Incorporating Structural Information into Legal Case Retrieval

YIXIAO MA, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China
YUEYUE WU, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China
QINGYAO AI, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China
YIQUN LIU*, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China
YUNQIU SHAO, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China
MIN ZHANG, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China
SHAOPING MA, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, China

Legal case retrieval has received increasing attention in recent years. However, compared to ad-hoc retrieval tasks, legal case retrieval has its unique challenges. First, case documents are rather lengthy and contain complex legal structures. Therefore, it is difficult for most existing dense retrieval models to encode an entire document and capture its inherent complex structure information. Most existing methods simply truncate part of the document content to meet the input length limit of PLMs, which will lead to information loss. Additionally, the definition of relevance in the legal domain differs from that in the general domain. Previous semantic-based or lexical-based methods fail to provide a comprehensive understanding of the relevance of legal cases. In this paper, we propose a **S**tructured **L**egal case **R**etrieval (SLR) framework, which incorporates internal and external structural information to address the above two challenges. Specifically, to avoid the truncation of long legal documents, the internal structural information, which is the organization pattern of legal documents, can be utilized to split a case document into segments. By dividing the document-level semantic matching task into segment-level subtasks, SLR can separately process segments using different methods based on the characteristic of each segment. In this way, the key elements of a case document can be highlighted without losing other content information. Secondly, towards a better understanding

*Corresponding author

Authors' addresses: Yixiao Ma, mayx20@mails.tsinghua.edu.cn, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084; Yueyue Wu, wuyueyue1600@gmail.com, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084; Qingyao Ai, aiqingyao@gmail.com, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084; Yiqun Liu, yiqunliu@tsinghua.edu.cn, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084; Yunqiu Shao, shaoyunqiu14@gmail.com, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084; Min Zhang, z-m@tsinghua.edu.cn, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084; Shaoping Ma, msp@tsinghua.edu.cn, Department of Computer Science and Technology, Institute for Internet Judiciary, Tsinghua University. Quan Cheng Laboratory, Beijing, China, 100084.

1046-8188/2023/7-ART
https://doi.org/10.1145/3609796

of relevance in the legal domain, we investigate the connections between criminal charges appearing in large-scale case corpus to generate a charge-wise relation graph. Then, the similarity between criminal charges can be pre-computed as the external structural information to enhance the recognition of relevant cases. Finally, a learning-to-rank algorithm integrates the features collected from internal and external structures to output the final retrieval results. Experimental results on public legal case retrieval benchmarks demonstrate the superior effectiveness of SLR over existing state-of-the-art baselines, including traditional bag-of-words and neural-based methods. Furthermore, we conduct a case study to visualize how the proposed model focuses on key elements and improves retrieval performance.

CCS Concepts: • **Information systems** → **Structured text search**; **Retrieval models and ranking**; • **Applied computing** → **Law**.

Additional Key Words and Phrases: legal case retrieval, structural information, relevance

1 INTRODUCTION

The legal case retrieval task aims to retrieve relevant cases from the legal case corpus given a query case. As a specific Information Retrieval (IR) task, legal case retrieval is essential to ensure justice in the legal domain. During a decision-making process, relevant prior cases can be crucial references or even directly involved in the final judgment [17]. With the continuous growth of case documents, automatically retrieving legal cases using IR techniques has received growing attention in recent years [3, 52].

Compared to traditional IR tasks, retrieving legal cases has encountered several challenges. The first challenge is the long length of legal case documents. Although pre-trained language models (PLMs) have achieved outstanding performance in various NLP and IR tasks, they are challenged in legal case retrieval given the length limitation. For example, a document in COLIEE2021 [19] (an international legal competition) contains 4,843 words on average, which is longer than the maximum encoding length of Longformer [2], let alone the mainstream PLMs [11, 27] taking up to 512 tokens as one input. To tackle this problem, Xiao et al. [58], Zhong et al. [65] truncate the exceeding part of the text, but this method results in the information loss of a document. Shao et al. [44] separate a legal document into paragraphs and retrieve relevant documents through paragraph-level interactions. This method reaches a better result than simple truncation, but its algorithm complexity is relatively high.

The second challenge is the difference in relevance between legal case retrieval and traditional ad-hoc retrieval. Specifically, two legal case documents may be unrelated even if their semantic or lexical similarity is high. Most existing models retrieve legal cases through lexical or semantic matching between two case documents. However, without domain knowledge, a model cannot understand the relevance in the legal domain comprehensively. For example, if there are two cases where the defendants are both convicted of causing traffic accidents, one accident is caused by drunk driving, while the other is an intentional injury to someone else. The writing of these case documents may be similar, but their entailed motives are completely different. Without domain knowledge, the similarity score between these two cases is often overrated by traditional retrieval models because of their high similarity in terms of term frequency or passage-level embedding. A few previous works try to utilize domain knowledge in their models. Ma et al. [30] identify significant paragraphs of a document with special placeholders. However, this method cannot be applied to other datasets because such placeholders are added manually by annotators to indicate the places for precedent references. Yue et al. [62] incorporate the hierarchical structure in criminal law of the civil law system [54] to differentiate confusing cases. However, this model merely calculates charge similarities with semantic-level word embeddings.

User study conducted by Shao et al. [45] reveals the importance of *objective* and *subjective* factors for the understanding of legal relevance, where *subjective* is the legal domain expertise, and *objective* is the case document property. As shown in this paper, we observe that those two factors can be directly captured by modeling the internal and external structural information of legal case documents. Specifically, the internal structural information (i.e., the inherent structure of the case document) entails the property of legal case documents,

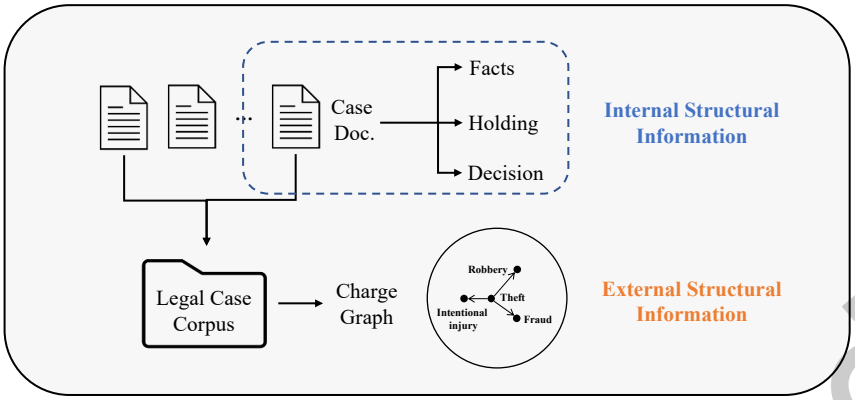

Fig. 1. Structural information in legal case retrieval.

Title: XXX's Case of Hazardous Driving
Court: People's Court of XXX Zone, Shandong Province
Case Number: Criminal Judgment (2018) Lu (XXXX) Criminal No. XXX

Parties' Information
Public Prosecutor: Shandong Province XXX People's Procuratorate
Defendant Background: Defendant XXX, male, was born in XXX , Shandong Province on September 10, 1986 …
Fact:
(Accusation of the Prosecutor) On December 18, 2017, the defendant XXX was drunk and drove an ordinary two-wheeled motorcycle with a license along Fushun Road in Weihai City near the entrance of New Century Home District when it collided with a small car… After identification, the defendant XXX was detected ethanol content of 112.32 mg / 100 ml in his venous blood…
(Evidence) …and provided the following: 1. The arrest issued by the Public Security Bureau of XXX; 2. Letter of responsibility identification for road traffic accidents; 3. Testimony of witness XXX …
(Facts Confirmed by the Court) After investigation, it was found that on December 18, 2017, the defendant XXX was drunk and drove …The evidence provided by the public prosecutor was verified, and the facts alleged by the public prosecutor were also confirmed. After the accident, defendant XXX confessed to the crime of driving a motor vehicle while intoxicate…
Holding:
The court believes that the defendant XXX violated the transportation management regulations and drove a motor vehicle on the road drunk, and his behavior constituted the crime of hazardous driving…
Decision:
The defendant XXX is guilty of hazardous driving, sentenced to one month and fifteen days of detention, suspended for three months, and a fine of six thousand RMB…
For written appeal, one original appeal form and four copies should be submitted.

Judge XXX
March 30, 2018
Clerk XXX

Fig. 2. Example of a case document.

while the legal domain expertise is represented by an innovative external structural information (i.e., charge relationship graph) proposed in our work. An illustration of the two kinds of structural information is shown in Figure 1.

To address these two challenges, we propose the **S**tructured **L**egal case **R**etrieval (SLR) model, which incorporates internal and external structural information into the retrieval process, to address the challenges of legal case retrieval. Specifically, to tackle the challenge of long document length, we adopt internal structural information

to match case documents. Internal structural information functionally divides a legal case document into multiple segments. Figure 2 is a simple example of a legal case document. Intuitively, we can extract key information from each segment separately according to the internal structural information of case documents. In most law systems, a case document is written in a certain structural manner and consists of multiple segments [43] to cover important aspects of the case. Among all segments, there are three segments that play an important role in representing key information of a document:

- *Facts* are objective fact statements confirmed by the court based on the evidence provided by the defendant and plaintiff. For example, where, when, and how the case happened. This segment only includes fact statements highly related to the defendant's crime.
- *Holding* is a court's determination of a matter of law based on the issue presented in the particular case. It is the main part of the judgment, Holding is a court's determination of a matter of law based on the issue presented in the particular case. It illustrates the reasons for the court's opinion.
- *Decision* contains the final judgment of the defendant including the criminal charge, sentence, and involved legal articles. This segment is regarded as the official result of a case.

These three segments reflect the complete process of how a case is analyzed and judged. In other words, they are logically connected. *Holding* are written based on the confirmed fact statements in *Facts*. Similarly, the *Decision* of a case is made by the analysis sentences in *Holding*. In general, *Facts* is most helpful to retrieve relevant cases because the legal relevance between two cases mostly depends on their key elements entailed in the objective fact statements. Therefore, if we want to identify key elements in *Facts*, we should know what is the decisive content in *Holding*, which can be firstly inferred by *Decision*. To this end, we split the document-level retrieval task into three segment-level subtasks to mine key information from these three segments of a case document. Then, the key elements in *Facts* can be extracted according to the logical correlation mentioned above. Specifically, the charges in *Decision* are used to identify the corresponding articles. Then, as the main reference to the judge's analysis, these articles are regarded as queries to retrieve key passages in *Holding*. Finally, taking the key *Holding* passages as the abstraction of *Facts*, we compute the attention weight of every word in *Facts* and generate an attention mask that highlights the key elements of a case. By dividing the traditional document matching task into subtasks, the content exceeding the length limit would not be simply truncated or ignored.

To tackle the challenge of legal-oriented relevance modeling, we introduce the external structural information between criminal charges as the complementary domain knowledge to SLR. We believe that the difference of relevance between legal case retrieval and traditional ad-hoc retrieval can be solved by introducing external structural information, namely the relation between charges modeled from a large-scale case corpus, because cases with the same or legally related cause of action may be more related. In particular, we adopt two kinds of structural information between charges to construct a graph of criminal charges: subtopic division of charges and resentenced charge frequency. Then, the charge similarity between queries and candidate cases is represented via graph embedding methods. Finally, a learning-to-rank framework [26] is employed as the last step of SLR to integrate internal and external structural information.

To summarize, our main contributions are:

- We propose a **S**tructured **L**egal case **R**etrieval (SLR) framework, which combines both internal structural information and external structural information for the legal case retrieval task.
- The internal structural information of case documents is employed to highlight key elements without losing important content information in the case document at the semantic level.
- Two kinds of external structural information in the legal domain are proposed to construct a charge graph modeling the relation between criminal charges in cases.
- Experimental results on different legal case retrieval datasets demonstrate the effectiveness of SLR compared to existing competitive models.

The rest of our paper is organized as follows: Section 2 reviews related works of this paper. Then, the design and implementation of our proposed model are introduced in Section 3. Section 4 describes the experimental results and analysis on different datasets. Finally, Section 5 concludes our work and discusses directions for future work.

2 RELATED WORK

2.1 Models for Legal Case Retrieval

The models for legal case retrieval can be categorized into two classes [3]: Expert Knowledge-based models and Natural Language Processing (NLP) models. For Expert Knowledge-based models, Zeng et al. [63] propose a knowledge representation model which extends traditional legal elements of issues by a new set of sub-elements for legal case representations. Saravanan et al. [42] design an ontological framework which enables inferences based on domain knowledge to enhance users' queries for retrieving relevant legal cases. However, recent effective models are mainly based on NLP and IR techniques. Galgani et al. [13] propose an approach utilizing both incoming and outgoing citation information to summarize keywords in legal documents for retrieval tasks. Tran et al. [49, 50] encode documents into a continuous vector space by phrase scoring algorithm. With the rapid development of deep learning, applying pre-trained language models (PLMs) to legal case retrieval has received huge success. Shao et al. [44], Westermann et al. [56] retrieve legal cases directly through a fine-tuned BERT [11]. In addition to the original PLMs pre-trained by multiple resources, Gururangan et al. [16] demonstrate that domain-adaptive pre-training improves the performance of PLMs in domain-specific tasks. For example, Zhong et al. [65] and Xiao et al. [58] use large Chinese legal corpus to pre-train BERT and Longformer, respectively. Both models outperform their precedent PLMs in legal tasks. Lyu et al. [29] extract and distinguish criminal elements in legal cases by a reinforcement learning framework called Criminal Element Extraction Network (CEEN). In addition to the legal domain, the idea of element extraction can transfer to other domains. For example, Wang et al. [55] present a bi-directional feedback clause-element relation network to tackle the cross-domain contract element extraction task. Sun et al. [48] propose Law-Match, a model-agnostic causal learning method that tackles the legal case matching task using the knowledge in law articles. The idea of utilizing law articles is also proposed by Ge et al. [14], which uses the random forest to parse articles into premise-conclusion pairs for further model training. Bhattacharya et al. [4] incorporate domain knowledge-based document similarity into PCNet [33] and propose a heterogeneous network called Hier-SPCNet which obtains competitive performance in the representation of document similarity. There are also works combining PLMs and traditional statistical methods to retrieve legal cases. Askari et al. [1] ensemble different ranking models including BM25 [40] and Longformer [2] to rerank a set of pooled candidates. Leveraging the input length and overall performance of models, in this paper, we adopt Lawformer [58] as the backend transformer of SLR.

2.2 Dense Retrieval for Long Documents

In some professional domains such as legal [3] or patent [60] search, documents usually exceed the input length of mainstream PLMs. To address this problem, two kinds of dense retrieval models adaptive to long text input are proposed. The first kind of model processes documents from left to right. For example, Dai et al. [10] enables capturing long-term dependency by a recurrence mechanism. Sukhbaatar et al. [46] proposed a model that dynamically selects an optimal span of the attention window. The second kind of model adopts a sparse attention pattern to avoid the computation of a large attention matrix regarding long texts. A representative work is proposed by Beltagy et al. [2], which combines global and sliding attention windows to comprehend contextual information in a long document. Child et al. [7] employ the dilated sliding window as attention blocks to deal with long texts. Li et al. [25] select key blocks in a long document by pre-ranking methods and aggregate these blocks to be a shorter document that can be processed by BERT. Although these works manage to adapt their

models to long documents, they are still unable to contain the full text of some legal case documents. Besides, the computational cost grows as the scale of input expands. Training long dense retrieval models with full documents is inefficient and challenging to capture key information. Therefore, in this paper, we leverage the internal structural information of legal case documents to identify key information. The model input will be only part of a case document with attention weights indicating key information.

2.3 Structural Information and Graph Methods

Lu et al. [28] utilize structural information in a long document to generate better text representations, but the "structural information" mentioned in their work is similar to metadata (e.g., abstract, keywords, authors, etc.), which is different from the concept of legal structural information in this paper. In addition, the "structural information" in Lu et al. [28], such as abstract and keywords, do not exist in all kinds of legal case documents. Therefore, this method can not be applied to legal search well.

The idea of utilizing internal and external structural information in legal documents has been attempted in previous works. Trompper and Winkels [51] split a court judgment into different parts by automatically assigning role labels to text elements in court judgments based on the document structure. With such labels, a parse tree is generated to represent the section hierarchy of legal documents. The split of legal documents facilitates the further application of internal structural information. For example, Li et al. [23] propose a structure-aware PLM which uses the structural information of legal documents to distinguish legal cases committed to different charges, but the structural information is not fully utilized as legal knowledge to guide the processing of different parts in legal documents. Li et al. [24] design a structure-aware PLM with heuristic pre-processing and post-processing methods to reduce the influence of irrelevant messages in legal documents for legal case retrieval, but such a structure-aware model design is task-oriented and can not be applied to legal documents with different structures. Ge et al. [14] adopt internal structure information and propose a multi-level matching network to help model better understand law articles in fine-grains for article recommendation. However, the structural information in this work is merely based on legal articles. By comparison, our proposed charge graph combines judicial decisions with legal articles, reflecting a more substantial connection between charges.

Graph-based methods can also capture structural relationships among data [64]. Even in domains without explicit structural features, reasoning on extracted structures is still a promising direction to study the latent information [66]. Du et al. [12] construct semantic relations between data labels by a graph and adopt Graph Convolutional Networks (GCN) [22]. As for the legal-specific tasks, Hartel et al. [18] conduct a case study to investigate the usage of criminological research (e.g., detecting specific crimes within court judgments) in the legal domain, which indicates the potential of utilizing charges as a kind of external structural information. Wang et al. [54] propose a hierarchical structure over charges and corresponding articles for charge classification. Yang et al. [61] propose LegalGNN, a legal case recommendation model based on a graph neural network. The whole framework unifies the content in legal documents and structural features generated by a knowledge graph. Chen et al. [6] predict judgment with a unified legal graph that fuses charge information in the dataset. These works only construct a graph qualitatively, while our method evaluates the importance of edges quantitatively through the analysis of a large-scale criminal case corpus.

3 METHOD

3.1 Task Definition

For each query, our task aims to retrieve relevant cases from a candidate case corpus. To be specific, given a query case q and a candidate list $C = \{c_1, c_2, ..., c_M\}(M \in \mathbb{N}^+)$, the task is to retrieve the most relevant candidate cases from C. For the legal case retrieval datasets used in this paper, each candidate case has a label indicating its

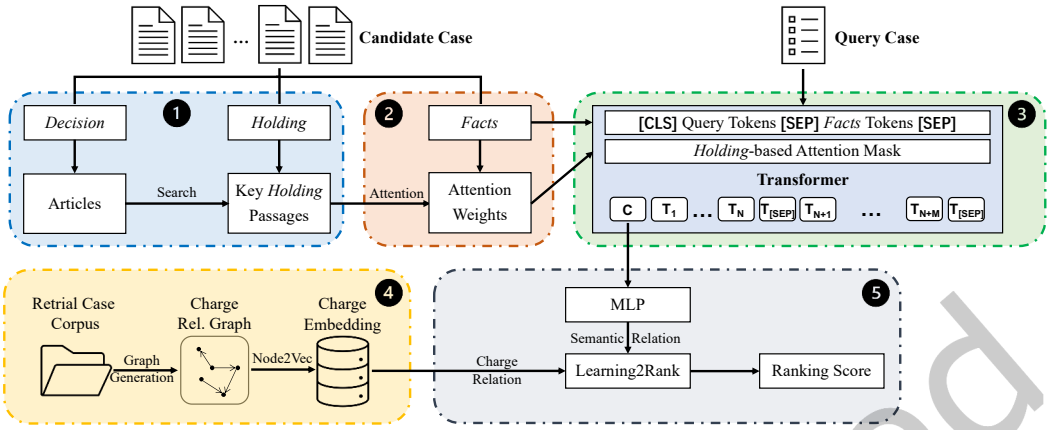

Fig. 3. The overall framework of SLR. Numbers denote the five steps: 1) Retrieval of key passages in *Holding*. 2) Attention mechanism based on key passages in *Holding*. 3) Semantic Matching of query-candidate pairs by the pre-trained model. 4) Charge relevance representation by external structural knowledge. 5) Learning to rank method to generate results.

relevance to the query. Therefore, the most desirable output for the task is a sorted list of all candidate cases $C_{sorted} = \{c_1^{'}, c_2^{'}, ..., c_M^{'}\}$ ranked by the relevance label.

In most practical legal case retrieval needs, q only consists of the case fact description and charges, while candidates are complete case documents. In detail, a criminal case document consists of the following parts: title, court, case name, case number, public prosecutor, defendant background, fact, holding, decision, and related articles of the case. In this paper, we mainly focus on the retrieval of cases under criminal law. All experiments are also conducted on criminal case documents.

3.2 Framework Overview

The overall framework of SLR is shown in Figure 3. For each query, SLR takes a candidate case document as the input each time and then predicts the relevance score between the query-candidate pair. Specifically, there are five steps in SLR. Step 1 to Step 3 aims to capture key elements from long case documents without losing too much textual information, while Step 4 considers the external structural information of charges to understand the relevance in the legal domain better. Finally, Step 5 integrates the features of these two kinds of structural information. In the following five subsections, we elaborate on the design and implementations of the five steps, respectively.

3.3 Step 1: Retrieval of key passages in Holding

A case document in the legal domain has three core segments: *Facts*, *Holding*, and *Decision*. Specifically, *Facts* is the description of a case such as where, when, and how the case happened. *Holding* is the opinion on case issues from judges. *Decision* states the final judgment including sentenced charges and related articles of the case. Of all segments, *Facts* contains most case details beneficial to relevance judgment. *Holding* and *Decision* work as the basis for the relevance in *Facts*. As the input length of Transformer is limited, SLR converts *Holding* and *Decision* to attention weights to highlight key elements in *Facts*.

To this end, the first step of SLR is to retrieve passages representing key circumstances of the case from *Holding*. In this paper, we extract a list of sentenced charges L_{charge} from *Decision* by regular expression matching. Since all charges are written in their official names, a high recall rate can be guaranteed in the step of charge extraction.

Then, we gather the criminal articles of corresponding charges in L_{charge}. All articles are from the official website [1]. The collected article set is denoted as S_a.

Given S_a, we can extract key passages from *Holding* as the basis to highlight key elements in *Facts*. Considering the trade-off between computational cost and retrieval performance, we employ a bag-of-words language model [36]. After splitting passages into words by Jieba [47] and removing stopwords, the set of key passages S_{psg} is denoted as:

$$S_{psg} = \{\text{argmax}\left(P(a_i \mid psg)\right) \mid a_i \in S_a\} \tag{1}$$

$$P(a_i \mid psg) \propto \prod_{t \in a_i} P(t \mid psg) \tag{2}$$

$$P(t \mid psg) = \lambda F\left(t \mid M_{psg}\right) + (1-\lambda)F\left(t \mid M_c\right) \tag{3}$$

where t is a token in an article, *psg* is a passage in *Holding*, $P(a_i \mid psg)$ represents the probability of generating article a_i given *psg*. and $P(t \mid psg)$ represents the probability of generating token t given *psg*. $P(t \mid psg)$ is a linear combination of two aspects. $F\left(t \mid M_{psg}\right)$ is the token frequency in the passage, while $F\left(t \mid M_c\right)$ is the token frequency in *Holding*. λ is a smoothing hyper-parameter ranging from 0 to 1. Notably, neural methods are not adopted in the first step, because the target of the first step is to search for the most relevant passages instead of accurate relevance scores or rankings of all passages. A traditional bag-of-word method is more efficient.

3.4 Step 2: Key Holding-based Attention

Key passages in *Holding* directly indicate the reasons for judgment. Therefore, the importance of sentences in *Facts* can be distinguished by their consistency with S_{psg}. In Step 2, we compute the attention weights of each token in *Facts* using the extracted key passages S_{psg} in Step 1. As discussed in Section 1, current mainstream PLMs can only take part of the document as input due to the length limitation. This is not desirable because truncating text essentially increases the probability of losing important information from the input data. Besides, the attention mask in previous transformers [53] only takes binary inputs indicating whether the model should ignore the token or not. In other words, for all input tokens valued 1 in the attention mask, the transformer regards them equally. This is incompatible with the actual legal situation because there are key elements clearly distinguished from other words in *Facts*. In this paper, the truncated part of the document is utilized as a supervisory signal in the attention mask. To this end, we rewrite the attention function to take float numbers inputs representing the weight of different tokens. According to the "SelfAttention Function" [2] in the transformer, the attention weights $probs_{att}$ is defined as:

$$probs_{att} = \text{softmax}\left(\frac{scores}{\sqrt{N}} + K(mask_{att} - 1)\right) \tag{4}$$

where *scores* are the self-attention scores computed in the transformers, N is the length of input tokens, $mask_{att}$ is the attention mask at the input stage, and $K = 10000$ is a large integer to magnify the influence of zero value in $mask_{att}$. $probs_{att}$, *scores*, and $mask_{att}$ are all vectors of shape (B, N), where B is the batch size of input. In order to: a) approximate the attention weights to the values in $probs_{att}$ linearly, and b) ensure the values of $mask_{att}$ and *scores* are of the same magnitude, we rewrite the definition of attention mask as:

$$mask_{att} = 1 + \frac{\ln(N \times \text{softmax}\left(\frac{v_t \cdot v_{psg}}{\sqrt{N}}\right))}{K} \tag{5}$$

[1] http://www.npc.gov.cn/wxzl/wxzl/2000-12/17/content_4680.html

[2] https://github.com/huggingface/transformers/blob/b1198a8440cc05f569b0bc22038993a1e5e707ab/src/transformers/models/roberta/modeling_roberta.py#L263

where v_t are token-level features of *Facts*, v_{psg} are passage-level features of key *Holding* passages, and $\cdot$ is the matrix multiplication. Following Equation 4, K is set to 10000. In this paper, v_{psg} and v_t are encoded by a shorter transformer, BERT-Crime [65]. Specifically, *Facts* and key *Holding* passages are first tokenized by the default tokenizer of Chinses-RoBERTa [27] separately. Then, a [*CLS*] token is prepended to the input tokens and multiple [*PAD*] tokens extend the input length to 512. v_{psg} is the final hidden state feature of the [*CLS*] token output given key *Holding* passages as input. v_t are stacked outputs of all tokens given *Facts* as input. The shape of v_t is (B, N, H) and the shape of v_{psg} is $(B, H, 1)$. $H = 768$ is the dimension of the transformer feature output.

3.5 Step 3: Semantic Matching

After generating the attention mask $mask_{att}$, a transformer is employed with $mask_{att}$ to present the semantic-level similarity between queries and candidates. In this paper, we adopt three different transformers as the backbone of our model: BERT-Crime [65], Chinese-Roberta [27], and Lawformer [58]. They are all proved effective in either legal or general domains in previous works. The selected backbone transformer is fine-tuned by the Next Sentence Prediction (NSP) task. Each query and candidate are concatenated to construct a query-candidate pair as the input. However, even if the Lawformer takes up to 4,096 tokens as input, there are still candidates whose *Facts* exceeds the length limit. Consequently, we filter out duplicate and less informative content from *Facts*. Specifically, general *Facts* consists of three parts organized in order: *accusation of the prosecutor*, *facts confirmed by the court*, and *evidence*. Among these parts, *evidence* is personal and not related to the key elements in *Facts* when solving charge-related legal tasks (e.g., legal case retrieval and charge prediction). Therefore, *evidence* is removed from the input during preprocessing. Besides, although the *accusation of the prosecutor* and the *facts confirmed by the court* both claim case facts with minor differences, final decisions are made according to the latter one. Therefore, *accusation of the prosecutor* is also removed unless it contains complementary information. Next, the query and filtered *Facts* are tokenized by the default tokenizer of Chinses-RoBERTa [9] and separated by a [*SEP*] token. A [*CLS*] token is added to the beginning of query tokens. Finally, the input tokens together with the attention mask $mask_{att}$ are fed into the Lawformer. As the output of the Lawformer, the final hidden state vector of the [*CLS*] token v_{cls} is passed into a fully-connected layer to make predictions:

$$\hat{y} = W_{fc} \cdot v_{cls} + b_{fc} \tag{6}$$

where $W_{fc} \in \mathbb{R}^{|C| \times H}$, $b_{fc} \in \mathbb{R}^{|C|}$, and C denotes the number of classes of the input data. In practice, $|C|$ is set to 2, which means every query-candidate pair is labeled by the positiveness or negativeness of the candidate's relevance to the query. The cross-entropy loss function is employed in SLR. For each query-candidate pair, given the label y and predicted $\hat{y}$:

$$\mathcal{L}(y, \hat{y}) = -\sum_{i=1}^{|C|} y_i \log(\hat{y}_i) \tag{7}$$

3.6 Step 4: External Structural Information-based Matching

In addition to case description, the charge information in *Decision* is often ignored by previous works, as discussed in Section 1. Given a query-candidate pair, the similarity between query charges and candidate charges also reflects the relevance of the candidate for a given query. Therefore, Step 4 represents the relevance of different criminal charges using external structural information in the legal domain. Intuitively, the external structural information can be modeled as a graph (denoted as G). Each node represents a criminal charge, and each edge represents a relationship between the charges of two ends. Furthermore, each edge has a weight indicating the extent of this relationship. In this paper, we combine two kinds of external structural information in the legal domain to construct G: 1) Subtopic division of charges and 2) Resentenced charge frequency. As shown in Figure 4, the edges in blue and orange are generated by the two kinds of external structural information, respectively.

3.6.1 Subtopic division of charges. Charges are categorized into ten subtopics in Chinese criminal law. Figure 4 is an example of how different charges are organized by subtopics. Charges in the same subtopic are concerned about the same legal interests. Therefore, the first kind of external structural information is from the division of criminal charges. In this paper, we adopt the charge frequency in case documents to model the relation inside the subtopic. Edges in blue indicate relevance between charges in the same subtopic. Given two charges u, v in the same subtopic, the weight w_{uv} of edge e_{uv} is defined as:

$$w_{uv} = w_{vu} = \sum_{c \in C; u, v \in \mathrm{J}_v(c)} \min\left(0.1, \frac{1}{|\mathrm{J}_v(c)| - 1}\right) \tag{8}$$

where C is the whole case document corpus, c is a candidate case and $\mathrm{J}_v(c)$ denotes the set of charges that are in the same subtopic as v in the *Decision* of case c. In other words, w_{uv} depends on the frequency of candidate cases containing charge u and charge v. As a certain charge, v may have connections between the other charges in the same subtopic, the weight (1 per case) is equally divided among all charges $\in \mathrm{J}_v(c)$. A min function is employed to ensure the upper bound of w_{uv} in case of extremely frequent charges. For instance, if there are ten cases containing both u and v without other charges $\in \mathrm{J}_v(c)$, then $w_{uv} = 10 * \min(0.1, \frac{1}{|2\text{-}1|}) = 1$.

3.6.2 Resentenced charge frequency. When considering the possible charges of a case from the perspective of legal justice, the final judgment may refer but not be limited to charges in one subtopic. Charges in different subtopics also have connections. In this paper, we utilize the information in resentenced cases (cases that have changed their previous judgment) as the second external structural information to represent relevance between charges without subtopic restriction. In such cases, charges in the first trial are confused with charges in the retrial. Therefore, these two kinds of charges have relevance to a certain extent. For example, the edge in orange in Figure 4 illustrates that the charge *Theft* has a possibility of being revised to *Intentional Injury* in resentenced documents. Suppose C_r are the set of resentenced cases which has charge u in the previous judgment J_{old} and v in the revised judgment J_{new}. Given an edge e_{uv} in orange, its weight w'_{uv} is defined as:

$$w'_{uv} = \sum_{c \in C_r} W_c(u, v) \tag{9}$$

$$W_c(u, v) = \begin{cases} \frac{1}{|\mathrm{J}_{\text{old}}(c)|} & \mathrm{J}_{\text{old}}(c) \subset \mathrm{J}_{\text{new}}(c) \\ \frac{1}{|\mathrm{J}_{\text{old}}(c) - \mathrm{J}_{\text{new}}(c)|} & u, v \notin \mathrm{J}_{\text{old}}(c) \cap \mathrm{J}_{\text{new}}(c) \\ 0 & Others \end{cases} \tag{10}$$

where W_c is the weight function of candidate case c.

After initiating the nodes, edges, and weights of all edges in G, Node2Vec [15], an algorithmic framework for representational graphs learning, is adopted to encode all nodes (charges) in G into vectors. The encoding algorithm includes multiple iterations. For each iteration, the algorithm randomly selects starting nodes and conducts a random walk on nodes through their edges. When reaching a new node, the choice of the next walk depends on the probability of all possible edges. In this paper, the weights w_{uv} and w'_{uv} are normalized to be the probability of edges:

$$p_{uv} = \frac{w_{uv} + w'_{uv}}{\sum_{\tilde{v} \in G}(w_{u\tilde{v}} + w'_{u\tilde{v}})} \tag{11}$$

where p_{uv} is the probability for the random walk to pass through edge e_{uv} and $\tilde{v}$ is the end of an edge starting from u. All charges appearing in one walk are regarded as one "passage". All such passages are aggregated together as a corpus, which is then fed into a Word2Vec [32] model. Finally, all criminal charges are represented by 32-dimensional word embeddings. For each document, all charge embeddings in *Decision* are averaged to be

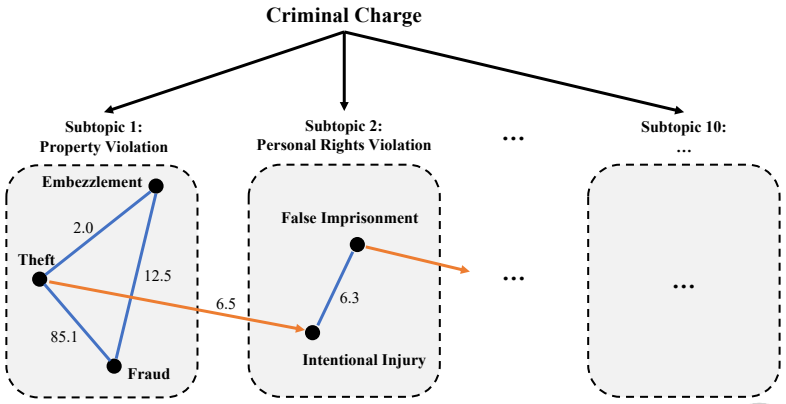

Fig. 4. An illustration of the graph generated by external structural information in the domain. Charges in a rounded rectangle belong to the same charge subtopic. Edges in blue and orange are generated by subtopic division and resentenced charge frequency, respectively.

the final charge embedding of the document. By computing the cosine similarity between charge embeddings of the query and candidate case, we have the final charge relevance score r.

3.7 Step 5: Learning to Rank

From Step 1 to Step 4, our model presents the semantic-level and charge-level relevance using internal and external structural information separately. In the final step, we aggregate the class predictions $\hat{y}$ and the relevance score r using Lightgbm [20], a light yet effective learning-to-rank model. The output $\hat{y}$ in Step 3 is a vector of length $|C|$. Each dimension is the probability of the candidate being classified as a specific category. r in Step 4 is the cosine similarity between the charge embeddings of the query and the candidate case. Consequently, a total of $(|C| + 1)$-dimensional feature is collected and passed into the learning-to-rank model to train and output predictions:

$$y_{out} = \text{lgb}\left(\text{Concat}\left(\hat{y}, r\right)\right) \tag{12}$$

Where lgb is the Lightgbm [20] model and y_{out} is the final ranking score of SLR. In practice, the training objective is set to NDCG-based ranking, and the evaluation metric is set to NDCG@30. Candidates in the candidate list are sorted by y_{out} as the final ranking list. Further evaluations are all based on this ranking list.

4 EXPERIMENTS

4.1 Datasets

In this paper, we conduct our experiments on two datasets: LeCaRD and CAIL2021-LCR. The **Le**gal **Ca**se **R**etrieval **D**ataset (LeCaRD) [31] is the first criminal case retrieval dataset under the Chinese law system. **C**hallenge of **AI** in **L**aw (CAIL) [59] is a competition held annually under the guidance of the Supreme People's Court and the Chinese Information Processing Society of China to promote AI technology and a higher level of digital justice since 2018. As one of the seven tasks in CAIL2021 [3], the legal case retrieval task provides a dataset named CAIL2021-LCR as the test set.

[3]http://cail.cipsc.org.cn/index.html

Table 1. The description of relevance label.

Label	Description
3	Both key elements and key circumstances are relevant.
2	Key elements are relevant but key circumstances are irrelevant.
1	Key elements are irrelevant but key circumstances are relevant.
0	Both key elements and key circumstances are irrelevant.

Table 2. Statistics of LeCaRD and CAIL2021-LCR.

Statistic	LeCaRD	CAIL2021-LCR
# Documents	43823	50000
# Queries	107	130
# Candidate cases per query	100	100
Avg. length per case document	8275	2707
# Avg. relevant cases per query	10.33	11.53
# Charges in query cases	20	20

Statistics of LeCaRD and CAIL2021-LCR are shown in Table 2. Case documents in both datasets are criminal decisions sampled from the China judgments Online website [4]. Every candidate case document has a four-level relevance label annotated by criminal law experts. The detailed label description is shown in Table 1. In practice, the test set of LeCaRD consists of 20 queries of different charges (one query per charge), the validation set contains 10 queries, and the remaining 77 queries are the training set. Since the size of CAIL2021-LCR is larger than LeCaRD, the test set of CAIL2021-LCR consists of 40 queries (two queries per charge), the validation set contains 10 queries, and the remaining 80 queries are the training set.

4.2 Baselines

To evaluate the performance of SLR on the legal case retrieval task, we adopt state-of-the-art baselines in two categories: traditional bag-of-words IR models and neural-based models.

- **Bag-of-words IR methods** regard the whole dataset as a term-level corpus and predict the relevance of queries and candidates by computing the term frequency, document frequency, and other statistical factors. In this paper, we select three representative traditional IR models as baselines: TF-IDF + SVM [34, 41], BM25 [40], and Language Models for Information Retrieval (LMIR) [36].
- **Chinese-RoBERTa** [9] is the Chinese version of RoBERTa [27]. RoBERTa is a replication of BERT [11] which extensively studies the choices of hyper-parameters and training data size. Compared to BERT, RoBERTa can match or exceed the performance of many previous models on various tasks.
- **BERT-Crime** [65] is a new release of BERT [11] pre-trained by Chinese criminal data. Compared to the original BERT, the BERT-Crime mainly focus on legal domain-specific tasks.
- **Lawformer** [58] is the first law-oriented and longformer-based [2] language model which can encode up to 4096 tokens as an input. Lawformer combines task-motivated global attention and local sliding window attention to capture long-distance information. It is proficient in dealing with long legal texts.

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

- **BERT-PLI** [44] represent case relevance by aggregating paragraph-level semantic similarity between two case documents. The model is based on a BERT fine-tuned by case law entailment dataset in COLIEE2019 [37] to adapt it to the legal domain.
- **CAIL2021 leaderboard** [5] shows top results in the CAIL2021 legal case retrieval task. This year, 251 teams from universities and companies signed up for the task and submitted the results. The results on the leaderboard can be regarded as the latest baselines.

4.3 Evaluation Metrics

We evaluate the performances of models using both ranking and precision metrics. For retrieval metrics, we adopt NDCG@(10,20,30) representing the NDCG score of top 10, top 20, and top 30 candidates in the retrieved list. For precision metrics, we adopt Mean Average Precision (MAP) and P@(5, 10) representing the precision of top 5 and top 10 candidates in the retrieved list. Notably, we merge the four-level label in legal case retrieval datasets into binary when measuring precision metrics. Only cases with the highest relevance label are regarded as relevant cases and the rest are regarded as irrelevant. Because according to the labeling criteria in Ma et al. [31], cases labeled 3 meet diverse search needs and can be an ideal reference in real judgment.

4.4 Experiment Setup

In this subsection, we describe our experimental setup. SLR and all baselines are written in Python3. The traditional bag-of-words models such as BM25 and TF-IDF are implemented by Gensim [39] and parameters are set to best values on the training set. Given a query, each traditional bag-of-words model regards all candidate cases of the query as a corpus. In other words, each query and its corresponding candidates constitute a corpus independent from other queries and candidates. Specifically, we adopt Jieba [47] to split sentences in candidate cases into words and remove stopwords based on an open-source stopword list [6]. The remaining words construct a word-level corpus for the initialization of traditional bag-of-words models. Then, the query is preprocessed similarly to the sentences in candidates. Finally, traditional bag-of-words models take such query words as input and retrieve the candidates in the corpus by the similarity score.

The neural-based models are implemented by PyTorch [35] Framework and HuggingFace Transformer Library [57]. We set the learning rate to $1 * 10^{-5}$, weight decay to zero, loss function to cross-entropy, and optimizing strategy to Adam [21]. For original transformer-based models such as Chinese-RoBERTa and BERT-Crime, the batch size is set to 24, and the maximum sequence length is set to 512. For long document transformer-based models such as Lawformer and our model, the batch size is set to 4, the maximum sequence length is set to 4096, and the attention window size is set to 512. Similar to the structure described in Section 3.5, all neural-based models adopt a cross-encoder manner to retrieve legal cases. The retrieval task is regarded as a binary classification task, which ranks the retrieved legal cases by sorting $\Delta = pred_1 - pred_0$, where $pred_1$ is the probability of the candidate to be predicted as label=3 and $pred_0$ is the probability to be predicted as label=0, 1, or 2. The input of neural-based models is the concatenation of the full query text and *Facts* of the candidate text. All text exceeding the maximum input length (i.e., 512 for BERT-Crime, BERT-PLI, Chinese-RoBERTa, and 4.096 for Lawformer) is truncated. The embedding obtained in the [*CLS*] token in the last hidden layer will be fed into an MLP classifier to make final predictions.

In terms of Node2Vec [15] framework, the number of iterations is set to 100, random walk length to 80, output dimension to 32, hyper-parameter p to 1, and q to 0.5. All models are trained and evaluated on four 32G NVIDIA V100 GPUs.

[5] http://cail.cipsc.org.cn/datagrid.html?raceID=1
[6] https://github.com/goto456/stopwords

4.5 Experimental Results

4.5.1 Comparison against baselines. In this subsection, we mainly demonstrate the experimental results of SLR and baselines on the legal case retrieval task. The overall performance on LeCaRD [31] and CAIL2021-LCR datasets are shown in Table 3a and Table 3b, respectively. Because both LeCaRD and CAIL2021-LCR have an official split of the training set and test set, in this paper, all PLM-based models are fine-tuned on the official training set and then evaluated on the test set. Through the overall experimental results, we have the following observations:

1) On both datasets, SLR achieves the best results in terms of all evaluation metrics except P@10. In addition, three versions of SLR with different PLM backbones consistently outperform traditional bag-of-words methods and neural-based methods. On both datasets, the best results of SLR have statistical significance over baselines on most NDCG metrics and part of precision metrics. This phenomenon illustrates the effectiveness and flexibility of our proposed SLR in legal case retrieval. The application of structural knowledge is significantly beneficial to the performance of retrieval tasks in the legal domain. In addition, since the backbone transformer works as an isolated block in SLR and can be replaced by any other semantic matching method effortlessly, SLR is easy to be integrated with other cross encoders to enhance the matching performance.
2) Although three versions of SLR have top-3 overall performances, they are not completely consistent on the two datasets. Specifically, SLR (Chinese-RoBERTa) has the best performance on LeCaRD but is surpassed by SLR (Lawformer) on CAIL2021-LCR. One possible explanation is the heterogeneity of data sources. For example, LeCaRD contains legal cases from the recent 20 years, while CAIL2021-LCR only includes legal cases since 2018. Besides, as a benchmark for legal competition, CAIL2021-LCR removes extremely long or short cases, while LeCaRD does not take document length into consideration. It can be inferred from Table 2 that the average length of LeCaRD is around three times longer than CAIL2021-LCR. Therefore, the heterogeneity of data sources results in the performance difference on different datasets.
3) As a retrieval task, SLR improves $NDCG@(10, 20, 30)$ by (8.11%, 6.91%, 2.94%) on LeCaRD and (5.28%, 4.47%, 2.73%) on CAIL2021-LCR compared to the existing best baseline, which proves that SLR has a better ranking ability than existing methods. Notably, the results of Lawformer on LeCaRD are different from the results reported in Xiao et al. [58] because they randomly split LeCaRD into a training and test set and adopted 5-fold cross-validation. By comparison, we adopt the official training set and test set published by the author of LeCaRD for a fair comparison.
4) The results of traditional bag-of-words baselines are not competitive as neural-based methods, because they only take word-level features, such as term frequency, into consideration. In addition to word-level features, understanding contextual information and legal elements is important for retrieval tasks in the legal domain. Even so, among all traditional bag-of-words baselines, LMIR has the best overall performance on both datasets. This result is consistent with the experimental results in other works [30, 31] because LMIR is a competitive bag-of-words baseline in the legal domain.
5) As a PLM that already exceeds the performance of many previous models in the general domain, Chinese-RoBERTa has the best overall performance among all neural-based baselines in legal case retrieval. Although Chinese-RoBERTa is pre-trained on data in the general domain, it still outperforms PLMs pre-trained specifically on legal data. This result demonstrates that there is potential for models with transfer learning ability to be applied in the legal domain even with the knowledge gap of domains. Also, the overall performance of BERT-PLI on both datasets is not good as expected, because the BERT-PLI model proposed in Shao et al. [44] utilizes the dataset from entailment task in COLIEE2020 [38] as an external legal data source to fine-tune the model before the final prediction. For a fair comparison, no external data is imported for any further training in this paper. Therefore, the performance of BERT-PLI declines compared to its previous results reported in Shao et al. [44].

6) Although BERT-Crime and Chinese-RoBERTa can only process 512-length input texts, their performances both exceed Lawformer which can process input texts up to 4,096 words. Although the longer length of input text increases the information of legal documents fed into the model, neither the pre-training tasks nor the structure of Lawformer is designed to capture key legal elements in such a long input. Consequently, the performance of Lawformer is surpassed by other PLM baselines with shorter input lengths. On the other hand, Lawformer enhanced by structural information (i.e., SLR(Lawformer)) has the most improvement compared to other versions of SLR and even has the best overall performances on CAIL2021-LCR. This result illustrates the effectiveness of structural information in capturing key legal elements. In addition, the application of internal structural information has more potential on PLMs with long text inputs, because internal structural information allows the model to contain more information without losing focus on key information.

Table 3. Evaluation results on (a) LeCaRD and (b) CAIL2021-LCR. SLR(BERT-Crime), SLR (Chinese-RoBERTa), and SLR (Lawformer) are all our presented methods, but with different backbone transformers adopted in Step 3. †\‡\§ denote statistical significance compared to the best version of SLR at a level of p-value < 0.05\0.01\0.005 using Wilcoxon test.

Model	NDCG@10	NDCG@20	NDCG@30	P@5	P@10	MAP
TF-IDF + SVM	0.6824§	0.7490§	0.8682§	0.3400†	0.3550†	0.4569†
BM25	0.7259§	0.7858§	0.8794§	0.3900	0.3900†	0.4753‡
LMIR	0.7316§	0.7911§	0.8847§	0.4500	0.3950†	0.5395
BERT-Crime	0.7719‡	0.8174‡	0.8805†	0.4300	0.3850‡	0.5566
BERT-PLI	0.7138§	0.7874§	0.8809§	0.3500†	0.3400§	0.4547‡
Chinese-RoBERTa	0.8257	0.8669‡	0.9276†	0.4600	0.4350	0.5505
Lawformer	0.7675§	0.8191§	0.9090§	0.4600	0.4000†	0.4890
SLR (BERT-Crime)	0.8931	0.9200	0.9513	0.5100	**0.4800**	0.6090
SLR (Chinese-RoBERTa)	**0.8927**	**0.9268**	**0.9549**	**0.4900**	0.4700	**0.6165**
SLR (Lawformer)	0.8809	0.9147	0.9496	0.4800	0.4600	0.5481

(a) LeCaRD

Model	NDCG@10	NDCG@20	NDCG@30	P@5	P@10	MAP
TF-IDF + SVM	0.6819§	0.7504§	0.8485§	0.4000§	0.4200§	0.4735§
BM25	0.7806§	0.8247§	0.9004§	0.4900§	0.4600§	0.5440§
LMIR	0.8189§	0.8372§	0.9094§	0.5050‡	0.4975§	0.5302§
BERT-Crime	0.8651	0.8789†	0.9270	0.5650	0.5275	0.5927†
BERT-PLI	0.7827§	0.8194§	0.8967§	0.5050†	0.4875§	0.5334§
Chinese-RoBERTa	0.8612	0.8951	0.9362	0.5750	0.5275†	0.6020
Lawformer	0.8264§	0.8721§	0.9224§	0.5450	0.5125†	0.5852†
SLR (BERT-Crime)	0.9013	0.9262	0.9585	0.5950	0.5625	0.6686
SLR (Chinese-RoBERTa)	0.9059	0.9240	0.9557	0.5950	0.5675	0.6646
SLR (Lawformer)	**0.9108**	**0.9351**	**0.9618**	**0.6250**	**0.5700**	**0.6815**

(b) CAIL2021-LCR

4.5.2 Complexity and Efficiency Analysis. In addition to the effectiveness of models, the complexity and efficiency are also important for the application of retrieval models. In this subsection, we further compare these two factors between SLR and neural-based baselines on the CAIL2021-LCR dataset in Figure 5. The algorithm complexity is

measured by the number of parameters in the models, while the efficiency is measured by the average fine-tuning time per epoch on the CAIL2021-LCR dataset. For a fair comparison, all models are fine-tuned on four 32G NVIDIA V100 GPUs with batch size = 8. Note that traditional bag-of-words baselines are not taken into consideration because there is no trainable parameter or training stage in bag-of-words baselines. In other words, traditional bag-of-words baselines have a significantly faster inference speed and worse overall performance compared to neural-based models including SLR, but such a comparison has no practical meaning.

According to Figure 5 (a), we have the following findings:

1) For all versions of SLR, the model parameter number is almost the same as their corresponding backbone PLMs, but the retrieval performance is significantly better than the backbone PLMs. This result indicates that the internal structural information improves the performance of legal case retrieval without adding extra computing costs to the model.
2) SLR(Lawformer) has the best retrieval result on CAIL2021-LCR and the most model parameters. This is because its backbone, Lawformer, has the largest model scale and longest input length compared to other PLM backbones. Although the performance of SLR(RoBERTa) is slightly worse than SLR(Lawformer), its parameter number is 25 million less than SLR(Lawformer). Therefore, SLR(RoBERTa) is an efficient model for the application in real searching systems.
3) The parameter numbers of models other than SLR(Lawformer) and Lawformer are close, because their backbone PLMs are similar. However, BERT-PLI is not competitive as other models. This phenomenon is explained in Section 4.5.1.

According to Figure 5 (b), we have the following findings:

1) For all versions of SLR, the fine-tuning time per epoch slightly exceeds their corresponding backbone PLMs. This is because SLR computes the key Holding-based attention weight for each input token of the transformer layer in advance. The time spent on this step is acceptable considering the performance improvement.
2) Similar to the finding in Figure 5 (a), SLR(Lawformer) has the best retrieval result on CAIL2021-LCR but requires most fine-tuning time. This is because its backbone, Lawformer, takes up to 4,096 tokens each time as input. Therefore, the inference and attention computation of SLR(Lawformer) and Lawformer is more time-consuming compared to other PLM-based models taking 512 tokens each time as input.
3) The fine-tuning time per epoch of BERT-PLI is over 50 times longer than other models, because BERT-PLI adopts paragraph-level interaction to represent the similarity between two documents. For each training step, both the query case document and candidate case document are split into paragraphs and concatenated into multiple *[CLS] query paragraph [SEP] candidate paragraph* pairs as input to be fed into BERT-PLI. Since a legal case document usually has dozens of paragraphs, such an interaction is very time-consuming. On the other hand, the fine-tuning time of BERT-PLI can be greatly reduced if paragraphs of legal case documents are split and encoded in advance. In this way, the fine-tuning stage only requires time spent on the recurrent network as described in Shao et al. [44].

4.5.3 Selection of attention weight function. The attention weight function in Section 3.4 is a key factor to affect the overall performance of SLR. Therefore, we conduct a comparative experiment to select the attention weight function. In this paper, we adopt two commonly used attention weight functions in this experiment: bilinear function and dot product. Specifically, given v_t and v_{psg} in Equation 5, the bilinear function introduces a trainable weight matrix W_{att}, and the attention weight is defined as $v_t \cdot W_{att} \cdot v_{psg}$. By comparison, dot product simply represents the attention weight by $v_t \cdot v_{psg}$. Notably, self-attention is not included in our experiments because the selected key *Holding* passages are supervisory information in SLR. In addition, concatenating the key *Holding* passages and the input of the transformers is also considered as a possible alternative to replace the attention weight function. The concatenation strategy does not combine key *Holding* passages into the attention mask.

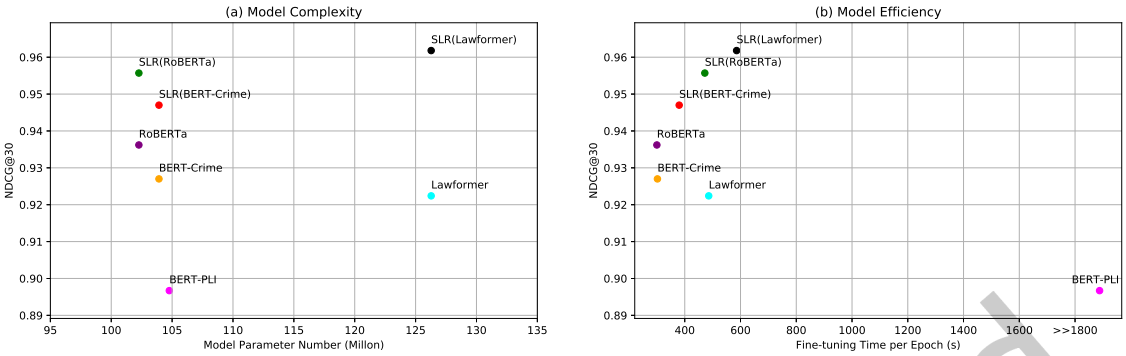

Fig. 5. Comparison between SLR and baselines in terms of (a) complexity and (b) efficiency on the CAIL2021-LCR dataset.

Table 4. Ranking performance of SLR using different attention strategies in Step 2.

Strategy	NDCG@10	NDCG@20	NDCG@30
None (Lawformer)	0.7675	0.8191	0.9090
Concatenation	0.7573	0.7935	0.8961
Weight matrix	**0.8169**	0.8364	0.9176
Dot product (Ours)	0.8140	**0.8488**	**0.9215**

Instead, it appends the key *Holding* passages to the input text of transformers in Section 3.5:

$$input = [CLS]\ query\ [SEP]\ facts\ [SEP]\ psg_{Holding}\ [SEP] \tag{13}$$

where $psg_{Holding}$ are tokenized key *Holding* passages.

The comparative experiment evaluates the performance of different strategies through NDCG metrics on the LeCaRD dataset. To control the variable of irrelevant modules, we conduct the comparative experiment using SLR without Step 4 and Step 5. The results are shown in Table 4. The first line of Table 4 is the result of Lawformer without any additional attention strategy. Among the remaining three strategies, both attention-based strategies improve all NDCG scores compared to the baseline, while the model with concatenated key *Holding* passages has even worse performance than the baseline. This result shows that simple concatenation does not learn key information from *Holding*, nor does it improve the overall ranking performance of candidates. Besides, although both attention strategies are beneficial to retrieving legal cases, using dot product as the attention weight function has the best results on two metrics (NDCG@20 and NDCG@30) while using a weight matrix only has one. The difference between these two attention-based methods on NDCG@10 is less than on NDCG@20 or NDCG@30. Consequently, we choose dot product as the function to compute the weights in Step 2.

4.5.4 Comparison against the CAIL2021 leaderboard. In addition to published competitive baselines, we also compare SLR against models in the latest competition. The CAIL2021 legal case retrieval task has three stages. Since the dataset for the third stage is not released yet, SLR is compared to models submitted to the second stage of the legal case retrieval task, which was held from September 10th, 2021, to October 10th, 2021. The second stage uses CAIL2021-LCR as the test set and an official open-source dataset as the training set. For a fair comparison, we follow the same training settings as the second stage to train SLR. Then, SLR is evaluated on the same test set as other submitted models. Figure 6 represents the results of SLR and top-15 models on the

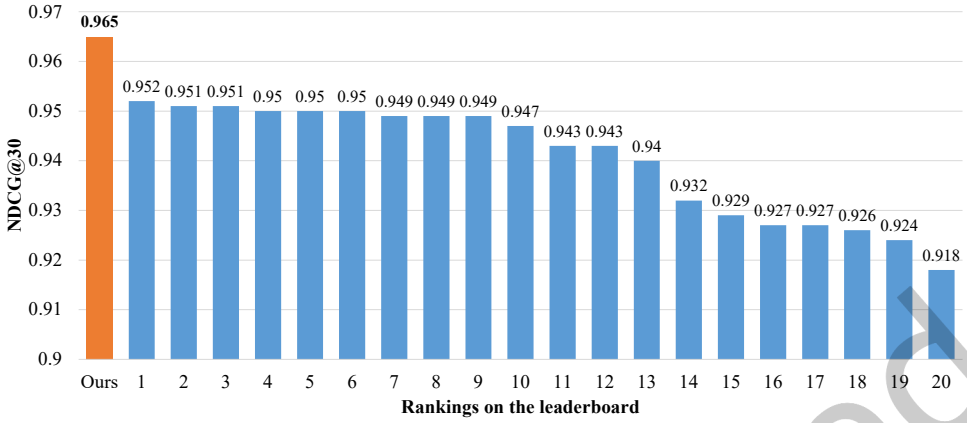

Fig. 6. Comparison with results on the leaderboard of CAIL2021 legal case retrieval task.

CAIL2021 leaderboard. The competition adopts NDCG@30 to measure submitted models, and SLR achieves the NDCG@30 score of 0.965, which outperforms the best result on the leaderboard by 1.3%. As the committee of CAIL2021 hasn't released the code of participating models to the public yet, we are unable to conduct further analysis or significance tests. Nevertheless, it can be concluded from Figure 6 that SLR has the best performance compared to submitted models in the CAIL2021 legal case retrieval task. Introducing structural information to the retrieval model is an effective method for improving the legal case retrieval task.

4.5.5 Selection of Learning-to-rank Framework. As the final step in SLR, Step 5 aggregates features from internal and external structural information to predict the final ranking score, which is directly related to the ranking results of the legal case retrieval task. In this paper, such aggregation is achieved by a learning-to-rank framework. Specifically, this learning-to-rank framework is implemented by an ensemble learning method, which integrates a cluster of weak learners and promotes them to strong learners for better prediction performance. So far, there are a number of ensemble learning methods available for the learning-to-rank framework in Step 5. Therefore, in order to choose the most suitable method for the aggregation of features in SLR, in this subsection, we conduct a comparative experiment on a series of existing competitive ensemble learning methods:

- Random Forest [5] is a classifier containing multiple decision trees. The number of features as well as the training data are all chosen randomly, and the prediction label with the most occurrences for the same prediction data is taken as the final prediction label.
- GBDT (Gradient Boosting Decision Tree) is an ensemble algorithm based on CART [8] decision trees. Among the algorithm, Gradient Boosting iterates the new learner through gradient descent. GBDT works well in data analysis and classification tasks.
- DART (Dropouts meet Multiple Additive Regression Trees) utilizes the trick of dropout setting in deep neural networks to randomly discard the generated decision trees and then iteratively optimize the boosted trees from the remaining set of decision trees.
- GOSS (Gradient-based One-Side Sampling) [20] aims to solve one limitation of GBDT that can not focus more on the last misclassified sample point at each iteration because there is no inherent weight in GBDT to implement such a process. GOSS, on the other hand, preserves large gradient samples and randomly selects some small gradient samples, compensating them with a constant weight. In this way, GOSS pays more attention to the undertrained samples, while not changing the overall distribution of the original data.

All the ensemble learning methods take the same features from previous steps as input and output the final ranking score $\hat{y}$. The overall results are shown in Table 5. According to Table 5a GBDT and DART prevail on the LeCaRD dataset, and their performance is similar. On the other hand, results in Table 5b show that GOSS has the best overall performance regarding different evaluation metrics, which suggests that one ensemble learning method can be applied to a specific dataset for a better ranking result. Nevertheless, no matter which dataset, the performance of GBDT, DART, and GOSS is better than Random Forest to a relatively large extent. Therefore, traditional boosting methods are proved more reasonable to be applied in the learning-to-rank framework, and the choice of boosting method depends on the specific dataset. Considering the design of the boosting method in SLR is not our main focus in this paper, we adopt GBDT as the method of the learning-to-rank framework in Step 5.

Table 5. Comparison between different boosting methods on the (a) LeCaRD and (b) CAIL2021-LCR datasets.

Boosting Method	NDCG@10	NDCG@20	NDCG@30	P@5	P@10	MAP
Random Forest	0.8600	0.8998	0.9393	0.4700	0.4400	0.5234
GBDT	0.8809	**0.9147**	**0.9496**	**0.4800**	**0.4600**	0.5481
DART	**0.8855**	0.9134	0.9495	0.4800	0.4600	**0.5506**
GOSS	0.8790	0.9122	0.9462	0.4900	0.4550	0.5317

(a) LeCaRD

Boosting Method	NDCG@10	NDCG@20	NDCG@30	P@5	P@10	MAP
Random Forest	0.8804	0.8844	0.9416	0.5900	0.5700	0.6204
GBDT	0.9108	0.9351	0.9618	**0.6250**	0.5700	0.6815
DART	0.9102	0.9324	0.9597	0.6150	**0.5775**	0.6701
GOSS	**0.9144**	**0.9393**	**0.9654**	0.6250	0.5700	**0.6885**

(b) CAIL2021-LCR

4.5.6 The influence of hyper-parameters on the experimental results. There are some hyper-parameters directly influencing the experimental results of the legal case retrieval task during the implementation of SLR. In this subsection, we elaborate on the choice of key hyper-parameters and analyze how different values of these hyper-parameters affect SLR.

The first hyper-parameter is the max depth of the decision tree in Step 5. This is used to control over-fitting when the size of input data is small. Notably, the decision tree in this paper still grows leaf-wise no matter how deep it is. In the default setting of Ke et al. [20], the max depth is set to four. However, the input of the decision tree only consists of two features (i.e., output vector $\hat{y}$ of length $|C|$ and charge embedding similarity score r). In other words, the total input dimension is only $|C| + 1$, where $|C| = 4$ is the number of label categories. Therefore, adopting a deep decision tree for learning to rank in Step 5 will cause model over-fitting. In this paper, we enumerate all possibilities of max depth less than or equal to four to discover the best decision tree depth for SLR. The value of the hyper-parameter mainly depends on NDCG metrics. The experimental result is shown in Figure 7. On the LeCaRD dataset, a decision tree with only depth one has the best performance, because the size of LeCaRD is relatively small. The aggregation of simple one-layer classifiers is enough to make predictions. However, results are different on CAIL2021-LCR, which contains a test set twice the size of that in LeCaRD. Therefore, a deeper decision tree is beneficial to the legal case retrieval task on CAIL2021-LCR. According to the results in Figure 7, in this paper, the depth of all decision trees is one for LeCaRD and two for CAIL2021-LCR. This value is dynamic considering further datasets with a larger size.

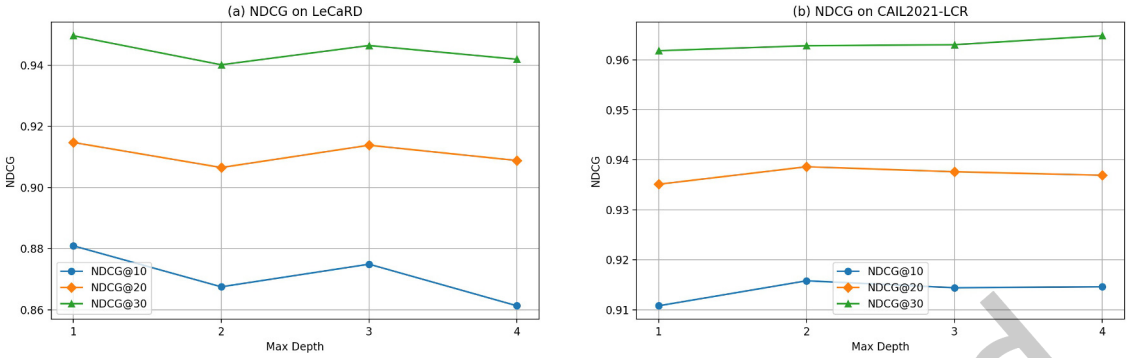

Fig. 7. The effect of max depth in decision trees in Step 5 on the (a) LeCaRD and (b) CAIL2021-LCR dataset.

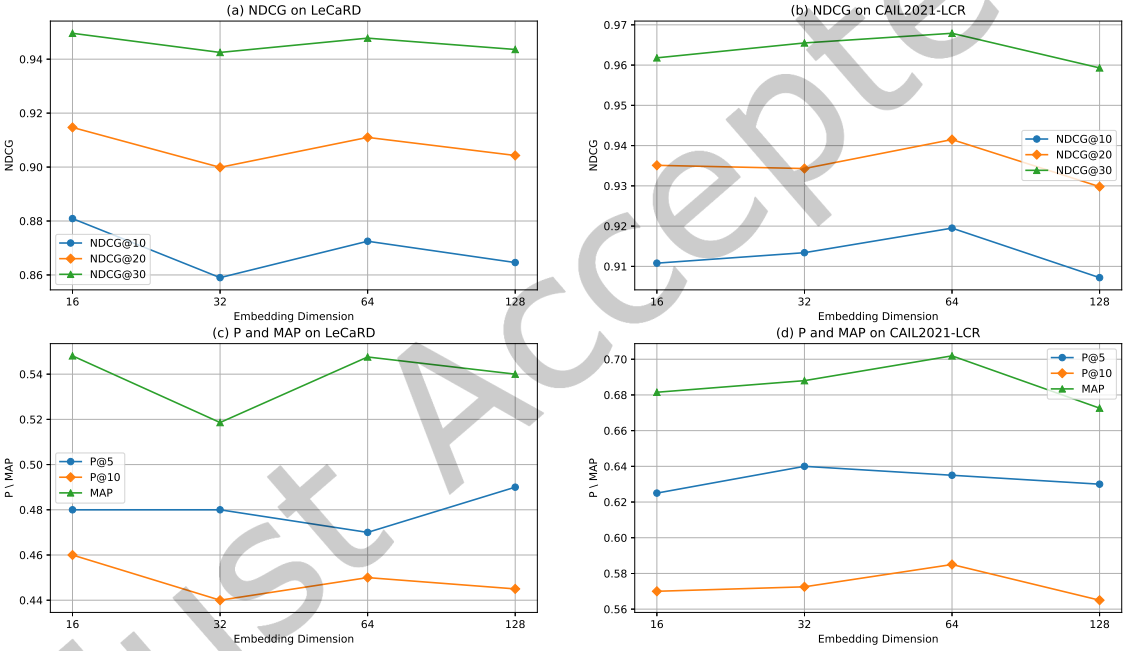

Fig. 8. The effect of charge embedding dimension in terms of different metrics on different datasets: (a) NDCG metrics on LeCaRD. (b) NDCG metrics on CAIL2021-LCR. (c) Precision metrics on LeCaRD. (d) Precision metrics on CAIL2021-LCR.

In addition, the dimension of charge embeddings used in Step 4 is also a key hyper-parameter, because the quality of charge embedding represents the effect of external structural information in SLR. Low-dimensional embeddings will result in information loss in the latent space of legal knowledge. On the contrary, embeddings with too high dimension are too sparse for charge representation, which is also a waste of storage resources. Therefore, we validate the performance of charge embeddings with different dimensions by changing the dimension of charge embeddings in Step 4 and observing the overall performance of SLR. The results are illustrated in Figure 8. The variation of the results on LeCaRD does not show a clear pattern. Specifically, in terms of NDCG metrics,

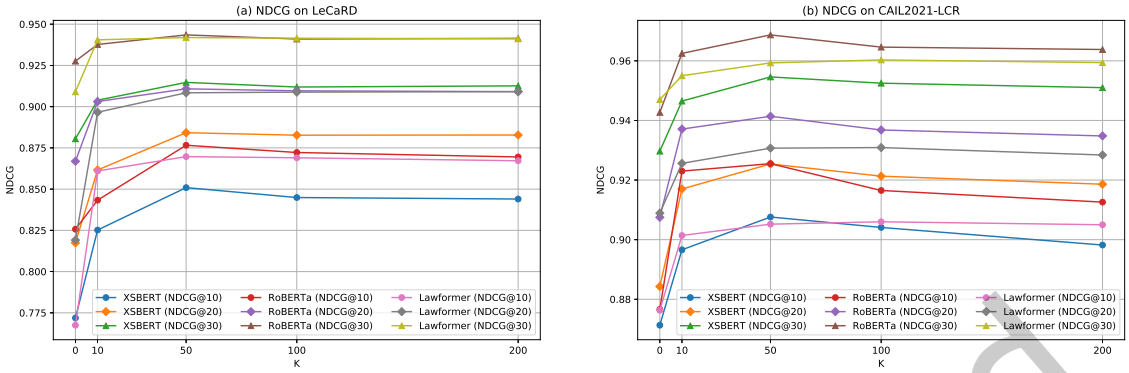

Fig. 9. The effect of hyper-parameter K on the (a) LeCaRD and (b) CAIL2021-LCR dataset.

embeddings of dimension 16 and 64 have better performance than embeddings of dimension 32 and 128. Also, the correlation between the dimension of embeddings and precision metrics is limited. Unlike LeCaRD, results on CAIL2021-LCR indicate that 64-dimensional charge embeddings have the best performance on both NDCG and precision metrics. Consequently, we adopt charge embeddings of dimension 64 for the representation of external structural information in SLR.

According to the design of SLR, two isolated scores, $\hat{y}$ and r are combined to predict the final ranking score. In this subsection, we demonstrate how much these two scores influence the final ranking result. To this end, the learning-to-rank framework in Step 5 is replaced with linear combination $y_{out} = \hat{y} + K * r$ to explicitly control the ratio between $\hat{y}$ and r. The experiment is conducted using several models on the LeCaRD and CAIL2021-LCR datasets. In practice, multiple values of K varied from zero to 1000 are adopted in this experiment. In this paper, the results of five representative values are shown in Figure 9 to reveal how the final ranking performance changes with K. When the value of K is small, the final ranking performance (i.e., NDCG score) increases as K increases. However, if $K > 50$, the final ranking performance declines when K continues to grow. In conclusion, the charge embedding relevance score r contributes more to the final ranking performance than $\hat{y}$, but the performance will decline if r takes up too much weight in Step 5.

4.6 Ablation Study

To investigate the effectiveness of internal and external structural information, we further conduct a series of ablation studies. In detail, we apply internal structural information to the attention mechanism from Step 1 to Step 3. Therefore, the generated attention mask described in Equation 5 is replaced with the binary mask in the original transformer to eliminate the effect of internal structural information, which is denoted as 'att' in Table 6. Similarly, graph G in Step 5 together with its generated charge embeddings are removed to validate the effect of external structural information, which is denoted as 'charge' in Table 6. The results in Table 6 demonstrate that the overall performance of SLR decrease without the attention layer (i.e., internal structural information) or charge relevance matching (i.e., external structural information). In addition, the model without the attention layer is close to the model without charge relevance. In addition, the results from both datasets show that by removing the charge relevance matching, SLR will lose more performance than by removing the attention layer. Although both structural information contributes to the effectiveness of SLR, external structural information is more important to the recognition of relevant legal cases. If both kinds of structural information are removed from SLR, the overall performances will significantly decrease to a more significant extent. Therefore, both attention

Table 6. Ablation study on (a) LeCaRD and (b) CAIL2021-LCR. 'w/o Att' denotes SLR without key *Holding*-based attention, 'w/o Charge' denotes SLR without charge relevance matching, and 'w/o L2R' denotes replacing the learning to rank method with simple linear combination in Step 5.

Model	NDCG@10	NDCG@20	NDCG@30	P@5	P@10	MAP
SLR	**0.8927**	**0.9268**	**0.9549**	**0.4900**	**0.4700**	**0.6165**
w/o Att	0.8794	0.9158	0.9493	0.4600	0.4500	0.5726
w/o Charge	0.8336	0.8659	0.9277	0.4600	0.4500	0.5740
w/o Att + Charge	0.8257	0.8669	0.9276	0.4600	0.4350	0.5505
w/o L2R	0.8397	0.8744	0.9316	0.4700	0.4550	0.5821

(a) LeCaRD

Model	NDCG@10	NDCG@20	NDCG@30	P@5	P@10	MAP
SLR	**0.9108**	**0.9351**	**0.9618**	**0.6250**	0.5700	**0.6815**
w/o Att	0.9009	0.9245	0.9545	0.5900	**0.5775**	0.6777
w/o Charge	0.8981	0.9213	0.9545	0.5650	0.5550	0.6145
w/o Att + Charge	0.8264	0.8721	0.9224	0.5450	0.5125	0.5852
w/o L2R	0.8691	0.8999	0.9378	0.5550	0.5350	0.6358

(b) CAIL2021-LCR

and charge relevance matching improve the performances of SLR in the legal case retrieval task. Combining the internal structural information and external structural information into the model can further improve the overall performance.

In addition to structural information, we validate the effectiveness of the learning-to-rank method in Step 5. To this end, Equation 12 in SLR is replaced with a linear combination: $y_{out} = \hat{y} + r$. The overall results are at the bottom of Table 6. The results of the ablation study on LeCaRD are consistent with CAIL2021-LCR. Compared to the original model, the performances of SLR without learning to rank slightly decrease. Although it is possible to add an extra weight parameter to the linear combination: $y_{out} = \hat{y} + K * r$, experiments in Figure 9 prove that the linear combination is still surpassed by the learning-to-rank strategy. On the other hand, the model without learning to rank methods outperforms all baselines. Note that the model without any of the structural information does not have the aggregation step in the ablation study. Therefore, the models in Table 6 all adopt the linear combination instead of the learning-to-rank method in Step 5.

4.7 Case Study

To have an intuitive understanding of the improvements made by structural information, we conduct a case study on SLR. First, in order to study what words or phrases the attention layer focuses on, we visualize the attention weights of words in a candidate case (label=3) in Step 2. As shown in Figure 10, the attention layer can focus on the words describing the key elements in a case. In particular, the words in dark red include key circumstances (e.g., *Before the contract expires*), key actions (e.g., *blocking*), and key entities (e.g., *6,000 RMB*). The words in light red, on the other hand, include concepts that are not crucial as dark red words but still rather important to a case. For example, key time (e.g., *2019*), key locations (e.g., *an abandoned fishpond*), and defendant (e.g., *Defendant Wen*) are all given the second-highest weight value by the attention layer in Step 2. Therefore, the extracted key *Holding* passages can address the problem of treating input tokens of a transformer equally without distinction.

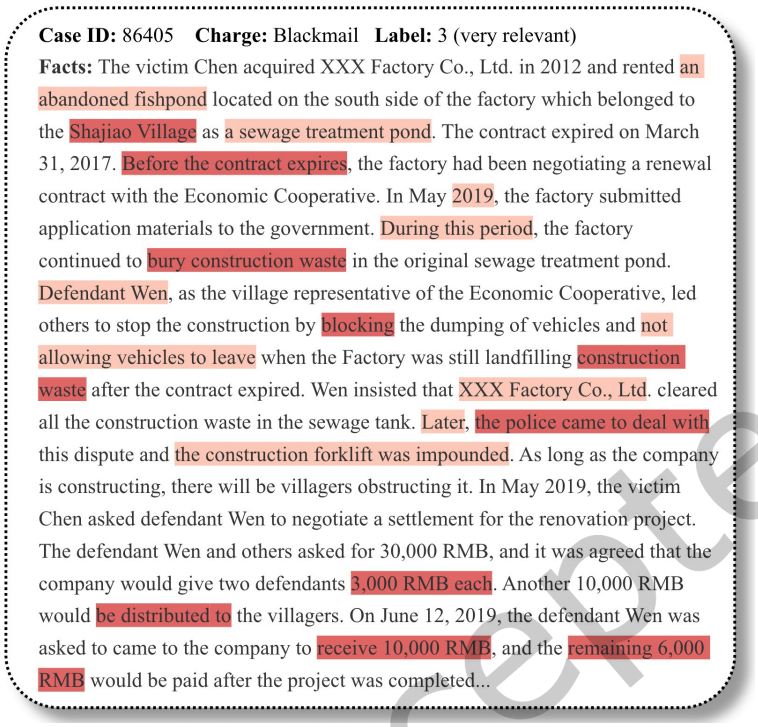

Fig. 10. Attention visualization of *Facts* in a candidate case. The words in the darker color refer to a higher attention weight than the words in the lighter color. Words on the white background denote the lowest attention weight. The first line contains basic information about the candidate case.

The attention layer in Step 2 gives prominence to the key information in candidate documents. As a result, the downstream task has a better convergence performance with the internal structural information.

Figure 11 is an illustration of how the structural information optimizes the process of reranking the retrieved candidate cases. Given *Query 84556* in the CAIL2021-LCR dataset as an example, the model without any structural information (i.e., the Lawformer backbone) first retrieves two negative candidate cases, *Doc1* and *Doc2* at the top of the list. Meanwhile, a highly relevant candidate case, *Doc16*, is in the middle of the list. As shown in Figure 10, when the key *Holding*-based attention is applied to Lawformer, the key elements in *Facts* will receive more attention. Consequently, cases whose highlighted words match the query more semantically (e.g., *Doc14* and *Doc16*) get a higher rank. At this moment, the ranking list still has potential for improvement. After introducing charge relevance matching to the model, the ranking list is optimized again. For instance, given *Blackmail* as the charge of the query case, the relevance score between charges of the candidate case in blue (i.e., *Forced Transaction*) and *Blackmail* is 0.8297. Besides, although the irrelevant case, *Doc3*, is in the third place, its charge (i.e., *Fraud*) only reaches a relevance score of 0.4915. By comparison, *Doc16* has the same charge as the query. Therefore, the rank of *Doc16* is increased to the first place when the model takes the charge relevance into consideration. In conclusion, the ranking list is optimized twice through internal and external structural information.

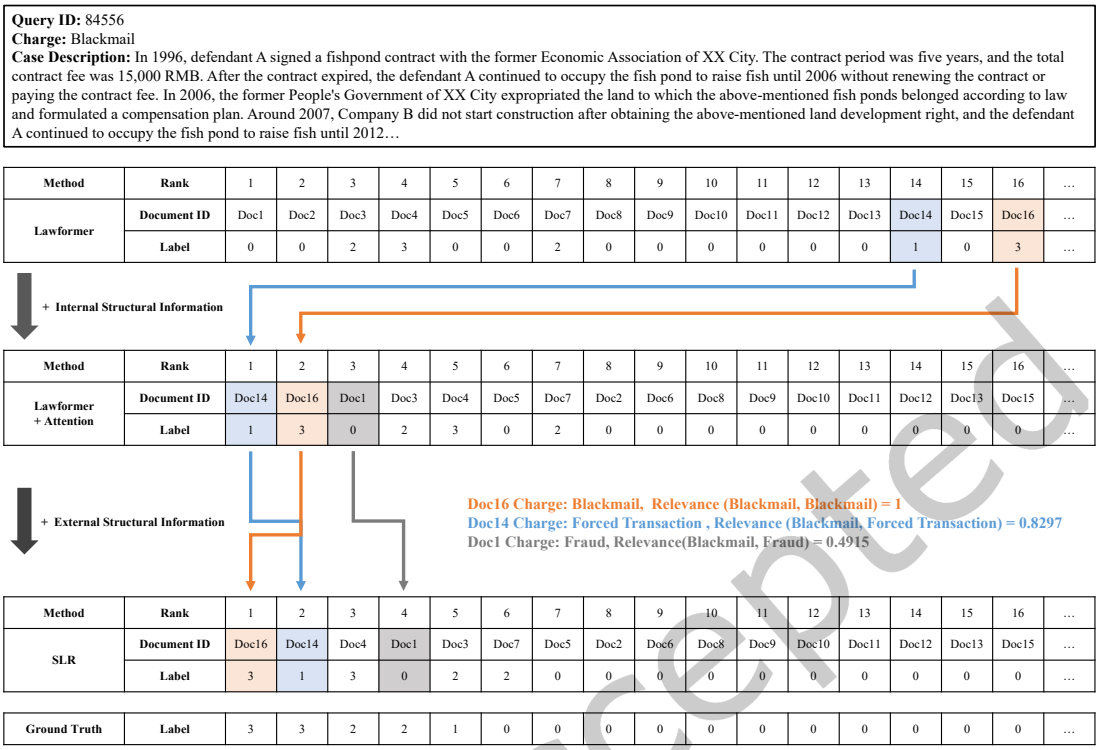

Fig. 11. Model optimization process of candidate rankings given Query 84556 in the CAIL2021-LCR dataset. Three tables at the top represent retrieval performance during different stages of SLR. Each table contains a ranking list of retrieved candidate cases, where numbers in the label line represent the relevance (e.g., 0: irrelevant 3: highly relevant) to the query case. The line at the bottom is the ground truth ranking results sorted by labels of all candidate cases. Relevance denotes the cosine similarity score between charge embeddings.

5 CONCLUSION

In this paper, we propose SLR, which incorporates internal and external structural information into the legal case retrieval task. Regarding the long length of legal documents, we employ the document-level internal structural information that divides case documents into segments to highlight key elements without losing information in the truncated case content. Besides, to address the difference of relevance between legal case retrieval and traditional ad-hoc retrieval, we further introduce external structural information which reveals the connections between charges in queries and candidate cases. Experiments on different legal case retrieval datasets demonstrate the effectiveness of our model. Through the experimental results, we can conclude that the internal structural information helps the model focus on key elements in legal documents, while the external structural information provides a charge-level feature to enhance the recognition of relevant cases. By combining internal and external structural information into the PLM, the performance of retrieving legal cases is significantly improved. We also conduct a case study to visualize how SLR focuses on key elements and improves the retrieval task.

In the future, we will mine more fine-grained and extensive structural information to further improve the effectiveness of legal information retrieval. Although this paper validates the effect of SLR on the datasets from the Chinese legal system, the idea of using internal and external structural information can also be applied to

other national legal systems, as it is designed based on the generic features of legal case documents. Our future work will consider working on cross-lingual legal case retrieval datasets or other tasks in the legal domain. In addition to the legal domain, documents in some other fields such as patent and medicine also have potential structural information to instruct retrieval tasks. We will try to apply the structured search method to other areas of IR, especially the retrieval tasks with specific domain information.

ACKNOWLEDGMENTS

This work is supported by the Natural Science Foundation of China (Grant No. 62002194).

REFERENCES

[1] AA Askari, SV Verberne, O Alonso, S Marchesin, M Najork, and G Silvello. 2021. Combining lexical and neural retrieval with longformer-based summarization for effective case law retrieva. In *Proceedings of the second international conference on design of experimental search & information REtrieval systems*. CEUR, 162–170.

[2] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[3] Trevor Bench-Capon, Michał Araszkiewicz, Kevin Ashley, Katie Atkinson, Floris Bex, Filipe Borges, Daniele Bourcier, Paul Bourgine, Jack G Conrad, Enrico Francesconi, et al. 2012. A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. *Artificial Intelligence and Law* 20, 3 (2012), 215–319.

[4] Paheli Bhattacharya, Kripabandhu Ghosh, Arindam Pal, and Saptarshi Ghosh. 2022. Legal case document similarity: You need both network and text. *Information Processing & Management* 59, 6 (2022), 103069.

[5] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[6] Si Chen, Pengfei Wang, Wei Fang, Xingchen Deng, and Feng Zhang. 2019. Learning to predict charges for judgment with legal graph. In *International Conference on Artificial Neural Networks*. Springer, 240–252.

[7] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509* (2019).

[8] Hugh A Chipman, Edward I George, and Robert E McCulloch. 1998. Bayesian CART model search. *J. Amer. Statist. Assoc.* 93, 443 (1998), 935–948.

[9] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101* (2019).

[10] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860* (2019).

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[12] Yichao Du, Pengfei Luo, Xudong Hong, Tong Xu, Zhe Zhang, Chao Ren, Yi Zheng, and Enhong Chen. 2021. Inheritance-guided Hierarchical Assignment for Clinical Automatic Diagnosis. *arXiv preprint arXiv:2101.11374* (2021).

[13] Filippo Galgani, Paul Compton, and Achim Hoffmann. 2012. Citation based summarisation of legal texts. In *Pacific Rim International Conference on Artificial Intelligence*. Springer, 40–52.

[14] Jidong Ge, Xiaoyu Shen, Chuanyi Li, Wei Hu, Bin Luo, et al. 2021. Learning Fine-grained Fact-Article Correspondence in Legal Cases. *arXiv preprint arXiv:2104.10726* (2021).

[15] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[16] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964* (2020).

[17] Hanjo Hamann. 2019. The German Federal Courts Dataset 1950–2019: From Paper Archives to Linked Open Data. *Journal of Empirical Legal Studies* 16, 3 (2019), 671–688.

[18] Pieter Hartel, Rolf + Wegberg, and Mark van Staalduinen. 2022. Investigating sentence severity with judicial open data: A case study on sentencing high-tech crime in the Dutch criminal justice system. *European Journal on Criminal Policy and Research* (2022), 1–21.

[19] Rabelo Juliano, Goebel Randy, Kano Yoshinobu, Kim Mi-Young, Yoshioka Masaharu, and Satoh Ken. 2021. Summary of the competition on legal information extraction/entailment (coliee) 2021. In *Proceedings of the COLIEE Workshop in ICAIL*.

[20] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017), 3146–3154.

[21] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[22] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[23] Haitao Li, Qingyao Ai, Jia Chen, Qian Dong, Yueyue Wu, Yiqun Liu, Chong Chen, and Qi Tian. 2023. SAILER: Structure-aware Pre-trained Language Model for Legal Case Retrieval. *arXiv preprint arXiv:2304.11370* (2023).
[24] Haitao Li, Weihang Su, Changyue Wang, Yueyue Wu, Qingyao Ai, and Yiqun Liu. 2023. THUIR@COLIEE 2023: Incorporating Structural Knowledge into Pre-trained Language Models for Legal Case Retrieval. arXiv:2305.06812 [cs.IR]
[25] Minghan Li, Diana Nicoleta Popa, Johan Chagnon, Yagmur Gizem Cinar, and Eric Gaussier. 2023. The power of selecting key blocks with local pre-ranking for long document information retrieval. *ACM Transactions on Information Systems* 41, 3 (2023), 1–35.
[26] Tie-Yan Liu. 2011. Learning to rank for information retrieval. (2011).
[27] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).
[28] Yonghe Lu, Yuanyuan Zhai, Jiayi Luo, and Yongshan Chen. 2019. MLPV: Text Representation of Scientific Papers Based on Structural Information and Doc2vec. *American Journal of Information Science and Technology* 3, 3 (2019), 62–71.
[29] Yougang Lyu, Zihan Wang, Zhaochun Ren, Pengjie Ren, Zhumin Chen, Xiaozhong Liu, Yujun Li, Hongsong Li, and Hongye Song. 2022. Improving legal judgment prediction through reinforced criminal element extraction. *Information Processing & Management* 59, 1 (2022), 102780.
[30] Yixiao Ma, Yunqiu Shao, Bulou Liu, Yiqun Liu, Min Zhang, and Shaoping Ma. 2021. Retrieving Legal Cases from a Large-scale Candidate Corpus. (2021).
[31] Yixiao Ma, Yunqiu Shao, Yueyue Wu, Yiqun Liu, Ruizhe Zhang, Min Zhang, and Shaoping Ma. 2021. LeCaRD: A Legal Case Retrieval Dataset for Chinese Law System. *Information Retrieval (IR)* 2 (2021), 22.
[32] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[33] Akshay Minocha, Navjyoti Singh, and Arjit Srivastava. 2015. Finding relevant indian judgments using dispersion of citation network. In *Proceedings of the 24th International Conference on World Wide Web*. 1085–1088.
[34] William S Noble. 2006. What is a support vector machine? *Nature biotechnology* 24, 12 (2006), 1565–1567.
[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019), 8026–8037.
[36] Jay M Ponte and W Bruce Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. 275–281.
[37] Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2019. A summary of the COLIEE 2019 competition. In *JSAI International Symposium on Artificial Intelligence*. Springer, 34–49.
[38] Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2020. COLIEE 2020: methods for legal document retrieval and entailment. In *JSAI International Symposium on Artificial Intelligence*. Springer, 196–210.
[39] Radim Řehůřek, Petr Sojka, et al. 2011. Gensim—statistical semantics in python. *Retrieved from genism. org* (2011).
[40] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995), 109.
[41] Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24, 5 (1988), 513–523.
[42] Manavalan Saravanan, Balaraman Ravindran, and Shivani Raman. 2009. Improving legal information retrieval using an ontological framework. *Artificial Intelligence and Law* 17, 2 (2009), 101–124.
[43] Jaromir Savelka, Hannes Westermann, Karim Benyekhlef, Charlotte S Alexander, Jayla C Grant, David Restrepo Amariles, Rajaa El Hamdani, Sébastien Meeùs, Aurore Troussel, Michał Araszkiewicz, et al. 2021. Lex rosetta: transfer of predictive models across languages, jurisdictions, and legal domains. In *Proceedings of the eighteenth international conference on artificial intelligence and law*. 129–138.
[44] Yunqiu Shao, Jiaxin Mao, Yiqun Liu, Weizhi Ma, Ken Satoh, Min Zhang, and Shaoping Ma. 2020. BERT-PLI: Modeling Paragraph-Level Interactions for Legal Case Retrieval.. In *IJCAI*. 3501–3507.
[45] Yunqiu Shao, Yueyue Wu, Yiqun Liu, Jiaxin Mao, and Shaoping Ma. 2022. Understanding Relevance Judgments in Legal Case Retrieval. *ACM Transactions on Information Systems* (2022).
[46] Sainbayar Sukhbaatar, Edouard Grave, Piotr Bojanowski, and Armand Joulin. 2019. Adaptive attention span in transformers. *arXiv preprint arXiv:1905.07799* (2019).
[47] J Sun. 2012. Jieba chinese word segmentation tool.
[48] Zhongxiang Sun, Jun Xu, Xiao Zhang, Zhenhua Dong, and Ji-Rong Wen. 2022. Law Article-Enhanced Legal Case Matching: a Model-Agnostic Causal Learning Approach. *arXiv preprint arXiv:2210.11012* (2022).
[49] Vu Tran, Minh Le Nguyen, Satoshi Tojo, and Ken Satoh. 2020. Encoded summarization: summarizing documents into continuous vector space for legal case retrieval. *Artificial Intelligence and Law* 28, 4 (2020), 441–467.

[50] Vu Tran, Minh Le Nguyen, and Ken Satoh. 2019. Building legal case retrieval systems with lexical matching and summarization using a pre-trained phrase scoring model. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*. 275–282.
[51] Maarten Trompper and Radboud Winkels. 2016. Automatic assignment of section structure to texts of Dutch court judgments. In *Legal knowledge and information systems*. IOS Press, 167–172.
[52] Marc Van Opijnen and Cristiana Santos. 2017. On the concept of relevance in legal information retrieval. *Artificial Intelligence and Law* 25, 1 (2017), 65–87.
[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.
[54] Pengfei Wang, Yu Fan, Shuzi Niu, Ze Yang, Yongfeng Zhang, and Jiafeng Guo. 2019. Hierarchical matching network for crime classification. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 325–334.
[55] Zihan Wang, Hongye Song, Zhaochun Ren, Pengjie Ren, Zhumin Chen, Xiaozhong Liu, Hongsong Li, and Maarten de Rijke. 2021. Cross-domain contract element extraction with a bi-directional feedback clause-element relation network. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1003–1012.
[56] Hannes Westermann, Jaromir Savelka, and Karim Benyekhlef. 2020. Paragraph similarity scoring and fine-tuned BERT for legal information retrieval and entailment. In *JSAI International Symposium on Artificial Intelligence*. Springer, 269–285.
[57] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* (2019).
[58] Chaojun Xiao, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. Lawformer: A Pre-trained Language Model for Chinese Legal Long Documents. *AI Open* (2021).
[59] Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Yansong Feng, Xianpei Han, Zhen Hu, Heng Wang, et al. 2018. Cail2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478* (2018).
[60] Xiaobing Xue and W Bruce Croft. 2009. Automatic query generation for patent search. In *Proceedings of the 18th ACM conference on Information and knowledge management*. 2037–2040.
[61] Jun Yang, Weizhi Ma, Min Zhang, Xin Zhou, Yiqun Liu, and Shaoping Ma. 2021. Legalgnn: Legal information enhanced graph neural network for recommendation. *ACM Transactions on Information Systems (TOIS)* 40, 2 (2021), 1–29.
[62] Linan Yue, Qi Liu, Binbin Jin, Han Wu, Kai Zhang, Yanqing An, Mingyue Cheng, Biao Yin, and Dayong Wu. 2021. NeurJudge: A Circumstance-aware Neural Framework for Legal Judgment Prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 973–982.
[63] Yiming Zeng, Ruili Wang, John Zeleznikow, and Elizabeth Kemp. 2005. Knowledge representation for the intelligent legal case retrieval. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*. Springer, 339–345.
[64] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks* 6, 1 (2019), 1–23.
[65] Haoxi Zhong, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. 2019. *Open Chinese Language Pre-trained Model Zoo.* Technical Report. https://github.com/thunlp/openclap
[66] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.