ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20210416055250
Preliminary communication

Intelligent Case Assignment Method Based on the Chain of Criminal Behavior Elements

Shaolin AO, Yongbin QIN*, Yanping CHEN, Ruizhang HUANG

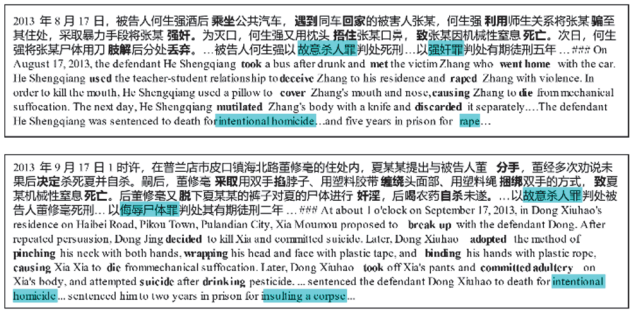
Abstract: The assignment of cases means the court assigns cases to specific judges. The traditional case assignment methods, based on the facts of a case, are weak in the analysis of semantic structure of the case not considering the judges' expertise. By analyzing judges' trial logic, we find that the order of criminal behaviors affects the final judgement. To solve these problems, we regard intelligent case assignment as a text-matching problem, and propose an intelligent case assignment method based on the chain of criminal behavior elements. This method introduces the chain of criminal behavior elements to enhance the structured semantic analysis of the case. We build a BCTA (Bert-Cnn-Transformer-Attention) model to achieve intelligent case assignment. This model integrates a judge's expertise in the judge's presentation, thus recommending the most compatible judge for the case. Comparing the traditional case assignment methods, our BCTA model obtains 84% absolutely considerable improvement under P@1. In addition, comparing other classic text matching models, our BCTA model achieves an absolute considerable improvement of 4% under P@1 and 9% under Macro F1. Experiments conducted on real-world data set demonstrate the superiority of our method.

Keywords: intelligent case assignment; neural networks; text matching; text representation

1 INTRODUCTION

With the rapid development of big data and artificial intelligence technology, in the judicial field, countries all over the world are advancing judicial intelligence research. In this process, judicial intelligence assistance is a hot research issue of smart justice. In the complicated trial procedure, the assignment of cases is the starting point for cases to enter the trial procedure and the key to the reasonable allocation of judge resources. With the increasingly prominent contradiction of "more cases and fewer people", the study of an intelligent case assignment method, which is to protect the interests of the parties and ensure the compatibility between cases assigned and judges' professional ability, has great theoretical significance and application value.

According to surveys, there are two modes of traditional case assignment mechanism: manual and random case assignment. In manual case assignment, the chief judge appoints judges, and in random case assignment, a computer assigns judges. Thus, the traditional mechanism has some shortcomings: Firstly, it is weak to analyse the semantic structure of the case. Exactly, based on the facts of cases, these methods are easy to ignore the order of criminal behaviors, because the same criminal behaviors in a different order often led to different trial results. As in the example of two Chinese cases shown in Fig. 1, both only involved murder and rape, but the different order of occurrence led to different trial results.


2013 年 8 月 17 日，被告人何生强酒后 **乘坐**公共汽车，**遇到**同车 **回家**的被害人张某，何生强 **利用**师生关系将张某 **骗**至其住处，采取暴力手段将张某 **强奸**。为灭口，何生强又用枕头 **捂住**张某口鼻，**致**张某因机械性窒息 **死亡**。次日，何生强将张某尸体用刀 **肢解**后分处 **丢弃**。…被告人何生强以 故意杀人罪 判处死刑…以 强奸罪 判处有期徒刑五年 … ### On August 17, 2013, the defendant He Shengqiang **took** a bus after drunk and **met** the victim Zhang who **went home** with the car. He Shengqiang **used** the teacher-student relationship to **deceive** Zhang to his residence and **raped** Zhang with violence. In order to kill the mouth, He Shengqiang used a pillow to **cover** Zhang's mouth and nose, **causing** Zhang to **die** from mechanical suffocation. The next day, He Shengqiang **mutilated** Zhang's body with a knife and **discarded** it separately…The defendant He Shengqiang was sentenced to death for intentional homicide…and five years in prison for rape…

2013 年 9 月 17 日 1 时许，在普兰店市皮口镇海北路董修亳的住处内，夏某某提出与被告人董 **分手**，董经多次劝说未果后 **决定**杀死夏并自杀。嗣后，董修亳 **采取**用双手 **掐**脖子、用塑料胶带 **缠绕**头面部、用塑料绳 **捆绑**双手的方式，**致**夏某机械性窒息 **死亡**。后董修亳又 **脱**下夏某某的裤子对夏的尸体进行 **奸淫**，后喝农药 **自杀**未遂。…以 故意杀人罪 判处被告人董修亳死刑… 以 侮辱尸体罪 判处其有期徒刑二年 … ### At about 1 o'clock on September 17, 2013, in Dong Xiuhao's residence on Haibei Road, Pikou Town, Pulandian City, Xia Moumou proposed to **break up** with the defendant Dong. After repeated persuasion, Dong Jing **decided** to kill Xia and committed suicide. Later, Dong Xiuhao **adopted** the method of **pinching** his neck with both hands, **wrapping** his head and face with plastic tape, and **binding** his hands with plastic rope, **causing** Xia Xia to **die** frommechanical suffocation. Later, Dong Xiuhao **took** off Xia's pants and **committed adultery** on Xia's body, and attempted **suicide** after **drinking** pesticide. ... sentenced the defendant Dong Xiuhao to death for intentional homicide... sentenced him to two years in prison for insulting a corpse...

Figure 1 Two examples of different accusations caused by a different order of behaviour

In the first case, the defendant first raped and then murdered the victim and was convicted of intentional homicide and rape. While in the second one, the defendant first murdered and then raped the victim and was convicted of insulting a corpse and intentional homicide. Secondly, judges' professional ability is not taken into consideration when cases are assigned, which can easily lead to incompatibility between cases assigned and judges' professional ability. Therefore, to solve the above problems, the development of an intelligent method of case assignment has great theoretical significance and application value.

In the long-term judicial practice, the court has accumulated a large amount of judgement document data. A judgement document is composed of the facts of a case, the results of judgements, the judges, and so on. Judgment documents are usually presented in text form, which contains important case information and knowledge value. How to extract valuable information from the text data to serve the intelligent case assignment is the key issue in the current research. By analyzing judges' trial logic, it is found that judges focus on the order of criminal behaviors and then make a final judgment. In this process, the behavioral elements of a case play a key role. The behavioral elements of a case describe behavior words of case process and they also describe the key elements of the case which related to the behavior words. Generally, a case can be regarded as being composed of a series of temporal behavioral elements. Thus, analyzing the elements of criminal behavior with temporal relationship is helpful to model the semantic information of cases, on which our work is based. The main contributions of this paper are as follows:

(1) This is the first work regarding intelligent case assignment as a text-matching problem between cases and judges and introducing a deep neural network to realize intelligent case assignment in the legal domain.

(2) This is the first work focusing on the importance of the elements of criminal behavior and the order of criminal behaviors. We propose to build the chain of criminal behavior elements with temporal relationships to model case structured semantic information, which enhances the ability to represent cases.

(3) We build a BCTA (Bert-CNN-Transformer-Attention) model to achieve intelligent case assignment. The model

integrates information about a judge's expertise in the judge's representation recommending the most compatible judge for the case. Experiments show that this method can significantly improve the accuracy of the assignment of cases.

The rest of this paper is organized as follows. Section 2 presents related works. Section 3 provides the essential definitions of intelligent case assignment. Section 4 describes our models. Section 5 discusses the experiments and results. The conclusion is given in Section 6.

2 RELATED WORKS

Our work is related to several research areas, including text representation, text matching models, and judicial intelligence research.

Text representation is an important basic task for many NLP tasks. The text representation model can be roughly classified into three categories. The first category is focused on text features, and the second category is based on topic features. Examples include VSM [1], LDA [2], LSI [3] and SSI [4]. These two types of text representation methods cannot model the position information and context information of words. The third category is a text representation model based on neural networks. Compared with previous methods, the neural network-based text representation method solves the problems of high-dimensional sparseness and lack of semantic association in the representation text. Word vectors are also called word embedding. Hinton et al. [5] proposed the concept of distributed representation, and Bengio et al. [6] proposed a neural network language model (NNLM), which opened the study of distributed representation methods for words based on neural networks. The most commonly used model is Word2Vec [7, 8]. The distributed representation of words has been greatly developed since the development of Word2Vec. Pennington et al. [9] proposed glove to learn word vectors in a global sense. Bojanowski et al. [10] proposed FastText to learn the morphological information of words. Prior to the FastText model, the representation of words was independent of the top and bottom. As ELMo [11], *BERT* [12] and other models have been proposed, text representation not only considers the morphological information of the word but also takes into account the context and semantic information. Recently, in the field of artificial intelligence and law, various neural network architectures such as CNN [13] and RNN [14] have been used for document embedding. Jiang et al. [15] use deep reinforcement learning methods to improve classification accuracy. Kang et al. [16] use CNN and GRU to improve the performance of the experimental results. Legal cases are often represented in text form. The key to our work is how to represent judges and cases. In view of the superiority of neural network-based text representation methods, we use neural network-based text representation methods to represent judges and cases.

In intelligent case assignment, the facts of a case are input, and all judges in the court are matched. This task can be regarded as a text-matching task. Many NLP tasks can be formulated as a matching problem between two texts. There have been many deep learning models proposed for text matching and ranking. These text matching models could be classified into three categories. The first category is a deep learning model based on single semantic document expression, which first learns the vector representations of two documents independently and then uses functions (such as vector dot product, cosine similarity function and MLP network) to calculate the similarity between the learned feature vectors. Typical models include DSSM (deep structured semantic models) [17]. However, they cannot capture the interactive information between texts and usually cannot achieve good performance. The second category is a deep learning model based on multi-semantic document expression, which is a multi-angle and multi-granularity generation of text vector representation for matching that can effectively reduce information loss. For example, MV-LSTM (multi-variable LSTM) [18], which uses Bi-LSTM to encode the text, interacts with the vectors at each moment of two sentences. However, the intrinsic structural properties of texts are not utilized by these models. The third category is the direct modelling matching mode, which pays attention to how to represent the text and pays attention to the dependence between text pairs. For example, Yin et al. [19] construct a similarity matrix of two sentences and then apply convolution to the matrix to extract features. Radford et al. [20] use RNN to build a Siamese network and use attention to capture the interactive information of two sentences. The third category considers the matching degree and matching structure at the same time, they achieve significant improvements in multiple text matching tasks. Following the work of text matching models, we build an interaction-based matching model to fully capture the semantic information and inherent structural information of texts.

The application of artificial intelligence technology in law has become an important aspect of judicial intelligence research. To date, some achievements have been made in judicial intelligence research, which focuses on the task of legal judgment prediction (LJP). For a given case, the task of LJP aims to empower machines to predict the judgment results (e.g., law articles, charges, and prison terms) of the case. Inspired by the success of deep learning techniques [13, 14, 21] on NLP tasks, researchers attempt to employ neural models to handle judgment prediction tasks. Some popular neural network methods are used in an automatic charge prediction task [22-24], and there are some works focusing on identifying applicable law articles for a given case [25-27]. In addition, some researchers focus on other areas of justice such as entity recognition [28, 29], court opinion generation [30] and analysis [31].

3 PROBLEM FORMULATION

In this section, some notations and terminologies will be introduced, followed by the essential definitions of intelligent case assignment.

Legal Cases Legal cases are ultimately presented in the form of judgment documents. By analyzing judges' trial logic and the composition of a judgment document, it is found that the facts of a case are the key information of the case. Therefore, we extract the facts of the case to represent the case. Supposing the facts of a case as a word sequence $fact = \{w_1, w_2, \ldots, w_n\}$, where n is the number of words.

Intelligent Case Assignment The purpose of intelligent case assignment is to recommend judges automatically for cases. First, for each case, we construct the chain of

criminal behavior elements $cbe = \{e_1, e_2, \ldots, e_m\}$, where m is the number of criminal behavior elements in the case. We use the language technology platform (LTP); (https://github.com/HIT-SCIR/ltp) to extract m behavior elements from the facts of the case. The LTP is developed by the Social Computing and Information Retrieval Research Center of Harbin Institute of Technology. Then, the feature vector of the facts of the case v_{fact} and the feature vector of the criminal behavior element chain v_{chain} are joined together to represent the case v_{case}. Formally, let $v_{case} = v_{fact} \otimes v_{chain}$ denote the feature vector of the case, and $v_{judge} = [fg_1, fg_2, \ldots, fg_i]$ denote the judge's matrix vector, where fg_i represents the feature vector of the i-th judge, and k is the number of judges in the court. Given a training dataset $D \doteq [<v_{case}, v_{judge}>]$, we aim to train a model $F(\cdot)$ so as to recommend a trail judge for any test case.

4 OUR METHODOLOGY

In this paper, we build a BCTA (Bert-Cnn-Transformer-Attention) model to realize intelligent case assignment. The architecture of this model is shown in Fig. 2. Our method consists of three parts: the representation of the case, the representation of the judge, and the match between the case and the judge. The representation of the case is generated by the behavior chain encoder and the fact encoder. The judge's representation is generated by the judge encoder. The match between the judge and the case is realized by case assignment module. In the following subsections, we discuss our method in detail.

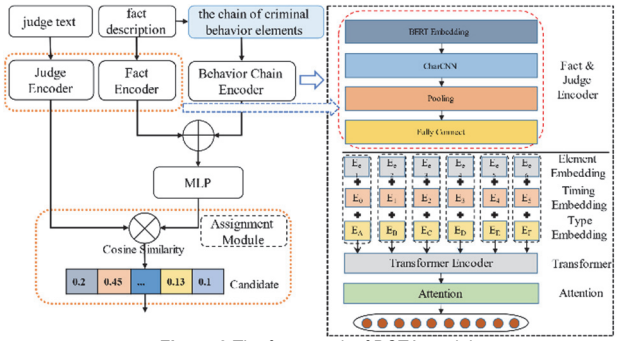


Figure 2 The framework of BCTA model

4.1 Case Representation Method

To solve the problem of weakly structured semantic analysis of the case in the traditional case assignment method, we construct the chain of criminal behavior elements with temporal relationships to enhance the ability to represent cases. We first extract the behavior elements from the facts of the case and then construct the chain of criminal behavior elements according to the order of behavior elements. Below, we first introduce the construction method of the chain of criminal behavior elements and then the representation method of the case.

The process of constructing the chain of criminal behavior elements is shown in Fig. 3. We use LTP to extract elements of criminal behavior and to build the relationships between elements. The specific steps are: first, we use LTP to preprocess the facts of the case; second, we use the semantic role tagging toolkit in LTP to perform semantic analysis on sentences; third, we filter elements according to Chinese grammar rules; and finally, according to the order of the elements of criminal behavior, we construct the chain of criminal behavior elements with temporal relationships. For instance, as shown in Fig. 4, LTP can extract three criminal behavioral words, namely, selling, arrested, and confiscated, and mark the semantic relationship between the elements as "A0", "A1", etc. Here "A0" represents the agent of the behavior, "A1" means the recipient of the behavior, "TMP" means the time when the behavior occurs, "LOC" means the place where the behavior occurs, "BNF" means the beneficiary of the act, and "ADV" means the adverbial modifier behavior. Therefore, according to the temporal relationship of the behavior elements, we can obtain the chain of criminal behavior elements in this case as (selling drugs, the public security organs arrested the defendant Wei Pengpeng, the public security organs confiscated 2 grams of drugs).

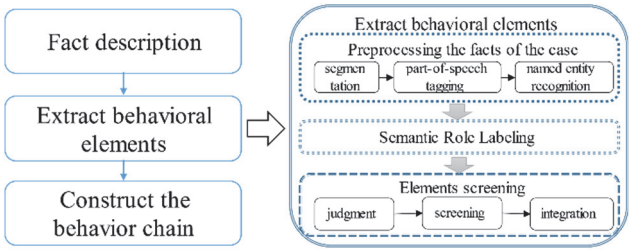


Figure 3 Flow chart of the construction of the chain of criminal behavior elements

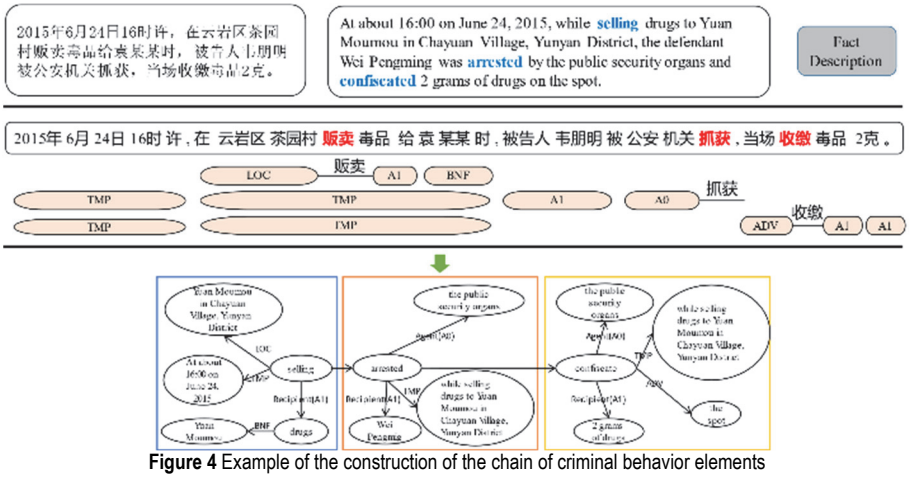


Figure 4 Example of the construction of the chain of criminal behavior elements

4.1.1 Behavior Chain Encoder

For a given case, $fact = \{w_1, w_2, \ldots, w_n\}$, is a word sequence of the facts of the case, where n is the number of words in the facts of the case, and $cbe = \{e_1, e_2, \ldots, e_m\}$, is the chain of criminal behavior elements of the case, where m is the number of behavior elements, such as "the public security organs". The behavior chain encoder is shown in Fig. 2. Next, we will introduce the layers.

BERT Layer To transform each element in the behavior element chain into a vector, a commonly used word embedding methods include a random look-up table and a pre-trained language model. To capture stronger semantic information, *BERT* is used here to obtain the feature representation of each behavior element and learn the semantic information within each element. For the chain of criminal behavior elements $cbe = \{e_1, e_2, \ldots, e_m\}$, each element is randomly mapped to obtain a sequence matrix $v = \{v_1, v_2, \ldots, v_m\}$, where $v_i \in R^k$ represents the vector representation of the i-th behavior element. After the *BERT* layer, we can obtain the feature representation $x = [x_1, x_2, \ldots, x_m]$, where $x_i \in R^h$ is the feature vector output by the *BERT* hidden layer, and h is the dimension of the *BERT* hidden layer. This process can be formalized as:

$$x = BERT(cbe) \tag{1}$$

CharCNN Layer To model the semantic relationship between behavior elements and the global dependency of the chain, the CharCNN layer is followed by the *BERT* layer. The dependency between the elements is learned through the convolution window. The convolution operation can be denoted as:

$$c = Conv(x) \tag{2}$$

Usually, multiple convolution windows of different sizes are set to obtain feature vectors with different granular information. If there are three convolution windows of different sizes, the obtained feature vector matrix is $C = [c_1, c_2, c_3]$.

Pooling Layer After the CNN layer, to obtain more valuable features in the behavior element chain, a pooling operation is performed on the output results of the convolution. Pooling operations include maximum pooling, average pooling, and so on. In this paper, a max operation is implemented for the behavior element chain. The pooling operation can be formalized as:

$$s = MaxPooling(c) \tag{3}$$

Fully Connected Layer After the *BERT* embedding operation, the convolution operation, and the pooling operation, the raw input sequence of the behavior element chain is transformed into a high-level abstract feature vector. Then, a fully connected layer can be adopted to give a global regulation, denoted by *Conn*. The process can be summarized as:

$$O = Conn(s) \tag{4}$$

BERT and CharCNN can learn the dependencies within and between behavioral elements. Here, we regard *BERT*, CharCNN, the pooling layer and the fully connected layer as the embedding layer of the behavior chain encoder as a whole. The output of the embedding layer is denoted by *element embeddings*. Inspired by the idea of *BERT*'s position embedding, we add temporal relationships in the order in which behavior elements occur, denoted by *timing embeddings*. And the type of each behavior element is also marked when extracting behavior elements and their relationships. In this paper, we aggregate the typical feature of the elements as external features into the representation of the behavior element chain, denoted by *type embeddings*. We use a randomly initialized lookup table to generate temporal series feature representations $TE = [te_1, te_2, \ldots, te_m]$ and type feature representations $TY = [ty_1, ty_2, \ldots, ty_m]$ of the behavior elements, where $te_i \in R^h$ represents the temporal series vector of each behavior element and $ty_i \in R^h$ represents the type vector of each behavior element. Thus, the feature vector of the criminal behavior element chain can be represented as:

$$ACT = O + TE + TY \tag{5}$$

Transformer Layer A transformer can be regarded as a graph attention network (GAT) [32]. To model better the temporal relationship of the criminal behavior element chain, we perform a transformer encoder operation on the output of the embedding layer. A transformer encoder computes the representation of each word through an attention mechanism with respect to the surrounding words. For the given behavior chain $cbe = \{e_1, e_2, \ldots, e_m\}$, we can obtain a semantic vector $behave = \{b_1, b_2, \ldots, b_m\}$, after the embedding layer and the transformer encoder, where $b_i \in R^h$. The process can be formalized as:

$$behave = TransformerEncoder(ACT) \tag{6}$$

Attention Layer The attention mechanism [33-35] lets the model capture the whole traffic dynamics in the input sequence. Inspired by [36, 37], we use the attention mechanism as an attention pooling mechanism, which can pay attention to the key element in the chain of criminal behavior elements and maintains the most meaningful information of the facts of the case. The formula is defined as follows:

$$u_i = \tan h\left(W_s b_i^T\right) \tag{7}$$

$$\alpha_i = \frac{\exp\left(u_i\right)}{\sum_j \exp\left(u_j\right)} \tag{8}$$

$$v_{chain} = \sum_{i=0}^{m} \alpha_i behave \tag{9}$$

where W_s is a learnable parameter. In summary, we obtain the final vector representation of the criminal behavior element chain as:

$$v_{chain} = [act_1, act_2, \ldots, act_m] \tag{10}$$

4.1.2 Fact Encoder

In this section, the fact encoder is used to obtain the semantic vector of the facts of a case. For a given case, *fact* = {w_1, w_2, …, w_n} is a word sequence of the facts of the case, where n is the number of words in the facts of the case. To capture more valuable features, here, we also use CNN and *BERT* to encode the facts of the case. The fact encoder is the embedding layer in the behavior chain encoder. On the basis of the above description, the encoding process of the fact encoder can be formally described as follows:

$$v_{fact} = Conn(Pooling(Conv(BERT(case)))) \quad (11)$$

Representation of the case in summary, by applying the behavior chain encoder and the fact encoder, we obtain the feature vector of the chain of criminal behavior elements v_{chain} and the feature vector of the facts of the case v_{fact} respectively. Then, we v_{chain} and v_{fact} to obtain the final feature vector of the case $v_{case} \in R^{2h}$. However, both v_{chain} and v_{fact} are high-order feature vectors obtained independently, and it is not feasible to model the interaction between them. Therefore, a multilayer perceptron (MLP) is used to provide global regulation between them. The MLP layer map v_{case} from *2h* to *h*. The process can be formalized as:

$$v_{case} = MLP(v_{fact} \otimes v_{chain}) \quad (12)$$

where $\otimes$ represents the concatenation operation.

4.2 Judge Representation Method

To solve the problem of the traditional case assignment methods without considering the expertise of judge, we integrate the feature of judges who are good at trial cases in the judges' representation to highlight the judges' expertise. Judges have heard numerous cases in the past and been involved in different causes. The differences in judges' expertise and experience have resulted in different cases with different trial quality. We assume that cases with high-quality judges are those that judges are good at. Using the feature of such cases can reflect judges' expertise.

In 2011, the Supreme People's Court of China published 31 indicators to evaluate the overall quality of court trials. Based on this, we consider how to evaluate the quality of individual judges' trials. After analysis, we select 3 indicators from 31 indicators to evaluate the quality of individual judges' trials: the rate of first-instance revised judgment and retrial(一审发改重审率), denoted by α, the average trial time of cases(案均审理时间), denoted by β, and the rate of cases closed within the statutory normal trial period(法定正常审限内结案率), denoted by θ. The weight calculation method for judges on the quality of any type of case can be formalized as:

$$w_{ij} = \frac{\theta_j + 1}{\alpha_j + \beta_j + 1} \quad (13)$$

where w_{ij} represents the weight of the judge's trial quality for any type of case, i = 1, …, n represents the number of judges, and j = 1, …, m represents the number of causes. One is added to the numerator and denominator to smooth the formula and prevent the result from being zero. For any judge, in any type of case, the trial quality weight w can be calculated according to Eq. (13). As seen by comparing different trial quality weights, the case type with the highest trial quality weight is the case type that judges are good at. In summary, we can determine the types of cases that any judge is good at. Then, we can generate the experimental labeled data set, which is in the form of <*case*, *judge*>.

We extract the feature of judge who is good at trial cases to represent the judge, so as to integrate judge's expertise into the judge's representation. As shown in Fig. 2, we use the judge encoder to encode the judge text to obtain the judge's feature vector. Because the judge's text also includes the facts of cases of the judge's good at cases, the combination of *BERT* and CNN is also used here to obtain more valuable feature vectors. As mentioned in the fact encoder, for a given the *i*-th judge's text $f_i = s_1, s_2, s_3, \ldots$, the process of the judge encoder can be formalized as:

$$fg_i = Conn(Pooling(Conv(BERT(f_i)))) \quad (14)$$

Here, $fg_i \in R^h$ represents the feature vector of the *i*-th judge. If there are *k* judges in the court, we can obtain the feature vector matrix $v_{judge} = [fg_1, fg_2, \ldots, fg_k]$ by means of the judge encoder.

4.3 Case Assignment Module

Given a training dataset $D \doteq [<v_{case}, v_{judge}>]$, we aim to maximize the accuracy of recommending judges based on the facts of a case. Based on v_{case} and $v_{judge} = [fg_1, fg_2, \ldots, fg_k]$, we can calculate the matching degree between the case and each judge through the cosine similarity [38]. This process can be formalized as:

$$\vartheta = soft\,max\left(\frac{v_{case} v_{judge}}{|v_{case}||v_{judge}|}\right) \quad (15)$$

Each value of $\vartheta \in R^k$ reflects the matching degree between the case and each judge. For any case, the judge with the largest matching value is the best judge to try the case.

For training, we use a cross-entropy loss function that is computed as follows:

$$\mathcal{L} = -\sum_{i=1}^{k} \vartheta_i \log\left(\overline{\vartheta_i}\right) \quad (16)$$

We employ Adam [39] for optimization, and apply dropout on every sematic vector to prevent overfitting.

5 EXPERIMENTS

5.1 Dataset and Settings

Dataset Currently, there is no publicly available datasets for intelligent case assignment. In this paper, we collect and construct an intelligent case assignment dataset.

The dataset consists of criminal cases published by the Chinese government from China Judgements Online(http://wenshu.court.gov.cn). The data generation is divided into four steps: Step 1, rule-based methods are used to extract the facts of a case and judges. By analyzing the data, it is found that the facts of cases of criminal cases are between "the trial has been completed" (现已审理终结) and "this court considers" (本院认为). Thus, the facts of a case can be extracted directly by means of rule matching. The judge of the case is identified by the presiding judge" (审判长), so the name of the judge comes after the presiding judge is extracted. Step 2, the data are cleaned. First, content irrelevant to the case is deleted; second, processing is normalized; and finally, the data with empty or garbled case facts are deleted. Step 3, the weight of trial quality is calculated. The source data are structured data stored in a dictionary, which contains the fields "whether to send back for retrial", "whether to revise the judgment at the second instance", and "trial duration". From these field values, we can use Eq. (13) to calculate the judge's trial quality weight in various cases. Step 4, data are generated. Different trial quality weights for the same judge can be compared to obtain the highest quality of judges' trial quality. Then, we can generate the experimental labelled data set, which is the form of <*case*, *judge*>. There are a total of 10979 cases and 11 judges in the labelled data set. To prevent sample imbalance, we split the training set, test set and validation set from each judge at a ratio of 8:1:1. The experimental data are shown in Tab. 1.

Settings We employ Adam as the optimizer and set dropout as 0.1 to prevent overfitting. The maximum number of vocabularies is 5000. For *BERT*, we set the maximum sentence length as 512. For CNN, we set the number of filters as 128, and the filter widths as {2, 3, 5, 7}. For the chain of criminal behavior elements, we set the maximum number of elements in a case as 64, and the maximum length of each behavior element as 16. We set the batch size as 32 for all models. We train every model for 50 epochs. The learning rate is 2e-5. The dimension of the hidden size to 128. We employ macro-precision (MP), macro-recall (MR), macro-F1 (MF), and P@1 as our evaluation metrics.

Table 1 Data set situation

Dataset	Number of data points	Data set size
train data	8778	8.10M
dev data	1098	0.95M
test data	1103	1.06M

5.2 Comparison with Traditional Case Assignment Methods

Comparing our method with the traditional case assignment methods. The case assignment methods commonly used in Chinese courts are lottery case assignment and balanced case assignment. Lottery case assignment means that the court numbers all judges without repeating. For any newly accepted case, court utilizes a computer program to select a random number, and the judge corresponding to the number is the one assigned the case. Balanced case assignment is based on lottery assignment to ensure that the number of cases handled by each judge is consistent. Our experimental results compared with those of the traditional case assignment methods are shown in Tab. 2.

Table 2 Comparison of our method with traditional case assignment methods / %

Model	P@1	MP	MR	MF
the lottery case assignment	10.16	-	-	-
the balanced case assignment	11.03	-	-	-
BCTA(ours)	**95.47**	92.38	90.22	**91.09**

During the experiment, 100 sets of experiments are performed on the lottery case assignment and the balanced case assignment, and the average value of the 100 sets of experimental results is taken as the final P@1 value. It can be seen that the case assignment method proposed achieves the best experimental result, which is 84% higher than those of both the lottery case assignment and the balanced case assignment. The reason is that the case assignment method proposed in this paper integrates the expertise of judges in the judges' representation, so as to ensure the compatibility between cases assigned and judges' professional ability. The lottery case assignment and the balanced case assignment essentially generate random numbers to obtain the assignment results. There are 11 judges in total, and the probability of randomly selecting each of the 11 numbers from 0 to 10 is one in eleven, which is approximately 9.09%.

5.3 Comparison with Mainstream Matching Models

In this paper, intelligent case assignment is regarded as a text-matching problem. Many classic models have been proposed for text matching, such ESIM [40], BIMPM [41], and ABCNN [19]. These classic text matching models input two sentences and output a label to identify the relationship between the two sentences. This experiment reproduces three matching models, ESIM, BIMPM, and ABCNN, to achieve case assignment. The experimental results are shown in Tab. 3.

Table 3 Experimental results compared with the results of other matching models / %

Model	P@1	MP	MR	MF
ESIM	91.02	86.34	76.45	77.02
BIMPM	88.66	74.51	74.77	74.18
ABCNN	90.29	86.45	78.69	81.61
BCTA(ours)	**95.47**	92.38	90.22	**91.09**

It can be seen from Tab. 3 that the model proposed in this paper is far better than the classic text matching model in case assignment. The main reason is that we model the structured semantic information of the case by constructing the chain of criminal behavior elements, whereas other methods only use neural networks to extract the text features of the case, resulting in loss of the temporal information of the behavior elements. Moreover, due to the long case text, the ESIM model and the BIMPM model with LSTM as the core perform poorly on long-sequence texts. Because when extracting long text features, LSTM's loop mechanism determines that it pays more attention to the end of the sequence. However, the ending content of the facts of a case rarely contains key information of the case. Thus, the ESIM and BIMPM models have the worst case assignment effects. The case assignment performance of the ABCNN model is better than those of the ESIM

model and the BIMPM model because it can capture more global and key information. The ABCNN model takes both word representation and phrase-level representation as model input, and performs attention calculations on the results after convolution, while the ESIM model and the BIMPM model only take word representations as model input, resulting in loss of semantic information.

5.4 Ablation Analysis

To further illustrate the significance of considering the chain of criminal behavior elements and to explore the criminal behavior elements in how to influence the performance, we conduct three sets of ablation experiments on our model. The first is to remove the criminal behavior element chain, such as $-v_{chain}$ in Tab. 4. The second is to remove a single element in the criminal behavior element chain. For example, $-PRE$ in Tab. 4 means that when constructing the chain of criminal behavior elements, the behavior word elements will be removed. Similarly, we can build a BCTA($-type$) model, where "*type*" stands for "AO" or "A1". The third set of experiments is to remove temporal features from the chain of criminal behavior elements, such as $-TE$ in Tab. 4. The experimental results are shown in Tab. 4.

Table 4 Ablation experiments / %

Model	P@1	MP	MR	MF
BCTA(ours)	**95.47**	92.38	90.22	**91.09**
$-v_{chain}$	92.74	85.33	86.14	85.40
−PRE	93.47	87.93	88.04	87.71
−A0	93.38	87.44	89.47	88.36
−A1	94.19	89.98	87.94	88.51
−TE	93.02	88.06	86.24	86.64

It can be seen in Tab. 4 that if the criminal behavior element chain is removed, the accuracy of the assignment of cases is greatly reduced, with a drop of nearly 3% on P@1 and a drop of 6% on MF. Additionally, if one element or a single feature ("A0","A1","PRE","TE") is removed when constructing the criminal behavior element chain, when the criminal behavior words or temporal features are removed, the accuracy of the assignment of cases is much worse than when other elements are removed, but it is inferior to removing the criminal behavior element chain completely. The experimental results prove that constructing the chain of criminal behavior elements with temporal relationships can enhance the ability to represent cases and improve the accuracy of the assignment of cases.

6 CONCLUSIONS

In this paper, we propose an intelligent case assignment method based on the chain of criminal behavior elements. This method builds the chain of criminal behavior elements to model the structured semantic information of the case, avoiding the loss of the semantics of the case. We build a BCTA model to realize intelligent case assignment, which can integrate information about a judge's expertise in the judge's representation. The experimental results have proved that the method proposed in this paper can significantly improve the accuracy of the assignment of cases. When recommending a judge, the accuracy can reach 95.47%, and the macro average F1 value can reach 91.09%. Our method can effectively avoid manual intervention, recommend the most compatible judge for the case, and shorten the case trial process.

Acknowledgements

This work is supported by the Joint Funds of the National Natural Science Foundation of China under Grant No. U1836205, the Major Research Program of National Natural Science Foundation of China under Grant No. 91746116, National Natural Science Foundation of China under Grant No. 62066007 and No. 62066008, the Major Special Science and Technology Projects of Guizhou Province under Grant No. (2017) 3002, the Key Projects of Science and Technology of Guizhou Province under Grant No. (2020) 1Z055 and Project of Guizhou Province Graduate Research Fund (Qianjiaohe YJSCXJH (2019)102).

7 REFERENCES

[1] Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613-620. https://doi.org/10.1145/361219.361220

[2] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *The Journal of machine Learning research*, 3, 993-1022.

[3] Deerwester. S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, *41*(6), 391-407. https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

[4] Baim, B., Weston, J., Grangier, D., Collobert, R., & Sadamasa, K. (2009). Supervised semantic indexin. *Proceedings of the 18th ACM conference on Information and Knowledge Management*, New York, NY, United States,187-196. https://doi.org/10.1145/1645953.1645979

[5] Hinton, G. E. (1986). Learning distributed representations of concepts. *Proceedings of the eighth Annual Conference of the Cognitive Science Society*, 1, 12.

[6] Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. T*he Journal of Machine Learning Research*, 3: 1137-1155.

[7] Wang, H., Zhou, C. D., Li, & L. X. (2019). Design and application of a text clustering algorithm based on parallelized K-means clustering. *Revue d'IntelligenceArtificielle*, 33(6), 453-460. https://doi.org/10.18280/ria.330608

[8] Bodapati J. D., Veeranjaneyulu N., & Shaik S. (2019). Sentiment analysis from movie reviews using LSTMs, *Ingenierie des Systemes d'Information*, *24*(1), 125-129. https://doi.org/10.18280/isi.240119

[9] Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 1532-1543. https://doi.org/10.3115/v1/D14-1162

[10] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 135-146. https://doi.org/10.1162/tacl_a_00051

[11] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized

word representations.*arXiv preprint* arXiv:1802.05365. https://doi.org/10.18653/v1/N18-1202

[12] Shabbeer, S. & Reddy, E. S. (2021). Prediction of sudden health crises owing to congestive heart failure with deep learning models. *Revue d'IntelligenceArtificielle*, *35*(1), 71-76. https://doi.org/10.18280/ria.350108

[13] Özyurt, F., Avcı, E., & Sert, E. (2020). UC-Merced image classification with CNN feature reduction using wavelet entropy optimized with genetic algorithm. *Traitement du Signal*, *37*(3), 347-353. https://doi.org/10.18280/ts.370301

[14] Al-Qaisi, A., AlTarawneh, M. S., ElSaid, A., & Alqadi, Z. (2020). A hybrid method of face feature extraction, classification based on MLBP and layered-recurrent network. *Traitement du Signal*, *37*(4), 555-561. https://doi.org/10.18280/ts.370402

[15] Jiang, X., Ye, H., Luo, Z., & Chao, W. H. (2018). Interpretable rationale augmented charge prediction system. *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*,Santa Fe, New Mexico, USA, 146-151.

[16] Kang, L., Liu, J., Liu, L., Shi, Q., & Ye, D. (2019). Creating auxiliary representations from charge definitions for criminal charge prediction. *arXiv preprint* arXiv:1911.05202.

[17] Liang, Y. & Chen, N. (2020). A novel tourist attraction recommendation system based on improved visual bayesian personalized ranking. *Ingénierie des Systèmes d'Information*, *25*(4), 497-503. https://doi.org/10.18280/isi.250413

[18] Guo, T., Lin, T., & Lu, Y. (2018). An interpretable LSTM neural network for autoregressive exogenous model. *arXiv preprint* arXiv:1804.05251.

[19] Yin, W., Schütze, H., Xiang, B., & Zhou, B. (2016). Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4, 259-272. https://doi.org/10.1162/tacl_a_00097

[20] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training.

[21] Le, Y., Wang, Z. J., Quan, Z., & Yao, B. (2018) ACV-tree: A new method for sentence similarity modeling. *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 4137-4143. https://doi.org/10.24963/ijcai.2018/575

[22] Luo, B., Feng, Y., Xu, J., Zhang, X., & Zhao, D. (2017). Learning to predict charges for criminal cases with legal basis. *arXiv preprint* arXiv:1707.09168. https://doi.org/10.18653/v1/D17-1289

[23] Lin, W. C., Kuo, T. T., Chang, T. J., Yen, C. A., Chen, C. J., & Lin, S. D. (2012). Exploiting machine learning models for Chinese legal documents labeling, case classification, and sentencing prediction. *Proceedings of the Twenty-Fourth Conference on Computational Linguistics and Speech Processing*, 140-141.

[24] Sahin, M. E., Guler, H., & Hamamci, S. E. (2020). Design and realization of a hyperchaotic memristive system for communication system on FPGA. *Traitement du Signal*, *37*(6), 939-953. https://doi.org/10.18280/ts.370607

[25] Liu, Y. H., Chen, Y. L., & Ho, W. L. (2015). Predicting associated statutes for legal problems. *Information Processing & Management*, *51*(1), 194-211. https://doi.org/10.1016/j.ipm.2014.07.003

[26] Xu, N., Wang, P., Chen, L., Pan, L., Wang, X., & Zhao, J. (2020). Distinguish confusing law articles for legal judgment prediction. *arXiv preprint* arXiv:2004.02557. https://doi.org/10.18653/v1/2020.acl-main.280

[27] Zhong, H., Guo, Z., Tu, C., Xiao, C. J., Liu, Z. Y., & Sun, M. (2018). Legal judgment prediction via topological learning. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 3540-3549. https://doi.org/10.1016/j.ipm.2014.07.003

[28] Cardellino, C., Teruel, M., Alemany, L., & Villata, S. (2017). Legal NERC with ontologies, Wikipedia and curriculum learning. *15th European Chapter of the Association for Computational Linguistics (EACL 2017)*. 2017, 254-25. https://doi.org/10.18653/v1/E17-2041

[29] Chalkidis, I., Androutsopoulos, I., & Michos, A. (2017). Extracting contract elements. *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*. 19-28.9. https://doi.org/10.1145/3086512.3086515

[30] Ye, H., Jiang, X., Luo, Z., Chao, W. (2018). Interpretable charge predictions for criminal cases: Learning to generate court views from fact descriptions. *arXiv preprint* arXiv:1802.08504. https://doi.org/10.18653/v1/N18-1168

[31] Wang, W. Y., Mayfield, E., Naidu, S., & Dittmar, J. (2012). Historical analysis of legal opinions with a sparse mixed-effects latent variable model. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics 1*, Jeju, Republic of Korea, 740-749.

[32] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks.*arXiv preprint* arXiv:1710.10903.

[33] Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation.*arXiv preprint* arXiv:1508.04025. https://doi.org/10.18653/v1/D15-1166

[34] Yang, Z., Yang, D., Dyer, C., He, X. D., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. *Proceedings of NAACL-HLT 2016*, San Diego, California, 1480-1489. https://doi.org/10.18653/v1/N16-1174

[35] Wang, Y., Huang, M., Zhu, X., & Zhao, L. (2016). Attention-based LSTM for aspect-level sentiment classification. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, 606-615. https://doi.org/10.18653/v1/D16-1058

[36] Er, M. J., Zhang, Y., Wang, N., & Mahardhika, P. (2016). Attention pooling-based convolutional neural network for sentence modelling. *Information Sciences*, 373, 388-403. https://doi.org/10.1016/j.ins.2016.08.084

[37] Zhang, D., Hong, M., Zou, L., Han, F., He, F. Z., Tu, Z. G., & Ren, Y. F. (2019). Attention pooling-based bidirectional gated recurrent units model for sentimental classification. *International Journal of Computational Intelligence Systems*, *12*(2), 723-732. https://doi.org/10.2991/ijcis.d.190710.001

[38] Pang, L., Lan, Y., Guo, J., Xu, J., Wan, S., & Cheng, X. (2016). Text matching as image recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, *30*(1).

[39] Da, K. (2014). A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980.

[40] Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., & Inkpen, D. (2016). Enhanced LSTM for natural language inference. *arXiv preprint* arXiv:1609.06038. https://doi.org/10.18653/v1/P17-1152

[41] Wang, Z., Hamza, W., & Florian, R. (2017). Bilateral multi-perspective matching for natural language sentences. *arXiv preprint* arXiv:1702.03814. https://doi.org/10.24963/ijcai.2017/579

Contact information:

Shaolin AO, MS. candidate
College of Computer Science and Technology, Guizhou University,
Guiyang 550025, China
E-mail: alyanyi@foxmail.com

Yongbin QIN, PhD, professor
(Corresponding author)
College of Computer Science and Technology, Guizhou University,
Guiyang 550025, China
E-mail: ybqin@foxmail.com

Yanping CHEN, PhD, associate professor
College of Computer Science and Technology, Guizhou University,
Guiyang 550025, China
E-mail: ypench@gmail.com

Ruizhang HUANG, PhD, associate professor
College of Computer Science and Technology, Guizhou University,
Guiyang 550025, China
E-mail: cse.rzhuang@gzu.edu.cn