Hindawi
Complexity
Volume 2020, Article ID 3089189, 12 pages
https://doi.org/10.1155/2020/3089189

Research Article

Legal Judgment Prediction Based on Multiclass Information Fusion

Kongfan Zhu, Rundong Guo, Weifeng Hu, Zeqiang Li, and Yujun Li

School of Information Science and Engineering, Shandong University, Qingdao 266200, China

Correspondence should be addressed to Yujun Li; liyujun@sdu.edu.cn

Received 25 June 2020; Revised 20 August 2020; Accepted 4 October 2020; Published 26 October 2020

Academic Editor: Shirui Pan

Legal judgment prediction (LJP), as an effective and critical application in legal assistant systems, aims to determine the judgment results according to the information based on the fact determination. In real-world scenarios, to deal with the criminal cases, judges not only take advantage of the fact description, but also consider the external information, such as the basic information of defendant and the court view. However, most existing works take the fact description as the sole input for LJP and ignore the external information. We propose a Transformer-Hierarchical-Attention-Multi-Extra (THME) Network to make full use of the information based on the fact determination. We conduct experiments on a real-world large-scale dataset of criminal cases in the civil law system. Experimental results show that our method outperforms state-of-the-art LJP methods on all judgment prediction tasks.

1. Introduction

Legal judgment prediction (LJP) aims to predict the judgment results according to the information based on fact determination, which consists of the fact description, the basic information of defendant, and the court view. LJP techniques can provide inexpensive and useful legal judgment results to people who are unfamiliar with legal terminologies, and they are also helpful for the legal consulting. Moreover, they can serve as a handy reference for professionals (e.g., lawyers and judges), which can improve their work efficiency.

LJP is regarded as a classic text classification problem and has been researched for many years [1]. For example, Liu et al. proposed to extract shallow textual features (e.g., Chinese characters, words, and phrases) for charge prediction [2]. Katz et al. predicted the US Supreme Court's decisions based on efficient features from case profiles [3]. Luo et al. combined the fact description with the corresponding law articles to predict the charges [4]. Although great progress has been made in the LJP, there still exist some problems, such as multiple subtasks, topological dependencies between subtasks, and cases of similar descriptions with different penalties. Zhong et al. pointed out that law articles prediction was one of the fundamental subtasks in some countries (e.g., China, France, and Germany) with the civil law system, and these subtasks had a strict order in the real world [5]. Further, Yang et al. proposed a neural model for the interaction between subtask results [6].

Despite these efforts in designing efficient features and employing advanced Natural Language Processing (NLP) techniques, LJP still confronts two major challenges.

1.1. The Lack of External Information. Some existing works propose various mechanisms to extract information from the fact description, such as the Word Collection Attention mechanism. Some other works propose various frameworks to build the dependencies between subtasks, such as DAG Dependencies of Subtasks and MPBF. However, for the judgment document in Figure 1, there are many other information items that can be utilized except the fact description. Such information is called the external information including the basic information of defendant and the court view. Therefore, how to utilize the external information effectively is a major challenge.

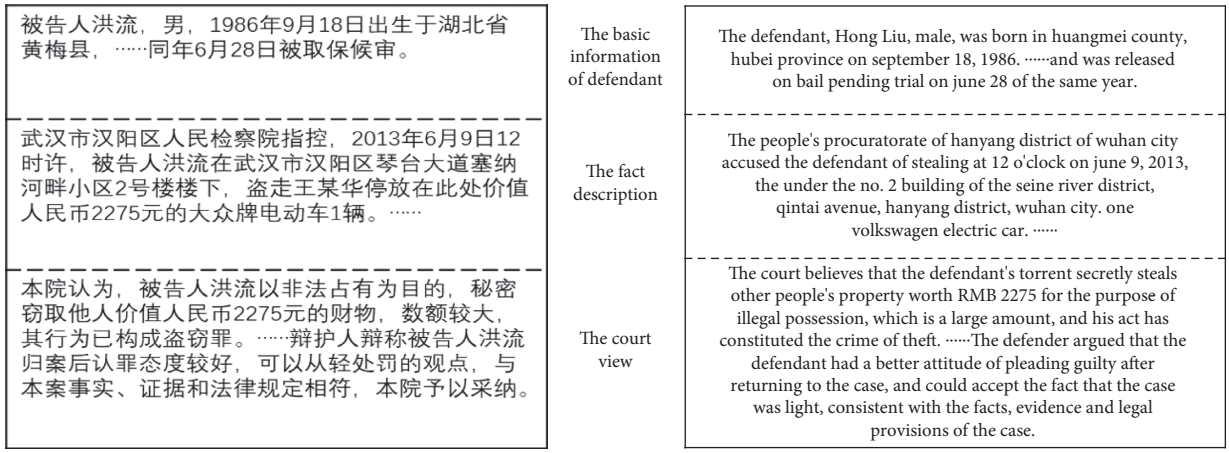

Figure 1: A judgment document example in China (original Chinese text and its English translation).

1.2. Encoding Long Document Is Difficult. The fact description in judgment document is often long document containing the long-term dependency problem. Many existing models, such as Recurrent Neural Network (RNN) [7] and Convolutional Neural Network (CNN) [8], which perform well in the text processing are unable to deal with the long-term dependency problem. There are some keywords in the judgment document that are very important for LJP. It is very difficult to find them in the judgment document.

In order to resolve the above challenges, in this paper, we propose the Transformer-HAN-Multi-Extra (THME) Network. It contains a structured data encoder to extract the semantics of the external information as well as a Transformer-Hierarchical Attention Network (TH) encoder to encode the fact description. Specifically, as shown in Figure 1, from the basic information of the defendant, we can get the defendant's gender, age, and education level and the content related to the criminal records of the defendant by using regular expressions. Similarly, we can get some objective attributes of a case, such as amount, plot, and consequences, from the court view. Based on the statistical analysis of large samples, we can find the relationship between the data and the terms of penalty as is shown in Table 1, where the symbol "+" represents "related." For example, given the same conditions, male's terms of penalty is longer than female's for certain cases. We use the symbol "↑" to denote positive correlation. For example, the more serious the case's plot is, the longer the defendant's terms of penalty will be. We use the symbol "↓" to denote negative correlation. For example, the better the defendant's guilty attitude is, the shorter the defendant's terms of penalty will be. It is worth noting that the case's conclusion in judgment document is significant for terms of penalty but it cannot be used as an input to predict the terms of penalty. If it is used as an input to predict the terms of penalty, it seems like that the cat shuts its eyes when stealing. Therefore, we first use the external information to predict the case's conclusion and then use it together with the external information to predict the terms of penalty. Meanwhile, according to the data attributes, we divide the data into continuous and discrete types. Then, we extract the required information via the continuous data encoder and the discrete data encoder. In order to reduce the information loss in the process of converting sentences into fixed-length vectors, an attention mechanism is adopted. But, it cannot solve the polysemy problem. Then, we choose a proper Transformer [9]. Transformer has attention structure; it has advantages over the RNN in solving long-term dependency problem and performs better than attention on polysemy. The Hierarchical Attention Network (HAN) can catch the keywords in a long document easily [10]. Thus, we can combine the Transformer with the HAN to solve the long-term dependency problem. Experimental results show that the performance of Transformer-HAN is better than Gate Recurrent Unit (GRU)-HAN.

The main contributions of this paper are summarized as follows:

(i) We propose a novel text processing structure, namely, Transformer-HAN, to improve the text encoding ability. This model can solve the long-term dependency problems better than the GRU-HAN. Transformer-HAN encoder uses the attention mechanism in addition to the necessary fully connected layer of the parameter matrix, and it works much faster than the encoder structure based on GRU and Long Short-Term Memory (LSTM).

(ii) We propose a structured data encoder. To introduce the external information as an auxiliary, we extract fact-related data from the defendant's basic information and the court view as supplementary information of the model. According to different attributes of data, we design both continuous and discrete data encoders. Experiments show that information based on fact determination can effectively improve the judgment prediction, especially for the prediction of the terms of penalty.

Table 1: Origins and categories of structured data.

Origin	Category
The basic information of defendant	Defendant's gender (+), defendant's education level (↑), defendant's previous convictions (+), defendant's number of previous convictions (↑), defendant's terms of penalty of previous convictions (↑), defendant's penalty of previous convictions (↑)
The court view	Case's amount of money involved (↑), case's conclusion (↑), case's amount (↑), case's plot (↑), case's consequence (↑), defender's guilty attitude (↓)

(iii) Experimental results show that the THME Network can effectively improve the prediction accuracy of few-shot data. The macro-average indicators of the three tasks of law article prediction, charge prediction, and terms of penalty prediction are relatively improved compared with other models, which indicates that the prediction accuracy of few-shot data has been greatly improved.

The rest of this paper is organized as follows. Section 2 briefly reviews the related work. In Section 3, we propose the overall THME framework and detailed methods. The experimental results and analyses are presented in Section 4. Finally, Section 5 contains the concluding remarks.

2. Related Work

2.1. Legal Judgment Prediction. With the development of Chinese legal digitalization process, as one of the most critical task steps in LegalAI, LJP has become more and more important. Thanks to the development of machine learning and text mining techniques, more researchers formalize this task under text classification frameworks. Most of these studies attempt to extract textual features [11–13] or introduce some external knowledge [4, 14]. However, these methods can only utilize shallow features and manually designed factors; usually the effect of these methods becomes worse when applied to other scenarios. Therefore, researchers take advantage of other technologies to improve the interpretability and generalization of the model. For example, Jiang et al. utilized the deep reinforcement learning to derive short snippets of documents from the fact descriptions to predict charges [15], and Chen et al. proposed a Legal Graph Network (LGN) to achieve high-precision classification of crimes [16]. Due to the rareness of some types of cases in real life, the few-shot problem is inevitable. While some researchers hardly solve this problem using machine learning, others find that neural networks have good results. For example, Chen et al. proposed a neural network model by embedding law articles and fact descriptions into the same embedding space in the same way [17]. Yang et al. proposed a repeated interactional mechanism to simulate the process of judge's decision [18].

2.2. Multitask Learning. Multitask models have many beneficial effects for deep learning tasks. Sulea et al. proposed multiple tasks, which include law articles predictions, charge predictions, and terms of penalty predictions, to test the application of machine learning in the judicial field [19]. Zhong et al. proposed a topological structure network, which can simulate the judge's judgment process to improve the performance of various tasks. Yang et al. designed a Multi-Perspective Bi-Feedback Network (MPBFN) to enhance the connection between tasks and allow tasks' results to flow in both directions. Wang et al. set the relationship between law articles as a tree structure via a Hierarchical Matching Network (HMN) and matched relevant law articles via a two-layer matching network [20], which can improve the work efficiency.

The emergence of multitask learning has promoted the development of LJP; however, due to the lack of external information, it has also resulted in unsatisfactory prediction of terms of penalty. In this work, we propose a framework to utilize the external information effectively. Different from most existing works, we extract the information from both the fact description and the external information and merge them together into a topological classifier to predict the three subtasks of LJP.

3. Method

In this section, we will describe the THME Network. We first give the essential definitions of the LJP task and the composition of THME Network in Sections 3.1 and 3.2, respectively. We describe a text encoder for fact descriptions in Section 3.3. We introduce the structured data encoder in Section 3.4. Finally, the classifier is proposed in Section 3.5.

3.1. Problem Formulation. In most tasks of the Chinese text processing, the char-granularity processing is superior to the word-granularity processing [21], so for each judgment document, we set each Chinese character as a token. The fact description is a token sequence $T = (t_1, t_2, t_3, \ldots, t_N)$, where N is the number of tokens. This can reduce the complexity of model and make it fit easier. Besides the input T, the basic information of the defendant and the court view are also deemed as external inputs of the structured data encoder. Given these inputs, we will predict the judgment results of applicable law articles, charges, and terms of penalty, which is a multitask classification problem.

3.2. Overview. Our THME consists of three parts, i.e., the text encoder, the structured data encoder, and the classifier. The text encoder is composed of text embedding layer, text convolution layer, main encoder layer, and information extraction layer. Due to different attributes of the structured

data, we divide structured data into discrete data and continuous data, for which we propose discrete data encoder and continuous data encoder, respectively. The classifier is implemented with a topological structure, which utilizes the topological dependencies between subtasks in LJP. The general framework of the THME is shown in Figure 2.

We employ a text encoder to extract the information from the fact description; the fact description is embedded into CNN, so that advanced features are gradually extracted from the shallow textual features. c_{ij} represents the j-th Chinese character in the i-th sentence. The main encoder layer is actually Transformer-HAN, which includes two layers: the first layer aggregates token-level features into sentence-level features, and the second layer aggregates sentence-level features into text-level features. Finally, we generate four hidden-layer states T_1, T_2, T_3 corresponding to three subtasks of LJP and T_4 corresponding to the case's conclusion which is critical in predicting the terms of penalty through the information extraction layer. Next, we employ the regular expression to extract the discrete data and the continuous data from the external information. Then, we standardize the continuous data, embed the discrete data, and input them into the discrete data encoder and continuous data encoder, respectively. The outputs of these two encoders are combined to generate the structured data vector T_m'. T_m' and the hidden-layer state T_4 are concatenated into a full connection network to predict the case's conclusion T_{dc}. The case's conclusion vector T_{dc} and the structured data vector T_m' make up the output of the structured data encoder T_m. Finally, T_m and the hidden-layer state of all subtasks in LJP T_1, T_2, T_3 are concatenated into the classifier with topological structure to predict the law articles, charges, and terms of penalty.

3.3. Text Encoder for Fact Description. We employ a text encoder to generate the vector of fact description as the input of the classifier. We will briefly introduce this encoder which is composed of lookup layer, convolution layer, Transformer-HAN layer, and information extraction layer.

3.3.1. Lookup and Convolution. Taking a token sequence T as input, the encoder computes a simple text representation through two layers, i.e., lookup layer and convolution layer.

(1) Lookup. We first convert each token t_i in T into a natural number $d_i \in N$ by preprocessed dictionary mapping. The token sequence T is converted into an integer sequence $D = (d_1, d_2, d_3, \ldots, d_N)$. Next, we propose an initialized word embedding sequence $E = (e_0, e_1, e_2, \ldots, e_s)$, $e_i \in R^k$, where s is the size of dictionary. d_i is mapped to x_i via the word embedding sequence E. Thus, we can obtain the text embedding sequence $X = (x_1, x_2, x_3, \ldots, x_N)$, $x_i \in R^k$, where k is the length of word embedding.

(2) Convolution. For X, we make a convolution operation with the convolution matrix $W \in R^{m\times(l\times k)}$ given by

$$c_i = W \cdot x_{i:i+l-1} + b_c, \tag{1}$$

where $x_{i:i+l-1}$ is the concatenation of word embeddings in the i-th window, $b_c \in R^m$ is the bias vector, m is the number of filters, and l is the size of a sliding window. We apply the convolution over each window i and finally obtain $C = (c_1, c_2, c_3, \ldots, c_N)$. The Chinese character vector after convolution has n-gram features; that is to say, the Chinese character vector after convolution has context features and is no longer isolated.

3.3.2. Transformer-HAN Encoder and Information Extraction. (1) Transformer-HAN encoder. Transformer is currently the most mainstream information extractor, mainly due to its unique attention mechanism, which achieves the true bidirectional encoding. However, the number of parameters of the multilayer Transformer encoder is very huge. In order to fully take advantage of Transformer and meanwhile constrain the number of parameters, we design the Transformer-HAN as our main encoder.

Transformer-HAN encoder is divided into two layers: the first layer uses Transformer for Chinese character-granularity coding, then uses the attention mechanism to extract the most important information in each word embedding, and combines them into sentence vectors. The second layer uses Transformer for sentence-granularity coding, then uses the attention mechanism to extract the most important information in sentence vectors, and combines them into a chapter-granularity vector. Therefore, the fact description is divided into m sentences $C = (c_1, c_2, \ldots, c_m)$, and the i-th sentence consists of n Chinese characters $c_i = (c_{i1}, c_{i2}, c_{i3}, \ldots, c_{in})$, where $m \times n = N$.

Since the Transformer encoder is less sensitive to the position of Chinese characters, we need to add the position embedding to the word embedding before input. For Chinese character in the j-th sentence c_j, we calculate its position vector P_j as

$$\begin{aligned} P(\mathrm{pos}, 2i) &= \sin\left(\frac{pos}{10000^{2i/d\,\mathrm{model}}}\right), \\ P(\mathrm{pos}, 2i+1) &= \cos\left(\frac{pos}{10000^{2i/d\,\mathrm{model}}}\right), \end{aligned} \tag{2}$$

where pos is the position of this Chinese character in the sentence, i is the index of the i-th value in its word embedding, and d model is the dimension of its word embedding. The position vectors of all Chinese characters in the sentence c_j form the sequence P_j. Then, we merge the position sentence P_j with c_j to obtain the sentence sequence with the information of position C_{pj} given by

$$C_{pj} = P_j \oplus c_j, \tag{3}$$

where $\oplus$ is an element-wise addition operation.

The Transformer encoder is composed of Multihead Attention (MHA), Add & Norm Layer, and Feed Forward (FF). Multihead Attention is composed of Self-Attention, for which the inputs Q, K, and V are the same. Multihead

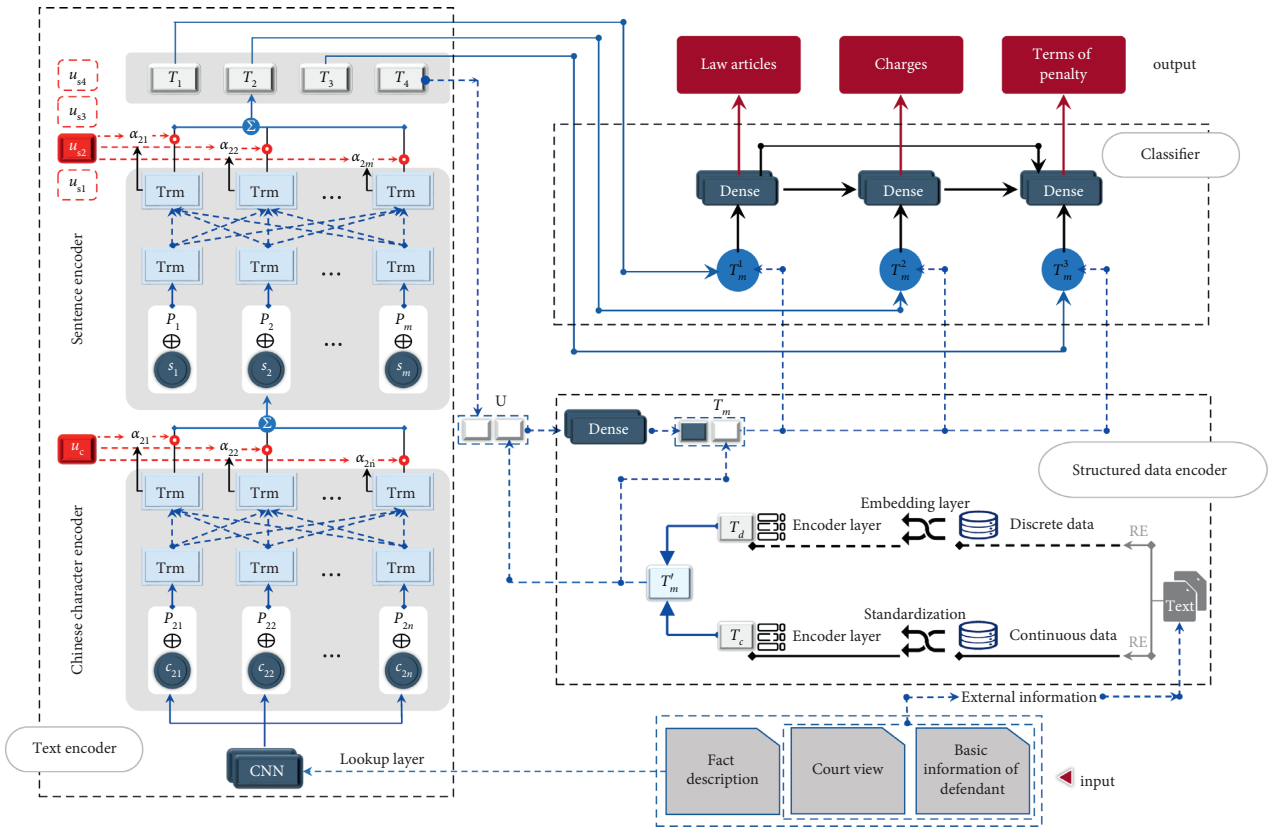

FIGURE 2: General THME framework. Trm denotes the Transformer, and RE denotes the regular expression. u_{ic} represents attention vector in the i-th sentence; u_{s1}, u_{s2}, u_{s3} represent attention vectors which extract textual features of three subtasks of LJP $(T)_1$, T_2, T_3 from sentence-level sequence; and u_{s4} represent attention vectors which extract textual features of case's conclusions $(T)_4$ similarly.

Attention converts Q, K, and V into Q', K', and V' through linear transformation by using a parameter matrix. Next, we apply the Self-Attention mechanism to extract the semantic information. This process is repeated h times. The results are concatenated together, and then the linear transformation is performed. The calculation process is given as follows:

$$Q = C_{pj}, K = C_{pj}, V = C_{pj},$$

$$Q_i' = QW_i^Q, K_i' = KW_i^K, V_i' = VW_i^V,$$

$$\text{Head}_i = \text{Attention}\left(Q_i', K_i', V_i'\right) = \text{softmax}\left(\frac{Q_i' K_i'^T}{\sqrt{d_k}}\right)V_i',$$

$$\text{MHA}\left(Q, K, V\right) = \text{concat}\left(\text{Head}_1, \text{Head}_2, \ldots, \text{Head}_h\right)W^0, \tag{4}$$

where concat() is the vector concatenation operation, d_k is the size of head, and $W^0, W_i^Q, W_i^K, W_i^V \in R^{k\times(k/h)}$ are the parameter matrices.

Add & Norm Layer contains the Add layer and the Norm layer. First, we merge the input of Multihead Attention C_{pj} with the output of MHA and obtain the fact semantic vector M_j as

$$M_j = C_{pj} \oplus \text{MHA}. \tag{5}$$

There are two reasons for this: First, it can make up for the lack of information. Second, it is equivalent to introducing a highway in the network. When the network is backpropagating, a part of it can be directly propagated into the original information without going through the complex network, preventing gradient explosion or gradient disappearance. Then, we employ the Layer Normalization [22] to normalize M_j and obtain $M_j' = (m_{j1}', m_{j2}', m_{j3}', \ldots, m_{jn}')$, $m_{ji}' \in R^k$. Therefore, we obtain the sentence sequence $M_j = (m_{j1}, m_{j2}, m_{j3}, \ldots, m_{jn})$ as

$$M_j = \text{Relu}\left(\left(M_j'W^1 + b_1\right)W^2 + b_2\right), \tag{6}$$

where $W^1, W^2 \in R^{k\times k}$ are the parameter matrices and b_1, b_2 are the basic vectors. Then, we use the attention vector to extract the main information. In order to get the sentence vector s_j, we initialize an attention vector $u_w \in R^n$ and obtain s_j as

$$a_i = u_{ic} \cdot m_{ji},$$

$$a_i' = \text{softmax}\left(a_i\right) = \frac{\exp\left(a_i\right)}{\sum_{j=1}^{n}\exp\left(a_j\right)}, \tag{7}$$

$$s_j = \sum_{i=1}^{n} a_i' m_{ji}.$$

Similarly, we get the sentence sequence $S = (s_1, s_2, \ldots, s_m)$. The sentence encoder is basically the same as the

Chinese character encoder. The difference is that the token vector is replaced with a sentence vector which is produced by the Chinese character encoder.

Since we still use the Transformer to encode the sentence sequence, we first calculate the sentence's position vector P_s and merge it with the sentence sequence S by

$$C_{p_s} = P_s \oplus S. \tag{8}$$

As the input of the Transformer, C_{p_s} passes the Transformer's MHA, Add & Norm Layer, and Feed Forward to obtain a new sentence sequence $S' = (s_1', s_2', \ldots, s_m')$, which has higher-level characteristics and more comprehensive and useful information.

(2) Information extraction. Finally, for our three subtasks of LJP and case's conclusion, we need four different attention vectors to extract four different kinds of information from the same information sequence. We first initialize four attention vectors $u_{s1}, u_{s2}, u_{s3}, u_{s4} \in R^m$ and obtain the vector $T_j \in R^n$ as

$$\begin{aligned} s_i'' &= \tanh\left(s_i' W^{T_j} + b_{T_j}\right), \\ t_i &= u_{sj} \cdot s_i'', \\ t_i' &= \operatorname{softmax}(t_i) = \frac{\exp(t_i)}{\sum_{j=1}^{l} \exp\left(t_j\right)}, \\ T_j &= \sum_{i=1}^{l} t_i' s_i'', \end{aligned} \tag{9}$$

where W^{T_j} is the fully connected matrix and b_{T_j} is the bias vector.

3.4. Structured Data Encoder. The deep learning model is like a judge. We train the model and keep feeding data to the model, just like constantly showing different cases to the judge and training the professional quality of the judge. However, most of the previous work only gave the model to "see" the fact description. In practice, the judge would not sentence the defendant only based on the fact description at the time of judging. In the process of judgment prediction, we sometimes need some explicit data to convict and sentence the defendant. For example, information such as the defendant's guilty attitude, whether to commit recidivism, and the amount of money involved directly affect the final judgment. Based on the above facts, we use the regular expression to extract discrete data and continuous data from the external information, as shown in Tables 2 and 3. In order to well integrate data into THMA, we design both the discrete data encoder and the continuous data encoder, as shown in Figure 3.

3.4.1. Continuous Data Encoder. We normalize each category of continuous data as

$$c_i' = \frac{c_i - \mu_c}{\sigma_c}, \tag{10}$$

where μ_c is the mean of continuous data and σ_c is the variance. We can obtain the continuous data sequence $C' = (c_1', c_2', c_3', \ldots, c_g')$, where g is the number of types of continuous data. Then, we employ a full connection network to fuse different types of continuous data and obtain the continuous data vector T_c as

$$T_c = \operatorname{Relu}\left(C' W^c + b_c\right), \tag{11}$$

where W^c is the fully connected matrix, b_c is the bias vector, and $T_c \in R^p$.

3.4.2. Discrete Data Encoder. Since there are few discrete data categories, we use the word embedding method to create a discrete data vector space for each category of discrete data. We convert each category of discrete data into its word embedding $d_i' \in R^w$. Similarly, we obtain the discrete data vector T_{di} as

$$T_{\mathrm{di}} = \operatorname{Relu}\left(d_i' W^{\mathrm{di}} + b_{\mathrm{di}}\right), \tag{12}$$

where W^{di} is the fully connected matrix, b_{di} is the bias vector, and $T_{\mathrm{di}} \in R^p$. The discrete data sequence is then represented as

$$T_d = \left(T_{d1}, T_{d2}, \ldots, T_{dq}\right), \tag{13}$$

where q is the number of categories of discrete data.

3.4.3. Case's Conclusion Prediction. The specific content of the case's conclusion is presented in Table 4.

In order to predict the case's conclusion, we firstly obtain the combination of discrete data sequence and continuous data vector as T_m', given by

$$T_m' = \operatorname{Concat}\left(T_c, T_d\right). \tag{14}$$

Case's conclusion is very helpful for LJP, especially for the prediction of terms of penalty. For prediction of case's conclusion, the input U is the concatenation of the case's conclusion corresponding vector T_4 and T_m'. Similarly, we obtain the vector of case's conclusion T_{dc} as

$$T_{\mathrm{dc}} = \operatorname{Relu}\left(U W^{\mathrm{dc}} + b_{\mathrm{dc}}\right), \tag{15}$$

where W^{dc} is the fully connected matrix, b_{dc} is the bias vector, and $T_{\mathrm{dc}} \in R^{\mathrm{dc}}$. Finally, we obtain the output of the structured data encoder as

$$T_m = \operatorname{Concat}\left(T_m', T_{\mathrm{dc}}\right). \tag{16}$$

3.5. Classifier. When a judge decides a case, he/she often first searches for the legal basis related to this case such as the fact description. Then, according to the relevant laws, the conviction is made. Finally, intergrating all the evidence and facts, the judge passes the sentence. Therefore, there are topological dependencies among multitask results [5]. We evaluate the performance on three LJP subtasks, including law articles (denoted as t_1), charges (denoted as t_2), and

Table 2: Continuous data.

Category	Case's amount of money involved	Number of previous convictions	Terms of penalty of previous convictions	Penalty of previous convictions
Attribute	Case's information	Information of previous convictions	Information of previous convictions	Information of previous convictions

Table 3: Discrete data.

Category	Case's amount	Case's plot	Case's consequences	Defendant's guilty attitude	Defendant's previous convictions
Attribute	Huge amount Larger amount Extremely huge amount Huge quantity Larger quantity Extremely huge amount	The plot is lighter The plot is bad The plot is serious The plot is particularly bad The plot is particularly serious The plot is slight	Causes serious consequences Causes significant losses Cause particularly serious consequences Causes particularly significant losses	Surrenders oneself Admits actively Guilty attitude is good Guilty attitude is very good	Several

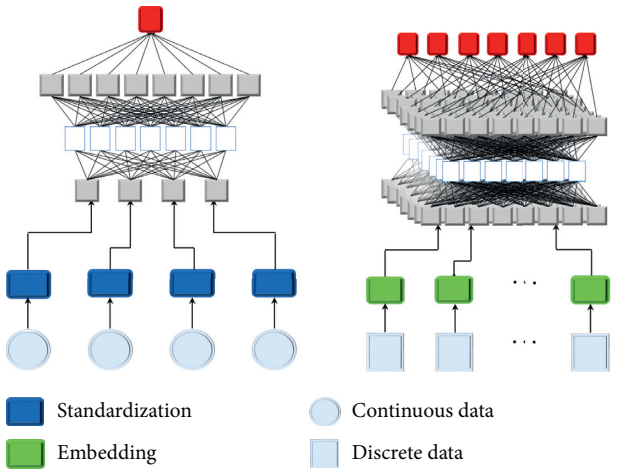

Figure 3: Continuous data encoder and discrete data encoder.

Table 4: Case's conclusion.

Catagory	Case's conclusion
Attribute	Light punishment Lenient punishment Mitigated punishment Severe punishment Heavier punishment

terms of penalty (denoted as t_3). Note that we implement the classifier with dependency in Figure 2; i.e.,

$$\begin{aligned} H_1 &= \phi, \\ H_2 &= t_1, \\ H_3 &= t_1, t_2, \end{aligned} \tag{17}$$

where H_i represents the input of t_i and ϕ is the empty set. This means that the charge prediction depends on law articles, and the terms of penalty prediction depend on both law articles and charges. Such explicit dependencies conform to the judicial logic of human judges, which will be verified in later sections. In order to combine the fact description and the structured data, we concatenate the structured data vector T_m and the i-th subtask's corresponding vector T_i to obtain the vector T_m^i as

$$T_m^i = \text{Concat}\left(T_m, T_i\right), \quad i = 1, 2, 3. \tag{18}$$

Considering the topological dependencies between subtasks, we predict the law article first, then the charge, and finally the terms of penalty. We obtain the law article's vector T_l as

$$\begin{aligned} T_l^1 &= \text{Relu}\left(T_m^1 W_m^1 + b_m^1\right), \\ T_l &= \text{softmax}\left(\text{Re}lu\left(T_l^1 W_l^1 + b_l^1\right)\right). \end{aligned} \tag{19}$$

The processes of charge prediction and terms of penalty prediction are similar with the law article prediction. Different from the law article prediction, the input of the charge prediction is the concatenation of T_m^2 and T_l^1, while the input of terms of penalty prediction is the concatenation of T_m^3, T_l^1, and T_{ch}^1. Finally, we obtain $T_l \in R^x, T_{\text{ch}} \in R^y$, and $T_p \in R^z$, where x, y, z are the number of categories of label for subtasks 1, 2, 3, respectively. In order to learn parameters of THME model, we use the Adam algorithm [23]. We adopt the cross-entropy loss in the training process as follows:

$$L_l^i = -[y \log \hat{y} + (1 - y)\log(1 - \hat{y})], \tag{20}$$

where $\hat{y}$ is the prediction result, y is the real result, l is the law articles prediction, and i is the i-th sample. Equation (20) represents the loss function of one sample in the prediction of the law articles. When there are multiple samples, we add

all the losses together to form the total loss of the law articles. We have three subtasks, so the sum of losses of the three subtasks constitutes the final loss of the model. We train our model in an end-to-end fashion and utilize the dropout [24] to prevent overfitting.

4. Experiments

In this section, we verify the effectiveness of our proposed model. We first introduce the datasets and the data processing. Then, we provide the necessary parameters of our model. Finally, we did some experiments to verify the advantage of our model and the importance of external information.

4.1. Dataset Construction. Since there are no publicly available LJP datasets in previous works, we collect and construct an LJP dataset CJO. CJO consists of criminal cases published by the Chinese government from China Judgment Online[1]. The data used in this experiment is all from the judgment documents published by the Supreme People's Court of China. Before the formal data processing, we first clean the data. Our experiment aims at criminal offense, so other types of judgment documents except criminal offense are screened out. Then, we filter out the multi-criminal judgment documents. The structure of the multi-criminal judgment documents is complicated, and we will research it in our future work. The terms of penalty for a single-criminal judgment document are up to 25 years, so we screen out the judgment documents with the terms of penalty more than 25 years (except death penalty and life imprisonment). Finally, we screened 5480000 judgment documents and obtained 750000 available data pieces. We used the selected 750000 pieces of data for experiments.

Our model's inputs include the token sequence T, the discrete data, and the continuous data. However, we find that our processing approach is not suitable for the terms of penalty of previous convictions. It cannot solve the problem of uneven distribution. Therefore, we discretize the terms of penalty. The specific method is shown in Table 5.

For the majority data in the CJO dataset, their terms of penalty are no longer than 12 months. Meanwhile, the amount of data decreases as the terms of penalty increase. Especially for those with terms of penalty longer than 3 years, the amount of data has dropped significantly. In order to solve the problem of uneven distribution, we use small intervals where data is dense and large intervals where data is sparse, so as to ensure the stability of the amount of data in each interval.

Table 5: Terms of penalty conversion table.

Terms of penalty	Conversion result
No penalty	0
0–6 months	1
6–9 months	2
9–12 months	3
1-2 years	4
2-3 years	5
3–5 years	6
5–7 years	7
7–10 years	8
10–25 years	9
Death or life imprisonment	10

4.2. Baselines. To evaluate the performance of our proposed THME framework, we employ the following text classification models and judgment prediction methods as baselines:

(i) Fact-Law Attention Model [4]: It was proposed by Luo et al. in 2017. The main idea is embedding the law article into the model and then using the fact descriptions to extract the relevant law article to help the model get good results.

(ii) TOPJUDGE [5]: It was proposed by Zhong et al. in 2018. The main idea is using the topological dependencies between subtasks to improve the task effect.

(iii) MPBFN-WCA [6]: It was proposed by Yang et al. in 2019. The main idea is that repeated iterations between subtasks can reduce the error accumulations, thereby improving the effectiveness of the tasks.

4.3. Experimental Settings. We set the word embedding size k as 256. For the discrete data encoder, the dimension of the discrete data embedding w is 32. The dimension of the output vector of the discrete data encoder q is 64, the dimension of the output vector of the continuous data encoder p is 64, and the dimension of the case's conclusion's vector dc is 256.

We use the TensorFlow framework to build neural networks. In the training part, we set the learning rate of Adam optimizer as 0. 0001 and the dropout probability as 0.5. The padding length of the text N is 320 tokens, the length of each sentence n is 16 tokens, and each text is divided into 20 sentences. We set the batch size as 256 for all models. We train each model for 256 epochs, and if overfitting occurs, we will terminate the training early.

We employ accuracy (Acc.), macro − precision(MP), macro − recall(MR), and macro $-F_1$ (F_1) as evaluation metrics. Here, the macro-precision/recall/F_1 is calculated by averaging the precision/recall/F_1 of each category.

4.4. Results and Analysis. All the models are repeated 3 times, and we evaluate the performance on three LJP subtasks, including law articles, charges, and terms of penalty and report the average values as the final results for clear illustration. Experimental results on the test set of CJO are shown in Table 6. It is shown that THME achieves the best performance on all metrics. Thus, the effectiveness and robustness of our proposed framework are verified. Compared with TOPJUDGE and MPBFN-WCA, THME takes advantage of the information of the fact determination and

thus achieves promising improvements. It indicates that the external information enables the model to learn rules that are not in the original fact description. Compared with Fact-Law Model, our model takes advantage of the correlation among relevant subtasks and achieves significant improvements. Thus, it is important to properly model topological dependencies between different subtasks.

4.5. Ablation Study. To further illustrate the significance of modules in our framework. Compared to THME, we designed the following models:

(i) Transformer-HAN-Single-Extra (THSE): We decompose the multitask model into a single-task model to verify the superiority of the multitask model.

(ii) Transformer-HAN-Single (THS): In order to reflect the role of continuous data and discrete data based on the fact description in a single-task, we design THS to compare the effect with THSE.

(iii) Transformer-HAN-Multi (THM): In order to reflect the role of continuous data and discrete data based on the fact description in multitasking, we design THM to compare the effect with THME.

(iv) GRU-HAN-Multiextra (GHME): In order to prove the role of Transformer in the model, we design the GHME model and the THME to compare their effects.

As shown in Table 7, compared with THS, THM can improve the performance by 1.52%, 4.8%, and 4.53% for law article prediction, charge prediction, and terms of penalty prediction in our dataset, respectively. Thus, multitask model is beneficial to improve the performance of each task. THSE performs better than THS, especially in terms of penalty prediction. THSE has enhanced the performance by 2.51%. Thus, the structured data based on the fact description plays an important role, even if the single-task model is also significantly better than the multitask model without the addition of structured data. Hence, the structured data plays a more important role compared with the multitask structure.

Through comparing GHME and THS, we can see that THS performs better, which indicates that the performance of Transformer is better than the traditional GRU model in handling long documents and the effect of Transformer-HAN on LJP is greater than that of the multitask topological structure and the external information. This also proves that the proposed Transformer-HAN is a state-of-the-art model to deal with long-term dependency problems.

4.6. Information Source Study. To further show the significance of the external information and explore the impacts of the information source, we evaluate the performance of THME under various information sources. We remove all the external information (fact), court view (-court view), defendant's information (-defendant's information), and case's conclusion (-case's conclusion), respectively. Results are summarized in Table 8.

It is shown that the performance of THME gets worse for all tasks after removing either origin of information. More specifically, when we remove all the external information, tremendous decrease is observed for the terms of penalty prediction. This demonstrates that the external information is beneficial for terms of penalty prediction. When we remove the defendant's information, the performance is better than when removing the court view. This also demonstrates that the court view is more significant than the defendant's information and it plays a decisive role in LJP. The case's conclusion comes from the court view. When we remove the case's conclusion, the effect of THME is worse than the situation of removing the defendant's information, which is similar to the situation of removing the court view. This demonstrates that the case's conclusion plays a very important role in LJP.

4.7. Error Analysis and Solution. Prediction errors induced by our proposed model can be traced down into the following causes.

4.7.1. Data Imbalance. Data imbalance is a natural phenomenon, because the number of cases with long terms of penalty is significantly less than those with short terms of penalty. Although we have adopted effective techniques to discretize the terms of penalty to reduce the impact of data imbalance, for the subtasks of law articles and charges, our model achieves more than 90% on accuracy, while only about 75% for macro-F1. This issue is much more severe on the subtask of the terms of penalty, for which our model yields a poor performance of only 40% macro-F1. The bad performance is mainly due to the imbalance of category labels; e.g., there are only a few training instances where the term is "life imprisonment or death penalty." Most judgment prediction approaches perform poorly (especially for recall) on these labels as listed in Figure 4.

4.7.2. Terms of Penalty Problem. It can be seen from the results that although our model surpasses other models in terms of penalty prediction, the effects of terms of penalty prediction is still very poor. The accuracy rate is only 56.89%, and the macro-average index is even less than 50%. Such an index is far from meeting the actual needs. The actual cases are often multiple criminal cases, which are much more complicated than the cases we are analyzing, but complex cases often contain more information, which also provide us with ideas for solving the problem of terms of penalty prediction. In multiple criminal cases, we can split the case into multiple subcases and then comprehensively consider the categories of subcases, the number of subcases, and the severity of subcases to provide more information for terms of penalty prediction. The specific implementation method remains to be explored.

Table 6: Judgment prediction results on CJO.

	Tasks	Law articles				Charges				Terms of penalty			
	Metrics	Acc.	F1	MP	MR	Acc.	F1	MP	MR	Acc.	F1	MP	MR
Baselines	Fact-Law	88.95	70.56	72.31	71.22	92.35	78.67	80.47	78.83	52.15	37.62	38.51	38.43
	TOPJUDGE	89.55	63.38	63.78	65.72	93.90	66.51	66.60	67.26	53.71	38.71	37.43	41.32
	MPBFN-WCA	88.35	71.54	74.09	70.90	89.78	57.54	66.78	55.81	46.00	20.40	23.74	22.83
Ours	**THME**	**90.93**	**77.26**	**80.24**	**76.43**	**96.36**	**81.77**	**83.05**	**81.04**	**56.88**	**45.71**	**47.66**	**46.07**

Table 7: Ablation study on CJO.

	Tasks	Law articles				Charges				Terms of penalty			
	Metrics	Acc.	F1	MP	MR	Acc.	F1	MP	MR	Acc.	F1	MP	MR
	THS	87.69	72.88	75.31	71.26	92.23	74.71	75.01	71.33	52.10	31.89	34.30	32.01
	THM	89.40	74.13	76.10	73.60	95.14	78.97	80.07	78.28	52.78	37.78	38.73	39.14
	THSE	90.57	69.75	70.55	70.09	96.12	75.02	75.59	74.83	53.41	38.48	41.46	40.09
	GHME	86.02	50.03	55.81	50.26	94.11	53.92	59.14	53.85	49.88	27.59	32.17	28.52
Ours	**THME**	**90.93**	**77.26**	**80.24**	**76.43**	**96.36**	**81.77**	**83.05**	**81.04**	**56.88**	**45.71**	**47.66**	**46.07**

Table 8: Comparable results of the effect of different information sources in the model.

	Tasks	Law articles				Charges				Terms of penalty			
	Metrics	Acc.	F1	MP	MR	Acc.	F1	MP	MR	Acc.	F1	MP	MR
Baselines	Fact	89.51	73.72	76.75	74.43	95.11	78.53	79.44	78.41	50.94	36.95	39.35	39.66
	-Court view	90.00	74.99	78.03	74.60	95.39	79.48	80.89	78.92	55.57	42.40	45.98	42.55
	-Defendant's information	90.90	76.02	**81.41**	75.52	96.35	81.35	82.74	80.71	55.61	43.86	**48.05**	43.84
	-Case's conclusion	90.40	75.13	77.68	75.06	96.13	80.15	81.85	79.36	55.08	43.22	46.75	43.19
Ours	**THME**	**90.93**	**77.28**	80.26	**76.44**	**96.36**	**81.77**	**83.06**	**81.04**	**56.89**	**45.72**	47.68	**46.08**

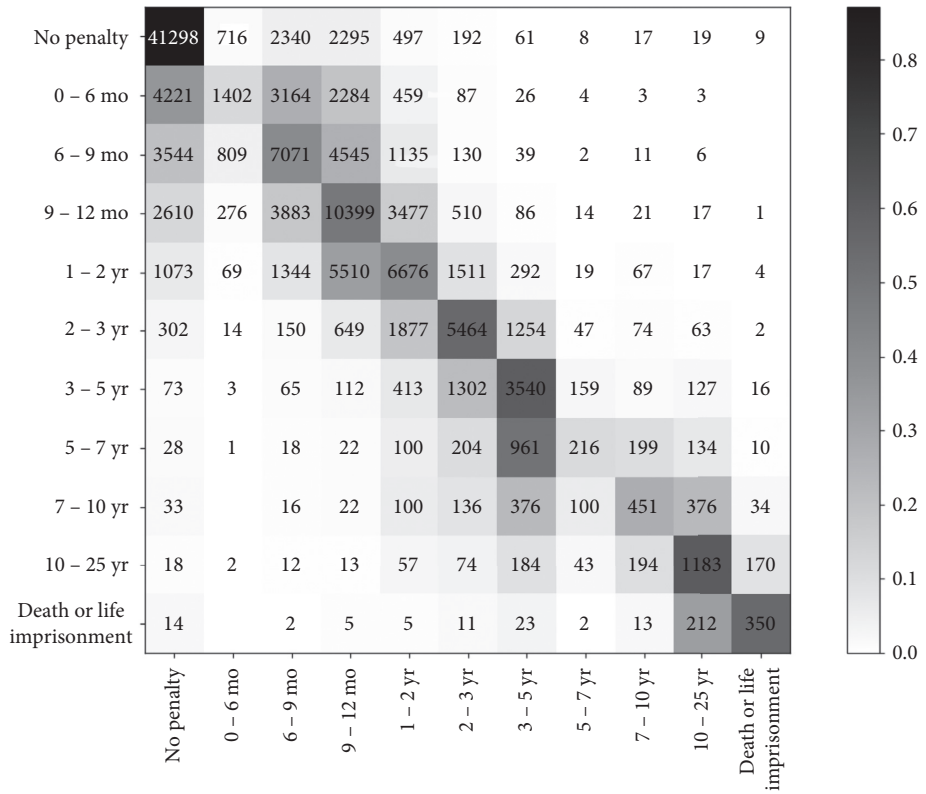

Figure 4: The confusion matrix in the subtask of predicting the terms of penalty. The rows denote the prediction truth while the columns denote the ground truth.

5. Conclusion

In this paper, we have studied the multi-extra and multi-task of LJP with topological dependencies between subtasks and address the problem of insufficient information and insufficient coding in LJP. Based on the topological structure between multiple tasks, we extract the information from the fact description via the Transformer-HAN encoder, extract the external information from the judgment document by the structured data encoder, and then integrate them into the classifier to reduce the misjudgment of penalty prediction. Experimental results show that our model achieves significant improvements over baselines for all judgment prediction tasks.

In the future, we will seek to explore the following directions: (1) It is interesting to explore the multitask legal prediction with multiple labels and multiple defendants. In recent years, the rise of knowledge graphs and graph neural networks (GNN) has made this possible [25–28]. (2) We will explore how to incorporate various factors into LJP, such as defendant's subjective viciousness, defendant's criminal means, and defendant's identity, which are not considered in this work. (3) When a judge decides a case, similar cases are crucial to the judgment result for this case. Therefore, we can also recommend similar judgment documents to judges [29–31]. (4) With more and more research on the transfer learning, GPT, Bert, and other natural language models are also produced and continuously improve the ability to extract information from the text. The use of transfer learning in the process of dealing with the fact descriptions may improve the effectiveness of models [32–34].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Kongfan Zhu and Rundong Guo contributed equally to the paper.

Acknowledgments

This work was supported in part by the Key Research and Development Program of China under Grant no. 2018YFC0831000 and no. 2017YFC0803400.

References

[1] J. A. Segal, "Predicting supreme court cases probabilistically: the search and seizure cases, 1962–1981," *American Political Science Review*, vol. 78, no. 4, pp. 891–900, 1984.

[2] C. Liu, C. Chang, and J. Ho, "Case instance generation and refinement for case-based criminal summary judgments in Chinese," *Journal of Information Science and Engineering*, vol. 20, no. 4, pp. 783–800, 2004.

[3] D. M. Katz, M. J. Bommarito, 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.

[4] B. Luo, Y. Feng, J. Xu, X. Zhang, and D. Zhao, "Learning to predict charges for criminal cases with legal basis," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2727–2736, Copenhagen, Denmark, September 2017.

[5] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, and M. Sun, "Legal judgment prediction via topological learning," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3540–3549, Brussels, Belgium, November 2018.

[6] W. Yang, W. Jia, X. Zhou, and Y. Luo, "Legal judgment prediction via multi-perspective bi-feedback network," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence IJCAI-19*, pp. 4085–4091, Melbourne, Australia, August 2019.

[7] W. Zaremba, I. Sutskever, and O. Vinyals, *Recurrent Neural Network Regularization*, https://arxiv.org/abs/1409.2329.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[9] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

[10] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, San Diego, CA, USA, June 2016.

[11] Y.-H. Liu, Y.-L. Chen, and W.-L. Ho, "Predicting associated statutes for legal problems," *Information Processing & Management*, vol. 51, no. 1, pp. 194–211, 2015.

[12] Q. Bao, H. Zan, P. Gong, J. Chen, and Y. Xiao, "Charge prediction with legal attention," in *Proceedings of the CCF International Conference on Natural Language Processing and Chinese Computing*, pp. 447–458, Dunhuang, China, October 2019.

[13] Y. Shen, J. Sun, X. Li, L. Zhang, Y. Li, and X. Shen, "Legal article- aware end-to-end memory network for charge prediction," in *Proceedings of the 2nd International Conference on Computer Science and Application Engineering*, pp. 1–5, Hohhot, China, October 2018.

[14] H. Chen, D. Cai, W. Dai, Z. Dai, and Y. Ding, "Charge-based prison term prediction with deep gating network," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, no. 1, pp. 6361–6366, Hong Kong, China, November 2019.

[15] X. Jiang, H. Ye, Z. Luo, W. Chao, and W. Ma, "Interpretable rationale augmented charge prediction system," in *Proceedings of the Coling 2018*, pp. 146–151, Santa Fe, NM, USA, August 2018.

[16] S. Chen, P. Wang, W. Fang, X. Deng, and F. Zhang, "Learning to predict charges for judgment with legal graph," in *Artificial Neural Networks and Machine Learning–ICANN 2019*, pp. 240–252, Springer, Berlin, Germany, 2019.

[17] Y. S. Chen, S. W. Chiang, and T. Y. Juang, "A few-shot transfer learning approach using text-label embedding with legal attributes for law article prediction," *EasyChair, Technical Representative*, vol. 1344, 2019.

[18] Z. Yang, P. Wang, L. Zhang, L. Shou, and W. Xu, "A recurrent attention network for judgment prediction," in *International Conference on Artificial Neural Networks*, pp. 253–266, Springer, Berlin, Germany, 2019.

[19] O.-M. Sulea, M. Zampieri, S. Malmasi, M. Vela, P. L. Dinu, and V. J. Genabith, *Exploring the Use of Text Classification in the Legal Domain*, ASAIL@ICAIL, London, UK, 2017.

[20] P. Wang, Y. Fan, S. Niu, Z. Yang, and J. Guo, "Hierarchical matching network for crime classification," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 325–334, Paris, France, July 2019.

[21] X. Li, Y. Meng, X. Sun, Q. Han, A. Yuan, and J. Li, "Is word segmentation necessary for deep learning of Chinese representations?," in *Proceedigs of the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3242–3252, Florence, Italy, July 2019.

[22] J. L. Ba, R. Kiros, and E. G. Hinton, *Layer Normalization*, https://arxiv.org/abs/1607.06450.

[23] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," https://arxiv.org/abs/1412.6980.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdi- nov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] S. Pan, R. Hu, S.-f. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2475–2487, 2019.

[26] J. Shaoxiong, P. Shirui, C. Erik, M. Pekka, and P.S. YU, "A survey on knowledge graphs: representation, acquisition and applications," https://arxiv.org/abs/2002.00388.

[27] S. Pan, J. Wu, X. Zhu, C. Zhang, and S. Y. Philip, "Joint structure feature exploration and regularization for multi-task graph classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 3, pp. 715–728, 2015.

[28] T. Guo, S. Pan, X. Zhu, and C. Zhang, "Cfond: consensus factorization for co-clustering networked data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 706–719, 2018.

[29] F. Xiong, W. Shen, H. Chen, S. Pan, X. Wang, and Z. Yan, "Exploiting implicit influence from information propagation for social recommendation," *IEEE Transactions on Cybernetics*, vol. 50, no. 10, pp. 4186–4199, 2019.

[30] F. Xiong, X. Wang, S. Pan, H. Yang, H. Wang, and C. Zhang, "Social recommendation with evolutionary opinion dynamics," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 10, pp. 1–13, 2018.

[31] Z. Li, F. Xiong, X. Wang, H. Chen, and X. Xiong, "Topological influence-aware recommendation on social networks," *Complexity*, vol. 2019, Article ID 6325654, 12 pages, 2019.

[32] X. Ben, P. Zhang, Z. Lai, R. Yan, X. Zhai, and W. Meng, "A general tensor representation framework for cross-view gait recognition," *Pattern Recognition*, vol. 90, pp. 87–98, 2019.

[33] X. Ben, C. Gong, P. Zhang, R. Yan, Q. Wu, and W. Meng, "Coupled bilinear discriminant projection for cross-view gait recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 3, pp. 734–747, 2020.

[34] X. Ben, C. Gong, P. Zhang, X. Jia, Q. Wu, and W. Meng, "Coupled patch alignment for matching cross-view gaits," *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 3142–3157, 2019.