Artificial Intelligence and Law
https://doi.org/10.1007/s10506-023-09377-4

ORIGINAL RESEARCH

Integrating legal event and context information for Chinese similar case analysis

Jingpei Dan[1] · **Lanlin Xu**[1] · **Yuming Wang**[2]

Accepted: 28 September 2023

Abstract

Similar case analysis (SCA) is an essential topic in legal artificial intelligence, serving as a reference for legal professionals. Most existing works treat SCA as a traditional text classification task and ignore some important legal elements that affect the verdict and case similarity, like legal events, and thus are easily misled by semantic structure. To address this issue, we propose a Legal Event-Context Model named LECM to improve the accuracy and interpretability of SCA based on Chinese legal corpus. The event-context integration mechanism, which is an essential component of the LECM, is proposed to integrate the legal event and context information based on the attention mechanism, enabling legal events to be associated with their corresponding relevant contexts. We introduce an event detection module to obtain the legal event information, which is pre-trained on a legal event detection dataset to avoid labeling events manually. We conduct extensive experiments on two SCA tasks, i.e., similar case matching (SCM) and similar case retrieval (SCR). Compared with baseline models, LECM is validated by about 13% and 11% average improvement in terms of mean average precision and accuracy respectively, for SCR and SCM tasks. These results indicate that LECM effectively utilizes event-context knowledge to enhance SCA performance and its potential application in various legal document analysis tasks.

Keywords Natural language processing · Legal artificial intelligence · Similar case analysis · Event detection

✉ Jingpei Dan
danjingpei@cqu.edu.cn

✉ Yuming Wang
ymwang@mail.hust.edu.cn

Lanlin Xu
xulanlin@cqu.edu.cn

[1] College of Computer Science, Chongqing University, Chongqing, China

[2] School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, Hubei, China

Published online: 25 October 2023

1 Introduction

Similar case analysis (SCA) aims to perform semantic analysis on similar legal cases and find similar cases. SCA has two main tasks: similar case matching (SCM) and similar case retrieval (SCR). SCM aims to determine whether legal case documents are similar, and SCR aims to find cases similar to the target case from candidate cases and sort them by similarity. SCA plays a significant role in the legal domain. In common law systems, like in the United States, Canada, and India, case law dominates, which means the judgments are made according to similar and representative cases in the past (Zhong et al. 2020). Although statutes are the primary in civil law systems, like in China, Germany, and Italy, similar cases still serve as references for legal professionals. As a large number of legal documents are generated and accumulated, how to retrieve similar cases efficiently from vast amounts of legal document data is a big challenge.

Theoretically, the SCR task can be solved using a collaborative-based approach, such as collaborative filtering, thus avoiding the similarity calculation. However, it is difficult to model the new cases due to the lack of user-item interactions in the recommendation scenario (Yang et al. 2022). Therefore, in the SCR task, we need to utilize the semantic information of the text to model the similarity and complete the case retrieval task. SCM aims to determine whether legal case documents are similar or not, which is based on similarity calculation. To sum up, the core problem of solving SCA is to calculate the similarity between legal documents.

With the development of deep learning in natural language processing (NLP), exploiting NLP techniques to assist legal tasks has drawn increasing attention rapidly. SCA is a crucial component of legal tasks, and as a result, there has been a lot of research work that has explored the possibility of NLP and SCA. For example, Mandal et al. (2021) convert legal documents into embedding vectors and calculate the text similarity between the embedding vectors. Wehnert et al. (2021) combine BERT word embeddings with TF-IDF vectors to enrich the document representations. Wu et al. (2021) improve search effectiveness by matching judgments to queries at the semantics level rather than at the keyword level. Shao et al. (2020) and Ma et al. (2021a) propose a BERT model based on paragraph-level semantic information. There have been several benchmark efforts in the field of legal artificial intelligence, including SCA, such as CAIL (Xiao et al. 2019), Legal TREC (Oard and Webber 2013), AILA (Bhattacharya et al. 2019) and COLIEE (Rabelo et al. 2020a). In the context of Chinese corpus, CAIL has released benchmarks for Chinese legal tasks, such as CAIL-2019 (Xiao et al. 2019) and LeCaRD (Ma et al. 2021b). These datasets have served as a crucial foundation for research in Chinese SCA, such as Lawformer (Xiao et al. 2021) and LFESM (Hong et al. 2020). However, it is still confronted with the following challenges in the Chinese SCA task:

Challenges 1: Similarity in semantic structure is not equivalent to case similarity. The existing methods have primarily focused on semantic structures, neglecting the significance of legal elements that can impact both the verdict and

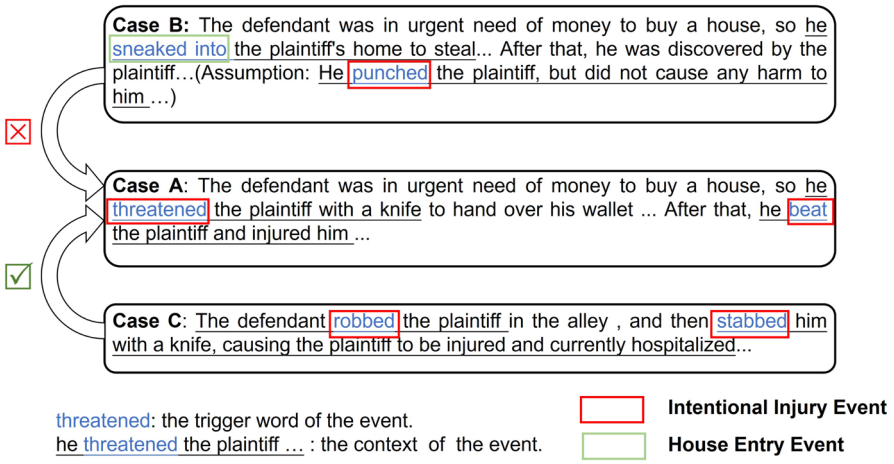

Fig. 1 An illustration of similar case matching. A and B are more similar in semantic structure, but in terms of the case, A and C are more similar because their events and the severity of events are more similar

the similarity between cases. This includes crucial elements like legal events. Taking the SCM task as an example, as depicted in Fig. 1, let's temporarily set aside the assumptions in Case B. While the fact statements of Case A and Case B may be semantically similar, and they are not actually similar because Case A involves violence events while Case B does not. Consequently, Case B should be classified as theft, whereas Case A should be classified as robbery. Traditional SCA methods relying solely on semantic similarity can be easily misled by semantic structures and often wrongly determine that Case A is more similar to Case B, when in fact the ground truth is that Case A is more similar to Case C. However, if we extract and compare the violent incidents in Case A, Case B, and Case C, it becomes evident that the correct conclusion can be reached. Figure 1 illustrates that the events of being threatened and beaten in Case A correspond to intentional injury events, which correspond to being robbed and stabbed in Case C. However, there are no such events in Case B. By incorporating such improvements, we can address the limitations of traditional language models that overly rely on semantic similarity.

Some researchers have solved this problem by extracting legal elements via human design. Hong et al. (2020) leverage regular expression to incorporate legal elements into text parsing. Hu et al. (2018) add attributes for charges manually in legal judgment prediction. However, manual rule-based approaches heavily rely on domain-specific prior knowledge and require significant human effort, which can be inefficient. Additionally, extracting legal elements as a single entity can lead to challenging problems. As shown in Fig. 1, if the assumption content is added to Case A, violent events would also be present. At this point, if similarity is solely judged based on the event sequence, it would fall into the semantic similarity trap. In Case A, although a violent incident occurred, its severity was much lower than in Cases B and C. Therefore, when considering the events that occur in a case, it is essential to

take into account not only the sequence of events but also the context in which they occur.

Overall, when it comes to legal judgments, events are crucial in determining case similarity for several reasons. Firstly, events directly impact case outcomes and significantly influence the legal analysis and decision-making process. Focusing on events allows us to capture the key actions and incidents that shape the case outcome. Secondly, events provide contextual relevance by reflecting the environment and background of the case, helping to identify similarities and differences. Furthermore, events serve as objective and identifiable elements that can be documented, analyzed, and compared, facilitating a systematic and consistent approach to case analysis. Gathering comprehensive information about events is often more feasible than quantifying complex legal elements or abstract concepts. However, it is important to note that other legal elements may hold greater importance in specific fields of law, which can be explored in our future work.

Besides, legal documents are professional and must contain many common language structures, and these parts will cause certain interference for SCA analysis. For example, a civil case document will often contain the following information:

- Personal information of plaintiff and defendant.
- Description of the facts of the case and the plaintiff's claims.
- The analysis of the court based on the factual description.

The above information usually tends to be mixed in an actual legal document. However, only the fact description is the key to SCA, and the rest parts would interfere with the SCA analysis. Moreover, many professional terminologies are used in legal documents, such *as found through trial*, *the focus of the dispute in this case*. These terminologies may interfere with similarity calculation since they rarely contain the key features required by similarity calculation and are challenging to be removed by data cleaning.

Challenge 2: Combine multiple datasets for training. The existing methods usually train their models only on the dataset of specific tasks, such as SCA. Thus, the models cannot utilize the knowledge from another dataset, like the event detection (ED) dataset. For example, to make the model have the ability of SCA, we usually train our model on a dedicated SCA dataset, such as (Xiao et al. 2019). However, as mentioned above, the event labels, which are important for SCM, are not involved in the existing SCM dataset. Thus, if we want to perform multi-task training of SCM and ED, we need to manually label events on the SCM dataset, which takes much time. Therefore, it remains an unsolved problem to leverage the existing event detection dataset, like LEVEN (Yao et al. 2022), to assist the SCA tasks.

Challenge 3: Properly integrating event information and text semantic features is a big challenge. ED is an information extraction task which can effectively capture the sequence of events contained in the text. It aims to automatically extract the event triggers from text and then classify their corresponding event types, which has been formalized as a sequence labeling task. Chinese characteristic law systems divide a legal criminal case into the event sequences and the penalties corresponding to the event (Feng et al. 2022). There is a clear causal relationship between incidents

and penalties. When an event is detected, a penalty must be imposed. Therefore, events play a crucial role in the penalty system, and judgment forms the basis for assessing the similarity of cases. While the concept of constituent elements is commonly associated with criminal law, the comparative analysis of case facts and legal provisions is universally applicable in legal practice. This analysis extends to civil cases as well, where comparing facts between resolved cases and ongoing cases is an indispensable component. Even without the requirement for criminal punishment, events can provide valuable insights and help establish the sequence of actions, identify responsibilities, and determine liability in civil disputes. This practice is also widespread in common law jurisdictions.

There have been many studies on ED. Li et al. (2021) propose a method that consists of a semantic feature extractor, a statistical feature extractor and a joint event discriminator to avoid being confused by the varied contexts. Si et al. (2022) introduce the prompt-based learning strategy to the domain of ED. Although the field of ED is developing rapidly, most of the current research is based on public datasets such as the common domain ED dataset ACE2005,[1] and has not explored the downstream tasks of ED. Moreover, just locating events is not enough to support the judgments of similar cases. For example, the same event with different contexts will reflect different information, like the severity of the event. Therefore, integrating both events and their corresponding contexts for judging similar cases may be a better choice.

To summarize, this study addressed the following research questions (*RQs*):

RQ1: How to introduce events into SCA as legal elements instead of just considering semantic similarity at semantic level?

RQ2: How to combine multiple legal datasets for joint training to leverage knowledge from other datasets?

To address these issues, we propose a legal event-context model named LECM for SCA tasks.

Firstly, to integrate the event and context information, an event-context integration mechanism is proposed to formalize the events and their context semantic features based on the attention mechanism. By highlighting the contextual features related to events, it helps alienates the features of the same event in different contexts and reduce the impact of legal terminology, narrative structure and other features on similarity calculation. Based on this event-context integration mechanism, the proposed LECM model for SCA can leverage semantic and event features for inference, thus improving accuracy and interpretability. Then, to help the LECM model locate key information of events, we use ED as an auxiliary task for the SCA tasks. Specially, an ED module is pre-trained on an ED dataset to locate event types and locations. As a bridge between the ED task and the SCA tasks, the ED module can improve efficiency by avoiding labeling event annotation manually for SCA tasks. Finally, the event information obtained by the ED module and related intermediate layer features will be used for subsequent similarity calculations for more accurate SCA. We conduct experiments on some real-world SCA datasets to investigate the

[1] http://projects.ldc.upenn.edu/ace/.

effectiveness of our model. The experiment results show that our method outperforms the competitive baselines. Specifically, LECM achieves the highest performance in precision and accuracy.

The main contributions of this paper can be summarized as follows:

(1) We propose a novel legal event-context model named LECM with three characteristics: (1) can improve the accuracy and interpretability of SCA by detecting events and extracting event context features based on the proposed event-context integration mechanism. (2) Can help SCA task to locate event key information by integrating event detection; (3) Can improve efficiency by utilizing a pre-trained ED module instead of labeling events manually for the target dataset, like SCA datasets in this paper.

(2) To evaluate the proposed LECM model, we conduct extensive experiments on two tasks of SCA, i.e., SCM and SCR. Comparing the competitive baselines, LECM achieves the highest performance in precision and accuracy. The experiments show that LECM yields substantial improvements in SCA tasks. Further ablation tests and the case study demonstrate the effectiveness of our method.

The rest of the paper is structured as follows: the related work on SCM and ED is introduced in Sect. 2. Section 3 elaborates on the problem definition and the proposed model. Experiment settings and results are discussed in Sect. 4. Finally, we conclude our work in Sect. 5.

2 Related works

2.1 Similar case analysis

SCA is an essential topic in legal artificial intelligence, consisting of SCM and SCR. SCM aims to measure the similarity between legal case documents, which is a particular form of semantic matching. SCR aims to find cases similar to the target case from candidate cases and sort them by similarity, an information retrieval task.

There are two broad approaches for SCA task: graph-based methods and semantic-based methods. The graph-based methods (Minocha et al. 2015; Bhattacharya et al. 2020; Bi et al. 2022; Yang et al. 2022) aim to construct a graph neural network based on the existing correlation information between cases, and uses the similarity of nodes to represent the similarity of cases. However, it is not easy to model the new cases due to a lack of user-item interactions. Therefore, we mainly consider semantic-based methods in this paper.

Traditional semantic-based methods for SCA tasks often rely on bag-of-words models, such as TF-IDF(Salton and Buckley 1988), BM25(Robertson and Walker 1994), and LMIR(Ponte and Croft 2017), which prioritize term-level similarities using statistical models. Traditional methods capture key features of text by comparing the frequency or weight of words and have achieved good results in certain tasks. In the study conducted by (Souza et al. 2021) on Brazilian legal document retrieval, various variants of the BM25 algorithm and language models were compared. The study demonstrated the effectiveness of the bag-of-words models in SCA tasks, highlighting its excellent performance in the legal domain.

Mandal et al. (2017) perform extensive experiments on a large dataset of Indian Supreme Court cases to compare various methodologies (TF-IDF, topic modelling, neural network) for measuring the textual similarity of legal documents. Although traditional methods have achieved significant research results, they may encounter certain challenges when dealing with complex texts, such as legal documents. Some of these challenges include high dimensionality and inaccurate context capture (Kusner et al. 2015; Zhao and Mao 2018; Ali et al. 2019). More recently, deep learning has been widely used in semantic matching. Based on the idea of representation learning, researchers (Wang et al. 2017; Jiang et al. 2019) began using latent space vectors of texts based on deep learning models, and the similarity score between texts is calculated based on their latent space vectors. Pre-trained language models, which are trained on unlabeled corpora, have been proven to benefit various NLP downstream tasks (Choi et al. 2020; Röttger and Pierrehumbert 2021).

Researchers from various countries have made significant contributions to the field of SCA tasks using deep learning and neural networks. Bench-Capon et al. (2012) explore different statistical methods, learning techniques, logical analysis, and expert knowledge in this area. Saravanan et al. (2009) propose an ontological framework to improve user queries for retrieving truly relevant legal judgments. Liu et al. (2022) apply a conversational agent workflow, originally designed for web search, to legal case retrieval. Furthermore, Opijnen and Santos (2017) identify several limitations of general information retrieval methods in the legal domain and propose a unique framework with six dimensions to capture the concept of relevance in legal information retrieval. Shao et al. (2020) utilize Bert to capture the relationships at the paragraph-level and then aggregate the paragraph-level representations to infer the relevance between two legal cases. Rabelo et al. (2019) apply a transformer-based technique to tackle identifying entailment relationships between a decision and candidate entailing paragraph.

Several approaches have been explored in previous research of SCA. A number of benchmarks have been published, such as CAIL(Xiao et al. 2019), Legal TREC(Oard and Webber 2013), AILA(Bhattacharya et al. 2019), and COLIEE(Rabelo et al. 2020a). In the previous COLIEE competition, a BERT-based language model, Legal-BERT(Chalkidis et al. 2020) is presented, which is pre-trained on a collection of several fields of English legal text. It is mentioned that Kim et al. (2017) introduce judicial document retrieval as an upstream task in the judicial question answering task, and achieved excellent retrieval performance through Siamese networks. Furthermore, in COLIEE 2020 (Rabelo et al. 2020b), a combination of the universal sentence encoder, TF-IDF, and a support vector machine has been proposed, which achieved a good performance for the case law retrieval task.

The SCA task based on the Chinese corpus has also attracted a lot of attention. Xiao et al. (2021) propose Lawformer, a Longformer-based language model, which is pre-trained on large-scale Chinese legal documents. It is demonstrated that Lawformer achieves improvement in a variety of legal artificial intelligence tasks. Hong et al. (2020) leverage regular expressions to extract auxiliary information and combine the Siamese network architecture to complete the semantic analysis of legal cases.

2.2 Event detection

Event Detection (ED) is a crucial information retrieval task in the NLP field. ED aims to extract the event triggers from texts and then classify their corresponding event types.

Existing ED methods can be categorized into two classes, feature-based methods and representation-based methods. Early works mainly focus on feature-based methods (McClosky et al. 2011; Li et al. 2013). Recently, representation-based methods have raised more attention. Li et al. (2021) introduce word-event co-occurrence frequencies into ED, to reduce the impact of similar contexts on ED. Deng et al. (2021) define links for different events to improve the model's performance on rare events. Besides, some methods are also developed for the legal domain. Feng et al. (2022) manually label events and use them for downstream fine-tuning. Wang et al. (2019) apply adversarial training to the task of event detection and employ dynamic pooling layers to obtain trigger-specific representation for each candidate. Researchers also propose several deep neural network for legal documents. Chen et al. (2020) extract the entities and the semantic relations for drug-related legal documents. Devlin et al. (2019) propose a document-level event-argument link method. HGEED (Lv et al. 2021) introduce a document graph to model sentence-to-sentence dependencies. For Chinese legal text ED tasks, a BiLSTM-CRF-based ED model (Li et al. 2020) is proposed. Shen et al. (2020) propose a pedal attention mechanism to extract semantic relations in long-distance. Li et al. (2019)present a mechanism to define focus events and a two-level labeling approach to automatically extract focus events from case materials. Similar to other specific domains, a legal ED dataset (Yao et al. 2022) is developed.

Despite the great success of ED, few studies have explored the downstream tasks of event detection in the legal domain. Event is an important feature of legal case documents, and it is an important basis for inferring the relevance between legal cases. Therefore, we take SCA as a downstream task of ED in this paper. However, since most of the current ED methods lead to excessive computational complexity, we adopt the BERT + CRF (Lafferty et al. 2001; Devlin et al. 2019) method to reduce the computational cost.

3 Method

In this section, we will elaborate on the proposed LECM in detail. First, we give the definition of the SCA tasks. Then, we give an overview of LECM as shown in Fig. 2 and describe the details of each component respectively. Notably, to adapt to different SCA tasks, some minor changes are required for LECM, as we will describe in detail.

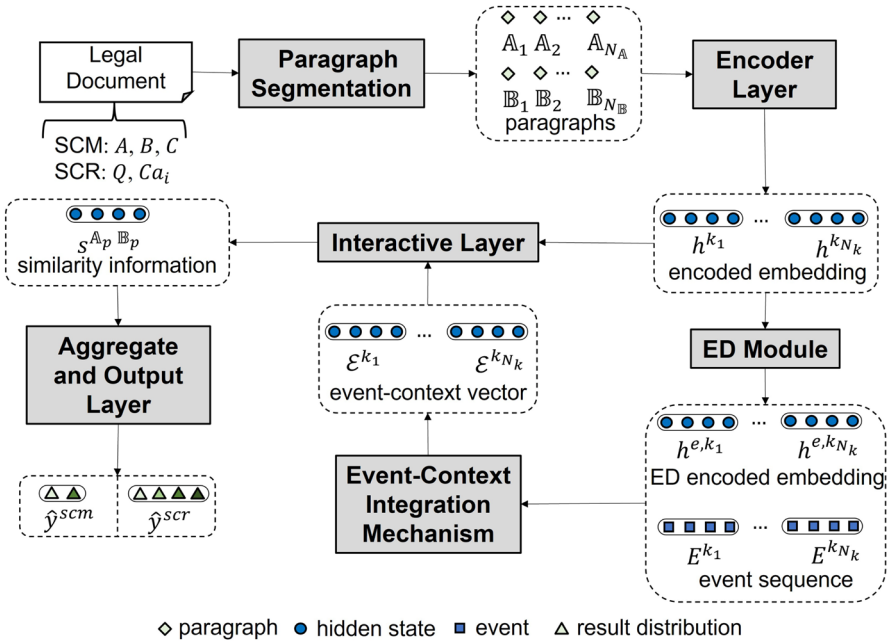

Fig. 2 The framework of LECM. $k \in \{\mathbb{A}_p, \mathbb{B}_p\}$ here

3.1 Problem definition

We evaluate the capability of the model for SCA through two specific tasks:

SCM: SCM aims to measure the similarity among legal documents and select the case most similar to the target case. In this paper, for simplicity, the input of SCM is supposed to be a triplet. To be specific, for a given triplet (A, B, C), case A, case B, and case C represent the different legal case fact descriptions. We use word sequences to denote the triplet: $A = \left[w_1^a, w_2^a, \ldots, w_{l_a}^a\right], B = [w_1^b, w_2^b, \ldots, w_{l_b}^b]$ and $C = [w_1^c, w_2^c, \ldots, w_{l_c}^c]$, where l_j is the length of word sequence, $w_j^i \in V$ denotes a word, and V is the pre-set fixed vocabulary. The SCM task can be represented as predicting the label $y_{scm} \in \{0, 1\}$, where $y = 1$ indicates that the similarity between A and B (denoted as $sim_{A,B}$) is less than the similarity between A and C (denoted as $sim_{A,C}$). Conversely, y = 0 indicates that the similarity between A and B is greater than the similarity between A and C. LECM will output probability values for label 0 and label 1. Typically, the label with the higher probability value is considered as the model's final prediction.

SCR: Given a query case, SCR task is to retrieve relevant cases from a pool of candidate cases. Unlike general recommendation tasks, SCR focuses solely on text similarity since there is a lack of user-item information. In this study, we transform the SCR task into a query and candidate matching problem that involves calculating similarity. To be specific, given a query case Q and a set of candidate case $Ca = \{Ca_1, Ca_2, \ldots, Ca_N\}$, where $Q = \left[w_1^Q, w_2^Q, \ldots, w_{l_q}^Q\right], Ca_i = \left[w_1^c, w_2^c, \ldots, w_{l_{ca}}^c\right]$. During the training phase, SCR

aims to calculate the similarity between the query and candidates, denoted as$y_{scr} = \{sim_{Ca_1,Q}, \ldots, sim_{Ca_N,Q}\}$. For a given query, after calculating the similarity for all candidate cases, we sort them based on their similarity scores and evaluate the model's performance based on this ranking.

Since the task forms of SCM and SCR are similar, LECM can be applied to both tasks with only a few modifications. Therefore, for the sake of brevity, unless otherwise specified, in SCM task, we use $\mathbb{A}$ to represent the case A and use $\mathbb{B}$ to represent the case B. In SCR task, we use $\mathbb{A}$ to represent query Q and use $\mathbb{B}$ to represent the candidate Ca_i.

In addition to the target task, we utilize ED as an auxiliary task, so we give the problem definition of ED here:

ED: Given a token sequence $= [x_1, x_2, \ldots, x_l]$, where l is the maximum number of tokens and x_i is the i-th token. ED needs to first identify the trigger word and then determine the corresponding event type. A trigger word refers to a keyword or phrase that initiates a specific event. These words are responsible for causing or triggering specific events in the text. Any tokens in the statement that do not qualify as trigger words are classified as non-trigger words. By identifying these trigger words, the occurrence of an event can be determined. Usually, these trigger words are associated with specific event types. Initially, the model needs to determine whether x_i is a trigger word. If it is, the model predicts the event type e_i for it, where$e_i \in \{Event_0, Event_1, \ldots, Event_N\}$, and E_i represents a specific event type while N is the total number of event types. In this paper, for all non-trigger words that do not themselves represent any type of event, we still define them as a class of events denoted as$Event_0$.

3.2 LECM model overview

The proposed legal event-context model learns to extract the representation of the event, and the fact description representation, which could be applied to the downstream task, such as SCA task. The architecture of LECM is shown in Fig. 2.

In LECM, the ED module is proposed to capture the information about legal events, which is firstly pre-trained on the LEVEN dataset (Yao et al. 2022) that is an auxiliary dataset in this paper to assist the model in downstream tasks. After that, the ED module will be a part of the LECM model to complete SCA tasks jointly. In the encoder layer, words are mapped to continuous vectors. We use the BERT (Devlin et al. 2019) to obtain the contextual representation of legal fact description. Inspired by BERT-PLI (Shao et al. 2020), the document is segmented into paragraphs and encode each paragraph separately. Since the number of paragraphs is variable, the model should be able to handle long or short legal documents. Then, in the event-context integration mechanism, the pre-trained ED module is utilized to extract the context features of the event. More specifically, the attention weights are calculated in a specific range of contexts based on the embedding representation of the event and the hidden layer vector of the ED module. Next, the interactive layer will capture the interactive semantic information between paragraphs based on the original semantics of paragraphs and

event-context information. Finally, the aggregate and output layer are adopted to aggregate the paragraph-level features and predict the final results of SCA. For the SCM task, the similarity between case A and case B, case A and case C will be output here, while for the SCR task, this layer will output the similarity between the query document and the candidate document.

3.3 Detail of LECM model

3.3.1 Paragraph segmentation

Most of the text in the SCA datasets exceeds the maximum input length of BERT, and truncating the text will result in information loss. To tackle this challenge, the legal document is segmented into paragraphs, and the interactive features are modeled at the paragraph-level, then the paragraph-level features will be aggregated in the aggregate and output layer. Since the input forms of SCM task and SCR task are different, the two tasks are processed slightly differently at this layer. To be specific, for SCM task, we first break the triplet into paragraphs and the length of each paragraph is the maximum input length of BERT:

$$A = \left[A_1, A_2, \ldots, A_{N_A}\right] \tag{1}$$

$$B = \left[B_1, B_2, \ldots, B_{N_B}\right] \tag{2}$$

$$C = \left[C_1, C_2, \ldots, C_{N_C}\right] \tag{3}$$

where N_i is the total number of paragraphs. For the SCR task, we break query case Q and candidate case Ca_i into paragraphs, similar to the SCM task:

$$Q = \left[Q_1, Q_2, \ldots, Q_{N_Q}\right] \tag{4}$$

$$Ca_i = \left[Ca_{i,1}, Ca_{i,2}, \ldots, Ca_{i,N_C}\right] \tag{5}$$

The following core work is to model the interaction features between the paragraphs. For the SCM task, it is to calculate the interaction characteristics between A and B, or A and C. For the SCR task, it is to calculate the interaction characteristics between Q and Ca_i. The procedure of these two tasks is the same until the output step. Therefore, to make the expression more concise, we use $\mathbb{A}$ and $\mathbb{B}$ to represent the case pair and $\mathbb{A}_p$ and $\mathbb{B}_p$ to represent the paragraphs in the two tasks that need to model the interaction features in the following parts. Any pair of paragraphs in $\mathbb{A}$ and $\mathbb{B}$ will feed into the ED module, the encoding step and the interactive information calculation step.

3.3.2 ED module

ED aims to predict the event label e_i on each individual token, taking into account the context and potential variations within each statement. In this paper, we consider events as legal elements that play an important role in subsequent SCA tasks. Although there are many successful ED models, using it as an upstream task will lead to the excessive computational complexity of LECM. Taking DMBERT (Wang et al. 2019) as an example, the input of this method needs to specify the position of the token to be predicted in the sentence. If there are m sentences and each sentence contains n tokens, then the time complexity after predicting all events is $O(mn)$. Since our model needs to complete downstream tasks on the basis of ED, such time complexity is unacceptable. Therefore, we chose BERT + CRF, a low time complexity ED model. It performs the ED task on m sentences, and the time complexity is only $O(m)$, independent of the specific length of text.

The ED module is pre-trained on the ED task before training LECM on the SCA task so that the event information can be leveraged by LECM in SCA task. Formally, denoting an input sequence $k = \left[w_1^{ed}, w_2^{ed}, \ldots, w_{l_e}^{ed}\right]$, ED aims to predict the event label e_i on w_i^{ed}.

Considering pre-trained language model has been proven to benefit various NLP downstream tasks (Devlin et al. 2019; Choi et al. 2020; Röttger and Pierrehumbert 2021), we employ BERT, a general pre-trained language model, as our basic encoder in the ED module to generate the embeddings of each token dynamically. Since all the legal documents of the datasets in this work are written in Simplified Chinese, the OpenCLap (Zhong et al. 2019) model is adopted as our BERT model, a pre-trained BERT model based on a large legal Chinese corpus.

In the encoder of the ED module, BERT learns the representation of the legal text as follows:

$$h^{ed,k} = BERT(k) \in \mathbb{R}^{l_{ed} \times d_s} \tag{6}$$

where $h^{ed,k}$ represents the embedded representation of paragraph $k \in \{\mathbb{A}_p, \mathbb{B}_p\}$ encoded by BERT and d_s is the size of hidden states generated by BERT. In this way, the legal knowledge from the pre-trained corpus is brought into the text embedding $h^{ed,k}$. Then, we employ a fully-connected layer to make the final prediction of ED task:

$$\hat{y}^{ed} = \sigma\left(W^{ed} h^{ed,k} + b^{ed}\right) \tag{7}$$

where W^{ed} and b^{ed} are the parameter of the linear transformation, σ is the nonlinear activation function, $\hat{y}^{ed}$ is the probability distribution predicted by the ED module, which can be specifically expressed as $\hat{y}^{ed} = [\hat{y}_1^{ed}, \hat{y}_2^{ed}, \ldots, \hat{y}_{l_{ed}}^{ed}]$.

For the training procedure of ED module, referring to the work of (Lample et al. 2016), the loss function of the model is built based on CRF. To be specific, we assign one of the paths of $\hat{y}^{ed}$ to be $e = \left[e_1, e_2, \ldots, e_{l_e}\right]$, where $e \in E$ and E is the set of all possible paths. Then, we define the score of input text k and

prediction path E as the combination of the transition probability matrix and the emission probability matrix:

$$score(k, e) = \sum_{i=0}^{l_e} T_{e_i, e_{i+1}} + \sum_{i=0}^{l_e} F_{k, e_i} \tag{8}$$

where F is the emission matrix, F_{k,e_i} represents the score of event label e_i at the i-th position. T is the transition matrix and $T_{e_i,e_{i+1}}$ represents the transition matrix score from state e_i to state e_{i+1}. Given an input text k, the probability of an event label sequence E is:

$$p_{crf}(e|k) = softmax(score(k, e))\#(9) \tag{9}$$

The current most probable path e^* is calculated as:

$$e^* = argmax_{e \in E} p_{crf}(e|k) \tag{10}$$

The loss function is log-likelihood loss:

$$\mathcal{L}^{ed} = -log\big(p_{crf}(e|k)\big) \tag{11}$$

By employing BERT + CRF to complete the ED task, we fine-tune the BERT model on the ED task and obtain the prediction event sequence e^* of a given legal text k and the hidden state $h^{ed,k}$, which contains semantic features relevant to the ED task.

3.3.3 Encoder layer

The encoder layer maps the fact description of a case into continuous hidden states, which contain contextual features. Similar to the ED module, we apply the pre-trained BERT from OpenCLap (Zhong et al. 2019) to encode legal documents. Inspired by the Siamese network (Neculoiu et al. 2016), we design our encoder based on a shared-weight BERT to encode every paragraph, which is beneficial to reducing model parameters while fully considering the interaction information between different documents. Specifically, given a paragraph pair $\mathbb{A}_p$ and $\mathbb{B}_p$, a shared-weight BERT is used to capture contextual representations:

$$h^k = BERT(k) \in \mathbb{R}^{l_k \times d_s} \tag{12}$$

where $k \in \{\mathbb{A}_p, \mathbb{B}_p\}$.

3.3.4 Event-context integration mechanism

After encoding each paragraph, the ED module is utilized to capture the event features in this layer. Specifically, we propose the event-context integration mechanism to model the interaction between the events and the context. Our method is different from LFESM (Hong et al. 2020), which uses one-hot vectors to represent legal features. We treat events as legal features and map them into learnable vectors, integrating the contextual features of events with the semantic features of the original text.

First, we leverage the ED module to obtain the event features. We load the parameters of the pre-trained ED module. Then, $\mathbb{A}_p$ and $\mathbb{B}_p$ are fed into the ED module to obtain the event label sequences of cases as $E^k = \left[e_1^k, e_2^k, \ldots, e_{l_k}^k\right], k \in \{\mathbb{A}_p, \mathbb{B}_p\}$. To further extract the event information of fact description, we feed the event label E^k and the hidden states $h^{ed,k}$ of the ED module into the event-context integration layer. Notably, for efficiency consideration, the parameters of the ED module are frozen after pre-training. The effect of freezing the parameters will be discussed in detail in Sect. 4.5, corresponding to LECM/FT.

To map the event sequence E^k into a continuous vector space, a random lookup matrix *Emb* that stores embeddings of events is initialized. Here $Emb \in \mathbb{R}^{N_e \times d_s}$, N_e is the total number of event types. Before training begins, we randomly initialize EMB, where each event is assigned a randomly initialized embedding vector. Specifically, we set the embeddings of non-event labels to zero vectors to prevent interference with event fusion. Formally, the embedding $h^{e,k}$ of E^k is defined as:

$$h^{e,k} = Emb\left(E^k\right) \in \mathbb{R}^{l_k \times d_s} \tag{13}$$

where *Emb* will take out the vector of the corresponding column according to the index of E^k as the embedding vector of the event. $h^{e,k}$ is the embedding of each event in the sequence of events E^k, which can be represented as: $h^{e,k} = [h_1^{e,k}, h_2^{e,k}, \ldots, h_{l_k}^{e,k}]$. The lookup matrix *Emb* is updated through backward gradient propagation.

The second step is to capture the context features of events. If events are only used as features for interactive computation, the same event will still be represented as the same feature in different contexts. However, in fact, the context of the event contains the information related to the event (e.g., the severity of the event). Therefore, the features of events with their corresponding contextual features are integrated. Besides, the context related to the event is an important part affecting the similarity. Suppose there are no legal events in a piece of text. In that case, there is a high probability that this event is some general description (e.g., personal information of plaintiff and defendant) or has little to do with the whole legal case. As long as these parts of the text are not included in the event context, their impact on the similarity of the cases can be reduced. Based on the above assumptions, we propose the event-context integration mechanism.

The hidden states $h^{ed,k}$ of the ED module contain context semantic information related to ED. The interaction features between ED context semantic features $h^{ed,k}$ and SCA semantic features h^k are calculated by:

$$h^{es,k} = h^{ed,k} W^{es} h^k + b^{es} \tag{14}$$

where $W^{es} \in \mathbb{R}^{d_s \times l_k}$, $b^{es} \in \mathbb{R}^{d_s}$. $h^{ES,k}$ represents the integration of event features with SCA features, and it can also be expressed as $h^{es,k} = [h_1^{es,k}, h_2^{es,k}, \ldots, h_{l_k}^{es,k}]$. The symbol *es* identifies the vector related to this integration process, and its meaning is ***e**vent detection features integrating with **s**emantics*. Inspired by (Vaswani et al. 2017), we extract contextual features based on the attention mechanism. More specifically, the attention weights from the *i*-th position event to the *j*-th position token are represented as follows:

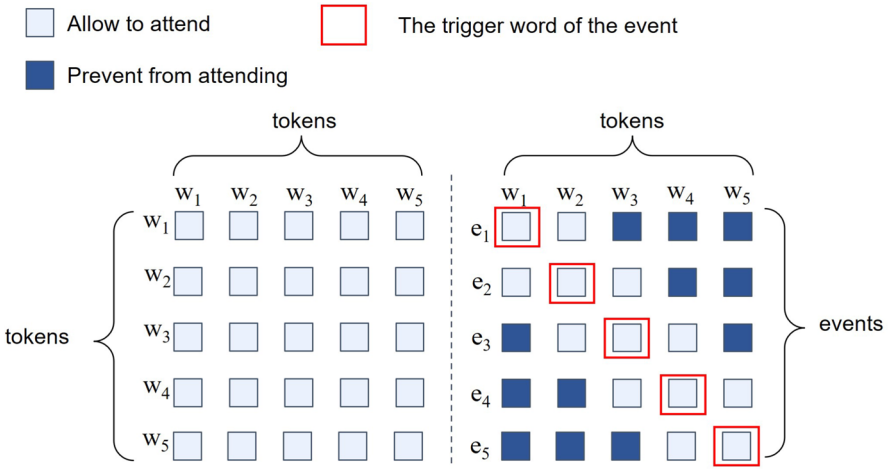

Fig. 3 The event-context integration mechanism. On the left is the standard self-attention mechanism. All the tokens from the document will be attended to. On the right is our event-context integration mechanism. We calculate the attention weights between events and tokens. The event only attends to the words close to the corresponding trigger words, and in the figure, the attention window size of the events is 2

$$\beth_{i,j}^{k} = softmax(\left(w_{i,j}^{e} h_{j}^{es,k}\right)^{T} \cdot h_{i}^{e,k} + m_{i,j}), \forall i,j \in \left[1,..,l_k\right] \cdot \tag{15}$$

$$m_{i,j} = \begin{cases} 0, allowtoattend \\ -\infty, preventtoattend \end{cases} \tag{16}$$

where $w_{i,j}^{e}$ is a learnable transformation parameter and $m_{i,j}$ controls the window size of event attention.

As Fig. 3 shows, the attention window of the event context depends on the trigger word position of the event. For the sake of logical clarity, the description of the event in the legal text usually appears around the trigger word. Thus, we set the center of the attention window as the trigger word position, and only tokens within the window will participate in the calculation of attention weights, corresponding to $m_{i,j} = 1$. If the token is outside the window, then it will not be noticed by the event, corresponding to $m_{i,j} = -\infty$. In this way, we can let the event only focus on the context related to it. Moreover, the text without events will lose the features of this part, making the model more focused on modelling the semantic features of the event-related context.

After that, the event-context vector is represented as $\mathcal{E}^k = [\mathcal{E}_1^k, \ldots, \mathcal{E}_{l_k}^k]$, where $\mathcal{E}_i^k$ is calculated by:

$$\mathcal{E}_i^k = \sum_{j=0}^{l_k} \beth_{i,j}^{k} h_j^{ES,k}, \forall i \in \left[1,..,l_k\right] \tag{17}$$

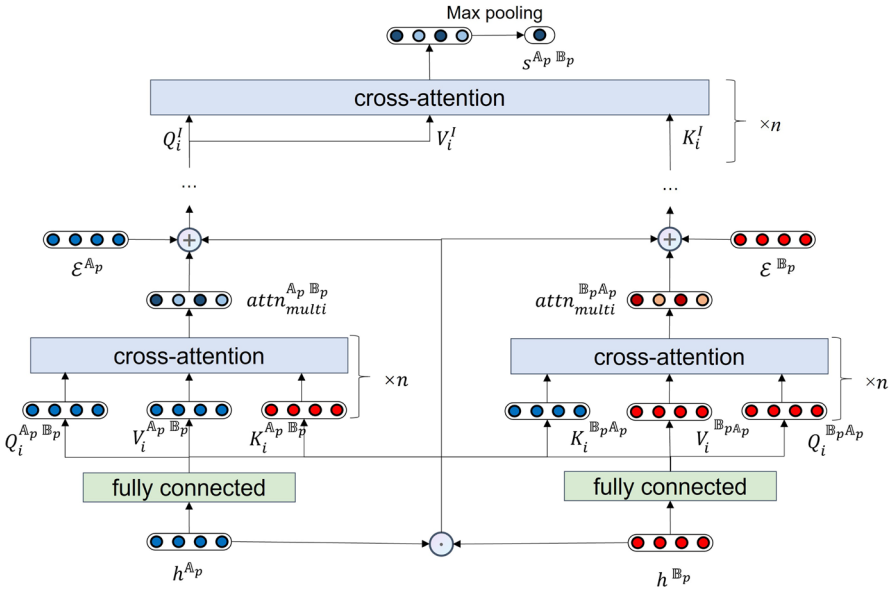

Fig. 4 The framework of the interactive layer. n denotes the total number of heads in multi-head attention

In (17), the context tokens within the attention window of event e_i^k is aggregated according to the corresponding attention weight $\sqsupseteq_{i,j}^k$. In this way, the context features of event e_i^k is compressed in $\mathcal{E}_i^k$.

3.3.5 Interactive layer

In the previous steps, we mainly model the internal information of the case. However, to calculate the similarity of case pairs, we also need to model the interactive semantic information between case pairs. In this layer, we will calculate the interactive semantic information between case pairs based on the multi-head attention mechanism. As Fig. 4 shows, we mainly utilize the two-layer cross-attention modules to realize the interactive semantic information of modelling case pairs. The difference between our cross-attention and most traditional self-attention lies in that the attention weight calculated by our cross-attention comes from paragraph $\mathbb{A}_p$ and paragraph $\mathbb{B}_p$, which reflects the interactive information between cases, while the traditional attention weight only comes from the query document.

More specifically, first we model the semantic information from $\mathbb{A}_p$ to $\mathbb{B}_p$. The query matrix $K_i^{\mathbb{A}_p\mathbb{B}_p}$, value matrix $V_i^{\mathbb{A}_p\mathbb{B}_p}$ and value matrix $Q_i^{\mathbb{A}_p\mathbb{B}_p}$ are constructed as follows:

$$K_i^{\mathbb{A}_p\mathbb{B}_p} = h^{\mathbb{A}_p} W_i^k \tag{18}$$

$$V_i^{\mathbb{A}_p\mathbb{B}_p} = h^{\mathbb{A}_p} W_i^v \tag{19}$$

$$Q_i^{\mathbb{A}_p\mathbb{B}_p} = h^{\mathbb{B}_p} W_i^q \tag{20}$$

where W_i^k, W_i^v, $W_i^q \in \mathbb{R}^{d_s \times d_s}$ and i represents the index of heads in multi head attention. Then, the cross-attention from case $\mathbb{A}$ to case $\mathbb{B}$ is calculated by:

$$attn_i^{\mathbb{A}_p\mathbb{B}_p} = softmax\left(\frac{Q_i^{\mathbb{A}_p\mathbb{B}_p}\left(K_i^{\mathbb{A}_p\mathbb{B}_p}\right)^T}{\sqrt{d_s}}\right)V_i^{\mathbb{A}_p\mathbb{B}_p} \tag{21}$$

For multi-head attention, the result of single attention will be concatenated together:

$$attn_{multi}^{\mathbb{A}_p\mathbb{B}_p} = attn_1^{\mathbb{A}_p\mathbb{B}_p} \oplus attn_2^{\mathbb{A}_p\mathbb{B}_p} \oplus \cdots \oplus attn_n^{\mathbb{A}_p\mathbb{B}_p} \tag{22}$$

where n denotes the total number of heads in multi-head attention, and $\oplus$ means the concatenation operation. To measure the original similarity information between case $\mathbb{A}_p$ and case $\mathbb{B}_p$, the difference and element-wise multiplication are calculated, then we concatenate event-context features and the interactive semantic features with the element-wise results together:

$$I^{\mathbb{A}_p\mathbb{B}_p} = \mathcal{E}^{\mathbb{A}_p} \oplus attn_{multi}^{\mathbb{A}_p\mathbb{B}_p} \oplus \left(h^{\mathbb{A}_p} \odot h^{\mathbb{B}_p}\right) \tag{23}$$

Here, $I^{\mathbb{A}_p\mathbb{B}_p}$ is concatenated by those vectors, and $\odot$ is the element-wise product between two vectors. $I^{\mathbb{A}_p\mathbb{B}_p}$ is considered as high-order interactive information from case $\mathbb{A}_p$ to case $\mathbb{B}_p$, which includes event-context features, interactive semantic features, and original similarity information. In this way, the semantic information of $\mathbb{A}_p$ can be fully utilized when calculating the similarity finally. The semantic information features from paragraph $\mathbb{B}_p$ to paragraph $\mathbb{A}_p$ are calculated in the same way as (18)–(23) shows and we can obtain $I^{\mathbb{B}_p\mathbb{A}_p}$.

Similar to (18)–(22), we utilize the attention mechanism to integrate $I^{\mathbb{A}_p\mathbb{B}_p}$ and $I^{\mathbb{B}_p\mathbb{A}_p}$. The query matrix K_i^I, value matrix V_i^I and value matrix Q_i^I are constructed as follows:

$$K_i^I = I^{\mathbb{A}_p\mathbb{B}_p} W_i^{k,I} \tag{24}$$

$$V_i^I = I^{\mathbb{A}_p\mathbb{B}_p} W_i^{v,I} \tag{25}$$

$$Q_i^I = I^{\mathbb{B}_p\mathbb{A}_p} W_i^{q,I} \tag{26}$$

where $W_i^{k,I}$, $W_i^{v,I}$, $W_i^{q,I} \in \mathbb{R}^{d_s \times d_s}$. We obtain the similarity features $attn_{mutil}^{s,\mathbb{A}_p\mathbb{B}_p} = [s_1^{\mathbb{A}_p\mathbb{B}_p}, s_2^{\mathbb{A}_p\mathbb{B}_p}, \ldots, s_{l_{\mathbb{A}_p}}^{\mathbb{A}_p\mathbb{B}_p}]$ between paragraph $\mathbb{A}_p$ and paragraph $\mathbb{B}_p$ as:

$$attn_i^{s,\mathbb{A}_p\mathbb{B}_p} = softmax\left(\frac{Q_i^I\left(K_i^I\right)^T}{\sqrt{d_s}}\right)V_i^I \tag{27}$$

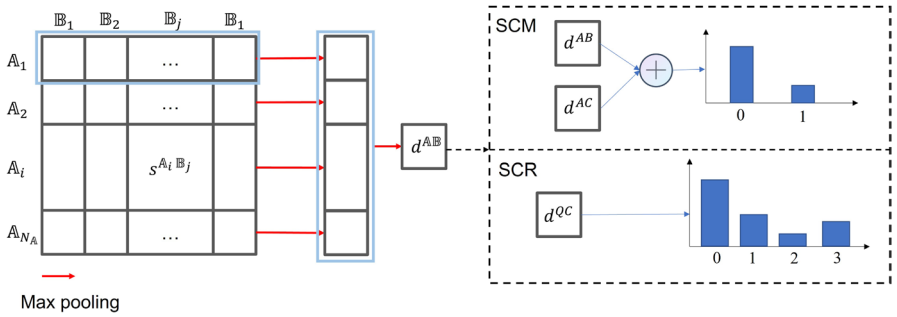

Fig. 5 The construction of the aggregate and output layer

$$attn_{multi}^{s,\mathbb{A}_p\mathbb{B}_p} = attn_1^{s,\mathbb{A}_p\mathbb{B}_p} \oplus attn_2^{s,\mathbb{A}_p\mathbb{B}_p} \oplus \cdots \oplus attn_n^{s,\mathbb{A}_p\mathbb{B}_p} \tag{28}$$

After that, it is fed into a max-pooling layer:

$$s^{\mathbb{A}_p\mathbb{B}_p} = Pooling\left(attn_{mutil}^{s,\mathbb{A}_p\mathbb{B}_p}\right) \tag{29}$$

where Pooling stands for the pooling operation over the dimension of sequence length. $s^{\mathbb{A}_p\mathbb{B}_p}$ represents the similarity information between paragraph $\mathbb{A}_p$ and paragraph $\mathbb{B}_p$.

3.3.6 Aggregate and output layer

The construction of the aggregate and output layer is shown in Fig. 5. After each paragraph pair from case $\mathbb{A}$ and case $\mathbb{B}$ pass through the event-detection layer and interactive attention layer, we can obtain the similarity information between any two paragraphs. They are combined as (30):

$$s^{\mathbb{A}\mathbb{B}} = \begin{bmatrix} s^{\mathbb{A}_1\mathbb{B}_1} & \cdots & s^{\mathbb{A}_1\mathbb{B}_{N_\mathbb{B}}} \\ \vdots & \ddots & \vdots \\ s^{\mathbb{A}_{N_\mathbb{A}}\mathbb{B}_1} & \cdots & s^{\mathbb{A}_{N_\mathbb{A}}\mathbb{B}_{N_\mathbb{B}}} \end{bmatrix} \tag{30}$$

where $N_\mathbb{A}$ and $N_\mathbb{B}$ represent the total number of paragraphs in case $\mathbb{A}$ and case $\mathbb{B}$. $s^{\mathbb{A}\mathbb{B}}$ aggregates the similarity information of all paragraphs in the case $\mathbb{A}$ corresponding to the paragraph in the case $\mathbb{B}$. Then, $s^{\mathbb{A}\mathbb{B}}$ is passed through a max-pooling layer to obtain the document-level similarity information as follows:

$$d^{\mathbb{A}\mathbb{B}} = Pooling\left(s^{\mathbb{A}\mathbb{B}}\right) \tag{31}$$

where *Pooling* represents performing a max-pooling operation on all paragraph-related dimensions. $d^{\mathbb{A}\mathbb{B}}$ represent the similarity information between case $\mathbb{A}$ and $\mathbb{B}$, and all the similarity features are compressed in it. Next, we need to construct $d^{\mathbb{A}\mathbb{B}}$ as the result required by different SCA tasks. Since the target output of different SCA tasks differs, we need to explain the output methods under various tasks separately.

For SCM task, taking the similarity features d^{AB} and d^{AC} as input, the predicted distribution y is calculated as follows:

$$R = d^{AB} \oplus d^{AC} \tag{32}$$

$$\widehat{y}^{scm} = softmax\left(W^{y}_{scm}R + b^{y}_{scm}\right) \tag{33}$$

Here, d^{AB} and d^{AC} are concatenated into the predicted result distribution R, and $\widehat{y}$ represents the probability distribution of the sample, which can be expressed as:

$$\widehat{y}^{scm} = \left[sim_{A,B}, sim_{A,C}\right] \tag{34}$$

Finally, we use the cross-entropy loss function to train our model:

$$\mathcal{L}^{scm} = -\sum_{i=0}^{|R|} y_i^{scm} log \widehat{y}_i^{scm} \tag{35}$$

where y_i^{scm} is the ground-truth label and $\widehat{y}_i^{scm}$ is the predicted result. R denotes the set of relevant labels.

For SCR task, the similarity information d^{QC} is passed through a fully-connected layer followed by a *softmax* function to make a prediction as follows:

$$\widehat{y}^{scr} = softmax\left(W^{y}_{scr}d^{QC} + b^{y}_{scr}\right) \tag{36}$$

The loss function is the same as the SCM task:

$$\mathcal{L}^{scr} = -\sum_{i=0}^{|R|} y_i^{scr} log \widehat{y}_i^{scr} \tag{37}$$

LECM takes legal events as the core basis for judging the similarity of cases and models the characteristics of legal events through events and event contexts. Legal case similarity is different from general textual similarity task, it needs to consider textual similarity from the legal professional point of view, and legal event features can reflect this well. We design models for two subtasks of SCA, which differ only slightly in detail. These differences are caused by the input and output forms of tasks. Theoretically, event legal features are not limited to calculating the similarity of legal texts, and event legal features are still needed in tasks such as crime prediction and sentence prediction. Therefore, in future work, we will explore more specific legal tasks.

4 Experiments

In this section, to investigate the effectiveness of LECM on similar case analysis, we carry out experiments on the public datasets and then compare the performance of our model with the baselines. Then, we conduct ablation experiments to investigate the effectiveness of each module in LECM. After that, we explore the impact of auxiliary datasets and attention window size on LECM. Finally, we select some typical cases from the datasets to illustrate the working mechanism of the model.

4.1 Datasets

SCA is formalized as two subtasks: SCM and SCR, both of which can be used to evaluate the SCA performance of the model. To evaluate the performance of LECM, we use CAIL-2019[2] dataset and LeCaRD[3] dataset, corresponding to the two subtasks. In addition, as mentioned in Sect. 3.3.2, LEVEN (Yao et al. 2022) dataset is used to train the ED module of LECM, which is an ED dataset.

CAIL-2019 is an open-source dataset that focuses on the SCM task. The input of CAIL-2019 is a triplet (A, B, C), where A, B, and C are fact descriptions of three cases. The objective is to determine whether case A is more similar to case B or case C, simplifying the task into binary classification. *Positive* or *Negative* labels are assigned based on the similarity between cases A and B. If case A is similar to B, it is recorded as *Positive*. Otherwise, it is recorded as *Negative*. All legal documents from CAIL-2019 are collected from China Judgments Online.[4] There are 8,138 samples in the dataset, of which 5,102 samples constitute the training dataset, 1,536 samples constitute the validation dataset, and the test dataset is composed of the rest 1,500 samples. All samples are related to civil, and the similarity between these documents is defined by legal professionals. Table 1 provides an overview of the dataset, demonstrating a balanced distribution of positive and negative samples. Notably, the average input length of this dataset is relatively short, enabling assessment of the LECM's performance on such text.

The LeCaRD dataset is a dataset for training the SCR task. LeCaRD is a legal case retrieval dataset in China's legal system, which is designed under the guidance of the official document published by the Supreme People's Court of China. LeCaRD consists of 107 query cases and 10,700 candidate cases, most of which are criminal cases. Each query will have about 100 candidate cases, and the model needs to sort the 100 candidate cases according to the similarity according to the text of the query. The higher the similarity, the higher the ranking. As evident from Table 1, the average query length is relatively shorter in comparison. However, the length of each candidate case is considerably longer, surpassing the maximum length capacity of general language models like Bert. This presents an opportunity to assess the performance of LECM when faced with a long document.

In addition, our ED module is trained using the LEVEN dataset, which consists of 8,116 legal cases and 150,997 human-annotated event mentions. Similar to CAIL-2019, LEVEN data is sourced from China Judgments Online, and events are marked by experienced legal experts. LEVEN encompasses 108 event types, covering various common categories such as deception, violence, accidents, and more. The cases in the CAIL-2019 and LeCaRD datasets mainly belong to civil and criminal cases, and LEVEN includes frequent events from these domains. As LEVEN serves as an auxiliary dataset and is not utilized for performance testing, the division rules for training and testing are not provided in Table 1. Importantly,

[2] https://github.com/thunlp/CAIL.
[3] https://github.com/myx666/LeCaRD/tree/main/data.
[4] https://wenshu.court.gov.cn/.

Table 1 Data statistic

SCM-Task	Statistic	Number
CAIL-2019	Total document	8,138
	Positive	4,236
	Negative	3,902
	Average length	676.24
	Training set size	5,103
	Test set size	1,536
	Valid set size	1,500
SCR-Task		
LeCaRD	Total document	10,700
	Query	107
	Candidate per query	100
	Average length of candidate	6,363.22
	Average length of query	444.59
	Training set size	8,560
	Test set size	2,140
ED-Auxiliary-Task		
LEVEN	Total document	8,116
	Event	150,997
	Event Type	108
	Average length	495.83

the average length of LEVEN is only 495.83, which falls within the maximum processing length of BERT. Consequently, our ED module does not require extensive processing of long text in LEVEN. Examples of text snippets for the above three datasets are shown in the Appendix.

4.2 Baselines

To verify the effectiveness of the proposed model, we compare our model with the following competitive baseline models:

- **TF-IDF**: As a robust classification model, term-frequency inverse document frequency (Salton and Buckley 1988) is used to extract features of inputs, and SVM (Suykens and Vandewalle 1999) is adopted as the classifier.
- **LMIR**: Language models for information retrieval (Ponte and Croft 2017) is a traditional retrieval model based on bag-of-words models.
- **TextCNN**: TextCNN (Kim 2014) is a classic CNN-based text classification model. We employ TextCNN with a single-layer convolution for fact encoding and classifier. Since TextCNN is not good at capturing long text features, we implement a Siamese network-based version, denoted as TextCNN^{S}.

- **SMASH-RNN**: Jiang et al. (2019) propose a hierarchical RNN based on attention, which uses the document structure to improve the representation of long-form documents.
- **Lawformer**: Lawformer (Xiao et al. 2021) is a longformer-based pre-trained language model, which is trained on large-scale legal case documents. Since lawformer can handle longer texts, we implement two versions of lawformer model: based on concatenation and based on the Siamese network, denoted as Lawformer^{C} and Lawformer^{S}, respectively.
- **BERT**: Bert (Devlin et al. 2019) is a mainstream pre-trained language model. It has demonstrated superior performance on various downstream tasks. Since the length of the input limits BERT, we only implement a Siamese network-based version, denoted as BERT^{S}.
- **BERT-PLI**: BERT-PLI (Shao et al. 2020) break the text into paragraphs and calculate similarity at the paragraph-level. In this way, BERT-PLI model the semantic interactions between paragraphs. Experiments show that it has good performance on legal texts.
- **LFESM**: Hong et al. (2020) extract legal elements via regular expressions and adopt BERT to capture long-range dependencies in the legal documents.

4.3 Experiment settings

For TF-IDF, we set the feature size to 2,000. The filter width of TextCNN is {2,3,4,5}, and each filter size was 25. For the SMASH-RNN, the hidden state size is 768. For the Bert-based model, we adopt the bert-base-chinese checkpoint from OpenCLap[5] as the basic encoder. The lawformer model can process longer sentences, thus we set the max length of each input to 700 for the lawformer-based model and for the rest model 512. Since the length 512 supported by Bert and LFESM is much smaller than the average document length of LeCaRD, we adopt the paragraph-segmentation method in Sect. 3.3.1 for the encoder and output head of Bert and LFESM, so that they can be adapted to longer text.

Note that the training procedure of our model is divided into two steps: pre-training the ED module and training the whole model with a frozen ED module. Each stage uses different hyper-parameters. Hyper-parameters are tuned on the validation set. We pre-train the ED module on the LEVEN dataset, and the dropout rate among each layer is 0.1. The batch size of the ED module is 16. The learning rate of the ED module is 1e−5. We take 20% of the data in LEVEN as the validation set, and the best performance model is adopted as our ED module. On the LEVEN dataset, we use accuracy to evaluate the model. The rest part of our model is trained on the CAIL-2019 and LeCaRD datasets. Hyperparameters are the same on both datasets unless otherwise specified. For the input document, we adopt a paragraph-segmentation approach mentioned in Sect. 3.3.1. Specifically, the number of paragraphs of query and candidate in the SCR task are 2 and 10, and the number of paragraphs

[5] http://zoo.thunlp.org/.

Table 2 Similar case analysis results on LeCaRD for SCR task

Metrics	P@5	P@10	MAP	NDCG@10	NDCG@20	NDCG@30
TF-IDF	42.00	43.50	49.05	66.68	72.77	80.43
LMIR	45.50	50.00	50.50	74.05	78.92	87.18
TextCNN^{S}	\	\	\	\	\	\
SMASH-RNN	\	\	\	\	\	\
Lawformer^{C}	40.50	36.00	49.93	75.73	77.54	81.70
Lawformer^{S}	41.50	42.00	46.53	73.36	77.81	83.04
BERT^{S}	42.50	42.50	52.21	76.01	78.47	81.09
LFESM	48.00	49.00	51.26	76.72	80.87	82.28
Bert-PLI	42.00	45.50	55.63	80.85	**86.19**	88.77
LECM	**48.50**	**51.50**	**62.24**	**85.07**	85.85	**89.63**

of cases in the SCM task are 2, respectively. The window size of the event-context integration mechanism is 64. Our method relies on event context features, and we believe that just the right window size can improve performance. As for the interactive layer, the hidden size of the multi-head attention layer is set to 768, and the number of heads in the multi-head attention layer is 4. Except for the ED module, the dropout rate per layer in the LECM model is 0.3. The batch size during training is 8 and 2 on CAIL-2019 and LeCaRD, respectively. We use Adam (Kingma and Ba 2015) as the optimizer to optimize the model, which is effective in neural model training. We set the learning rate to 1e−5 and the l2-normalization coefficient λ is 1e−5. In addition, we use NVIDIA Apex to accelerate the training procedure.

Since SCM is a binary classification task and CAIL-2019 is a balanced dataset, we employ accuracy (Acc.) as our evaluating metrics, which more objectively reflects the effectiveness of LECM and other baselines. Note that the validation set and the test set of CAIL-2019 are divided by the original author, so the validation set can also fairly reflect the performance of the model. Therefore, we utilize both the validation set and the test set as the evaluation results. For the SCR task, we utilize precision (P) and normalized discounted cumulative gain (NDCG) (Järvelin and Kekäläinen 2002) to evaluate the performance. Precision metrics include P@5, P@10 and mean average precision (MAP). NDCG metrics include NDCG@10, NDCG@20 and NDCG@30. P@k concerns whether the ground-truth case appears in the Top-K retrieval result list. NDCG@k concerns the position of the ground-truth cases in the retrieval result list. Following the literature (Ma et al. 2021b), we randomly sample 20% of the data from the LeCaRD dataset as the test set to evaluate the performance of the models. Other standard parameters follow the default settings of the Pytorch[6] framework.

4.4 Experimental results and discussion

We first evaluate the overall performance of all models on two SCA subtasks, including SCM and SCR. Tables 2 and 3 shows the comparative experimental

[6] https://pytorch.org/.

Table 3 Similar case analysis results on CAIL-2019 for SCM task

Metrics	Acc. (Valid)	Acc. (Test)
TF-IDF	52.97	53.58
LMIR	51.22	57.61
TextCNN^{S}	62.33	64.52
SMASH-RNN	64.80	66.53
Lawformer^{C}	63.79	64.21
Lawformer^{S}	67.70	70.09
BERT^{S}	65.02	67.69
LFESM	69.05	73.23
Bert-PLI	68.44	69.75
LECM	**71.24**	**76.35**

results of our model with baselines on the CAIL-2019 dataset and the LeCaRD dataset. The result in bold represent the best performing methods. The symbol of "\" indicates that the method cannot converge normally within a limited number of training epochs. According to the results, we can observe that LECM significantly outperforms all previous baselines on the SCM and SCR tasks. we discuss these experimental results in detail in following subsections.

(1) For the SCM task, among all of the baselines, LFESM achieves the highest performance, indicating that the legal feature captured by the regular expression is helpful for the SCM task. Compared to LFESM, our model achieves the highest accuracy on validation and test datasets, respectively, which verify the effectiveness of our model for SCM. Compared with manually designing regular expressions, the cooperation of the ED module and event-context integration mechanism can extract more comprehensive features. However, on the SCR task, LFESM does not perform well because LFESM mainly considers civil cases when designing regular expressions matching rules, while the LeCaRD dataset is dominated by criminal cases. Therefore, LFESM cannot extract helpful legal features from the LeCaRD dataset. This also confirms the importance of introducing legal features in the SCA task.
(2) For the SCR task, BERT-PLI is the best-performing baseline model, but it does not perform well on the SCM task. This shows that although BERT-PLI can capture the dependencies between paragraphs on long texts, such dependencies cannot help model inference on short texts because short texts pay more attention to the similarity between paragraphs. Except for NDCG@20, LECM achieves the highest performance in the remaining indicators of the SCR task. This indicates that LECM is more precise in selecting top-ranked candidate cases. Besides,

our method achieves better performance on the SCM task, indicating that the paragraph-level features in LECM can be leveraged in a long either short text.

(3) It is observed that Siamese-based models generally outperform concatenation-based models. It can be seen that the neural network model is more inclined to encode a single case rather than a concatenation of case triplet, thereby reducing interference information. It shows the importance and rationality of using Siamese-based architecture in the encoder layer of LECM.

(4) The baseline model based on the bag-of-words model performs much better on the SCR task than on the SCM task. For TF-IDF and LMIR, they take advantage of the whole legal document though they are weaker than the neural network-based model in semantic understanding. However, TextCNN and SMASHRNN cannot converge on the SCR task. Although TextCNN and SMASHRNN can accept long input text, they are not designed for long text tasks. Longer inputs will cause problems like exploding gradients or gradients disappearing. Therefore, it is hard for them to handle long text retrieval.

From the overall results, LECM is significantly superior over the best baseline for a large margin on adopted evaluation metrics, which indicates that LECM has excellent SCA performance. We summarize the reasons as follows: (1) In our model, the event-context integration mechanism will assign different context information to different events, alienating the impact of the same event in the same description. (2) While introducing events as legal features, we also incorporate event-related contextual features into the inference process. (3) We break long documents into paragraphs and perform event detection of the segmented texts. We use the pooling layer to aggregate the results at the end. This enables LECM to handle longer texts while also having excellent performance on short texts. Besides, the ED module cannot handle text that exceeds the input limit, and the paragraph-segmentation mechanism in LECM avoids this problem.

4.5 Ablation test

To study the impact of each layer in our model, we designed several ablation tests to investigate the performance of LECM. Some modifications of our method are listed as follows:

- LECM/EC: We remove the event-context integration mechanism and ED module.
- LECM + RE: This model removes the ED module, and the event sequence is randomly initialized.
- LECM/I: This model removes the interactive layer, and all hidden vectors are concatenated.
- LECM/FT: We remove the freeze on the ED module parameter. LECM will continue to update the parameters of this module when training on the SCA dataset.

Table 4 Ablation test results on LeCaRD for SCR task

Metrics	P@5	P@10	MAP	NDCG@10	NDCG@20	NDCG@30
LECM	48.50	**51.50**	**62.24**	85.07	**85.85**	**89.63**
LECM/EC	42.50	46.00	55.31	81.13	83.41	87.15
LECM + RE	43.50	45.00	54.27	77.77	81.06	84.69
LECM/I	43.50	44.00	55.29	78.03	81.28	85.09
LECM/FT	**49.00**	50.50	60.88	**85.25**	84.68	88.74

Table 5 Ablation test results on CAIL-2019 for SCM task

Metrics	Acc. (Valid)	Acc. (Test)
LECM	71.24	**76.35**
LECM/EC	68.16	71.33
LECM + RE	66.10	70.91
LECM/I	67.41	72.88
LECM/FT	**71.72**	75.77

Tables 4 and 5 shows the performance of these LECM variants. First, when we remove the event-context layer and ED module, our method loses the ability to capture events and their context in fact description. Due to the lack of information and context of the event, the performance of LECM/EC declined by a large margin, which shows that taking the event and its context as legal features can facilitate the model to capture the critical textual features. To further demonstrate the effectiveness of the context of an event, we replace the ED module with a random sequence of events. On the one hand, the decline of LECM + RE verified that the accuracy of ED would affect the model. On the other hand, the decline of LECM is not as apparent as we expected because LECM can reduce the accumulation of ED module errors by flexibly learning the embedded vectors of events. Second, we remove the interactive layer and feed the results of the event-context integration mechanism into the aggregate and output layer. The decrease in the results demonstrates that the interactive layer plays an irreplaceable role in our model. Third, the performance of LECM/FT is almost the same as the original model. However, since the model unfreezes the parameters of the ED module, the model incurs more computational costs when performing forward and backward propagation. Experimental results of LECM/FT show that these costs do not lead to improvement. Besides, since the parameters of the ED module have changed, the accuracy of the event detection task will not be guaranteed. The improvement of LECM /FT compared to the performance of the baseline models will mainly come from a more complex network structure rather than accurate events and their contexts. This also leads to poor interpretability of the model, which is vital in legal artificial intelligence.

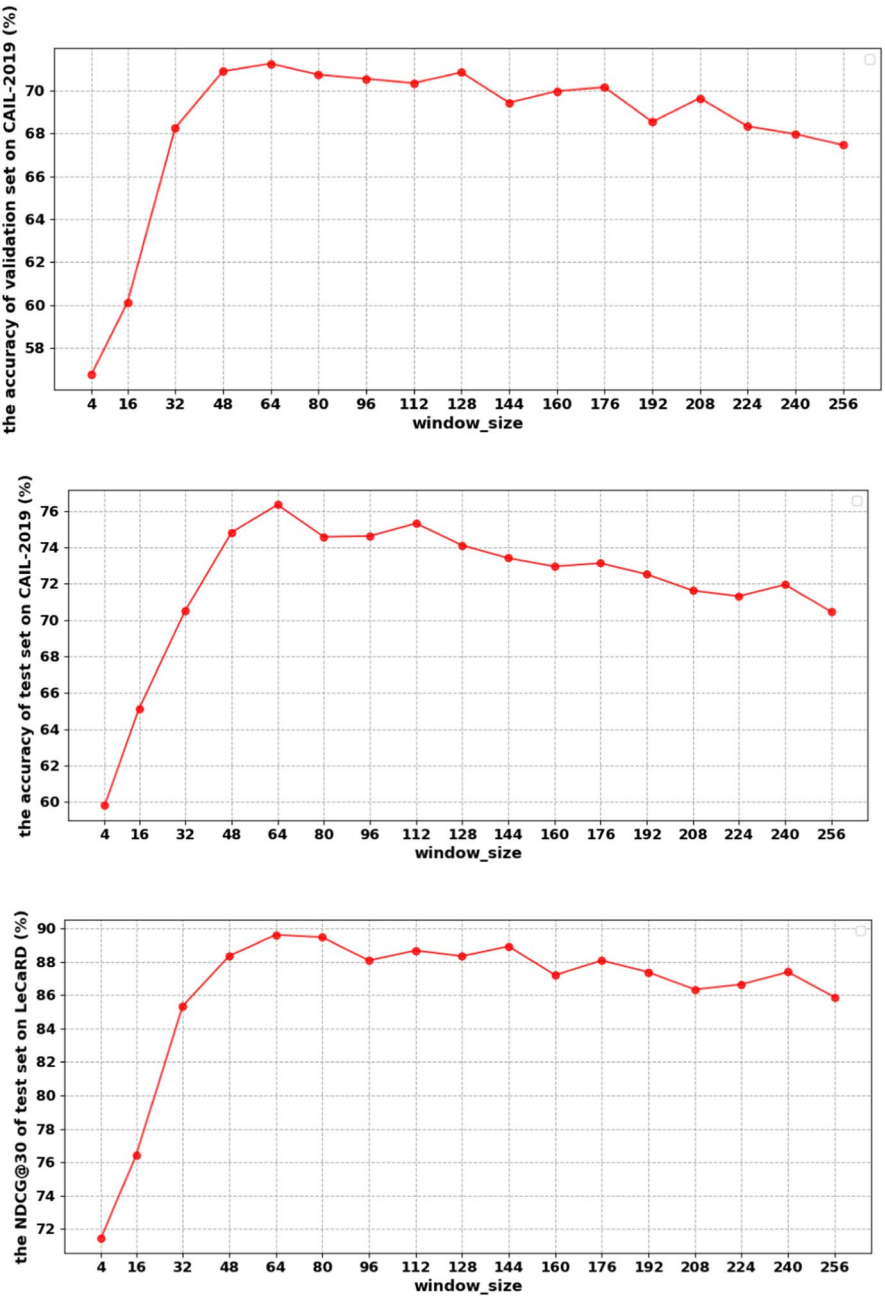

Fig. 6 The impact of attention window size

4.6 Impact of window size

To further explore the effectiveness of the event-context integration mechanism, we test our model with various attention window sizes. The core of LECM is the attention to computational events and their contexts. Therefore, the size of the attention window is an important hyperparameter for LECM. We gradually increment the window size by 2 and test the performance of LECM. Figure 6 shows the model performance concerning the context window size.

It can be observed that the LECM is very sensitive to changes in the window size. More specifically, we find that the performance of setting window size as 4 or 16 was not very ideal. The accuracy of the model is around 50% in the SCM task, which is approximately equal to the model making random guesses. We suppose that due to language habits, the adjacent words of trigger words are similar in a small range, so they cannot provide helpful level features, interfering with the original semantic information. As a result, this has a noticeable impact on the performance of the model. Therefore, when the window size is gradually increased from 4, the performance of the model is also significantly improved. When the attention window size of the model exceeds 64, the performance of the model starts to degrade slowly. When the window size is too large, event-context attention degrades to approximate global attention, and trigger words will attend to tokens that do not describe themselves, which will also affect the performance. The model achieves the best performance when the attention window size is 64 on three test sets. Although the datasets contain a large number of long texts, we take the paragraph-segmentation mechanism for the texts, and the maximum length of each paragraph is still limited to 512, which is the maximum input length of BERT. Thus, we speculate that when the length of input text is 512, the optimal size of the window size is about 64. In this case, the model can focus on the words that describe itself and avoid interference from other words. Therefore, we adopt the window size 64 in our method. There could be a correlation between the window size and the maximum input length, but further investigation will be conducted in our future research to explore this relationship.

4.7 Impact of auxiliary dataset

LECM involves two datasets during the training process: the main dataset of the SCA task and the auxiliary dataset LEVEN. To explore the impact of the auxiliary dataset on the performance of LECM, we make different transformations on the auxiliary dataset LEVEN as follows:

Data augmentation ($\mathbf{LEVEN_{DA}}$): This transformation represents augmenting the entire fact description, including swapping sentence positions in the fact description, deleting a sentence at random, and copying the fact description.
Keep civil events ($\mathbf{LEVEN_{+CE}}$): The legal case texts of the LEVEN dataset are divided into two categories: criminal cases and civil cases. To explore the

Table 6 Experimental results of ED Module on LEVEN

Dataset	Acc.
LEVEN_{DA}	**84.92**
$\text{LEVEN}_{\text{+CE}}$	76.82
$\text{LEVEN}_{\text{-CE}}$	77.18
LEVEN_{RD}	82.65
LEVEN_{CS}	79.43
LEVEN	83.17

Table 7 Experimental results of different auxiliary dataset on LeCaRD for SCR task

Metrics	P@5	P@10	MAP	NDCG@10	NDCG@20	NDCG@30
LECM with LEVEN_{DA}	45.00	**53.00**	**62.63**	83.72	85.37	88.36
LECM with $\text{LEVEN}_{\text{+CE}}$	43.00	42.00	54.63	74.58	77.63	82.49
LECM with $\text{LEVEN}_{\text{-CE}}$	44.00	51.00	59.48	82.07	84.54	88.08
LECM with LEVEN_{RD}	45.50	42.00	48.50	76.92	79.75	82.33
LECM with LEVEN_{CS}	48.00	48.00	61.35	84.26	82.50	89.09
LECM with LEVEN	**48.50**	51.50	62.24	85.07	**85.85**	**89.63**

Table 8 Experimental results of different auxiliary dataset on CAIL-2019 for SCM task

Metrics	Acc. (Valid)	Acc. (Test)
LECM with LEVEN_{DA}	70.79	75.37
LECM with $\text{LEVEN}_{\text{+CE}}$	70.18	74.40
LECM with $\text{LEVEN}_{\text{-CE}}$	66.85	66.36
LECM with LEVEN_{RD}	65.54	71.11
LECM with LEVEN_{CS}	70.90	75.41
LECM with LEVEN	**71.24**	**76.35**

impact of different case types in LEVEN on downstream tasks, we use regular expressions to remove the criminal case documents and keep only the civil case documents.

Remove civil event ($\text{LEVEN}_{\text{-CE}}$): We only keep the criminal cases documents of LEVEN, similar to $\text{LEVEN}_{\text{+CE}}$.

Random Delete (LEVEN_{RD}): Random delete documents from LEVEN.

Copy sentences (LEVEN_{CS}): As Tables 1 and 2 showed, the average case length of the LEVEN dataset is less than 512, while the CAIL-2019 and LeCaRD dataset both exceed 512. We randomly copy some sentences in the LEVEN dataset to make the length of the case reach 512 to observe whether the length of the case affects the Model performance.

Table 6 shows the test performance of the ED module on the LEVEN dataset after the above transformation. Tables 7 and 8 shows the test results of LECM corresponding to different auxiliary datasets. We can observe that:

(1) For LEVEN_{DA}, the ED module outperforms the original LEVEN dataset on the LEVEN_{DA} dataset. Data augmentation enriches the LEVEN dataset to a certain extent, so the performance of the ED module is improved. However, this did not result in a significant change in performance on the downstream SCA task. The ED module is more inclined to accept data with the same distribution as the LEVEN_{DA}. However, when performing the SCA task, the data accepted by the model comes from CAIL-2019 and LeCaRD, so the improvement of the ED module on the DA dataset cannot be generalized to the SCA dataset. We suspect that one of the reasons for the different distribution of the data is the average input length. As shown in Table 1, the average case text length of the LEVEN dataset is less than 512, while the average length of CAIl-2019 and LeCaRD is longer than 512 (the excess will be truncated). Therefore, we randomly replicate the sentences of the case text in the LEVEN dataset to make it longer than 512, thus constructing the LEVEN_{CS} dataset. The ED module shows a noticeable performance drop on the LEVEN_{CS} dataset, which is caused by the ED module overfitting events in repeated sentences. The effect of LEVEN_{CS} on ED did not significantly affect the SCA task. We speculate that the identical distribution of the data offsets the effect of overfitting.

(2) For LEVEN_{RD}, the ED module does not perform well on LEVEN, affecting the accuracy of LECM on CAIL-2019 and LeCaRD. In LEVEN_{RD}, due to the small amount of data, the generalization performance of the ED module is poor, resulting in a low accuracy rate on LEVEN, and this error will accumulate in LECM. This also shows that LECM will use a specific type of event during inference, and the wrong event type will lead to the degradation of model performance.

(3) From the experimental results of $\text{LEVEN}_{\text{-CE}}$ and $\text{LEVEN}_{\text{+CE}}$, it can be seen that whether LEVEN contains civil cases will have a more significant impact on the performance of LECM on CAIL-2019. In the absence of civil events (i.e., $\text{LEVEN}_{\text{-CE}}$), most events will be attributed to *other* type and this results in a significant drop in LECM performance on CAIL-2019. Although LECM will still learn the context information corresponding to other type of events, due to the lack of effective event embedding, it is difficult to capture the words that can really impact the context information. For $\text{LEVEN}_{\text{+CE}}$, since criminal events were low frequent in CAIL-2019, keeping criminal cases in the LEVEN dataset has a small impact. We speculate that the performance of LECM on the SCA task is mainly derived from related type events. For example, the performance of LECM on the CAIL-2019 dataset mainly depends on the related events of civil cases, and the performance on the LeCaRD dataset mainly depends on the related events of criminal cases. The absence of relevant events has a performance loss. To further verify this speculation, we also performed the same test on the LeCaRD dataset. For $\text{LEVEN}_{\text{-CE}}$, we removed civil cases and kept criminal cases, and the results show that LECM decreases to a certain extent on both CAIL-2019 and LeCaRD, but the reduction of LeCaRD is smaller, and the reduction of CAIL-2019 is more significant. There will be a substantial decline in LeCaRD only under $\text{LEVEN}_{\text{+CE}}$. This verifies our

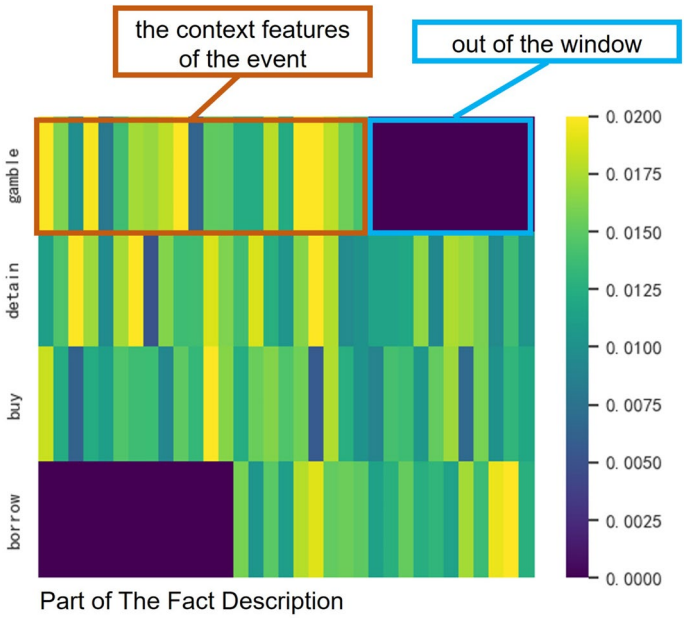

Fig. 7 The heatmap shows the event-context attention matrix between event sequence and fact description. We sample partial events from the complete sequence of events for presentation

conjecture that the key to auxiliary dataset selection is whether to include downstream SCA task-related events.

4.8 Case study

To understand how integrating event and context information benefits the SCA task, we show the inference process behind SCM.

The core process of LECM lies in the calculation of event and context attention weight. Thus, we visualize the event-context attention heat map to illustrate how LECM helps promote the performance of SCM. Figure 7 is a heatmap of the event-context attention matrix between the event sequence and legal case fact description. We intercept four representative events from the complete event sequence. First, the event only attends to words within the window to avoid attending to the context that does not belong to the description itself. Taking the event "gambling" as an example, the deep color part of "gambling" represents the tokens that are not in the attention window. The light color part represents the words that are allowed to be attended. The brighter the color, the more relevant the words are to the event. The attention ranges of different events may overlap to some extent. For example, the event "detain" and the event "buy" have the same attention window range in this part of the fact description. This indicates that the trigger word of the event is too close. Although the attention windows overlap, their attention weights are not the same. Because different events are mapped

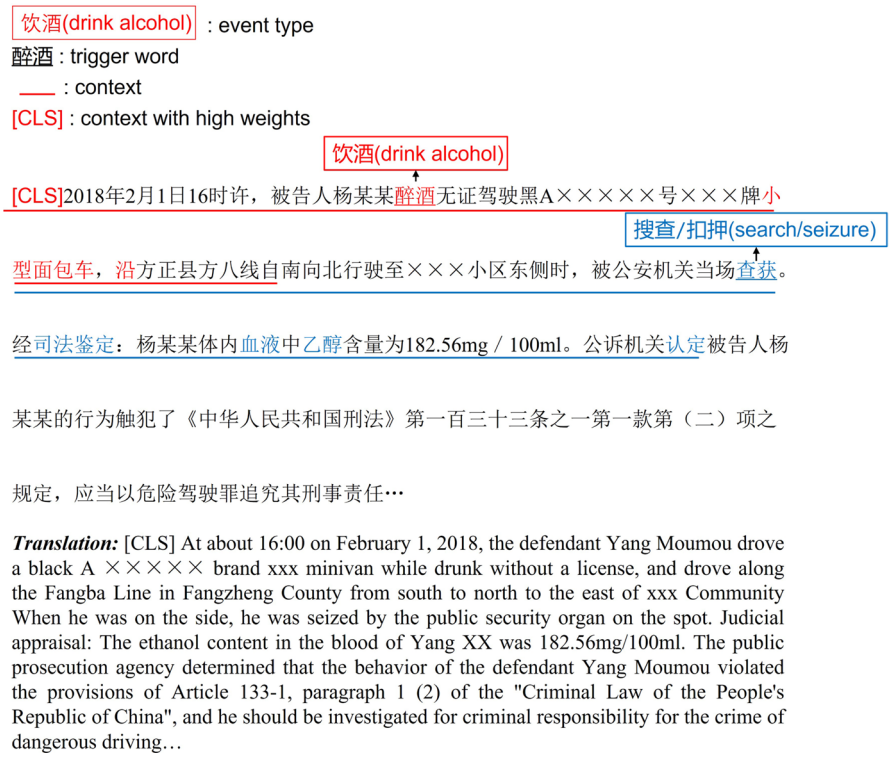

Fig. 8 A typical example from training dataset

to different embedding vectors, they will focus on different parts under the same attention window.

Furthermore, we cite a typical example from the training datasets to illustrate that our method works. As Fig. 8 shows, since the original text is in Simplified Chinese, the order and segmentation of the text cannot be reflected in the translation, so we did not add a callout symbol to the translation. First, there are two events in this paragraph: drink alcohol and search/seizure. In the context of these events, we highlight parts with high attention weight. Note that the event can pay attention to the relevant part of the context. In addition, for the general text in the second half of the paragraph, no event occurs in this part of the text, and they will lose the event-context features. Note that LECM does not involve case pairs when extracting event features, so the features are still suitable for single-text legal tasks. Therefore, we are considering exploring the application of LECM to more downstream legal tasks as our future work (Table 9).

4.9 Error analysis

Error analysis is the process of identifying, examining, and understanding the mistakes made by a model in order to gain insights into its performance and

Table 9 Example in error analysis

Error type	Example
Numeric dependency error	原告又分别于2013年8月30日、8月31日向被告×××汇款**1000000**元。双方未约定借款期限及借款利息。现原告起诉要求被告立即归还借款**1600000**元, 并要求按xx银行同期同类贷款利率支付自起诉之日起至借款实际履行之日止的逾期付款利息。 ***Translation:*** *The plaintiff was divided on August 30, 2013, and on August 31, 2013, Defendant×××Hyuga received **1000,000** yuan. Term and interest of both unpromised loans. At present, the plaintiff requested a loan of **1600,000** yuan for the defendant, and the interest rate for the same loan was paid by the bank for the same period of time*
Word sense disambiguation	其很生气, 觉得网站无缘无故封其账号黑钱, 就决定用网络洪水流量攻击的方式报复一下他们。其在网上找了一个网络攻击软件, 名字叫 "×××.rar", 然后其在网上黑了一个服务器, 其把这个IP地址输入到那个攻击软件里, 通过远程控制发起了洪水流量攻击, 其记得一共攻击了四次, 第一次和第二次攻击的是域名网络攻击软件存储在其网盘里。 ***Translation:*** *He was very angry and felt that the website had unreasonably blocked his account for money, so he decided to retaliate against them through online flood traffic attacks. He found a network attack software called "×××. rar" online, and then hacked a server online. He inputted this IP address into the attack software and launched a **flood** traffic attack through remote control. He remembered attacking a total of four times, with the first and second attacks being the domain name network attack software stored on his network disk*
Event missing error	在尝试确实能够用于入侵家庭摄像头之后, 于2017年8月3日使用QQ26×××65 (昵称: 穷途末路) 建立QQ30×××46群 (群名: 建哥ip(2)群), 与入侵家庭摄像头的爱好者在群内交流分享破解软件及ip地址。期间,×××在群内上传并免费共享×××.zip等入侵软件及相关文件。截止案发,×××.zip的破解软件程序共被群成员下载35人次。另查明, 被告人×××用入侵软件实际侵入并控制家庭摄像头13台。 ***Translation:*** *After attempting to hack into home cameras, QQ 26 was used on August 3, 2017×××65 (Nickname: Dead End) Establishing QQ30×××46 groups (group name: Jian Ge IP (2) group), communicate and share cracking software and IP addresses with enthusiasts who invade home cameras within the group. During this period,×××Upload and share for free within the group×××. Intrusion software and related files such as zip. As of the incident,×××. The zip cracking software program has been downloaded by 35 group members. Further investigation, defendant×××Use intrusion software to actually **invade** and control 13 home cameras*
Attention window error	我对原告主张的借款本金人民币10万元没有异议, 我现在没有能力偿还。对于原告主张的利息人民币5.4万元我不认可, 因为当时我向原告借款是为了放贷款, 其他人没有偿还我利息, 我也不能给付原告利息, 但是本金我认可。诉讼费我不同意承担。 ***Translation:*** *I have no objection to the plaintiff's claim for a loan principal of RMB 100000, and I am currently unable to repay it. I do not agree with the plaintiff's claim for an interest of 54,000 yuan, as I borrowed money from the plaintiff to **release the loan**. Other people did not repay my interest and I cannot pay the plaintiff interest, but I agree with the principal. I do not agree to bear the litigation costs*

Table 9 (continued)

Error type	Example
Misjudgment of event type	×××跟我说×××欠她的钱, 问我能不能帮她去讨钱, 因为我欠她人情就答应了这件事情, 之后我和×××一起去×××家向他讨钱。 ***Translation:*** *×××tells me×××I owe her money and asked if I could help her go and beg for money. Because I* ***owe*** *her a favor, I agreed to this matter. Afterwards, I×××Go together×××My family begged him for money*

improve its accuracy. It involves analyzing erroneous predictions and determining the underlying causes of these errors. We have conducted a thorough analysis of the erroneous predictions in the LECM and have identified the five most common types of errors. The detailed analysis is as follows.

1. **Numeric dependency error** Among the error cases, the most common error is related to numerical values. In the test set, there are finance-related cases where the similarity is closely tied to the monetary amount involved. For instance, when the amount involved is as high as one million RMB, it should significantly influence the judgment of case similarity. However, the LECM model lacks the ability to effectively perceive and understand the contextual information related to numerical values, such as the significance of different monetary amounts in financial cases. As a result, the LECM model fails to accurately calculate the similarity of cases based on important numerical types like the amount involved or the weight of drugs.
2. **Word sense disambiguation** Word sense disambiguation poses a common challenge in ED. For instance, in the given example, the key trigger word "flood" was originally associated with Distributed Denial-of-Service (DDoS) attacks in the original text. However, during the ED stage, the system mistakenly interpreted it as a natural disaster, resulting in an erroneous event classification as a flood. In event descriptions, the same words can be used to represent different events, and their meanings can vary depending on the context. This inherent ambiguity and uncertainty make it more difficult to accurately identify and classify events in ED.
3. **Event missing error** All events detected by the LECM are derived from the LEVEN dataset. While the LEVEN dataset offers a comprehensive overview of judicial events, there might be some inevitable omissions. In the given example, the term *invade* indicates a network attack event. However, due to certain reasons, this specific event type was not included in the LEVEN dataset, posing challenges for the LECM's recognition of such events. The fundamental reason for this error is that the LECM heavily relies on the quality of the auxiliary dataset. If the auxiliary dataset is small in size and has fewer event types, it becomes difficult for the LECM to make reasonable inferences and accurately identify events. To address this issue, it is crucial to continuously update and expand the auxiliary dataset with a wider range of event types to enhance the LECM's event recognition capabilities.

4. **Attention window error** Encoding context within the attention window based on detected events is a crucial step for LECM. Currently, LECM considers the proximity range of the words triggered by events as the attention window range. Previous experiments have demonstrated that this rule can accurately capture the relevant context. However, there are also special cases where the context is not within the adjacent range. In the given example, the plaintiff mentioned the keyword "release the loan" but did not provide a specific description of this event, resulting in the event context not being found in its adjacent text. Finding a more flexible approach to selecting context based on events will be one of our future areas of focus and improvement.
5. **Misjudgment of event type** This is a commonly encountered error. For our ED module, we opted to use BERT + CRF, which handles event complexity well but sacrifices some accuracy. However, when there is a widespread occurrence of event type errors, LECM struggles to provide precise answers based on the event and its context. In the specific example given, the term "owe" highlighted in bold fails to capture the concept of debt. In the Chinese context, the syntax of "owe" bears similarity to that of "debt" leading to incorrect judgment of the event by the ED module.

5 Conclusion and future work

In this paper, we explore the task of SCA and propose the legal event-context model (LECM) to solve it. First, we propose the event-context integration mechanism to formalize the event and the corresponding context information, which captures the contextual features related to events. The event-context integration mechanism introduces events as legal features into the reasoning process, which calculate the similarity of text pairs from a more legal perspective to improve the accuracy and interpretability of SCA. Then, we leverage ED as an auxiliary task for the SCA tasks to help the model locate events and provide the semantic features related to ED. The ED module acts as a bridge between the ED task and the SCA tasks while avoiding the difficulty of manual event annotation for SCA tasks. The experimental results show that LECM outperforms the state-of-the-art model in SCA tasks, which indicates that our model can effectively leverage event-context features from fact description to improve performance and is prospected to be applied to other downstream subtasks of legal intelligence.

5.1 Discussion

Our study has three important theoretical implications. First, we propose the event-context integration mechanism to integrate legal events with relevant contexts. Different from common semantic matching models, our method enables the model to calculate case similarity from two dimensions of legal elements and semantic features. Current researches on legal element extraction (Hong et al. 2020) are mainly based on manually pre-defined rules. However, the definition of rules is

often challenging, and the scope of the application needs to be more comprehensive. Compared with the rule definition method, based on the support of the LEVEN dataset, LECM covers the element types more comprehensively, can extract legal elements more accurately, and avoids the tedious rule-building work.

Second, although ED methods have been developed (McClosky et al. 2011; Li et al. 2013; Deng et al. 2021), the applications of these methods to downstream prediction tasks are rare. We build a legal incident detection model based on the LEVEN dataset and apply it to the SCA task. Our experimental results show that introducing events can effectively improve the accuracy of SCA. Besides, the downstream tasks of event detection are broader than the analysis of similar cases. We believe it can play an important role in more downstream judicial applications, which will be our future work.

Third, we introduce an additional ED dataset to avoid manually labeling events on the SCA dataset. Current multi-task learning methods (Sener and Koltun 2018; Hu et al. 2022) focus on a single dataset. In the legal artificial intelligence field, there is a correlation between legal datasets, so it is necessary to utilize existing datasets to avoid heavy manual labeling work. We adopt a two-stage training method to pre-train the ED task and integrate the ED model into the SCA model. Our findings reveal the regularity between the performance of the SCA model and ED datasets, which clarifies the basis for selecting the auxiliary dataset.

This study also provides noteworthy practical implications. First, due to a large number of cases, legal practitioners often need to spend a lot of time and energy screening similar cases to quickly focus on the core of the case and prepare for litigation ideas. LECM can accurately analyze similar cases and help legal practitioners judge whether they can refer to a particular case or select similar cases from candidate cases, thereby saving judicial resources. Second, due to the need for more judicial expertise, ordinary people often need help to inquire and understand similar cases accurately. The automated similar case analysis model allows ordinary people to understand the essence of cases through similar cases and simultaneously eases the pressure on judicial practitioners.

5.2 Future work

In future work, we will explore more downstream tasks (e.g., legal judgement prediction, legal question answering) to investigate the effectiveness of LECM. Moreover, we will optimize the ED module in LECM so that the ED task and downstream tasks can be better combined.

Appendix: Example text snippets in the datasets

Dataset	*Example*
CAIL-2019	**Case A:** …被告因需要资金, 向原告借款三次, 2015年6月18日第一次借款为人民币10000元, 2015年7月20日第二次借款人民币10000元, 2015年9月1日第三次借款人民币20000元, 并出具借条三张, 约定借期均为一个月。借款到期后, 被告未及时归还借款。借款期间, 被告将两辆车子的登记证书抵押在原告处。原告向被告多次催要借款未果, 故诉至法院, 请求依法支持原告诉求。被告×××未答辩, 也未提交书面证据。当事人围绕诉讼请求依法提交了证据…
	Translation: *…The defendant borrowed money from the plaintiff three times due to the need for funds. The first loan was RMB 10,000 on June 18, 2015, the second loan was RMB 10,000 on July 20, 2015, and the third loan was RMB 1 on September 1, 2015. 20,000 yuan, and issued three IOUs, with an agreed loan period of one month. After the loan was due, the defendant failed to repay the loan in time. During the loan period, the defendant mortgaged the registration certificates of the two vehicles with the plaintiff. The plaintiff repeatedly urged the defendant to borrow money but failed, so he appealed to the court, requesting to support the plaintiff's claim in accordance with the law. Defendant×××did not respond and did not submit written evidence. The parties submitted evidence according to the law around the claim…*
	Case B: …被告×××向原告借款25万元, 为原告出具了借条, 并口头约定按月息2%计算; 2016年6月27日, 被告×××向原告借款1万元, 为原告出具了借条, 并口头约定按月息2%计算; 2016年7月11日, 被告×××向原告借款2万元, 为原告出具了借条, 并口头约定按月息2%计算, 即被告×××共向原告借款28万元。借款后, 原告多次催要, 被告至今未支付借款本息。为此, 请求人民法院判令被告偿还原告本金20万元及利息…
	Translation: *…Defendant×××borrowed 250,000 yuan from the plaintiff, issued an IOU for the plaintiff, and verbally agreed to calculate the monthly interest at 2%; on June 27, 2016, the defendant×××borrowed 10,000 yuan from the plaintiff, and issued an IOU for the plaintiff, and verbally agreed to calculate the monthly interest at 2%; on July 11, 2016, the defendant×××borrowed 20,000 yuan from the plaintiff, issued an IOU for the plaintiff, and verbally agreed to calculate the monthly interest at 2%, that is, the defendant×××A total of 280,000 yuan was borrowed from the plaintiff. After the loan, the plaintiff repeatedly demanded it, but the defendant has not yet paid the principal and interest of the loan. For this reason, we request the people's court to order the defendant to repay the plaintiff's principal of 200,000 yuan and interest…*
	Case C: …原告×××诉称, 2014年11月28日, 被告×××向其借款2万元; 2015年2月13日, 被告×××又向其借款5万元; 2015年6月16日, 被告×××又向其借款5万元。因被告未能还款, 原告于2016年8月将被告诉至×××市人民法院。2016年10月9日, 原、被告达成还款计划。同日, 原告申请撤回对被告的起诉。其后, 被告未能按照还款计划的内容履行钱款, 现要求被告归还借款人民币3万元、自第一次起诉日2016年8月30日起按当期银行利息结算由被告归还…
	Translation: *…Plaintiff×××claimed that on November 28, 2014, defendant×××borrowed 20,000 yuan from him; on February 13, 2015, defendant×××borrowed another 50,000 yuan from him; on June 16, 2015, the defendant×××borrowed another 50,000 yuan from him. Because the defendant failed to repay the loan, the plaintiff will be notified to the People's Court of×××City in August 2016. On October 9, 2016, the plaintiff and the defendant reached a repayment plan. On the same day, the plaintiff applied to withdraw the lawsuit against the defendant. Afterwards, the defendant failed to fulfill the payment according to the content of the repayment plan. The defendant is now required to return the loan of RMB 30,000, which will be settled at the current bank interest from the date of the first lawsuit on August 30, 2016…*

Dataset	*Example*
LeCaRD	**Query:** …被告人×××饮酒发生急性酒精中毒, 头脑出现幻觉, 臆想自己近几年的不幸遭遇与国家有关领导和×××市×××等人有关。为了发泄其不满情绪, 被告人×××从自己住处拿柴刀步行到×××村便民服务中心办公室, 将办公室卷帘门损坏, 并用柴刀刀背将玻璃门打碎, 进入室内后, 拿柴刀在办公室内乱砸乱砍, 毁坏了室内摄像头1只、采集器1只、液晶显示器4台、尼康牌照相机1台及数据线、电源线数根… ***Translation:*** *…Defendant××× suffered from acute alcohol intoxication after drinking, and had hallucinations in his head, imagining that his misfortunes in recent years were related to relevant state leaders and××× city××× and others. In order to vent his dissatisfaction, defendant××× took a hatchet from his own residence and walked to the office of the convenience service center of××× village, damaged the rolling shutter door of the office, and smashed the glass door with the back of the hatchet. After entering the room, he took the hatchet Smash and hack in the office, destroying 1 indoor camera, 1 collector, 4 LCD monitors, 1 Nikon camera, and several data cables and power cables…* **Candidate1:** …被告人×××认为声音大, 已影响自己和家人休息, 便从家里拿一根木棍到×××住处, 将×××收音机打坏。同日上午7时许,×××手持山钩刀来到×××家门口,×××便从家里厨房拿出钢钎, 兄弟俩人再次发生争执并打架。期间,×××持钢钎打击×××头部等部位,×××被打后, 回到自己家中。之后,×××见×××伤势较重, 遂驾驶农用车载×××去医院治疗… ***Translation:*** *…Defendant××× believed that the sound was loud enough to interfere with the rest of himself and his family, so he took a wooden stick from home to××× 's residence and broke the radio of×××. At about 7 o'clock in the morning on the same day,××× came to the door of××× 's house with a mountain hook knife in hand, and××× took out a steel brazing rod from the kitchen at home, and the two brothers had another dispute and fight. During this period,××× beat××× 's head and other parts with a steel drill. After being beaten,××× returned to his home. Later,××× saw that××× was seriously injured, so he drove××× to the hospital for treatment…* **Candidate2:** …被告人×××和饮酒后将在×××市×××电业局工作人员×××、×××放置在摩托车上的两顶安全帽无故拿走, 后×××向被告人×××和讨要安全帽时,×××和一手持水果刀, 一手持菜刀, 表示拒绝归还。×××、×××讨要未果后, 遂报警。不久后, 接警民警×××带着辅警×××赶到现场, 劝说×××和归还安全帽, 但×××和仍拒不归还且情绪越发激动, 其双手持刀威胁要砍死接警民警, 并向接警民警及辅警逼近… ***Translation:*** *…The defendant××× took away without reason the two safety helmets placed on the motorcycle by the staff of××× Electric Power Bureau of××× City××××××after drinking, and then××× reported to the defendant When the person××× he asked for the helmet,××× he held a fruit knife in one hand and a kitchen knife in the other, refusing to return it.×××,××× reported to the police after asking for it but failed. Not long after, the police officer××× rushed to the scene with the auxiliary police officer×××, and persuaded××× he to return the helmet, but××× he still refused to return it and became more and more emotional. He threatened to hack to death with a knife in both hands Receive the police, and approach the police and auxiliary police…* (More candidates…)
LEVEN	…被告人×××母亲×××在×××市×××公司送餐期间, 因搭乘司机×××驾驶的车辆发生交通事故, 造成乘车人×××腰部左侧横突骨折, 其住院治疗共支付医疗费8754.55元… *Events*: (trigger word: 骨折 event: 受伤), (trigger word: 支付, event: 支付/给付) ***Translation:*** *…The defendant××× 's mother××× had a traffic accident in the vehicle driven by the driver××× during the delivery of meals at××× Company in××× City, resulting in the fracture of the left transverse process of the passenger××× 's waist, He paid a total of 8754.55 yuan in medical expenses for hospitalization…* *Events: (trigger word: fracture event: injury), (trigger word: paid, event: payment) …*

Acknowledgements This work was supported by Humanities and Social Science Planning Fund [Grant Numbers 21YJAZH013] from the Ministry of Education, China.

Declarations

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

References

Ali F, Kwak D, Khan P et al (2019) Transportation sentiment analysis using word embedding and ontology-based topic modeling. Knowl Based Syst 174:27–42. https://doi.org/10.1016/j.knosys.2019.02.033

Bench-Capon TJM, Araszkiewicz M, Ashley KD et al (2012) A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. Artif Intell Law 20:215–319. https://doi.org/10.1007/s10506-012-9131-x

Bhattacharya P, Ghosh K, Ghosh S, et al (2019) Overview of the FIRE 2019 AILA Track: Artificial Intelligence for Legal Assistance. In: Mehta P, Rosso P, Majumder P, Mitra M (eds) Working Notes of FIRE 2019 - Forum for Information Retrieval Evaluation, Kolkata, India, December 12–15, 2019. CEUR-WS.org, pp 1–12

Bhattacharya P, Ghosh K, Pal A, Ghosh S (2020) Hier-SPCNet: A Legal Statute Hierarchy-based Heterogeneous Network for Computing Legal Case Document Similarity. In: Huang JX, Chang Y, Cheng X, et al. (eds) Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25–30, 2020. ACM, pp 1657–1660

Bi S, Ali Z, Wang M et al (2022) Learning heterogeneous graph embedding for Chinese legal document similarity. Knowl Based Syst 250:109046. https://doi.org/10.1016/j.knosys.2022.109046

Chalkidis I, Fergadiotis M, Malakasiotis P, et al (2020) LEGAL-BERT: The Muppets straight out of Law School. CoRR abs/2010.02559:

Chen Y, Sun Y, Yang Z, Lin H (2020) Joint entity and relation extraction for legal documents with legal feature enhancement. In: Scott D, Bel N, Zong C (eds) Proceedings of the 28th international conference on computational linguistics, COLING 2020, Barcelona, Spain (Online), December 8–13, 2020. International Committee on Computational Linguistics, pp 1561–1571

Choi H, Kim J, Joe S, Gwon Y (2020) Evaluation of BERT and ALBERT sentence embedding performance on downstream NLP tasks. In: 25th International conference on pattern recognition, ICPR 2020, Virtual Event/Milan, Italy, January 10–15, 2021. IEEE, pp 5482–5487

Deng S, Zhang N, Li L, et al (2021) OntoED: Low-resource Event Detection with Ontology Embedding. In: Zong C, Xia F, Li W, Navigli R (eds) Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1–6, 2021. Association for Computational Linguistics, pp 2828–2839

Devlin J, Chang M-W, Lee K, Toutanova K (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Burstein J, Doran C, Solorio T (eds) Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers). Association for Computational Linguistics, pp 4171–4186

Feng Y, Li C, Ng V (2022) Legal judgment prediction via event extraction with constraints. In: Muresan S, Nakov P, Villavicencio A (eds) Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22–27, 2022. Association for Computational Linguistics, pp 648–664

Hong Z, Zhou Q, Zhang R, et al (2020) Legal feature enhanced semantic matching network for similar case matching. In: 2020 International joint conference on neural networks, IJCNN 2020, Glasgow, United Kingdom, July 19–24, 2020. IEEE, pp 1–8

Hu Z, Li X, Tu C, et al (2018) Few-shot charge prediction with discriminative legal attributes. In: Bender EM, Derczynski L, Isabelle P (eds) Proceedings of the 27th international conference on computational linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20–26, 2018. Association for Computational Linguistics, pp 487–498

Hu X, Wu X, Shu Y, Qu Y (2022) Logical form generation via multi-task learning for complex question answering over knowledge bases. In: Calzolari N, Huang C-R, Kim H, et al. (eds) Proceedings of the 29th international conference on computational linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12–17, 2022. International Committee on Computational Linguistics, pp 1687–1696

Järvelin K, Kekäläinen J (2002) Cumulated gain-based evaluation of IR techniques. ACM Trans Inf Syst 20:422–446. https://doi.org/10.1145/582415.582418

Jiang J-Y, Zhang M, Li C, et al (2019) Semantic text matching for long-form documents. In: Liu L, White RW, Mantrach A, et al. (eds) The world wide web conference, WWW 2019, San Francisco, CA, USA, May 13–17, 2019. ACM, pp 795–806

Kim M-Y, Lu Y, Goebel R (2017) Textual entailment in legal bar exam question answering using deep siamese networks. In: Arai S, Kojima K, Mineshima K, et al. (eds) New frontiers in artificial intelligence - JSAI-isAI workshops, JURISIN, SKL, AI-Biz, LENLS, AAA, SCIDOCA, kNeXI, Tsukuba, Tokyo, Japan, November 13-15, 2017, Revised Selected Papers. Springer, pp 35–48

Kim Y (2014) Convolutional Neural Networks for Sentence Classification. In: Moschitti A, Pang B, Daelemans W (eds) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. ACL, pp 1746–1751

Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Bengio Y, LeCun Y (eds) 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings

Kusner MJ, Sun Y, Kolkin NI, Weinberger KQ (2015) From word embeddings to document distances. In: Bach FR, Blei DM (eds) Proceedings of the 32nd international conference on machine learning, ICML 2015, Lille, France, 6–11 July 2015. JMLR.org, pp 957–966

Lafferty JD, McCallum A, Pereira FCN (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Brodley CE, Danyluk AP (eds) Proceedings of the eighteenth international conference on machine learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28–July 1, 2001. Morgan Kaufmann, pp 282–289

Lample G, Ballesteros M, Subramanian S, et al (2016) Neural architectures for named entity recognition. In: Knight K, Nenkova A, Rambow O (eds) NAACL HLT 2016, The 2016 conference of the North American chapter of the association for computational linguistics: human language technologies, San Diego California, USA, June 12–17, 2016. The Association for Computational Linguistics, pp 260–270

Li Q, Ji H, Huang L (2013) Joint event extraction via structured prediction with global features. In: Proceedings of the 51st annual meeting of the association for computational linguistics, ACL 2013, 4–9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers. The Association for Computer Linguistics, pp 73–82

Li C, Sheng Y, Ge J, Luo B (2019) Apply event extraction techniques to the judicial field. In: Harle R, Farrahi K, Lane ND (eds) Proceedings of the 2019 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2019 ACM international symposium on wearable computers, UbiComp/ISWC 2019 Adjunct, London, UK, September 9–13, 2019. ACM, pp 492–497

Li Q, Zhang Q, Yao J, Zhang Y (2020) Event extraction for criminal legal text. In: Chen E, Antoniou G (eds) 2020 IEEE international conference on knowledge graph, ICKG 2020, Online, August 9–11, 2020. IEEE, pp 573–580

Li R, Zhao W, Yang C, Su S (2021) Treasures outside contexts: improving event detection via global statistics. In: Moens M-F, Huang X, Specia L, Yih SW (eds) Proceedings of the 2021 conference on empirical methods in natural language processing, EMNLP 2021, Virtual Event/Punta Cana, Dominican Republic, 7–11 November, 2021. Association for Computational Linguistics, pp 2625–2635

Liu B, Wu Y, Zhang F et al (2022) Query generation and buffer mechanism: towards a better conversational agent for legal case retrieval. Inf Process Manag 59:103051. https://doi.org/10.1016/j.ipm.2022.103051

Lv J, Zhang Z, Jin L et al (2021) HGEED: Hierarchical graph enhanced event detection. Neurocomputing 453:141–150. https://doi.org/10.1016/j.neucom.2021.04.087

Ma Y, Shao Y, Liu B, et al (2021a) Retrieving legal cases from a large-scale candidate corpus. Proceedings of the Eighth International Competition on Legal Information Extraction/Entailment, COLIEE2021

Ma Y, Shao Y, Wu Y, et al (2021b) LeCaRD: a legal case retrieval dataset for Chinese law system. In: Diaz F, Shah C, Suel T, et al. (eds) SIGIR'21: The 44th international ACM SIGIR conference on research and development in information retrieval, virtual event, Canada, July 11–15, 2021. ACM, pp 2342–2348

Mandal A, Ghosh K, Ghosh S, Mandal S (2021) Unsupervised approaches for measuring textual similarity between legal court case reports. Artif Intell Law 29:417–451. https://doi.org/10.1007/s10506-020-09280-2

Mandal A, Chaki R, Saha S, et al (2017) Measuring similarity among legal court case documents. In: Chakraborty PP, Gupta M, Dey L, Roy S (eds) Proceedings of the 10th annual ACM India compute conference, Compute 2017, Bhopal, India, November 16–18, 2017. ACM, pp 1–9

McClosky D, Surdeanu M, Manning CD (2011) Event extraction as dependency parsing for BioNLP 2011. In: Tsujii J, Kim J-D, Pyysalo S (eds) Proceedings of BioNLP shared task 2011 workshop, Portland, Oregon, USA, June 24, 2011. Association for Computational Linguistics, pp 41–45

Minocha A, Singh N, Srivastava A (2015) Finding relevant Indian judgments using dispersion of citation network. In: Gangemi A, Leonardi S, Panconesi A (eds) Proceedings of the 24th international conference on world wide web companion, WWW 2015, Florence, Italy, May 18–22, 2015 - Companion Volume. ACM, pp 1085–1088

Neculoiu P, Versteegh M, Rotaru M (2016) Learning text similarity with Siamese recurrent networks. In: Blunsom P, Cho K, Cohen SB, et al. (eds) Proceedings of the 1st workshop on representation learning for NLP, Rep4NLP@ACL 2016, Berlin, Germany, August 11, 2016. Association for Computational Linguistics, pp 148–157

Oard DW, Webber W (2013) Information retrieval for E-discovery. Found Trends Inf Retr 7:99–237. https://doi.org/10.1561/1500000025

Ponte JM, Croft WB (2017) A language modeling approach to information retrieval. SIGIR Forum 51:202–208. https://doi.org/10.1145/3130348.3130368

Rabelo J, Kim M-Y, Goebel R (2019) Combining similarity and transformer methods for case law entailment. In: Proceedings of the seventeenth international conference on artificial intelligence and law, ICAIL 2019, Montreal, QC, Canada, June 17–21, 2019. ACM, pp 290–296

Rabelo J, Kim M-Y, Goebel R, et al (2020a) A summary of the COLIEE 2019 competition. In: New frontiers in artificial intelligence: JSAI-isAI international workshops, JURISIN, AI-Biz, LENLS, Kansei-AI, Yokohama, Japan, November 10–12, 2019, Revised Selected Papers 10. Springer, pp 34–49

Rabelo J, Kim M-Y, Goebel R, et al (2020b) COLIEE 2020: Methods for Legal Document Retrieval and Entailment. In: Okazaki N, Yada K, Satoh K, Mineshima K (eds) New Frontiers in Artificial Intelligence - JSAI-isAI 2020 Workshops, JURISIN, LENLS 2020 Workshops, Virtual Event, November 15-17, 2020, Revised Selected Papers. Springer, pp 196–210

Robertson SE, Walker S (1994) Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In: SIGIR'94. Springer, pp 232–241

Röttger P, Pierrehumbert JB (2021) Temporal adaptation of BERT and performance on downstream document classification: insights from social media. In: Moens M-F, Huang X, Specia L, Yih SW (eds) Findings of the association for computational linguistics: EMNLP 2021, Virtual Event/Punta Cana, Dominican Republic, 16–20 November, 2021. Association for Computational Linguistics, pp 2400–2412

Salton G, Buckley C (1988) Term-weighting approaches in automatic text retrieval. Inf Process Manag 24:513–523. https://doi.org/10.1016/0306-4573(88)90021-0

Saravanan M, Ravindran B, Raman S (2009) Improving legal information retrieval using an ontological framework. Artif Intell Law 17:101–124. https://doi.org/10.1007/s10506-009-9075-y

Sener O, Koltun V (2018) Multi-task learning as multi-objective optimization. In: Bengio S, Wallach HM, Larochelle H, et al. (eds) Advances in neural information processing systems 31: annual conference on neural information processing systems 2018, NeurIPS 2018, December 3–8, 2018, Montréal, Canada. pp 525–536

Shao Y, Mao J, Liu Y, et al (2020) BERT-PLI: modeling paragraph-level interactions for legal case retrieval. In: Bessiere C (ed) Proceedings of the twenty-ninth international joint conference on artificial intelligence, IJCAI 2020. ijcai.org, pp 3501–3507

Shen S, Qi G, Li Z, et al (2020) Hierarchical Chinese legal event extraction via pedal attention mechanism. In: Scott D, Bel N, Zong C (eds) Proceedings of the 28th international conference on

computational linguistics, COLING 2020, Barcelona, Spain (Online), December 8–13, 2020. International Committee on Computational Linguistics, pp 100–113
Si J, Peng X, Li C, et al (2022) Generating disentangled arguments with prompts: a simple event extraction framework that works. In: IEEE international conference on acoustics, speech and signal processing, ICASSP 2022, Virtual and Singapore, 23–27 May 2022. IEEE, pp 6342–6346
Souza E, Vitório D, Moriyama G, et al (2021) An information retrieval pipeline for legislative documents from the Brazilian chamber of deputies. In: Schweighofer E (ed) Legal knowledge and information systems - JURIX 2021: the thirty-fourth annual conference, Vilnius, Lithuania, 8-10 December 2021. IOS Press, pp 119–126
Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9:293–300. https://doi.org/10.1023/A:1018628609742
van Opijnen M, Santos C (2017) On the concept of relevance in legal information retrieval. Artif Intell Law 25:65–87. https://doi.org/10.1007/s10506-017-9195-8
Vaswani A, Shazeer N, Parmar N, et al (2017) Attention is all you need. In: Guyon I, Luxburg U von, Bengio S, et al. (eds) Advances in neural information processing systems 30: annual conference on neural information processing systems 2017, December 4–9, 2017, Long Beach, CA, USA. pp 5998–6008
Wang Z, Hamza W, Florian R (2017) Bilateral multi-perspective matching for natural language sentences. In: Sierra C (ed) Proceedings of the twenty-sixth international joint conference on artificial intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017. ijcai.org, pp 4144–4150
Wang X, Han X, Liu Z, et al (2019) Adversarial training for weakly supervised event detection. In: Burstein J, Doran C, Solorio T (eds) Proceedings of the 2019 Conference of the North American chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers). Association for Computational Linguistics, pp 998–1008
Wehnert S, Sudhi V, Dureja S, et al (2021) Legal norm retrieval with variations of the bert model combined with TF-IDF vectorization. In: Maranhão J, Wyner AZ (eds) ICAIL'21: eighteenth international conference for artificial intelligence and law, São Paulo Brazil, June 21–25, 2021. ACM, pp 285–294
Wu T-H, Kao B, Chan F, et al (2021) Semantic search and summarization of judgments using topic modeling. In: Schweighofer E (ed) Legal knowledge and information systems - JURIX 2021: the thirty-fourth annual conference, Vilnius, Lithuania, 8-10 December 2021. IOS Press, pp 100–106
Xiao C, Hu X, Liu Z et al (2021) Lawformer: a pre-trained language model for Chinese legal long documents. AI Open 2:79–84. https://doi.org/10.1016/j.aiopen.2021.06.003
Xiao C, Zhong H, Guo Z, et al (2019) CAIL2019-SCM: a dataset of similar case matching in legal domain. CoRR abs/1911.08962
Yang J, Ma W, Zhang M, et al (2022) LegalGNN: Legal Information Enhanced Graph Neural Network for Recommendation. ACM Trans Inf Syst 40:33:1–33:29. https://doi.org/10.1145/3469887
Yao F, Xiao C, Wang X, et al (2022) LEVEN: a large-scale Chinese legal event detection dataset. In: Muresan S, Nakov P, Villavicencio A (eds) Findings of the association for computational linguistics: ACL 2022, Dublin, Ireland, May 22–27, 2022. Association for Computational Linguistics, pp 183–201
Zhao R, Mao K (2018) Fuzzy bag-of-words model for document representation. IEEE Trans Fuzzy Syst 26:794–804. https://doi.org/10.1109/TFUZZ.2017.2690222
Zhong H, Zhang Z, Liu Z, Sun M (2019) Open Chinese language pre-trained model zoo
Zhong H, Xiao C, Tu C, et al (2020) How does NLP benefit legal system: a summary of legal artificial intelligence. In: Jurafsky D, Chai J, Schluter N, Tetreault JR (eds) Proceedings of the 58th annual meeting of the association for computational linguistics, ACL 2020, Online, July 5–10, 2020. Association for Computational Linguistics, pp 5218–5230