles within the same chapter, similar charges, and labels [8], this framework, despite considering the judge's decision-making process as much as possible, still appears somewhat singular in terms of the elements considered for the final judgment formed.

At the same time, the researcher noticed that the judgment prediction task in the judicial scenario is dependent, progressive, and sequential, so the model can be designed using the relevant learning method between multiple subtasks. The author who put forward the idea earlier used the directed acyclic topological graph to construct the relationship between various subtasks among judges [9]. After that, Yang et al. designed a multi-angle forward prediction and backward verification framework to utilize the dependency between multiple subtasks [10]; Yang et al. generated results by relational sequence of dependency between the articles of law and the charges [11]; Chen et al. viewed logically dependent tasks from the perspective of causal reasoning, tried to estimate the causal effect of tasks and use it to improve the performance of the model [12]; Dong et al. used the global consistency graph to model the LJP task as a node retrieval task on the graph [13], and produced a further generation method to optimize the joint learning of multiple dependent subtasks in the judicial scenario. These methods are useful to improve the prediction performance of some subtasks.

It is worth mentioning that some studies aimed to enhance the performance of multi-label classification by applying the sequence-to-sequence paradigm of generative models [14]. In contrast, our approach is exactly the opposite; we leverage the high accuracy of the classification model to improve the performance of the generation task. In this paper, our model also utilizes the dependencies between various judgment information on the merits' judgment subtasks. Unlike previous work, we take into account that the dependencies between multiple subtasks are more complex in real judicial scenarios. We distinguish between explicit and implicit subtasks, where explicit subtasks refer to the judgment information on the merits, while implicit subtasks (such as compulsory measures) serve as auxiliary tasks but may not be reflected in the judicial views. Furthermore, we introduce generative subtasks (such as sentencing circumstances) with topological dependencies, further enhancing the performance of the model.

3. Our approach

The methodology of model design is based on the characteristics of the text paragraphs of judicial views. Through analysis, we find the following characteristics:

(1) Judicial view is a semi-structured text. In the context of Chinese justice, including the legal documents produced by the courts, procuratorates, and other judicial authorities, the paragraphs related to judicial views are all the same, and all start with the judiciary reasoning section.

(2) Further analysis reveals that the generation of judicial views is a multi-step process, with each step generating judgment information on the merits, and the judiciary reasoning section is a collection of judgment information on the merits.

(3) If the judgment information on the merits is removed, the paragraph of the judicial view can be regarded as a reusable and learnable template between cases.

With the above analysis, the text generation of judicial view paragraphs targeted in our study is transformed into a structure of multiple classification subtasks and similar templates, which can improve the text generation of judicial view by combining the classification model of judgment information on the merits and the generation model of templates. We have analyzed the textual characteristics of this paragraph, summarized the judgment information on the merits, and referred to the removed part as a template for a similar template.

Specifically, we use the classification model with better prediction performance to predict the judgment information on the merits, use the text generation model to generate the templates, and finally merge them. Therefore, our judicial view generation model includes three modules: Classification generation information fusion encoder, Multi-angle topology judgment information on the merits' extractor, and Judicial view generator. The overall flow of the model is illustrated in Fig. 2.

3.1. Classification generation information fusion encoder

The information fusion encoder includes two core components: classification task encoder and generation task encoder. We have noticed that there is information useful for classification tasks in the template and sentencing circumstances. For example, the fact description contained in the template will contain many words closely related to the crime and the law. Naturally, we fuse the results of the generation task encoding shared by the two subtasks of template generation and sentencing circumstance generation with the results of the classification task encoding for the classified task. The information fusion encoder can help the model understand the fact description and sentencing statement from different angles, and learn a more comprehensive representation.

3.1.1. CNN encoder

For the classification task, we introduce the word vector pre-trained by word2vec [15] and use CNN [16] to extract text features. The CNN encoder mainly includes the following three components:

Word2vec: Firstly, fact description and sentencing instructions are reprocessed and expressed as word sequence, by retrieving word2vec, we can get the vector corresponding to each word, where each word w_i is converted to its corresponding vector word embedding $x_i, i \in [1, l]$, where l is the sentence length, where $\oplus$ is the vector splicing operator. The word vector is calculated by the following formula:

$$X_C = x_{1:l} = x_1 \oplus x_2 \oplus \cdots \oplus x_l \tag{1}$$

Convolution: We can take a sentence vector as a one-dimensional image and convolve it through a linear filter, where the width of the filter is equal to the dimension of word embedding. Then modify the window size h of the filter to obtain local features from different angles. Here, the window size refers to the number of words to be considered jointly, which is the vector formed by splicing h words in the window. The convolution operation can be performed through a convolution matrix and a deviation vector b. It can be calculated by the following formula. Where, $\cdot$ is the dot product operation, and c_i is the output vector of the ith convolution with h size window. We repeat the convolution operation and the word sequence can be expressed as $c^h = \{c_1^h, \dots, c_{l-h+1}^h\}$.

$$c_i^h = Convolution(x_{i:i+h-1}) = W \cdot x_{i:i+h-1} + b \tag{2}$$

Activating and pooling: Then we use the activation function RELU to reduce the over-fitting and reduce the parameter calculation. Because different convolution window sizes are used, we adopt the 1-max pooling strategy, that is, take the maximum value of the vector in the length dimension and add them together. Finally, we concat features from n different window size calculation and obtains $H_C = \{v^{h_1}, \dots, v^{h_n}\}$.

$$v^h = Max(RELU(c^h)) \tag{3}$$

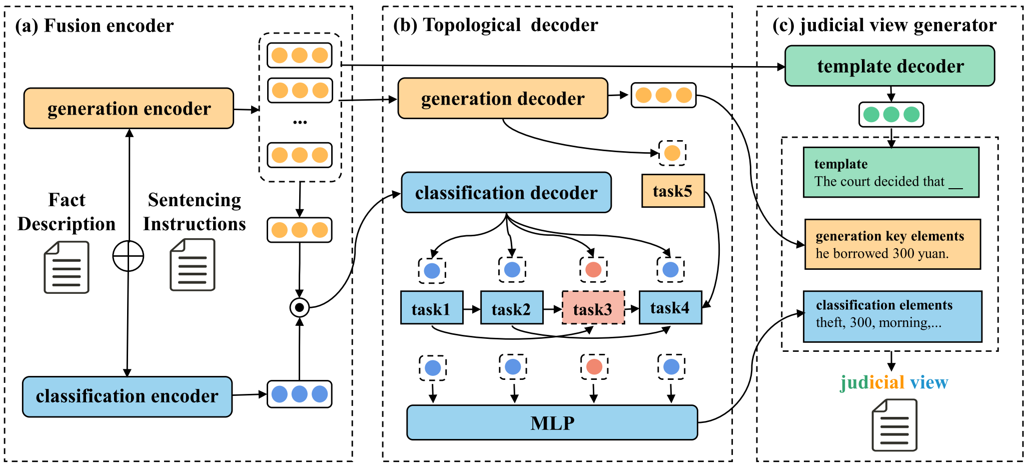


Fig. 2. Architecture of our model which consists of Fusion encoder, Topological decoder, and judicial view generator.

3.1.2. Transformer encoder

For the generation task, we use the Transformer encoder [17]. Transformer is a sequence of sequence network based on an attention mechanism. It can calculate in parallel, and will not forget important information with the increase of input length. It has more advantages in processing legal long text and generating tasks. Four core components in the Transformer are shown below:

Position encoding: Because the Transformer encoder is completely based on the self-attachment mechanism, the model does not possess the ability to capture sequential sequences like RNN [18]. We use the property of sin and cos to calculate the position code, which can save the number of parameters and take into account the absolute position and relative position information. The following formula can calculate the ith position encoding in a sequence. where d_{model} is the dimension of word embedding, and 2_j and 2_{j+1} denote even and odd dimensions-respectively. For each word embedding x_i, the final word vector representation is obtained by adding the position encoding of the corresponding position, as shown in the following equation. The final word sequence is represented as $X_T = \{p_1, p_2, \dots, p_l\}$

$$\begin{aligned} PE_{(i,2j)} &= sin(\frac{1}{10000^{\frac{2j}{d_{model}}}}) \\ PE_{(i,2j+1)} &= cos(\frac{1}{10000^{\frac{2j}{d_{model}}}}) \\ p_i &= x_i + PE_i \end{aligned} \tag{4}$$

Multi attention: Multiple attention builds different projection information in several different projection spaces through self-attention. The Transformer assigns three weights W_Q, W_K, W_V to obtain word–word relationships. W_Q finds the relationship between the current word and other input, W_K finds the relationship between other input and the current word, and W_V represents the value, and their dimensions are kept consistent. After linear mapping, the three weight W_Q, W_K, W_V are transformed into Q, K, V matrices, denoted as following:

$$\begin{aligned} Q &= Linear(X_T) = X_T W_Q \\ K &= Linear(X_T) = X_T W_K \\ V &= Linear(X_T) = X_T W_V \end{aligned} \tag{5}$$

Among them, $W_Q, W_K, W_V \in \mathbb{R}^{d_{model} \times d_{model}}$. Assuming that h is the number of heads in the multi-headed attention mechanism, the Q, K, V matrix is divided into h parts in the embedding dimension, and the changed dimension is d_k, satisfying $d_k \cdot h = d_m odel$, and the self-attention H_i for the ith head can be obtained by the following equation. In turn, the vector of input X_T after multi-head attention is expressed as the following equation. where W_O denotes the weight matrix of the attention obtained by different heads combined with the weight matrix of h results, $W_O \in \mathbb{R}^{d_{model} \times d_{model}}$.

$$\begin{aligned} H_i &= Attention(Q_i, K_i, V_i) = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}}) \\ X_{attn} &= MultiHead(Q, K, V) = [H_1, H_2, \dots, H_h] W_O \end{aligned} \tag{6}$$

Layer Normalization and Add: The skip connection expresses the output as a linear superposition of a nonlinear transformation of input and input. The skip connection can solve the problem of gradient dissipation and gradient explosion, and maintain effective backpropagation. LayerNorm normalizes the hidden layer in the neural network to standard normal distribution to speed up the training speed and accelerate the convergence. The following formula can calculate it:

$$\begin{aligned} X_{attn} &= X_T + X_{attn} \\ X_{attn} &= LayerNorm(X_{attn}) \end{aligned} \tag{7}$$

Feedforward: The feedforward consists of two layers of linear mapping and one layer with only the first layer having the RELU activation function. The operation of the feedforward layer can be expressed by the following formula:

$$\begin{aligned} H_T &= Activate(Linear(Linear(X_{attn}))) \\ &= Max(0, X_{attn} W_1 + b_1) W_2 + b_2 \end{aligned} \tag{8}$$

Among them, $W_1 \in \mathbb{R}^{d_{model} \times d_{model}}, W_2 \in \mathbb{R}^{d_{model} \times d_{model}}$ is the weight matrix, $b_1 \in \mathbb{R}^{d_{model}}, b_2 \in \mathbb{R}^{d_{model}}$ is bias.

3.1.3. Information fusion

The fact description and sentencing description become vector H_C after passing through the CNN encoder as input, and the input becomes vector H_T after passing through the transformer encoder. H_C contains different convolution window features and H_T contains the information of each token in the sequence. So we unify the vector size by using the linear layer to H_C and doing the max operation to H_T. Then add the two to obtain the fusion information, which is expressed as follows:

$$H_f = Linear(H_C) + Max(H_T) \tag{9}$$

3.2. Multi-angle topology judgment information on the merits generator

Existing research uses the topological dependency between the subtask results to solve the multi-task problems in the law, such as Topjudge [9], which uses the dependency between the three subtasks of law, charge, and sentence. However, due to ignoring the complexity and diversity of subtask types, their multi-task framework can only be used to solve multiple classification tasks from end to end. In this paper, we introduce a multi-angle topology network that leverages the characteristics of multiple subtasks and their interrelationships. We differentiate these subtasks based on whether they are explicit or implicit, as well as whether are classification or generation. By employing this framework, we enable subtasks to benefit from complex and diverse dependencies, resulting in a more comprehensive information gain.

3.2.1. *Construct multi-task topology*

By leveraging domain knowledge and drawing analytical conclusions from the judicial reasoning paragraph, we can systematically organize the information attributes of each key piece of information.

Distinguish between explicit and implicit tasks: Based on insights gained from the judicial domain, it is observed that the decision-making process of the judicial scenario is not fully reflected in the content of judicial reasoning. However, judgment information on the merits that does not appear in judicial reasoning can enhance judgment information on the merits' judgments. Thus, it is crucial to differentiate between explicit and implicit information within the judgment information on the merits. Explicit information refers to the judgment information on the merits that can be directly used to fill in the blanks in the template, and implicit task refers to the information that can help improve the explicit tasks.

Distinguish between classification and generation tasks: Based on the analysis of the judicial reasoning paragraph, the legal template script comprises the summary of the fact description and sentencing circumstances. These extracted pieces of information are rational and can aid in predicting judgment information on the merits. These elements, resembling rationales, exhibit greater flexibility and length compared to the key elements of classification tasks. Therefore, it is not suitable to categorize this information using classification tasks. Instead, it is more appropriate to consider them as generated subtasks. Once the subtask attribute is defined, the data flow between different subtasks can be determined based on logical order and the information gained from a judicial perspective, leading to the construction of a multi-subtask topology.

3.2.2. *Mutual constraint learning*

Inspired by Topjudge [9], we use LSTM to complete the information interaction between different subtasks. Each subtask uses one LSTM cell and updates the LSTM cell in turn according to the topological order. Due to the one-way topological relationships between legal subtasks, bidirectional LSTM [19] is not suitable for this task. In order to introduce generation subtasks and make them useful for classification subtasks, we use a transformer to extract important information and add it to the topology order of the topology map. The specific process is as follows: Based on the constructed topology and topological order, we can obtain a task list $T = \{t_1, t_2, \dots, t_n\}$ with an ordered subtask number of n. The predicted result of the jth subtask t_j is represented by the input representation H_f and the task set T' composed of dependent subtasks as conditions, where $T' \in T$ and for any subtask t_i.

(1) If t_i is a classification subtask, then the impact of t_i on the LSTM cell corresponding to the jth subtask t_j is represented by E_{t_i}:

$$E_{t_i} = W_{i,j} \begin{bmatrix} h_i \\ c_i \end{bmatrix} + b_{i,j} \tag{10}$$

where h_i, c_i are the hidden state and memory cell of task t_i, and h_j, c_j are the initialized hidden state and memory cell of task t_j. $W_{i,j}, b_{i,j}$ represent the weight matrix and bias in the transition process from task t_i to t_j.

(2) If t_i is a generation subtask, the input representation H_T obtained by the transformer encoder will be specifically used for the input representation of t_i subtask, but it cannot be used for subsequent subtasks. We use a transformer decoder to generate subtask representations simultaneously. The calculation method of the transformer decoder is similar to that of the transformer encoder. However, the query is generated by the token of the generation target to perform multi-head attention and layer normal operations similar to those in the transformer encoder. The K and V matrices are from the encoding results of the transformer encoder.

$$\begin{aligned} H_g &= MultiHead(X_g W'_Q, X_g W'_K, X_g W'_V) \\ H_g &= MultiHead(H_g W''_Q, H_T W''_K, H_T W''_V) \end{aligned} \tag{11}$$

where X_g represents the embedding of the target token of the generation subtask, W'_Q, W'_K, W'_V and W''_Q, W''_K, W''_V' represent the weight matrices of Q, K, V used in the two multi-head attention operations, and the layer norm and feedforward operations are omitted in the above formulas. After the max operation, H_g is used to obtain the effect of cell initialization on the jth subtask t_j. Here we only update the hidden state of the LSTM cell corresponding to the t_j subtask, but not the memory cell:

$$E_{t_i} = W_{i,j} \cdot Max(H_g) + b_{i,j} \tag{12}$$

After the influence of the dependent subtask set T' is taken into account, the LSTM cell initialization for the jth subtask t_j is updated as follows:

$$\begin{bmatrix} \bar{h}_j \\ \bar{c}_j \end{bmatrix} = \begin{bmatrix} h_j^\circ \\ c_j^\circ \end{bmatrix} + \sum_{t_i \in T'} (E_{t_i}) \tag{13}$$

where h_j°, c_j° represents the randomly initialized hidden state and the memory cell of the jth subtask.

3.2.3. *Independent representation and prediction of multiple subtasks*

(1) if t_j is a classification subtask, we use LSTM to process the input and initialized hidden state and memory cell. We use the final hidden state h_j as the representation of task t_j, as shown in the formula:

$$\begin{bmatrix} h_j \\ c_j \end{bmatrix} = LSTM(H_f, \begin{bmatrix} \bar{h}_j \\ \bar{c}_j \end{bmatrix}) \tag{14}$$

First, we apply a fully connected layer to obtain information about this subtask. Then, we use a sigmoid function to learn the probability distribution of the prediction:

$$\hat{y}_j = sigmoid(W_j h_j + b_j) \tag{15}$$

(2) If t_j is a generation subtask, we use the transformer decoder to process the input and use the output H_g^j as the representation of the specific task of subtask t_j. After softmax, it is mapped to the logits on the vocabulary to represent the probability distribution of the generated sequence:

$$\hat{y}_j = softmax(W_j H_g^j) \tag{16}$$

where W_j, b_j are the learnable weight matrix and bias for subtask t_j.

3.3. *Judicial view generator*

The judicial view generator mainly includes two modules: template generation and judgment information on the merits' filling. The less flexible and more versatile part of the judicial view can be obtained by generating the model, which can be regarded as a template without judgment information on the merits. Get more specific and accurate judgment information on the merits through the multi-angle topology judgment information on the merits' generator above, fill them in the appropriate position, and then get the final judicial view. This process can be illustrated in Fig. 3.

3.3.1. *Template generation*

We retain templates after removing the judgment information on the merits from judicial views. Specifically, each judgment information on the merits is assigned a specific word, which is then used to replace the corresponding judgment information on the merits. The purpose of the template generator model is to learn how to construct these templates. judgment information on the merits is predicted using the above Multi-angle topology judgment information on the merits' generator,

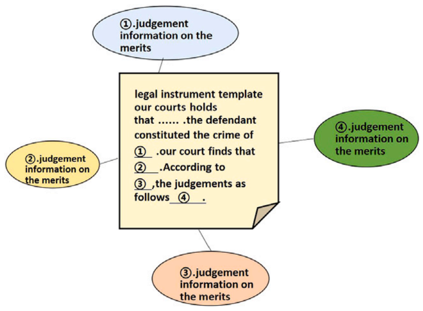

①.judgement information on the merits
legal instrument template our courts holds that ...... .the defendant constituted the crime of ① .our court finds that ② .According to ③ ,the judgements as follows ④ .
②.judgement information on the merits
④.judgement information on the merits
③.judgement information on the merits

Fig. 3. Generally, the judgment information on the merits and template contained in the first paragraph of a legal document.

while the template generation model generates the templates. Then we combine the two to form judicial views.

Similar to the previous generation subtask. We use the transformer model to generate templates. The same transformer encoder is used as the generation subtask, but different transformer decoders are used to obtain the specific representation of the template generation task H_g^{temp}. This enables template generation and subtask prediction to promote each other and improve the quality of judicial views. The probability distribution of the model generation template is calculated as follows:

$$\hat{y}_{temp} = softmax(W_{temp} H_g^{temp}) \tag{17}$$

Among them, W_{temp} is the learnable weight matrix for template generation.

3.3.2. Judgment information on the merits' filling template

The fact description and sentencing description are input to the model, and the judgment information on the merits and template is generated after the generation of multi-angle topology judgment information on the merits and template. The explicit classification subtask and explicit generation subtask in the judgment information on the merits is backfilled to the empty space of the template. Although the implicit subtask predicts the results. It does not participate in the formation of judicial views. Its role is to help improve the prediction or generation performance of additional explicit subtasks.

In summary, the algorithm of TBLIG is illustrated in Algorithm 1. $T = \{t_1, t_2, \dots, t_n\}$ is the set of all subtasks, containing n subtasks in total. t_i is the ith subtask. E_{t_i} is the initialization effect of the ith subtask on the next subtask in the topological order. $\hat{y}_{temp}$ is the generated judicial viewpoint template, and $\{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n\}$ is the results of the subtasks. **Steps 1–5:** Encode inputs using a fusion encoder to understand the fact description and Sentencing instruction comprehensively. **Steps 6–19:** Apply a multi-angle topology network to link classification and generation subtasks, capturing their interdependencies and obtaining subtask results. **Steps 20–21:** Generate a judicial view template and fill it with subtask results to form the final judicial view.

3.4. Training and inference

3.4.1. Training

In the training stage, we provide the model with the real hollowing template and the real label of each sub-task, focusing on the accuracy of the training model for template generation and the prediction or generation performance of each sub-task. For all classified subtasks, their total loss is calculated according to the formula, where n_c is the number of classification subtasks:

$$\mathcal{L}_c = -\sum_{i=1}^{n_c} (y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)) \tag{18}$$

Algorithm 1 TBLIG

```
1: Input: Fact description $F$, Sentencing instructions $S$
2: Output: Judicial view $V$
3: $H_T$ = TransformerEncoder $(F, S)$
4: $H_g^t emp$ = TransformerDecoder $(H_T)$
5: $H_g$ = TransformerDecoder$'(H_T)$
6: $H_C$ = CNN/ Bert $(F, S)$
7: $H_f = H_C + MaxPooling(H_T)$
8: for $t_i$ in $T = t_1, t_2, ..., t_n$ do
9:     if $t_i$ is classification task then
10:        $E_{t_i} = W_{i,j}[h_i, c_i] + b_{i,j}$
11:    else
12:        $E_{t_i} = W_{i,j} \cdot Max(H_g) + b_{i,j}$
13:    end if
14:    $[\bar{h}_j, \bar{c}_j] = [h_j^\circ, c_j^\circ] + \sum_{t_i \in T'} (E_{t_i})$
15:    $[h_j, c_j] = LSTM(H_f, [\bar{h}_j, \bar{c}_j])$
16:    if $t_i$ is classification task then
17:        $\hat{y}_j = sigmoid(W_j h_j + b_j)$
18:    else
19:        $\hat{y}_j = softmax(W_j H_g^j)$
20:    end if
21: end for
22: $\hat{y}_{temp} = softmax(W_{temp} H_g^{temp})$
23: $V$ = FillTemplate$(\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_n\}, \hat{y}_{temp})$
```

For all generation subtasks, their total loss is calculated according to the formula, where n_g is the number of generation subtasks, and l is the length of the target generated by the real subtask:

$$\mathcal{L}_g = -\frac{1}{l} \sum_{i=1}^{n_g} \sum_{t=1}^{l} y_{i,t} log(\hat{y}_{i,t}) - (1 - y_{i,t}) log(1 - \hat{y}_{i,t}) \tag{19}$$

For template generation, loss calculation is as follows, where l_{temp} is the length of the real template:

$$\mathcal{L}_{temp} = -\frac{1}{l_{temp}} \sum_{t=1}^{l_{temp}} y_t log(\hat{y}_t) - (1 - y_t) log(1 - \hat{y}_t) \tag{20}$$

So the overall loss is:

$$\mathcal{L}_{all} = \mathcal{L}_c + \mathcal{L}_g + \mathcal{L}_{temp} \tag{21}$$

3.4.2. Inference

In the inference stage, we do not provide model subtask labels and real templates, but input fact descriptions and sentencing scenarios to the trained model, and let the model generate templates and predict the results of each judgment information on the merits. Finally, as described in 3.3.2, the final judicial view is obtained by filling in key elements in the template.

4. Experiments

4.1. Data set collection

We test the feasibility of the experimental methods in the stage of review and prosecution of the Procuratorate. In the broad sense of "justice", it includes the review and prosecution of the Procuratorate and the judgment of the court. Unlike most current LJPs, which focus on the research of court data and experiments, we mainly test the effect of the "review and prosecution" stage according to the advantages of our existing database. Our experimental data comes from the publication of documents of the Supreme People's Procuratorate Net.[2] We have

[2] https://wenshu.court.gov.cn.

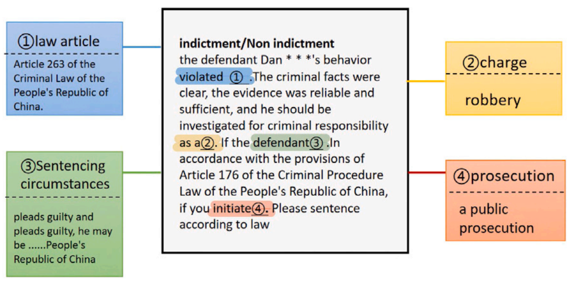

①law article
Article 263 of the Criminal Law of the People's Republic of China.
③Sentencing circumstances
pleads guilty and pleads guilty, he may be ......People's Republic of China
indictment/Non indictment
the defendant Dan * * *'s behavior violated ① .The criminal facts were clear, the evidence was reliable and sufficient, and he should be investigated for criminal responsibility as a②. If the defendant③ .In accordance with the provisions of Article 176 of the Criminal Procedure Law of the People's Republic of China, if you initiate④. Please sentence according to law
②charge
robbery
④prosecution
a public prosecution

Fig. 4. An example of the judgment information on the merits and inherent template contained in the paragraph of the Court's opinion in an indictment.

obtained two kinds of documents from the website: indictment and non-indictment. The dataset is mainly from four representative provinces, namely Beijing, Guangdong, Zhejiang, and Yunnan, to represent the eastern, western, southern, and northern regions of China, respectively. Our dataset reflects the various situation in China, as these regions have distinctive geographical, economic, and political characteristics. For example, Zhejiang Province is at the forefront of judicial reform, and Guangdong Province has the highest GDP, while Yunnan Province has relatively lower economic development. Beijing is the nation's capital.

We downloaded 90 386 indictments from five different provinces: Beijing, Guangdong, Zhejiang, Tianjin, and Yunnan. Then we cleaned the document and extracted the "basic information about the case" section of the procuratorial document, which completely describes the behavior of the perpetrator and various factors that may affect his crime or the severity of the sentence. At the same time, we also extracted the judiciary reasoning section paragraph, which includes the summary of the action, applicable charges, laws, sentencing circumstances whether to prosecute, and so on. The "basic information of the case" section is mainly used for the input information of the experimental model, and the judiciary reasoning section is used for the output information of the experimental model. We have deleted documents that lack the complete "basic information about the case" and the judiciary reasoning section. We randomly divide them into training, validation, and testing sets at the ratio of 8:1:1, i.e. 72 308 cases are used in the training set and 9039 cases are for validation, 9039 cases are for testing.

4.2. Construct a template for judicial view reasoning

The second step of the experiment is to construct the "judicial view" of the Procuratorate in the judiciary reasoning section. We invited experts in the field of law to sort out the judgment information on the merits of the judgment mainly involved in this paragraph and found that there are four explicit pieces of information, including charges, law articles, sentencing circumstances, whether to prosecute or not, as well as one implicit information of "compulsory measure".[3] We manually extracted the above five types of information from the judiciary reasoning section and the "Proceedings" paragraph. Table 1 shows the statistical results of our dataset, there are 62 types of crimes, 56 types of law articles, 2 types of compulsory measures, and 2 types of prosecution status, in which 58% of the cases of sentencing circumstances are specified.[4] In these types of information, the mutual constraints between the charges and law articles make the specific charges correspond to the corresponding law articles, and sentencing

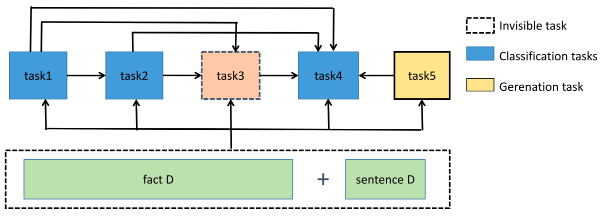


Fig. 5. Topological structure diagram between subtasks.

circumstances will affect the final result of whether to prosecute, while the compulsory measures will also potentially affect other subtasks.

In the reasoning template of the judicial view, different judicial subjects will have differences in different reasoning occasions. In terms of the judicial reasoning of the Procuratorate, the effect of the content of the judiciary reasoning section after removing the above judgment information on the merits is shown in the following Fig. 4. Due to the differences in the structure of the indictment and the non-indictment, the template styles are also divided into two types. The 1–4 sequence in the cloze is mainly predicted by the judgment information on the merits' generation, and the text part of the figure is realized by the template generation link in the judicial view generation module.

4.3. Construct judgment information on the merits' topology for specific scenarios

After constructing the cloze map, the next important work is how to predict the judgment information on the merits better than the direct generation model. Our experiment mainly uses the judgment dependency between this judgment information on the merits to connect this judgment information on the merits in the form of a topological graph and constructs the model based on this graph, as shown in Fig. 5. Generally speaking, in the real scene, the judicial personnel makes judgments based on the facts, determines the charges and laws, and then comprehensively considers whether to prosecute according to whether the actor pleads guilty to punishment, whether to commit recidivism, whether to surrender, and other circumstances. We determine the structure chart according to the prosecution process of the Procuratorate.

Compared with the previous method of using the relationship between multiple subtasks in the judiciary, the innovation of our experiment is that the potential implicit subtasks (precisely "compulsory measures") are fully considered, and the classification algorithm and generation algorithm are cross-used at the same time. Implicit judgment information on the merits on compulsory measures and the generation of sentencing circumstances will help improve the prediction performance of judgment information on the merits about related subtasks in this experiment.

4.4. Baselines

In order to evaluate the performance and interpretability of our model, we implemented several baselines to compare these two aspects. For predicting judgment information on the merits, We have developed some universal strong baselines and some outstanding-performing legal judgment prediction (LJP) baselines as follows:

(1) **CNN** [20] is a traditional convolutional neural network model for text classification.

(2) **Bert** [21] is a pretrained model that has been adjusted to Chinese.

For legal text generation, we implemented a universal generation baseline and a criminal judicial view generation baseline as follows:

(3) **Transformer** [22] is a state-of-the-art model in several text generation tasks.

[3] Although the "compulsory measure" are not directly stated in the judiciary reasoning section, the judgment of this information is helpful to predict other judgment information on the merits.

[4] Therefore, in the indictment/no indictment, the sentencing circumstances are not required.

Table 1
The number of various data in our data set, CM means compulsory measures, PR means prosecution, SC means no sentencing circumstances, SD means no sentencing description.

Name	Total	Training set	Validation set	Test set	Charge	Article	CM	PR	SC	SD
Number	90 386	72 308	9039	9039	62	56	2	2	38 301	47 473

Table 2
Comparison between the performance of judgment information on the merits' prediction.

Model	Charge				Law article				Compulsory measure				Prosecution			
	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
C3VG	93.02	77.19	76.43	74.94	–	–	–	–	–	–	–	–	–	–	–	–
Topjudge	92.82	77.31	75.91	76.11	93.77	79.00	76.12	77.00	–	–	–	–	–	–	–	–
LADAN	93.60	83.10	73.31	76.32	92.94	82.23	74.78	77.09	–	–	–	–	–	–	–	–
Neurjudge	92.89	78.82	73.63	74.83	93.98	80.14	76.23	77.23	–	–	–	–	–	–	–	–
CNN	93.47	82.46	76.75	78.56	94.16	83.82	76.76	78.99	87.56	87.45	87.68	87.51	97.94	96.82	95.37	96.08
Bert	94.74	81.61	80.08	79.87	95.46	83.95	80.56	81.32	89.14	89.89	89.48	89.27	99.12	99.44	99.13	99.25
TBLIG (CNN)	93.53	82.86	78.31	79.60	94.38	82.94	78.30	79.85	88.30	88.17	88.38	88.24	**99.97**	**99.95**	99.92	99.94
TBLIG (Bert)	**94.96**	**83.71**	**82.91**	**82.53**	**95.41**	**85.32**	**83.05**	**83.61**	**91.86**	**90.34**	**90.51**	**90.20**	99.89	99.92	**99.95**	**99.94**

Table 3
Compare the TBLIG (CNN) model with the directly generating model (Transformer).

Model	Charge				Law article				Prosecution			
	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
TBLIG	**93.53**	**82.86**	**78.31**	**79.60**	**94.38**	**82.94**	**78.30**	**79.85**	**99.97**	**99.95**	**99.92**	**99.94**
Generating	88.31	74.70	71.27	71.71	87.37	58.17	52.03	54.52	92.63	92.22	81.71	85.76

Table 4
Compare the TBLIG (CNN) with the ablation setting of generating Sentencing Circumstances Independently (SC-I).

Model	Charge				Law article				Compulsory measure				Prosecution			
	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF	Acc	MaP	MaR	MaF
TBLIG	**93.53**	82.86	**78.31**	79.60	94.38	82.94	**78.30**	**79.85**	**88.30**	**88.17**	**88.38**	**88.24**	**99.97**	**99.95**	**99.92**	**99.94**
SC-I	94.38	**84.47**	78.22	**80.37**	**94.70**	**83.06**	76.54	78.94	87.21	87.07	87.33	87.14	98.11	97.47	95.18	96.28

(4) **Topjudge** [9] is a topological multi-task learning model that captures the dependencies among subtasks in LJP.

(5) **LADAN** [23] uses a graph distillation to extract discriminative features.

(6) **NeurJudge** [24] splits the fact description into two parts and encodes them separately.

(7) **C3VG** [4] is a model following a two-stage architecture which is from extraction to generation. We also evaluated its performance in predicting charges.

We implemented all the aforementioned baselines according to the settings described in their original paper. For the sake of facilitating a more direct comparison with our method, we developed our TBLIG approach based on the two encoders mentioned above (CNN, Bert). Additionally, we conducted several ablation experiments to demonstrate the effectiveness of our method:

(8) **SC-I** removes the interdependencies between the "sentencing circumstance" subtask and other subtasks within TBLIG, essentially generating the "sentencing circumstance" independently.

(9) **SC-NS** directly eliminates the "sentencing circumstance" subtask from TBLIG.

4.5. Metrics

4.5.1. Generation metrics

In evaluating the quality of the judicial text finally generated, we use the widely used evaluation metrics, ROUGE and BLEU. They evaluate the similarity between the generated text and the actual text from different perspectives.

ROUGE [25] is a set of commonly used evaluation metrics in NLP. We utilized ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L (R-L) in this paper. R-1 and R-2 refer to the overlap of unigram and bigram between the generated and reference documents, respectively. R-L is a Longest Common Subsequence (LCS) based statistic.

Table 5
Comparison between the performance of judicial view generation.

Model	Court view generation					
	B-1	B-2	B-N	R-1	R-2	R-L
Transformer	77.58	74.07	73.21	85.20	78.70	83.62
C3VG	78.35	74.70	73.30	86.60	78.29	83.71
TBLIG (CNN)	78.39	75.83	75.02	86.98	80.80	85.67
TBLIG (Bert)	**79.32**	**76.41**	**76.05**	**87.53**	**81.61**	**86.37**

BLEU [26] is a method of automatic text-generation evaluation that highly correlates with human evaluation. We calculate these metrics in our experiments using the NLTK toolkit.[5] We use BLEU-1 (B-1) and BLEU-2 (B-2) to evaluate from the perspectives of unigram and bigram. BLEU-N (B-N) averages B-1, B-2, B-3, B-4.

4.5.2. Information accuracy metric

The generation metrics can only reflect the similarity between the target text and the generated text. Some models have the ability to generate legal templates, but the judgment information on the merits may be wrong. Therefore, we evaluate the performance of judgment information on the merits by accuracy (Acc), macro precision (MaP), macro recall (MaR), and macro F1 (MaF).

5. Results and analysis

5.1. Results of judgment information on the merits's prediction

Although the ultimate goal of this paper is text generation, from the perspective of the process sequence, the first step is to achieve

[5] https://www.nltk.org/modules/nltk/translate/bleuscore.html.

Table 6
Comparison between template generation and sentencing circumstances generation in TBLIG (CNN).

Generation	B-1	B-2	B-N	R-1	R-2	R-L
Template	84.36	81.42	80.86	89.69	84.73	89.81
Sentencing circumstance	59.52	56.95	55.94	68.97	62.33	66.28

Table 7
Compare the sentencing circumstances as gain information (SC-GI), the sentencing circumstances appear independently (SC-I), and the sentencing circumstances are not as subtasks (SC-NS).

Model	Generation	B-1	B-2	B-N	R-1	R-2	R-L
TBLIG (CNN)	Template	**84.63**	81.42	80.86	89.69	**84.73**	**89.81**
	SC	**59.52**	**56.95**	**55.94**	**68.97**	**62.33**	**66.28**
	Judicial view	78.39	**75.83**	**75.02**	**86.98**	**80.80**	**85.67**
SC-NS	Template	79.04	75.63	74.86	87.08	80.47	85.39
	Judicial view	**78.57**	75.24	74.33	86.05	79.60	84.29
SC-I	Template	84.34	**82.01**	**81.37**	**90.19**	84.69	89.32
	SC	58.12	55.61	54.68	67.49	61.27	64.82
	Judicial view	78.29	75.72	74.90	86.93	80.66	85.59

the prediction of high accuracy of judgment information on the merits. Therefore, from the perspective of analyzing the experimental results, we should first test the experimental effect of the judgment information on the merits prediction. We compare our method, TBLIG, with three control groups in terms of judgment information on the merits prediction. The three control groups consist of a method that does not reflect the dependencies between subtasks, a method that directly generates and extracts judgment information on the merits, and a method that does not generate gain information in sentencing circumstances.

5.1.1. *Comparison with the methods that do not utilize dependencies between subtasks*

As shown in Table 2, we draw the following conclusions: (1) Compared with all the baselines, TBLIG achieved the best performance, indicating the capability of our method to predict judgment information on the merits in judicial texts. (2) TBLIG based on two types of encoders significantly surpassed the performance of the original model, suggesting that establishing dependencies between prediction tasks is beneficial. (3) TBLIG exhibited varying improvements across different tasks. Compared to the strongest baseline (Bert), there was an increase of 2.66 in the "charge" task, an increase of 2.29 in the "law article" task, a 0.93 increase in the "compulsory measure" task, and a 0.69 increase in the "prosecution" task. This indicates that tasks positioned later in the topological order gradually benefit from preceding tasks.

5.1.2. *Comparison with the method that extracts judgment information on the merits from the directly generating model*

In this comparative experiment, we employ a transformer network to directly generate the entire judicial view. We then extract the judgment information on the merits from the generated results and compare it with our model. As shown in Table 3, we obtained the following conclusions: (1) Compared to the methods that directly generate results, TBLIG clearly achieved better performance. This indicates that structuring the task first and then using a classification model to predict judgment information on the merits leads to higher accuracy compared to extracting accuracy from directly generated results. (2) Our TBLIG model showed improvements across different tasks compared to the direct generation models. In terms of the metric MaF, the "charge" task improved by 7.89, the "law article" task improved by 25.33, and the "prosecution" task improved by 14.18. (3) Utilizing a classification model to predict judgment information on the merits significantly enhanced the prediction of "law article". This may be due to the task's position in the topological order, allowing it to acquire more information.

5.1.3. *Comparison with the method that does not gain information from sentencing circumstances*

In this comparative experiment, the model not acquiring information about sentencing circumstances means that the sentencing circumstance task is not part of the topological order but treated as an independent subtask. As shown in Table 4, we have reached the following conclusions: (1) Compared to the results without considering "sentencing circumstances" as gain information, TBLIG demonstrates better overall performance, indicating that incorporating "sentencing circumstances" as a consideration is beneficial for predicting judgment information on the merits. (2) Considering "sentencing circumstances" as gain information (TBLIG) outperforms not considering "sentencing circumstances" as gain information (SC-I). (3) Compared to not considering "sentencing circumstances", our method TBLIG shows improvements in most tasks. Specifically, the "compulsory measure" task improved by 1.1, and the "prosecution" task improved by 3.66. This suggests that incorporating "sentencing circumstances" as gain information can enhance the accuracy of predicting other tasks.

5.2. *Results of judicial views generation*

The ultimate goal of our study is to achieve the text generation of judicial reasoning paragraphs. Therefore, the second level of experimental data analysis is to analyze the actual effect of this experimental method from the perspective of the effect of text generation. Our experiment adopts a comparison with the direct text generation method and non-gain information to illustrate the innovative effect of our experimental method.

5.2.1. *Comparison between the performance of judicial view generation*

As shown in Table 5, we have obtained the following conclusions: (1) TBLIG, implemented with both types of encoders, outperforms C3VG and Transformer, indicating the capability of our method to generate higher-quality court opinions. Specifically, TBLIG (BERT) exhibits an improvement of 2.75 in the B-N metric and 2.66 in the R-L metric compared to C3VG. (2) When comparing TBLIG (CNN) and TBLIG (BERT), better performance is observed with BERT as the encoder, suggesting that TBLIG benefits from more powerful learning capabilities within the encoder.

5.2.2. *Comparison between template generation and sentencing circumstances generation*

For the two generation tasks in the TBLIG framework, we first compared their performances. In the framework of this experimental method, two models of text generation are used. From the perspective of generation performance, As shown in Table 6, the sentencing circumstances generation subtask is more difficult than the template generation task in the judicial view generation encoder. Compared to the sentencing circumstances generation, the judicial view generation has 24.92 and 23.53 higher B-N and R-L indicators, respectively.

5.2.3. *Comparison with the methods that do not gain information from sentencing circumstances*

As shown in Table 7, we have obtained the following conclusions: (1) Compared to the experiment without "sentencing circumstances", TBLIG has better performance and helps improve text generation. This indicates that the information on "sentencing circumstances" can indirectly influence the generation of judicial views by affecting the prediction of judgment information on the merits. (2) Compared to not treating "sentencing circumstances" as a subtask (SC-NS), our method TBLIG shows improvement in text generation effectiveness; compared to the independent occurrence of "sentencing circumstances" (SC-I), our method also improves text generation effectiveness. (3) Predicting with "sentencing circumstances" performs better than without this subtask, and using this subtask to predict other judgment information on the merits is better than treating it as a standalone task.

6. Discussion

The current research in legal AI faces challenges in generating judicial views. Judicial decisions place significant emphasis on procedural reasoning, which is presented as a distinct paragraph in legal documents. Judicial views aim to clarify the rationale behind the final decision, encompassing various key pieces of information and reasoning skills. Our TBLIG framework significantly enhances the quality of judgment information on the merits' prediction in judicial reasoning. This method aligns with the current trend in legal AI development, focusing on the synergy between domain knowledge and data algorithms.

At the same time, based on the understanding of some characteristics and laws of the generating task of judicial reasoning, we should reflect on the evaluation method of the task, which may not simply rely on the generation evaluation method. The reason is perhaps that the text of judicial reasoning is composed of "multiple judgment information on the merits and reasoning template", which is crucial for the accuracy of the text of judicial reasoning. Therefore, the evaluation of this task should be performed separately. Judgment information on the merits adopts the classification evaluation method, and the template adopts the generation evaluation method.

7. Conclusion

In this paper, we have addressed the challenging task of generating judicial views, which has been relatively underexplored in the field of legal AI. We recognized that conventional text generation algorithms may fall short in providing accurate and professional legal expressions, as well as correctly representing judgment information on the merits. To bridge this gap, we proposed the Template-Based Legal Information Generation (TBLIG) framework, drawing inspiration from how judges construct their views step by step. Our method consists of two essential steps: firstly, identifying multiple key pieces of information crucial to the judicial view, and secondly, synchronously generating a legal template based on the specific case. Through extensive experimentation on indictments and non-indictments datasets, we validated the effectiveness of our approach by comparing it with other existing models and conducting ablation experiments.

CRediT authorship contribution statement

Xiang Zhou: Supervision, Resources, Investigation, Conceptualization. **Yudong Wu:** Writing – original draft, Methodology, Conceptualization. **Ang Li:** Writing – original draft, Validation, Software, Methodology, Conceptualization. **Ming Cai:** Supervision, Resources, Investigation, Conceptualization. **Yiquan Wu:** Writing – review & editing, Supervision, Investigation, Conceptualization. **Kun Kuang:** Writing – review & editing, Investigation, Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work has been supported by the 2024 Zhejiang Provincial Philosophy and Social Science Special Project (24ZJQN064Y) and Information Technology Center, ZheJiang University.

References

[1] J. Cui, X. Shen, F. Nie, Z. Wang, J. Wang, Y. Chen, A survey on legal judgment prediction: Datasets, metrics, models and challenges, 2022, arXiv preprint arXiv:2204.04859.
[2] W. Huang, X. Liao, Z. Xie, J. Qian, J. Xiao, Generating reasonable legal text through the combination of language modeling and question answering, in: Twenty-Ninth International Joint Conference on Artificial Intelligence and Seventeenth Pacific Rim International Conference on Artificial Intelligence IJCAI-PRICAI-20, 2020.
[3] K. Atkinson, T. Bench-Capon, D. Bollegala, Explanation in AI and law: Past, present and future, Artificial Intelligence 289 (2020) 103387.
[4] L. Yue, Q. Liu, H. Wu, Y. An, L. Wang, S. Yuan, D. Wu, Circumstances enhanced criminal court view generation, in: SIGIR, 2021, pp. 1855–1859.
[5] Y. Wu, K. Kuang, Y. Zhang, X. Liu, C. Sun, J. Xiao, Y. Zhuang, L. Si, F. Wu, De-biased court's view generation with causality, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2020, pp. 763–780.
[6] H. Ye, X. Jiang, Z. Luo, W. Chao, Interpretable charge predictions for criminal cases: Learning to generate court views from fact descriptions, 2018, arXiv preprint arXiv:1802.08504.
[7] K.D. Ashley, A brief history of the changing roles of case prediction in AI and law, Law Context: Soc.-Leg. J. 36 (2019) 93.
[8] H. Zhang, Z. Dou, Y. Zhu, J.-R. Wen, Contrastive learning for legal judgment prediction, ACM Trans. Inf. Syst. 41 (4) (2023) 1–25.
[9] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, M. Sun, Legal judgment prediction via topological learning, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 3540–3549.
[10] W. Yang, W. Jia, X. Zhou, Y. Luo, Legal judgment prediction via multi-perspective bi-feedback network, 2019, arXiv preprint arXiv:1905.03969.
[11] Y. Feng, C. Li, J. Ge, B. Luo, Improving statute prediction via mining correlations between statutes, in: W.S. Lee, T. Suzuki (Eds.), Proceedings of the Eleventh Asian Conference on Machine Learning, in: Proceedings of Machine Learning Research, Vol. 101, PMLR, 2019, pp. 710–725.
[12] W. Chen, J. Tian, L. Xiao, H. He, Y. Jin, Exploring logically dependent multi-task learning with causal inference, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2020, pp. 2213–2225.
[13] Q. Dong, S. Niu., Legal judgment prediction via relational learning, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development, 2021.
[14] W. Liao, Y. Wang, Y. Yin, X. Zhang, P. Ma, Improved sequence generation model for multi-label classification via CNN and initialized fully connection, Neurocomputing 382 (2020) 188–195.
[15] K.W. Church, Word2Vec, Nat. Lang. Eng. 23 (1) (2017) 155–162.
[16] Y. Ji, H. Zhang, Z. Zhang, M. Liu, CNN-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances, Inform. Sci. 546 (2021) 835–857.
[17] A.T. Liu, S.-W. Li, H.-y. Lee, Tera: Self-supervised learning of transformer encoder representation for speech, IEEE/ACM Trans. Audio Speech Lang. Process. 29 (2021) 2351–2366.
[18] M. Hüsken, P. Stagge, Recurrent neural networks for time series classification, Neurocomputing 50 (2003) 223–235.
[19] G. Liu, J. Guo, Bidirectional LSTM with attention mechanism and convolutional layer for text classification, Neurocomputing 337 (2019) 325–338.
[20] J. Long, E. Shelhamer, T. Darrel, Fully convolutional networks for semantic segmentation.
[21] Y. Cui, W. Che, T. Liu, B. Qin, S. Wang, G. Hu, Revisiting pre-trained models for Chinese natural language processing, 2020, arXiv preprint arXiv:2004.13922.
[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[23] N. Xu, P. Wang, L. Chen, L. Pan, X. Wang, J. Zhao, Distinguish confusing law articles for legal judgment prediction, in: D. Jurafsky, J. Chai, N. Schluter, J.R. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, Association for Computational Linguistics, 2020, pp. 3086–3095, http://dx.doi.org/10.18653/v1/2020.acl-main.280.
[24] L. Yue, Q. Liu, B. Jin, H. Wu, K. Zhang, Y. An, M. Cheng, B. Yin, D.N. Wu, A circumstance-aware neural framework for legal judgment prediction, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Online, 2021, pp. 11–15.
[25] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, in: Text Summarization Branches Out, 2004, pp. 74–81.
[26] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, 2002, pp. 311–318.