*Article*

# A High-Precision Two-Stage Legal Judgment Summarization

**Yue Huang [1,2], Lijuan Sun [1,2,*], Chong Han [1,2] and Jian Guo [1,2]**

[1] College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[2] Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[*] Correspondence: sunlj@njupt.edu.cn

**Abstract:** Legal judgments are generally very long, and relevant information is often scattered throughout the text. To complete a legal judgment summarization, capturing important, relevant information comprehensively from a lengthy text is crucial. The existing abstractive-summarization models based on pre-trained language have restrictions on the length of an input text. Another concern is that the generated summaries have not been well integrated with the legal judgment's technical terms and specific topics. In this paper, we used raw legal judgments as information of different granularities and proposed a two-stage text-summarization model to handle different granularities of information. Specifically, we treated the legal judgments as a sequence of sentences and selected key sentence sets from the full texts as an input corpus for summary generation. In addition, we extracted keywords related to technical terms and specific topics in the legal texts and introduced them into the summary-generation model as an attention mechanism. The experimental results on the CAIL2020 and the LCRD datasets showed that our model achieved an overall 0.19–0.41 improvement in its ROUGE score, as compared to the baseline models. Further analysis also showed that our method could comprehensively capture essential and relevant information from lengthy legal texts and generate better legal judgment summaries.

**Keywords:** legal judgment; text summarization; generative network; attention mechanism

**MSC:** 68U15

check for updates

**Citation:** Huang, Y.; Sun, L.; Han, C.; Guo, J. A High-Precision Two-Stage Legal Judgment Summarization. *Mathematics* **2023**, *11*, 1320. https://doi.org/10.3390/math11061320

Academic Editor: Ioannis G. Tsoulos

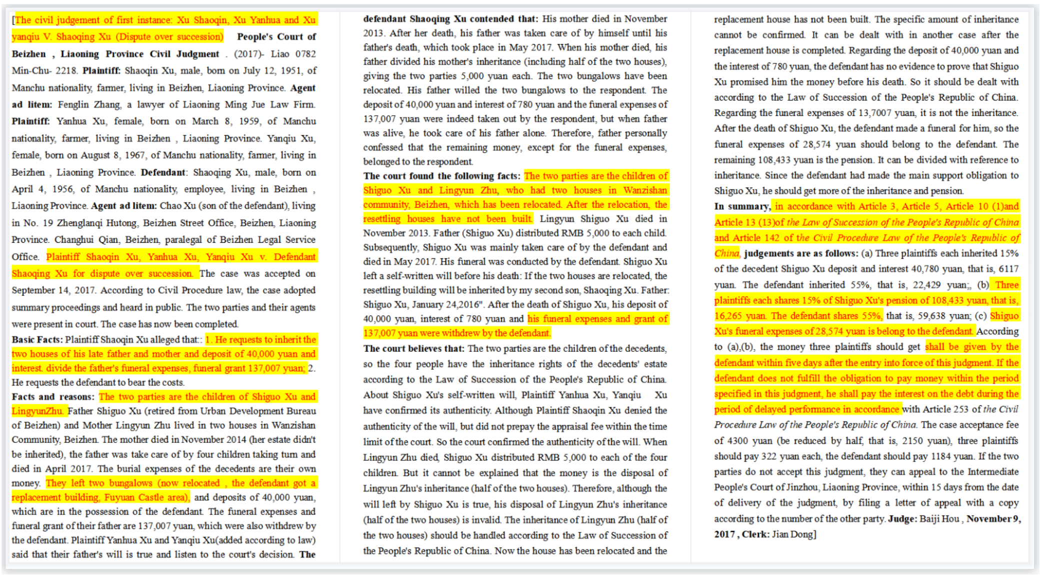
Received: 7 February 2023
Revised: 2 March 2023
Accepted: 7 March 2023
Published: 9 March 2023

## 1. Introduction

In the legal field, previous judgment documents are important references for legal practitioners to determine the measurements of the penalties and judgment scales. Legal practitioners must read through hundreds of judgment documents to identify precedents that conform to a specific case's characteristics. Therefore, the automatic summarizations of legal judgments are very important. Saravanan et al. proposed legal-field-specific approaches to implement automatic summarization for legal judgment documents, to a certain extent [1,2]. However, the complex textual structure of a legal document is different from that of an academic paper. An academic paper typically has a conclusion section that summarizes the work, while a legal judgment's conclusion may be scattered throughout the text. Figure 1 demonstrates how the key information was scattered throughout a specific case judgment document. Therefore, extracting key information from a lengthy text is crucial for the implementation of summarization automation on legal judgment documentation.

To promote this research direction, Erkan et al. employed LexRank to extract the summary of a judgment document [3]. LexRank is an extractive text summarization method that treats text summarization as a sentence-ranking task and can handle lengthy text input. However, the summaries generated by LexRank had sentence redundancies and incoherence, and LexRank could not fully cover the text. Rush et al. further adopted an abstract text summarization method [4]. They used the abstractive-summarization model to combine and arrange words in the lexicon to form generalized new sentences.

Although their method was flexible and conformed to the generation method of natural summarization, it had some issues, such as factual bias and textual repetition. Moreover, abstractive-summarization models usually are unable to handle long texts due to their high memory requirements.


[The civil judgement of first instance: Xu Shaoqin, Xu Yanhua and Xu yanqiu V. Shaoqing Xu (Dispute over succession) **People's Court of Beizhen , Liaoning Province Civil Judgment** . (2017)- Liao 0782 Min-Chu- 2218. **Plaintiff:** Shaoqin Xu, male, born on July 12, 1951, of Manchu nationality, farmer, living in Beizhen, Liaoning Province. **Agent ad litem:** Fenglin Zhang, a lawyer of Liaoning Ming Jue Law Firm. **Plaintiff:** Yanhua Xu, female, born on March 8, 1959, of Manchu nationality, farmer, living in Beizhen , Liaoning Province. Yanqiu Xu, female, born on August 8, 1967, of Manchu nationality, farmer, living in Beizhen , Liaoning Province. **Defendant**: Shaoqing Xu, male, born on April 4, 1956, of Manchu nationality, employee, living in Beizhen , Liaoning Province. **Agent ad litem:** Chao Xu (son of the defendant), living in No. 19 Zhenglanqi Hutong, Beizhen Street Office, Beizhen, Liaoning Province. Changhui Qian, Beizhen, paralegal of Beizhen Legal Service Office. Plaintiff Shaoqin Xu, Yanhua Xu, Yanqiu Xu v. Defendant Shaoqing Xu for dispute over succession. The case was accepted on September 14, 2017. According to Civil Procedure law, the case adopted summary proceedings and heard in public. The two parties and their agents were present in court. The case has now been completed.

**Basic Facts:** Plaintiff Shaoqin Xu alleged that:: 1. He requests to inherit the two houses of his late father and mother and deposit of 40,000 yuan and interest. divide the father's funeral expenses, funeral grant 137,007 yuan; 2. He requests the defendant to bear the costs.

**Facts and reasons:** The two parties are the children of Shiguo Xu and LingyunZhu. Father Shiguo Xu (retired from Urban Development Bureau of Beizhen) and Mother Lingyun Zhu lived in two houses in Wanzishan Community, Beizhen. The mother died in November 2014 (her estate didn't be inherited), the father was take care of by four children taking turn and died in April 2017. The burial expenses of the decedents are their own money. They left two bungalows (now relocated , the defendant got a replacement building, Fuyuan Castle area), and deposits of 40,000 yuan, which are in the possession of the defendant. The funeral expenses and funeral grant of their father are 137,007 yuan, which were also withdrew by the defendant. Plaintiff Yanhua Xu and Yanqiu Xu(added according to law) said that their father's will is true and listen to the court's decision. **The defendant Shaoqing Xu contended that:** His mother died in November 2013. After her death, his father was taken care of by himself until his father's death, which took place in May 2017. When his mother died, his father divided his mother's inheritance (including half of the two houses), giving the two parties 5,000 yuan each. The two bungalows have been relocated. His father willed the two bungalows to the respondent. The deposit of 40,000 yuan and interest of 780 yuan and the funeral expenses of 137,007 yuan were indeed taken out by the respondent, but when father was alive, he took care of his father alone. Therefore, father personally confessed that the remaining money, except for the funeral expenses, belonged to the respondent.

**The court found the following facts:** The two parties are the children of Shiguo Xu and Lingyun Zhu, who had two houses in Wanzishan community, Beizhen, which has been relocated. After the relocation, the resettling houses have not been built. Lingyun Shiguo Xu died in November 2013. Father (Shiguo Xu) distributed RMB 5,000 to each child. Subsequently, Shiguo Xu was mainly taken care of by the defendant and died in May 2017. His funeral was conducted by the defendant. Shiguo Xu left a self-written will before his death: If the two houses are relocated, the resettling building will be inherited by my second son, Shaoqing Xu. Father: Shiguo Xu, January 24,2016". After the death of Shiguo Xu, his deposit of 40,000 yuan, interest of 780 yuan and his funeral expenses and grant of 137,007 yuan were withdrew by the defendant.

**The court believes that:** The two parties are the children of the decedents, so the four people have the inheritance rights of the decedents' estate according to the Law of Succession of the People's Republic of China. About Shiguo Xu's self-written will, Plaintiff Yanhua Xu, Yanqiu Xu have confirmed its authenticity. Although Plaintiff Shaoqin Xu denied the authenticity of the will, but did not prepay the appraisal fee within the time limit of the court. So the court confirmed the authenticity of the will. When Lingyun Zhu died, Shiguo Xu distributed RMB 5,000 to each of the four children. But it cannot be explained that the money is the disposal of Lingyun Zhu's inheritance (half of the two houses). Therefore, although the will left by Shiguo Xu is true, his disposal of Lingyun Zhu's inheritance (half of the two houses) is invalid. The inheritance of Lingyun Zhu (half of the two houses) should be handled according to the Law of Succession of the People's Republic of China. Now the house has been relocated and the replacement house has not been built. The specific amount of inheritance cannot be confirmed. It can be dealt with in another case after the replacement house is completed. Regarding the deposit of 40,000 yuan and the interest of 780 yuan, the defendant has no evidence to prove that Shiguo Xu promised him the money before his death. So it should be dealt with according to the Law of Succession of the People's Republic of China. Regarding the funeral expenses of 13,7007 yuan, it is not the inheritance. After the death of Shiguo Xu, the defendant made a funeral for him, so the funeral expenses of 28,574 yuan should belong to the defendant. The remaining 108,433 yuan is the pension. It can be divided with reference to inheritance. Since the defendant had made the main support obligation to Shiguo Xu, he should get more of the inheritance and pension.

**In summary,** in accordance with Article 3, Article 5, Article 10 (1)and Article 13 (13)of *the Law of Succession of the People's Republic of China* and Article 142 of *the Civil Procedure Law of the People's Republic of China*, **judgements are as follows:** (a) Three plaintiffs each inherited 15% of the decedent Shiguo Xu deposit and interest 40,780 yuan, that is, 6117 yuan. The defendant inherited 55%, that is, 22,429 yuan;, (b) Three plaintiffs each shares 15% of Shiguo Xu's pension of 108,433 yuan, that is, 16,265 yuan. The defendant shares 55%, that is, 59,638 yuan; (c) Shiguo Xu's funeral expenses of 28,574 yuan is belong to the defendant. According to (a),(b), the money three plaintiffs should get shall be given by the defendant within five days after the entry into force of this judgment. If the defendant does not fulfill the obligation to pay money within the period specified in this judgment, he shall pay the interest on the debt during the period of delayed performance in accordance with Article 253 of *the Civil Procedure Law of the People's Republic of China*. The case acceptance fee of 4300 yuan (be reduced by half, that is, 2150 yuan), three plaintiffs should pay 322 yuan each, the defendant should pay 1184 yuan. If the two parties do not accept this judgment, they can appeal to the Intermediate People's Court of Jinzhou, Liaoning Province, within 15 days from the date of delivery of the judgment, by filing a letter of appeal with a copy according to the number of the other party. **Judge:** Baiji Hou , **November 9, 2017 , Clerk:** Jian Dong]

**Figure 1.** Example of case judgment. This shows how key information is scattered throughout this specific case judgment document.

With the development of deep learning, a neural-network-based summarization model using seq2seq has become a research hotspot. To handle the factual biases and textual repetition issues, See et al. proposed a pointer-based seq2seq model that solved the problem of unrecorded words and rare words, to a certain extent [5]. In addition, See et al. tackled sequence duplication using a coverage mechanism. Although the readability of the summaries generated by their seq2seq model was better, a large corpus was required for training. Due to legal restrictions, large volumes of legal judgment documentation and reference summaries are not freely available. Current legal case datasets have less than 30k training examples [6], which is one-tenth of academic datasets, such as arXiv and PubMed [7]. Therefore, unsupervised and self-supervised model-based text summarization methods are gaining more attention in the legal field.

The pre-trained language model was introduced into legal text summarization as a self-supervised neural-network-based model. Feijo et al. proposed a pre-training model-based method [8] called LegalSumm. LegalSumm created different views on the source text to handle lengthy legal judgments and added a hidden module to determine the authenticity of candidate summaries, relative to the source text. However, their method used the RulingBR dataset (which was in Portuguese), where the documentation and its summaries were much shorter in length than typical legal judgments. The lengthy legal judgment issue was, therefore, not entirely resolved. In addition, legal judgments usually contain an abundance of technical terms and keywords related to specific topics [9], which can be difficult to interpret. Although a pre-trained language summary model can better integrate the contextual information, its attention mechanism becomes too sparse to focus on relevant topics when processing a lengthy text. Therefore, integrating keyword information into the summary-generation model is also challenging.

To tackle with these issues, we used raw legal judgment documentation as information at different granularities, including the text, the sentences, and the keywords. To shorten the length of the input text and capture the key information scattered throughout the complete text, we treated the legal judgment documentation as a sequence of sentences and selected key sentence sets from the full text as an input corpus for abstractive summarization. At the same time, the keyword information, such as technical terms and specific topics, was extracted from the text as an attention mechanism and introduced into the text-generation model, so the final summary could better integrate key information.

In this study, we proposed a two-stage summary-generation model to handle different granularities of information. As shown in Figure 2, our model combined extractive and abstractive summarization. The model's first stage comprehensively extracted key sentences from legal judgment texts using extractive summarization. Specifically, we first divided the judgment documentation into a linear sequence of sentences annotated with $[x_1, x_2, \ldots, x_n]$ and introduced a hierarchical sentence-vector-representation method to realize rapid sentence vectorization. Next, we utilized the BERT+LSTM model to annotate the sequences of sentence vectors in order to indicate whether the extractive summary model should contain the corresponding sentences, to realize the extraction of key sentences. The model's second stage involved inputting the previously extracted key sentences into the abstractive summary model to complete the final summary generation. Our abstractive summarization method was based on the unified pre-trained language model (UNILM) [10], which only required a specific self-attention mask for the seq2seq text-generation task. In addition, we adopted a keyword extraction method based on a BiLSTM to extract keywords related to professional terms and specific topics in the judgment text. Moreover, we introduced these into the text-generation model as an attention mechanism to better integrate these keywords into the final generated summary. Furthermore, we introduced a sparse softmax and copy mechanism into our model to avoid over-fitting and any factual deviation in the generated summary. To summarize, our contributions were as follows:

- We proposed a two-stage summary-generation model. Our model generated a text summarization after extracting key information, thereby aggregating the relevant important information that was scattered throughout lengthy legal judgment texts.
- We introduced keywords related to technical terms and specific topics in legal judgment texts as an attention mechanism for our proposed model, so key information could be better integrated into the final generated summary.
- We demonstrated the effectiveness of our proposed approach through extensive experiments on datasets comprised of legal judgment documentation.

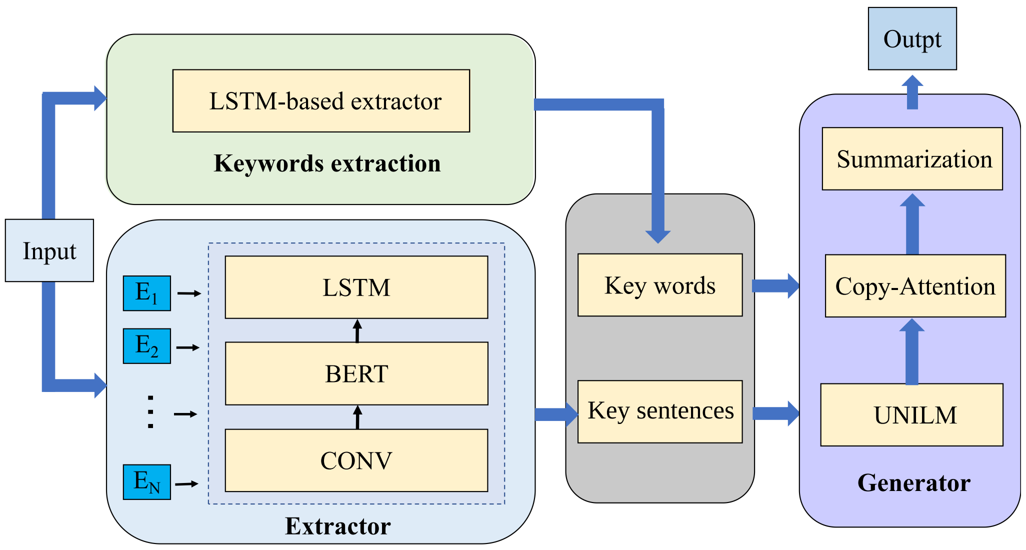


**Figure 2.** Overall framework of our proposed two-stage model. In stage one, we utilized the CONV + BERT + LSTM model to realize the extraction of key sentences and adopted an extractive method based on BiLSTM to extract keywords. In stage two, we sent this information to the abstractive model, which consisted of a UNILM model and an attention mechanism, which then could generate a summary.

The rest of this paper is organized as follows: First, in Section 2, we introduce the relevant work and the research status. Then, we introduce the proposed method in detail and analyze it theoretically in Section 3. We present our experimental design and result analysis in Section 4. Finally, we summarize the study in Section 5.

## 2. Related Work

In this section, we provide an overview the recent advances in pre-trained language models and text summarization.

Pre-training Methods: This was a model based on its predecessors to resolve similar issues. When solving a problem, researchers typically begin with a model that has been trained on similar issues, instead of training a new model from scratch. A pre-trained language model provides a state-of-the-art approach for many NLP tasks and saves considerable time and resources. The existing pre-trained methods generally fall into two categories: (1) language models, such as GPT [11], which is a sequential generation model that predicts the next word based on the previous context; and (2) masked language models, such as BERT [12], RoBERTa [13], and ALBERT [14], which randomly mask some input words and predict these masked words. Masked language models have the advantage of being bi-directional, rather than uni-directional, which means that they view a text from both the left and the right of the token being predicted.

Automatic Text Summarization (ATS): ATS can be divided into two categories: (1) extractive summarization and (2) abstractive summarization. The extractive summarization method attempts to create a summary by selecting essential sentences or phrases from the source documents. Zhang et al. [15] regarded the extraction of a summary as a latent variable-inference problem. When predicting an output summary, Liu et al. [16] applied the concept of structured attention to induce multiple dependent tree representations of documents. Narayan et al. [17] optimized the objective function to extract an importance statement and form a summary through reinforcement learning. Nallapati et al. [18] obtained concise sequences by using a recurrent neural network (RNN) as an encoder.

The abstractive summarization approach considers input documents as an intermediate representation, from which an output summary is then generated. It generates new texts, rather than simply selecting and combining sentences from the source document. Some encoder–decoder methods [4] have generated sentences using the deep-learning method RNN. Paulus et al. [19] proposed a deep reinforcement-learning model for abstractive summarization. Narayan et al. [20] proposed a convolutional neural network and further conditioned it on topic distributions.

In addition, recent studies [21] have focused on summarization via pre-trained language models. Zhang et al. and Liu et al. [22,23] first applied BERT to fine-tune text summarizations. By using BERT's word-embedding as input, Wang et al. [24] integrated extractive and generative networks into a unified model. This enabled the model to learn how to extract and generate information from the input text in an iterative and adaptive approach.

Inspired by the above research, we abandoned the limitations of a single method and combined extractive summarization with abstractive summarization in this study.

## 3. Method

### *3.1. Problem Definition*

The problem of a two-stage legal judgment summarization was defined as follows:

**Input:** The source document of legal judgment.

**Stage One:** Extract essential sentences and keywords: Firstly, we introduced a hierarchical sentence-vector-representation method to realize rapid sentence vectorization. Next, we utilized the BERT + LSTM model to annotate a sequence of sentence vectors to realize the extraction of key sentences. In addition, we adopted an extractive method based on a BiLSTM to extract keywords related to professional terms and specific topics in the judgment text.

**Stage Two:** Generate summarization: Firstly, we input the previously extracted key sentences into the abstractive summary model based on a UNILM. Then, we introduced the previously extracted keywords into the text-generation model as an attention mechanism to better integrate these keywords into the final generated summary.

**Output:** The generated legal judgment summary.

Next, we present the proposed model in detail. Particularly, we discuss the extraction of key sentences and keywords, the abstractive-summarization model, and the relevant designs of these two components.

### 3.2. Extractive Model

To identify the most expressive sentences in the full text, we first split both the source text and the reference summary into sentences. Then, for each sentence in the summary, we searched for the sentences with the closest meaning in the complete source text.

Specifically, we divided the text into a linear sequence of sentence units marked as $[x_1, x_2, ..., x_n]$. Extractive summarization was defined as a sequence-annotation task. It provided a label $y_i \in \{0,1\}$ to each sentence $x_i$ to indicate whether the summary should include the sentence. In the next step, we converted the original generative corpus into a sequence-labeled one. An overview of the extractive method is shown in Figure 3. The extractive-summarization model was a process, rather than a result, and the extracted results still had to be further optimized by the abstractive-summarization model. The principle of the extractive-summarization model was to be comprehensive, that is, to cover all the information required by the final summary.It was worth noting that extractive summarization methods also organized the extracted text to streamline the data, but these methods usually have discouraging performance in terms of text coherence. Therefore, we used the summary-generation approach to polish the summary.

#### 3.2.1. Hierarchical Sentence Representation

We employed a temporal convolutional model [25] to compute the representation of each individual sentence in the document, so that sentence coding could be realized quickly and simultaneously. BERT [26] was applied to the convolutional output to further capture the long-range semantic dependencies among sentences, and it ensured the context semantics. Moreover, it enabled a strong representation, denoted as $h_j$, for the $j$-th sentence in the document to be learned, and therefore, the contexts of all previous and future sentences were considered in the same document.

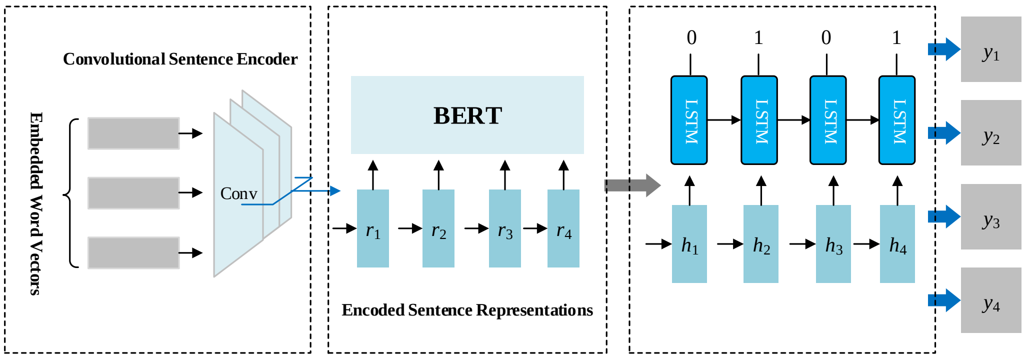


**Figure 3.** The overview of the extractive model. First, we employed a temporal convolutional model to compute the representation of each individual sentence in the document. Then, we annotated the sequence of the sentence vectors to indicate whether the extractive summary should contain the corresponding sentences, in order to realize the extraction of key sentences.

#### 3.2.2. Sentence Selection

For the purpose of selecting the extracted sentences on the basis of the above sentence representations, we annotated the sequences of the sentence vectors to indicate whether the extractive summary model should contain the corresponding sentences, in order to realize the extraction of key sentences. Chen et al. [27] further proved that the combination of long short-term memory(LSTM) with a transformer had unique advantages. In this paper, we employ an LSTM model to annotate the sequences of the sentence vectors. Furthermore, LSTM introduced a gating mechanism to remove and add information. Additionally, the gate determined the information that should be removed or updated through a sigmoid neural network layer, effectively avoiding the problem of long-term dependence on text processing. The LSTM had three of these gates, including a forget-gate, an input-gate, and an output-gate.

After the preceding sentence representations, the sentence-embedding $s = \{s_1, s_2, \ldots, s_n\}$ was obtained as input for the LSTM. At $t$-time, we assumed that the current input was $s_t$ and the output of the previous moment was $h_{t-1}$. The first step in the LSTM was to determine which information from the cell state would be discarded by the forget-gate, as shown below:

$$f_t = \text{sigmoid}\left(w_f[h_{t-1}, s_t] + b_f\right) \tag{1}$$

where $f_t$ is the forget-gate, and $w_f$ and $b_f$ are the weight and the bias, respectively, of the forget-gate.

Subsequently, the input-gate determined what new information we would store in the cell state. The cell-state-update value was obtained from $s_t$ and $h_{t-1}$ through an activation function of *tanh*. Furthermore, the process of the calculation was as follows:

$$i_t = \text{sigmoid}(w_i[h_{t-1}, s_t] + b_i) \tag{2}$$

$$\tilde{c}_t = \tanh(w_c[h_{t-1}, s_t] + b_c) \tag{3}$$

where $i_t$ is the input-gate, $w_i$ and $b_i$ are the weight and the bias, respectively, of the input-gate, $\tilde{c}_t$ is the cell state-update value, $w_c$ and $b_c$ are the weight and the bias, respectively, of the cell state.

In the next step, we combined the above two expressions to create an update of the cell state $c_t$.

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{4}$$

The output-gate determined the output information. The current output was obtained by filtering the new cell state through the output-gate; the calculation formula was the following:

$$o_t = \text{sigmoid}(w_o[h_{t-1}, s_t] + b_o) \tag{5}$$

$$h_t = o_t * \tanh(c_t) \tag{6}$$

where $o_t$ is the output-gate, $w_o$ and $b_o$ are the weight and the bias, respectively, of the output-gate, and $h_t$ is the current output.

Eventually, we adopted a sigmoid function to calculate the final prediction score $y_t$ of the input sentence, aiming to determine whether the extractive summary needed to contain this sentence, as indicated below:

$$y_t = \text{sigmoid}\left(w^{'} h_t + b^{'}\right) \tag{7}$$

#### 3.2.3. Keywords Extraction

Legal judgment texts usually contain many keywords related to technical terms and specific topics, which are difficult to interpret. In order to integrate the keyword information into the summary, we added a keyword-extraction component to obtain the keywords. At present, the term frequency–inverse document frequency (TF-IDF) method [28] has been widely used to extract keywords. The TF-IDF method indicated that if a word frequently appeared in an article but rarely in other articles, we could assume that the word was important. Although TF-IDF was simple and efficient, it measured the importance of words only by frequency, without considering the word order or semantics. To overcome this problem, we introduced a BiLSTM and an attention mechanism because the BiLSTM could learn contextual semantic information well, and the attention mechanism could identify the relevant information in the full text. In this study, we used a seq2seq network [29] to perform keyword-extraction tasks. An overview of the model is shown in Figure 4. Specifically, the BiLSTM encoded the input sequence $[x_1, x_2, \ldots, x_n]$ into a hidden layer state $[h_1, h_2, \ldots, h_n]$, respectively, where $x_i$ is the $i$-th input token and $h_i$ represent the $i$-th hidden state. Then, the hidden state sequence $[h_1, h_2, \ldots, h_n]$ could be updated by the attention mechanism and introduced to the decoder LSTM to generate an output sequence $[k_1, k_2, \ldots, k_n]$, respectively. The variable $k_t$ is the generated keyword at $t$ time, and it is according to the $k_{t-1}$ at the $t$-1 time, the current hidden state $\hat{h}_t$, and the context vector $c_t$, as shown below:

$$k_t = f(\hat{h}_t, k_{t-1}, c_t) \tag{8}$$

In the attention mechanism, the contribution degree of each hidden state $\hat{h}_i$ to the prediction $k_t$ was controlled by the attention weight $\alpha_t$.

$$\hat{h}_t = \alpha_t * f(h_1, h_2, \ldots, h_n) \tag{9}$$

where $\alpha_t$ is obtained according to the correlation between $k_{t-1}$ and the hidden layer state at $t$ time.

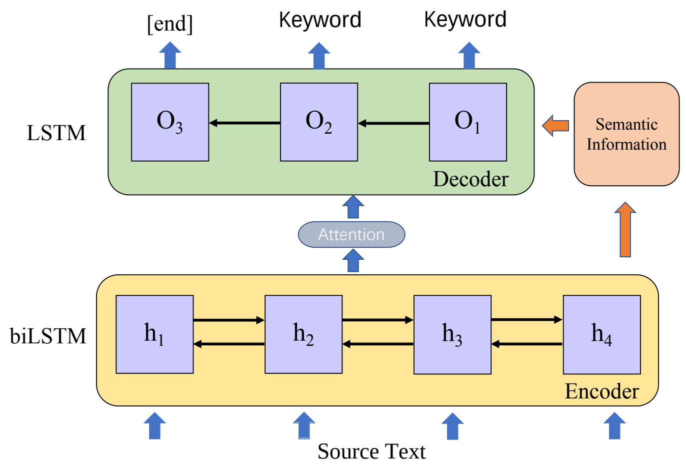


**Figure 4.** Keyword-extraction model. We introduced a BiLSTM and an attention mechanism to perform keyword-extraction tasks.

### 3.3. *Abstractive Model*

#### 3.3.1. seq2seq Learning of UNILM

Due to the good performance of the RoBERTa-wwm-ext [30] pre-trained language model, we chose its parameters to initialize our generation model and used the mask strategy of the UNILM mask. At the same time, we input the text sequence in the format of "document + reference summary". Here, the "document" was composed of the key sentences obtained from the model in the first step.

We input data in the format of "[cls] document [sep] reference summary [sep]", assuming that the input $x = [x_1, x_2, ..., x_n]$, $x$ was an n-dimensional vector. The model was composed of multi-layer transformers, assuming that the output of the transformer after the $L$ layer was expressed as $h^l$, as shown below:

$$h^l = \text{Transformer}_1\left(h^{l-1}\right) \tag{10}$$

$$h^l = \left[h_1^L, h_2^L, \ldots, h_n^L\right] \tag{11}$$

In UNILM, the calculation of the attention head $A_l$ was similar to that in BERT, except for introducing the self-attention mask matrix M.

$$\begin{cases} Q = h^{l-1}W_l^Q \\ K = h^{l-1}W_l^K \\ V = h^{l-1}W_l^V \end{cases} \tag{12}$$

$$M_{ij} = \begin{cases} 0 & allow \quad to \quad attend \\ -\infty & prevent \quad from \quad attending \end{cases} \tag{13}$$

$$A_l = Attention(Q,K,V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right)V_l \tag{14}$$

where the previous layer's output $h^{l-1} \in R^{|x|\times d_h}$ is linearly projected to a triplet of queries, keys, and values, using parameter matrices $W_l^Q, W_l^K, W_l^V \in R^{d_h \times d_k}$, respectively, and the mask matrix $M \in R^{|x|\times|x|}$ determined whether a pair of tokens could be related to each other.Different mask matrices $M$ were used to control attention for different contexts. As shown in (13), tokens in the first sequence (document) could be related to each other, while tokens in the second sequence (reference summarization) could only be related to the left tokens. For example, based on a document sequence "$x_1x_2x_3$" and its reference summary sequence "$x_4x_5$", the input sequence of the model was "$[cls]x_1x_2x_3[sep]x_4x_5[sep]$". Here, each token in "$[cls]x_1x_2x_3[sep]$" could related to each other, but "$x_4$" in "$x_4x_5[sep]$" could only relate to the five tokens on the left.

### 3.3.2. Generator Network

In order to generate the summary, the model decoder was decoded by a multi-layer transformer, and the output was connected to a linear classifier (such as the softmax classifier) to generate the probability distribution of the vocabulary, thereby generating the summary text. Specifically, we used the "document" sequence instead of the "document + reference summary" sequence as the input text. Here, the "document" was composed of the key sentences obtained from the first step of the extractive model. Assuming that the given input was "$[cls]x_1x_2x_3[sep]$", we first attached a "$[mask]$" token at the end of the sequence.

$$[cls]x_1x_2x_3[sep][mask] \tag{15}$$

Then, we adjusted its corresponding attention mask matrix so that the tokens in "$[cls]x_1x_2x_3[sep]$" could be related to each other but could not be related to "$[mask]$". Furthermore, "$[mask]$" could be related to all tokens in "$[cls]x_1x_2x_3[sep][mask]$". After the sequence was input into the UNILM model, the representation vectors $h_{[cls]}$,$h_1$,$h_2$,$h_3$,$h_{[sep]}$,$h_{[mask]}$ of each token were obtained. Then, the representation vector $h_{[mask]}$ made linear changes and fed them into the sparse softmax function to realize the probability distribution of the vocabulary:

$$P_{vocab} = \text{sparsemax}(w'(wh_{[MASK]} + b) + b') \tag{16}$$

where $w'$, $w$, $b'$, and $b$ are learning parameters.

Since the conventional softmax was prone to over-learning and over-fitting, we chose the sparse softmax function, sparsemax, to alleviate this problem [31]. The formula was defined as follows:

$$\text{sparsemax}(z)_j = \begin{cases} \frac{e^{z_i}}{\sum_{j\in\Omega_k} e^{z_j}}, i \in \Omega_k \\ \quad 0 \quad , i \notin \Omega_k \end{cases} \tag{17}$$

where $\Omega_K$ represents the subscript set of the first $k$ elements after $(z_1, z_2, \ldots, z_n)$ were arranged from large to small. In other words, when using sparsemax to calculate the probability, only the probability of the first $k$ elements needed to be retained, and the rest were directly set to 0.

We added the generated keyword semantic information vector $k$ as an additional input for the attention mechanism and modified it, according to (16):

$$P_{vocab} = \text{sparsemax}(w^{'}(wh_{[MASK]} + w_k k_j + b) + b^{'}) \tag{18}$$

where $w_k$ is a learning parameter. By extracting keywords from the legal judgment text, we could introduce them into the summary-generation model as an attention mechanism in order to integrate the keyword semantic information into the final generated summary.

### 3.3.3. Copy Mechanism and Loss function

Copy mechanism: Furthermore, the generation probability $P_{gen}$ was introduced to represent the probability of generating words from the vocabulary, and $1 - P_{gen}$ represented the probability of generating words from the original text, as shown in Figure 5. The $P_{gen}$ was calculated by the sigmoid function, as follows:

$$P_{gen} = \text{sigmoid}(w_h H_t^{'} + w_m h_{t-output} + w_x x_t) + b^{''}) \tag{19}$$

where $w_h$,$w_m$,$w_x$ and $b^{''}$ are learning parameters, $H_t^{'}$ is the context vector, $h_{t-output}$ is the output of the decoder at time $t$, and $x_t$ is the input of the decoder at the time $t$. Since $P_{gen}$ was obtained, the improved vocabulary probability distribution could be calculated:

$$P(w) = P_{gen} P_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \tag{20}$$

If $w$ was a word outside the vocabulary, $P_{vocab}(w) = 0$ indicated that words could not be generated from the vocabulary and would need to be copied from the source document. The expression $\sum_{i:w_i=w} a_i^t$ represented the probability of copying from the source document, according to the distribution of attention.

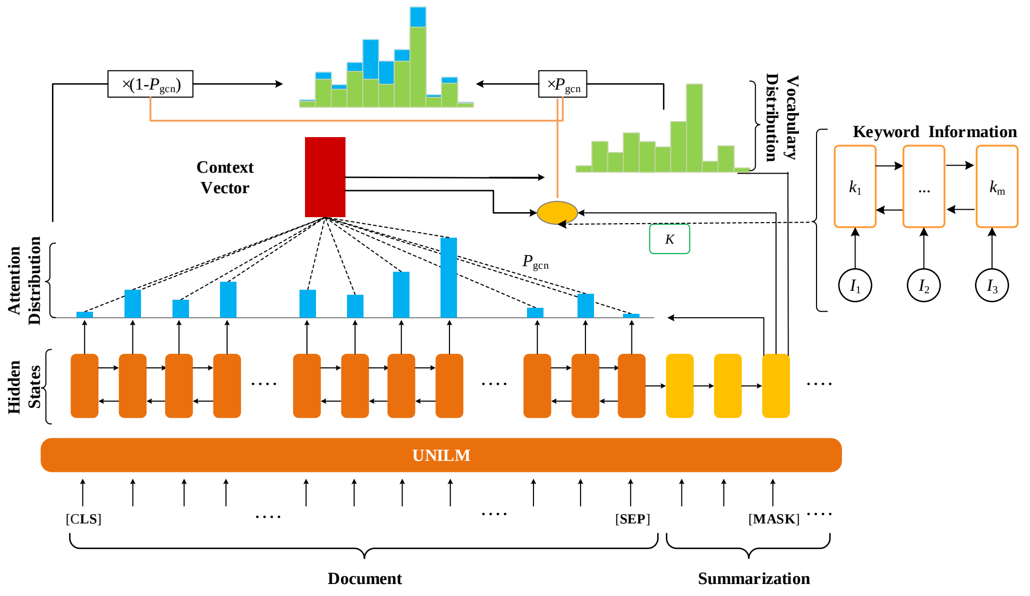


**Figure 5.** The overall of summary-generation model. Our model was based on a unified pre-training language model. We introduced keywords into the text-generation model as an attention mechanism. Furthermore, we introduced sparse softmax and a copy mechanism into our model to avoid overfitting and any factual deviation in the generated summary.

Loss function: In order to avoid any duplication of the results when generating the summary, we added a coverage loss to the original loss in the model. A coverage vector $c^t$ obtained by summing the attention weights of the previous time steps was calculated. The attention distribution of the original text at time $t$ was affected by the previous attention distribution, thus avoiding excessive repeated attention on the same position and preventing the repetition of the generated results.

$$c^t = \sum_{i=0}^{t-1} a^i \tag{21}$$

$$covloss_t = \sum_i \min\left(a_i^t, c_i^t\right) \tag{22}$$

where $c_i^t$ is the coverage vector at time $t$, $a_i^t$ represents the attention weight at time $t$, and the final loss of each step was obtained by adding the coverage loss to the original loss function:

$$loss_t = -\log P(w_t^*) + \lambda \sum_i \min\left(a_i^t, c_i^t\right) \tag{23}$$

## 4. Experimental Results

In this section, we first discuss the datasets used in this paper and the baselines in the existing text summarization research. Then, we describe the corresponding experiments on the datasets. Finally, we detail the ablation experiments that verified the effectiveness of every component.

### *4.1. Datasets*

Two datasets were used in this study for comparison. The first experimental dataset was sourced from the legal summarization competition, the China Legal Research Cup CAIL2020 (The CAIL2020 data used to support the findings of this study are available at https://www.kaggle.com/datasets/weipengfei/sfzy-small (accessed on 26 January 2023)).These data were provided and notated by the Beijing Judicial Big-Data Institute, including 9848 civil judgments in the first instance. The judgment document was divided into several sentences in advance, each sentence was annotated with a label of importance, and the corresponding reference summary was provided. The second was the Legal Case Reports Dataset (LCRD) [32], which contained Australian legal cases from the Federal Court of Australia (FCA). There were 4000 judgment texts with corresponding reference summaries in this dataset. The above two datasets were both long-text summary datasets. Before summary generation, we pre-processed the texts and reference abstracts in the datasets, including text de-noising, word segmentation, sentence segmentation, etc.

Due to the uncertainty and randomness of the datasets, splitting the datasets into training and testing sets only once could lead to a certain degree of bias in the evaluation results of the model. In order to solve this problem, we adopted a $k$-fold cross-validation method and randomly divided the datasets multiple times when training and evaluating the model. The datasets were divided into different training sets and testing sets each time to evaluate the stability and generalization abilities of the model. Specifically, we chose $k$ = 5 and divided the labeled data into five parts, four of which were used to train the extraction model. For each round of cross-validation, we randomly selected one as the testing set and the remaining four as the training set. We used the training set to train the model and then evaluated the model's performance on the testing set. For each of the five rounds, we selected a different testing set each time, and then we finally took the average of the five performance indicators as the model's performance index.

### *4.2. Implementation Details*

#### 4.2.1. Justification for the Applied Techniques

In this study, we applied various techniques to develop legal judgment summarization models, and we justified each technique in terms of its applicability to the corresponding task. First, we used a pre-trained language model based on the transformer architecture, which has shown promising results in various natural-language-processing tasks. Using the transformer in legal judgment summarization was justified because it could capture the domain-specific language and the structures of legal texts, which could be challenging for other general language models. Second, we adopted a two-stage summary-generation method. Our model extracted key sentences using an extractive method and then fed them into an abstractive model to generate text summaries. This technique was reasonable because it could effectively solve issues concerning lengthy texts and scattered important information. Third, we introduced technical terms and topic-specific keywords from legal judgment texts as an attention mechanism in the summarization model. It was reasonable because the attention mechanism provided the model with domain-specific knowledge to improve model performance on summarization tasks.

In summary, we carefully chose the techniques for our model of legal judgment summarization. Transformer architecture, two-stage summarization methods, and attention mechanisms were justified based on their contributions towards improving the performance and quality of the generated summaries.

#### 4.2.2. Parameter Setting

The UNILM model was constructed by using the bert4keras8 toolkit [33]. The number of transformer layers of the UNILM was set at 12, the hidden vector dimension was set at 768, num_training_steps was set at 10,000, and the epochs were set at 55. The learning rate of the Adam optimizer was $3 \times 10^{-5}$, and the linear decay function of the learning rate was used to ensure the learning rate decayed linearly, from the first step to the last step, at 50% of the initial learning rate. In the decoding stage, we used a beam search to generate the text summaries, and the beam size was set at 5.

### *4.3. Baselines*

We compared the performance of our model with three types of text-summarization models: an extractive model, an abstractive model, and a two-stage model. We chose these methods as the baselines because they have been commonly used for text summarization. All comparison models are described in detail, as follows.

- Lead-3: This model is an extractive summarization that selects the document's first three sentences as the text summary.
- TextRank: This model was proposed by Mihalcea et al. [34]. It uses the PageRank algorithm to determine significant sentences and then rank and reorganize them.
- SummaRuNNer: This model was proposed by Nallapati et al. [18]. It uses a recurrent neural network to capture the final subset of the source document.
- Pointer-Generator: This model was proposed by See et al. [5]. Using a pointer mechanism, it copies the words directly from the source document into the generated text.
- BERTSumABs: This model was proposed by Liu and Lapata et al. [23]. It learns the general language representation through pre-training and uses it for text summary generation.
- KESG: This model proposes a two-stage task and uses keyword extraction for text summarization to achieve the best performance [29].

### *4.4. Metrics*

ROUGE (recall-oriented understudy for gisting evaluation) [35] was a set of metrics for evaluating automatic summarization and machine translation. In general, ROUGE evaluated a summary based on the co-occurrence of information in $n$-grams in the summary and was an evaluation method for $n$-gram recall. ROUGE-N was an $n$-gram recall, computed as follows:

$$ROUGE - N = \frac{\sum\limits_{s \in ref} \sum\limits_{gram_n \in S} Count_{match}(gram_n)}{\sum\limits_{s \in ref} \sum\limits_{gram_n \in S} Count(gram_n)} \tag{24}$$

The denominator is the number of $n$-grams in the artificial summary (that is, the standard summary), and the numerator is the number of $n$-grams that co-occur (coincide) between the artificial summary and the machine-generated automatic summary. ROUGE was very similar to the definition of recall metrics. We chose ROUGE-1, ROUGE-2, and ROUGE-L (the indicator following "ROUGE" expressed the longest common sub-sequence between the two text units) as metrics in this paper.

### *4.5. Results*

In this subsection, we describe the series of experiments conducted to investigate the performance of our proposed model.The details are presented in Table 1, which shows the automatic evaluation results of our model and the comparison models on the datasets. The first three models in the table are the comparison baselines of the extractive-summarization models. The following two are the comparison baselines of the abstractive-summarization models. KESG [29] is the state-of-the-art Chinese two-stage text-summarization model; the last is our model. Based on our results, we could draw several general conclusions:

- The extractive method did not exhibit good performance on the legal dataset because the legal documents had complex text structures and the relevant elements had been scattered throughout the text. Furthermore, while the abstractive model performed better than the extractive model, it required a very long time to train.
- As compared to the traditional model, the performance of the model based on pre-training was better under different evaluation criteria, indicating that the pre-trained language model could capture key information and summarize texts effectively.
- As compared to the extractive and abstractive summarization models, the two-stage model had a higher score in the ROUGE index. This suggested that the two-stage summarization model could fully use the advantages of the above two models, prioritized the key information of the source text, and improved the accuracy and readability of the generated text.
- Our framework achieved the best performance, as compared to all the baselines. Our method realized a 0.19–0.41 improvement over the best results on the two datasets.

**Table 1.** Comparison results on two datasets. We compared the performance of our model with three types of text-summarization models.

| **Model** | **CAIL2020** | | | **LCRD** | | |
|---|---|---|---|---|---|---|
| | **ROUGE-1** | **ROUGE-2** | **ROUGE-L** | **ROUGE-1** | **ROUGE-2** | **ROUGE-L** |
| Lead-3 | 18.48 | 5.06 | 11.94 | 17.36 | 5.13 | 10.58 |
| TestRank | 20.57 | 5.15 | 13.53 | 20.24 | 5.62 | 13.06 |
| SummaRuNNer | 22.55 | 6.25 | 15.64 | 21.57 | 6.19 | 14.83 |
| Pointer-Generator | 38.37 | 21.09 | 30.26 | 37.59 | 20.14 | 29.56 |
| BERTSumABs | 43.23 | 26.94 | 35.15 | 42.02 | 26.07 | 34.56 |
| KESG | 43.51 | 26.91 | 35.21 | 42.86 | 26.10 | 34.63 |
| Our Model | **43.76** | **27.11** | **35.62** | **43.15** | **26.32** | **34.95** |

In summary, the experiments verified the superiority of our approach for text summarization. However, the distinctive effect of each individual mechanism had to be further clarified. To this end, we conducted ablation experiments, as described in the next subsection.

### *4.6. Ablation Experiments*

To prove the validity of each component, we removed certain modules from the framework separately and observed the performance. As discussed, our framework had two components (the extractive model and the abstractive model). We listed the baselines and compared their performance, as follows:

- Our-EM: This variant removed the extractive summarization component and used only keyword extraction.
- Our-RWE: This variant did not use the parameters of RoBERTa-wwm-ext to initialize our generation model and, instead, used the parameters of the basic BERT to initialize the model.
- Our-SS: This variant removed the sparse softmax function and used the traditional softmax function.
- Our-KE: This variant removed the keyword-extraction method.
- Our-CM: This variant removed the copy mechanism.

The performance of all ablation variants is displayed in Table 2, which revealed that all our proposed components achieved continuous progress in terms of ROUGE-1, ROUGE-2, and ROUGE-L. At the same time, the results showed that the integrated framework significantly outperformed all the alternative ablative variants on all three metrics. This verified the effectiveness of our complete framework. Specifically, we had the following findings based on the results of the ablation experiments:

- The extractive component had important significance in our summarization task and provided significant improvement for our model. It extracted the key sentences that were most relevant to the reference summary from the original text and provided complete contextual text information for the generation model.
- The sparse softmax component improved our model by 1–2%. Since our model used a pre-trained model, the application of the sparse softmax prevented over-fitting and enhanced the interpretability.
- Although the keyword extraction component had little impact on the improvement of our model, it introduced the keyword information, such as technical terms and specific topics in the legal judgment text, into the summary-generation model. Moreover, our model did not significantly regress if the keyword extraction was ablated.
- The copy mechanism improved our model by 0.5–1%. Although the improved score was not very high, it ensured the fidelity between the summary and the original text and mitigated professional errors. Therefore, it was quite necessary for practical use.

**Table 2.** Results of ablation experiments on two datasets. We removed specific modules from the framework separately and compared the performance.

| **Model** | **CAIL2020** | | | **LCRD** | | |
|---|---|---|---|---|---|---|
| | **ROUGE-1** | **ROUGE-2** | **ROUGE-L** | **ROUGE-1** | **ROUGE-2** | **ROUGE-L** |
| Our-EX | 40.52 | 23.95 | 32.27 | 40.23 | 23.54 | 32.68 |
| Our-RWE | 41.03 | 24.88 | 33.26 | 42.36 | 25.93 | 34.23 |
| Our-SS | 43.03 | 26.65 | 35.04 | 42.76 | 26.15 | 34.78 |
| Our-KE | 43.66 | 27.05 | 35.53 | 43.06 | 26.28 | 34.81 |
| Our-CM | 43.36 | 26.98 | 35.34 | 42.93 | 26.01 | 34.62 |
| Our Model | **43.76** | **27.11** | **35.62** | **43.15** | **26.32** | **34.95** |

Through the ablation experiments, we found that some methods improved the model significantly while others contributed only minor improvements to the model's performance.

In general, our integrated model achieved the best performance, as compared to all the baselines, indicating that our model integrated these components well.

### *4.7. Impact of Hyper-Parameters*

In this subsection, we investigate how different hyper-parameters affected the proposed framework. Figure 6 shows the results when we selected the hyper-parameter $k = 10$ of sparse softmax, as this model obtained the best performance on the CAIL2020 and LCRD datasets. At the same time, Figure 7 shows that the number of keywords that also had an impact on the final performance. For the CAIL2020 dataset, the model performed best when the number of keywords was 15. For the LCRD dataset, when the number of keywords was between 15 and 20, the model achieved the best performance.

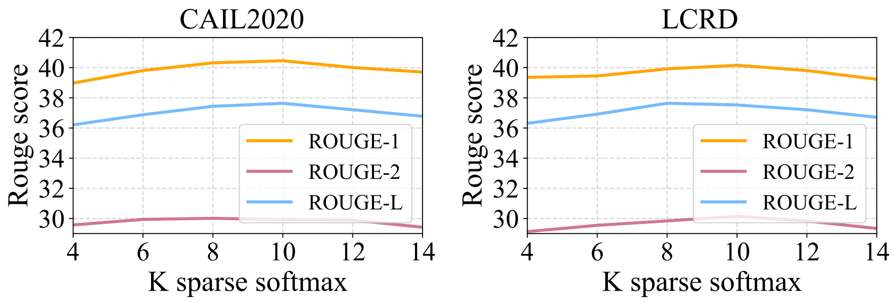


**Figure 6.** Experimental results for *k* sparse softmax.

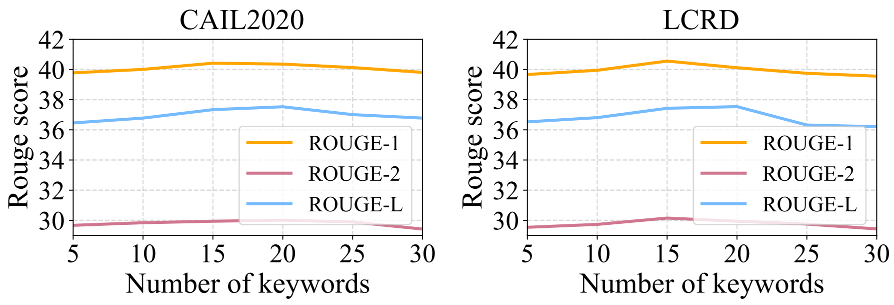


**Figure 7.** Experimental results for the number of keywords.

### 4.8. Uncertainty Analysis

This research aimed to develop a legal judgment summary model to summarize a given legal judgment's critical points effectively. Although our model achieved encouraging results on several evaluation indicators, several uncertainties had to be considered: Firstly, the uncertainty of the training data. Although we used a large corpus of legal judgments to train our model, the representativeness and relevance of these judgments could vary, and they may not have fully covered the range of data required for legal judgment summaries, which could also have affected the performance of our model. Secondly, the uncertainty associated with the model. For different types of legal judgments, our model may not have been able to capture all the nuances of legal judgments effectively or could have been biased towards certain types of legal judgments.

To reduce these uncertainties in future research, we plan to explore the quality and relevance of datasets further and evaluate the effectiveness of training with more diverse and representative datasets. In addition, we plan to explore alternative modeling approaches that could better capture the nuances of legal judgments and evaluate the impact of different modeling choices on summarization performance.

### 4.9. Complexity Analysis

The proposed legal judgment summarization model was based on the advanced transformer architecture proven effective for natural-language-processing tasks. However, such architectures are relatively complex and require significant computing resources to be trained and used effectively. We analyzed the primary sources of model complexity: first, the size of the model. Our model had to learn a large number of parameters during training, which could lead to long training times and high memory requirements. To address this complexity, we explored the use of computationally efficient techniques in our model, such as using sparse attention or pruning techniques to reduce the number of parameters. The second source of model complexity was the vast number of different legal domains. The model had to be trained on large amounts of labeled data to perform well, but not all of the data pertained to all legal judgment documents. Therefore, the model required extensive hyper-parameter tuning for different legal domains. To address this, we plan to explore techniques such as transfer learning. Third, the complexity of model evaluation was another source of model complexity. Our model required massive computing resources to train, and evaluating its performance on large datasets was time-consuming. To address this complexity, we plan to explore more efficient evaluation techniques.

In conclusion, while our legal judgment summarization model showed promising results, its complexity posed challenges for practical deployment. By exploring methods such as model compression, transfer learning, and optimized evaluation, we aim to reduce this complexity and enable more efficient and effective legal judgment summarization in the future.

## 5. Conclusions

In this study, we proposed a two-stage legal judgment summarization model that was used to comprehensively capture important and relevant information from lengthy legal texts. We treated legal judgments as a sequence of sentences and selected key sentence sets from the full texts as the input corpus for the abstraction summarization. We extracted keywords related to professional terms and specific topics in the judgment texts and introduced them into the summary-generation model as an attention mechanism so that these keywords could be integrated into the final generated summary. We confirmed the validity of our model on a variety of datasets and conducted a comprehensive evaluation to demonstrate the importance of each component of the presented model.

**Author Contributions:** Funding acquisition, L.S.; Methodology, Y.H. and L.S.; Resources, Y.H., L.S., C.H. and J.G.; Writing—original draft, Y.H.; Writing—review and editing, Y.H., L.S., C.H. and J.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the National Natural Science Foundation of China (No. 61873131, No. 61872194, No. 61902237), and the "Sailing Program" of Young Science and Technology Talents, supported by Shanghai Science and Technology Commission (19YF1418200).

**Data Availability Statement:** Restrictions apply to the availability of part of the data. Two datasets were used in this study. One was obtained from China Legal Research Cup and is available from https://www.kaggle.com/datasets/weipengfei/sfzy-small (accessed on 26 January 2023) with the permission of Pengfei Wei. Another is available in a publicly accessible repository that does not issue any DOI. This data can be found here: https://tianchi.aliyun.com/dataset/88798 (accessed on 24 January 2023).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Saravanan, M.; Ravindran, B.; Raman, S. Improving legal document summarization using graphical models. *Front. Artif. Intell. Appl.* **2006**, *152*, 51.
2. Saravanan, M.; Ravindran, B.; Raman, S. Automatic identification of rhetorical roles using conditional random fields for legal document summarization. In Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I, Hyderabad, India, 7–12 January 2008 .
3. Erkan, G.; Radev, D.R. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.* **2004**, *22*, 457–479. [CrossRef]
4. Rush, A.M.; Chopra, S.; Weston, J. A neural attention model for abstractive sentence summarization. *arXiv* **2015**, arXiv:1509.00685.
5. See, A.; Liu, P.J.; Manning, C.D. Get To The Point: Summarization with Pointer-Generator Networks. *arXiv* **2017**, arXiv:1704.04368.
6. Xu, H.; Savelka, J.; Ashley, K.D. Accounting for sentence position and legal domain sentence embedding in learning to classify case sentences. In *Legal Knowledge and Information Systems*; IOS Press: Amsterdam, The Netherlands, 2021; pp. 33–42.
7. Cohan, A.; Dernoncourt, F.; Kim, D.S.; Bui, T.; Kim, S.; Chang, W.; Goharian, N. A discourse-aware attention model for abstractive summarization of long documents. *arXiv* **2018**, arXiv:1804.05685.
8. de Vargas Feijo, D.; Moreira, V.P. Improving abstractive summarization of legal rulings through textual entailment. *Artif. Intell. Law* **2021**, *31*, 91–113 . [CrossRef]
9. Turtle, H. Text retrieval in the legal world. *Artif. Intell. Law* **1995**, *3*, 5–54. [CrossRef]

10. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified language model pre-training for natural language understanding and generation. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019 ; Volume 32.
11. Ethayarajh, K. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. *arXiv* **2019**, arXiv:1909.00512.
12. Tenney, I.; Das, D.; Pavlick, E. BERT rediscovers the classical NLP pipeline. *arXiv* **2019**, arXiv:1905.05950.
13. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
14. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **2019**, arXiv:1909.11942.
15. Zhang, X.; Lapata, M.; Wei, F.; Zhou, M. Neural latent extractive document summarization. *arXiv* **2018**, arXiv:1808.07187.
16. Liu, Y.; Titov, I.; Lapata, M. Single document summarization as tree induction. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 1745–1755.
17. Narayan, S.; Cohen, S.B.; Lapata, M. Ranking sentences for extractive summarization with reinforcement learning. *arXiv* **2018**, arXiv:1802.08636.
18. Nallapati, R.; Zhai, F.; Zhou, B. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
19. Paulus, R.; Xiong, C.; Socher, R. A deep reinforced model for abstractive summarization. *arXiv* **2017**, arXiv:1705.04304.
20. Narayan, S.; Cohen, S.B.; Lapata, M. Do not give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv* **2018**, arXiv:1808.08745.
21. Zhang, X.; Wei, F.; Zhou, M. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. *arXiv* **2019**, arXiv:1905.06566.
22. Zhang, H.; Xu, J.; Wang, J. Pretraining-based natural language generation for text summarization. *arXiv* **2019**, arXiv:1902.09243.
23. Liu, Y.; Lapata, M. Text summarization with pretrained encoders. *arXiv* **2019**, arXiv:1908.08345.
24. Wang, Q.; Liu, P.; Zhu, Z.; Yin, H.; Zhang, Q.; Zhang, L. A text abstraction summary model based on BERT word embedding and reinforcement learning. *Appl. Sci.* **2019**, *9*, 4701. [CrossRef]
25. Kim, Y. Convolutional Neural Networks for Sentence Classification. *arXiv* **2014**, arXiv:1408.5882.
26. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2018.
27. Chen, M.X.; Firat, O.; Bapna, A.; Johnson, M.; Macherey, W.; Foster, G.; Jones, L.; Parmar, N.; Schuster, M.; Chen, Z.; et al. The best of both worlds: Combining recent advances in neural machine translation. *arXiv* **2018**, arXiv:1804.09849.
28. Joachims, T. *A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization*; Technical report; Carnegie-Mellon University Pittsburgh pa Department of Computer Science: Pittsburgh, PA, USA, 1996.
29. Deng, Z.; Ma, F.; Lan, R.; Huang, W.; Luo, X. A two-stage Chinese text summarization algorithm using keyword information and adversarial learning. *Neurocomputing* **2021**, *425*, 117–126. [CrossRef]
30. Cui, Y.; Che, W.; Liu, T.; Qin, B.; Yang, Z. Pre-training with whole word masking for chinese bert. *IEEE/ACM Trans. Audio, Speech, Lang. Process.* **2021**, *29*, 3504–3514. [CrossRef]
31. Su, J. SPACES: An Extraction-Generation Summarization In Long Text (The summary of the "Legal AI Challenge"). 2021. Available online: https://spaces.ac.cn/archives/8046 (accessed 1 January 2021). (In Chinese)
32. Galgani, F.; Compton, P.; Hoffmann, A. Citation Based Summarisation of Legal Texts. In Proceedings of the PRICAI 2012, Kuching, Malaysia, 3–7 September 2012. Springer: Berlin/Heidelberg, Germany, 2012; Volume LNCS 7458, pp. 40–52.
33. Su, J. bert4keras. 2020. Available online: https://bert4keras.spaces.ac.cn (accessed on 25 January 2023 ).
34. Mihalcea, R.; Tarau, P. Textrank: Bringing order into text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–29 October 2004; pp. 404–411.
35. Lin, C.Y. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*; Association for Computational Linguistics: Barcelona, Spain, 2004; pp. 74–81.