BERT-PLI: Modeling Paragraph-Level Interactions for Legal Case Retrieval

Yunqiu Shao[1] , **Jiaxin Mao**[1] , **Yiqun Liu**[1*] , **Weizhi Ma**[1] , **Ken Satoh**[2] , **Min Zhang**[1] and **Shaoping Ma**[1]

[1]BNRist, DCST, Tsinghua University, Beijing, China
[2]National Institute of Informatics, Tokyo, Japan
yiqunliu@tsinghua.edu.cn

Abstract

Legal case retrieval is a specialized IR task that involves retrieving supporting cases given a query case. Compared with traditional ad-hoc text retrieval, the legal case retrieval task is more challenging since the query case is much longer and more complex than common keyword queries. Besides that, the definition of relevance between a query case and a supporting case is beyond general topical relevance and it is therefore difficult to construct a large-scale case retrieval dataset, especially one with accurate relevance judgments. To address these challenges, we propose BERT-PLI, a novel model that utilizes BERT to capture the semantic relationships at the *paragraph-level* and then infers the relevance between two cases by aggregating *paragraph-level interactions*. We fine-tune the BERT model with a relatively small-scale case law entailment dataset to adapt it to the legal scenario and employ a cascade framework to reduce the computational cost. We conduct extensive experiments on the benchmark of the relevant case retrieval task in COLIEE 2019. Experimental results demonstrate that our proposed method outperforms existing solutions.

1 Introduction

Precedents (case laws) are a primary source of laws in the common law system, which is fundamental for a lawyer's court preparations. With the rapid increase of digitized legal documents, it takes great efforts of legal practitioners to search for relevant cases manually. Given this situation, an automatic retrieval system that identifies relevant prior cases will greatly alleviate the heavy document works. Therefore, case law retrieval is an important research issue for both IR and legal communities. In recent years, there have been a number of benchmark efforts on the topic of Legal Information Retrieval, e.g., Legal TREC [Oard *et al.*, 2013], AILA [Bhattacharya *et al.*, 2019], COLIEE [Rabelo *et al.*, 2019], etc.

*Corresponding Author

The development of retrieval models sits at the core of IR researches. The legal case retrieval scenario, which aims to identify relevant prior cases given a query case, can be also viewed as a specific application of retrieval models. However, the legal case retrieval task is different from traditional ad-hoc text retrieval in several aspects, including the length of query and candidate texts, the definition of relevance, and the accessibility of legal datasets. Therefore, existing methods developed for IR tasks face a few serious challenges in this task, such as:

Challenge 1. Both the query and candidate cases involve extreme long texts. For example, cases in COLIEE 2019 Task 1 contain around 3,000 words on average. It is challenging for representation learning methods to represent the long document well in a limited semantic space, and it is also difficult for the matching function learning methods to construct and aggregate matching signals.

Challenge 2. The definition of relevance in the legal scenario is somehow beyond the general definition of topical relevance [Van Opijnen and Santos, 2017]. Relevant cases are those that can support the decision of the current case, which usually involve similar situations and suitable statutes. Therefore, it is crucial to identify the similarities in the aspects of legal issues and legal processes of the cases, which calls for semantic understanding of whole documents.

Challenge 3. Collecting a large dataset for this task can be challenging if not impossible. For one thing, downloading large-scale legal documents is restricted in many law systems. For another, it is quite expensive to obtain accurate relevance judgments since it requires expert knowledge in the legal domain. The lack of data brings obstacles to the training process of deep neural models.

To tackle the above challenges, we propose BERT-PLI, a novel model that utilizes BERT [Devlin *et al.*, 2018] to model **P**aragraph-**L**evel **I**nteractions for legal case retrieval. For modeling of long texts (**Challenge 1**), we break the documents into paragraphs and infer the relationship between cases from a fine-grained perspective, which allows exploiting the information of the full document instead of the truncated or summarized one. Beyond term-level matching, we model the semantic interactions between two paragraphs with BERT (**Challenge 2**). Moreover, we adapt the BERT model to the legal scenario by fine-tuning it on a sentence pair classi-

fication task with a small-scale legal case entailment dataset (**Challenge 3**). In practice, we employ the cascade framework to avoid high computational cost and arrange the model in a multi-stage pipeline for legal case retrieval. Specifically, we select top-K candidates according to the BM25 rankings and fine-tune BERT with an extra legal dataset before applying BERT-PLI to relevance prediction. Experiments are conducted on the COLIEE 2019 [Rabelo *et al.*, 2019] legal case retrieval task and the results demonstrate the effectiveness of our proposed method.

2 Related Work

A large number of retrieval models, especially for ad-hoc text retrieval, have been proposed in the past decades. Traditional bag-of-words IR models, including VSM [Salton and Buckley, 1988], BM25 [Robertson and Walker, 1994], and LMIR [Song and Croft, 1999], which are widely applied in search systems, are mostly based on term-level matching. Since the mid-2000s, LTR (Learning to Rank) methods [Liu and others, 2009], which are driven heavily by manual feature engineering, have been well studied and utilized by commercial web search engines as well.

In recent years, the development of deep learning has also inspired applications of neural models in IR. Generally, the methods can be categorized into two types [Xu *et al.*, 2018], methods of representation learning and methods of matching function learning. Based on the idea of representation learning, queries and documents are represented in the latent space by deep learning models, and the query-document relevance score is calculated based on their latent representations with a vector space scoring function, e.g., cosine similarity. Various neural network models have been applied to this task. For instance, DSSM [Hu *et al.*, 2014] utilized DNN in the early stage. Further, some studies exploit CNNs to capture the local interactions [Hu *et al.*, 2014; Shen *et al.*, 2014], while some studies apply RNNs to modeling text sequences [Palangi *et al.*, 2016]. However, most of the model structures are not designed for representing long documents. In particular, it is difficult for CNN models to represent the complex global semantic information while RNN models tend to forget important signals when dealing with a long sequence. On the other hand, matching function learning methods first construct a matching matrix or capture local interactions based on the word-level matching, and then use neural networks to discover the high-level matching patterns and aggregate the final relevance score. Well-known models include ARC-II [Hu *et al.*, 2014], MatchPyramid [Pang *et al.*, 2016], and Match-SRNN [Wan *et al.*, 2016]. Although these models work well for ad-hoc text retrieval, where the query is quite short, their performances are restricted in the scenario of legal case retrieval due to its quadratic time and memory complexity in constructing the whole interaction matrix. Since BERT [Devlin *et al.*, 2018] has reached state-of-art performance in 11 NLP tasks, pre-trained language models have drawn great attention in many fields related to NLP. Recently, some works have shed light on the application of BERT to ad-hoc retrieval [Dai and Callan, 2019; Yilmaz *et al.*, 2019] by modeling evidence in query-sentence or query-passage pairs. As for legal case retrieval, it is still worth investigating how to utilize BERT to model the relationship between long case documents.

Finding relevant materials is fundamental in the legal field. In countries following the common law system, prior cases are one of the primary sources of law. Thus, legal case retrieval is an important research topic. Meanwhile, legal case retrieval is challenging because it is different from ad-hoc retrieval in various aspects, including the definition of relevance in law [Van Opijnen and Santos, 2017] and the characteristics of legal documents, such as the document length, professional legal expressions, and the logical structures behind natural languages [Turtle, 1995]. A variety of approaches as well as expert knowledge are involved in this task [Bench-Capon *et al.*, 2012], e.g., logical analysis, lexical matching, distributed representation, etc. For instance, decomposition of legal issues [Zeng *et al.*, 2005], ontological frameworks [Saravanan *et al.*, 2009], and link analysis [Monroy *et al.*, 2013] have been explored. Generally speaking, methods can be grouped into two broad categories: those based on manual knowledge engineering (KE) and those based on natural language processing (NLP) [Maxwell and Schafer, 2008]. The competitions held recently, such as COLIEE, are mostly aimed at exploring the application of NLP-based methods and providing benchmarks for the legal case retrieval task. In COLIEE 2019, both traditional retrieval models and neural models are explored in the legal case retrieval task. Specifically, [Tran *et al.*, 2019] combined distributed representation with lexical matching features via LTR algorithms while [Rossi and Kanoulas, 2019] utilized BERT with the help of automatic summarization algorithms.

3 Method

3.1 Task Description

The legal case retrieval task involves finding prior cases that should be "noticed" concerning a given query case in the set of candidate cases [Rabelo *et al.*, 2019]. "Noticed case" is a legal technical term denoting that a precedent is relevant to a query case, in other words, it supports the decision of a query case. Formally, given a query case q, and a set of candidate cases $D = \{d_1, d_2, \ldots, d_n\}$, the task is to identify the supporting cases $D^* = \{d_i^* \mid d_i^* \in D \wedge noticed(d_i^*, q)\}$, where $noticed(d_i^*, q)$ denotes that d_i^* should be noticed given the query case q. Both the query and the candidates are legal documents containing long texts, which consist of the facts in a case.

3.2 Architecture Overview

In general, we deal with the legal case retrieval task within a multi-stage pipeline inspired by the cascade framework. As illustrated in Figure 1, it consists of three stages. In Stage 1, we select top-K candidates from the initial candidate corpus with respect to the query case q according to BM25 scores. In Stage 2, we fine-tune the BERT model on a sentence pair classification task with a legal case entailment dataset in order to adapt it to modeling semantic relationships between legal paragraphs. In the final stage, BERT-PLI conducts rele-

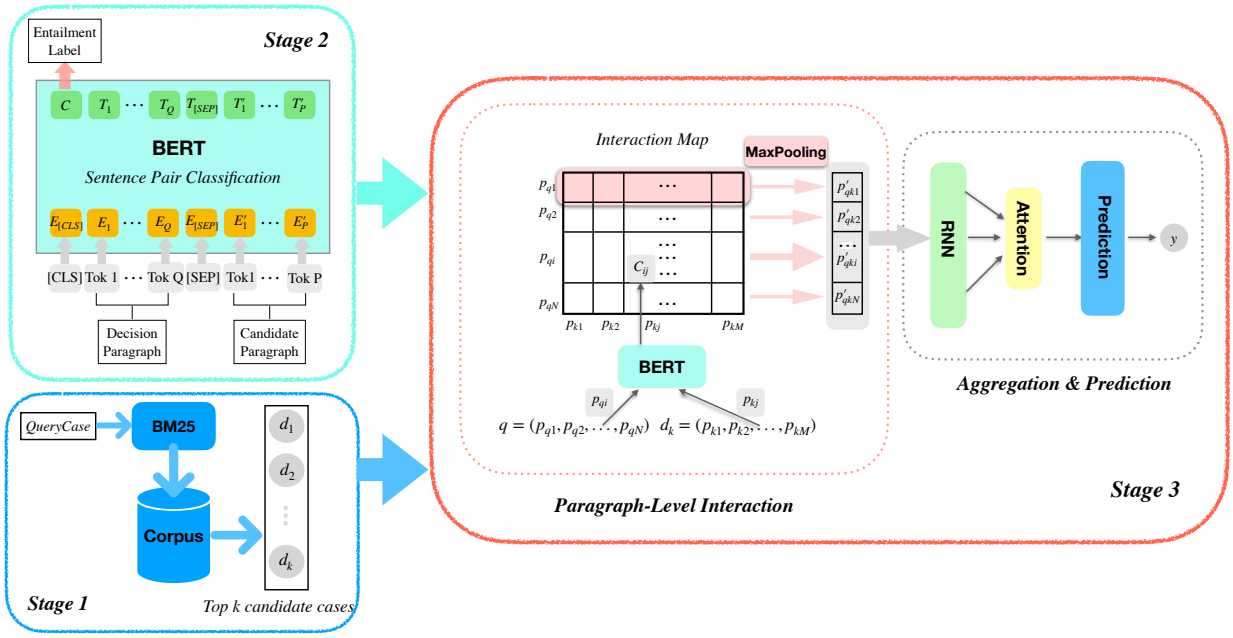

Figure 1: An illustration of the multi-stage pipeline.

vance prediction with the fine-tuned BERT (Stage 2) among the selected candidates (Stage 1).

3.3 Stage 1: BM25 Selection

Deep learning models are usually time-consuming and resource-consuming. Considering the computational cost, we employ the cascade framework which utilizes BM25 to prune the set of candidates. The BM25 model is implemented according to the standard scoring function [Robertson and Walker, 1994]. This stage inevitably hurts both recall and precision, and we mainly pay attention to optimizing recall at this stage since the downstream models can further discard irrelevant documents.

3.4 Stage 2: BERT Fine-tuning

Fine-tuning is relatively inexpensive compared with the pre-training procedure, which allows BERT to model specific tasks with small datasets [Devlin *et al.*, 2018]. Therefore, before applying BERT to infer the relationship of case paragraphs, we fine-tune it with a small-scale legal case entailment dataset provided by COLIEE 2019 Task 2 (**Challenge 3**). This task involves identifying the paragraphs that entail the decision paragraph of a query case from a given relevant case. Fine-tuning on this task enables BERT to infer the supportive relationships between paragraphs, which is useful for the legal case retrieval task.

We fine-tune all parameters of BERT on a sentence pair classification task in an end-to-end fashion. The input is composed of the decision paragraph of a query case and a candidate paragraph in the relevant case. The text pair is separated by the `[SEP]` token and a `[CLS]` token is prepended to the text pair. As for the output, we feed the final hidden state vector corresponding to the first input token (`[CLS]`) into a classification layer. In this task, we use a fully-connected layer to do binary classification, optimizing a cross-entropy loss, written as $-(y\log(\hat{y}) + (1-y)\log(1-\hat{y}))$.

3.5 Stage 3: BERT-PLI

To tackle the challenge brought by long and complex documents, we first break a document into paragraphs (**Challenge 1**) and model the interactions between paragraphs in the semantic level (**Challenge 2**). In Figure 1 (Stage 3), the query q and one of the candidate document d_k are represented by paragraphs, denoted as $q = (p_{q1}, p_{q2}, \ldots, p_{qN})$ and $d_k = (p_{k1}, p_{k2}, \ldots, p_{kM})$, where N and M denote the total numbers of paragraphs in q and d_k, respectively. For each paragraph in q and d_k, we construct a paragraph pair (p_{qi}, p_{kj}), where $1 \le i \le N$ and $1 \le j \le M$, as the input of BERT, along with the reserved tokens (i.e. `[CLS]` and `[SEP]`). The final hidden state vector of the token `[CLS]` is viewed as the aggregate representation of the input paragraph pair. In that way, we can get an interaction map of paragraphs, in which each component $\boldsymbol{C}_{ij}$, $\boldsymbol{C}_{ij} \in \mathbb{R}^{H_B}$, models the semantic relationship between the query paragraph p_{qi} and the candidate paragraph p_{kj}.

For each paragraph of the query, we capture the strongest matching signals with the candidate document using the max-pooling strategy, and hence get a sequence of vectors, denoted as $\boldsymbol{p}'_{qk} = [\boldsymbol{p}'_{qk1}, \boldsymbol{p}'_{qk2}, \ldots, \boldsymbol{p}'_{qkN}]$. Each component $\boldsymbol{p}'_{qki}$ aggregates the interactions of all of paragraphs in the candidate document corresponding to the i-th query paragraph as follows:

$$\boldsymbol{p}'_{qki} = \mathrm{MaxPool}(\boldsymbol{C}_{i1}, \boldsymbol{C}_{i2}, \ldots, \boldsymbol{C}_{iM}),\ \boldsymbol{p}'_{qki} \in \mathbb{R}^{H_B}. \quad (1)$$

We use an RNN structure to further encode the representation sequence. Assuming that the legal document always follows a certain reasoning order, we consider the forward

RNN in this model. In the forward pass, RNN generates a sequence of hidden states:

$$\boldsymbol{h}_{qk} = [\boldsymbol{h}_{qk1}, \boldsymbol{h}_{qk2}, \ldots, \boldsymbol{h}_{qkN}],\ \boldsymbol{h}_{qki} \in \mathbb{R}^{H_R}, \quad (2)$$

where $\boldsymbol{h}_{qki} = \mathrm{RNN}(\boldsymbol{h}_{qk(i-1)}, \boldsymbol{p}'_{qki})$ is generated by LSTM or GRU in practice.

The attention mechanism is also employed to infer the importance of each position. The attention weight of each position is measured by:

$$\alpha_{qki} = \frac{\exp(\boldsymbol{h}_{qki} \cdot \boldsymbol{u}_{qk})}{\sum_{i'} \exp(\boldsymbol{h}_{qki'} \cdot \boldsymbol{u}_{qk})}, \quad (3)$$

where $\boldsymbol{h}_{qki}$ is the i-th hidden state given by the forward RNN and $\boldsymbol{u}_{qk}$ is generated as follows:

$$\boldsymbol{u}_{qk} = \boldsymbol{W}_u \cdot \mathrm{MaxPool}(\boldsymbol{h}_{qk}) + \boldsymbol{b}_u, \quad (4)$$

where $\boldsymbol{W}_u \in \mathbb{R}^{H_R \times H_R}$, $\boldsymbol{b}_u \in \mathbb{R}^{H_R}$. We can then get the document-level representation via attentive aggregation:

$$\boldsymbol{d}_{qk} = \sum_i \alpha_{qki} \boldsymbol{h}_{qki}. \quad (5)$$

Finally, the representation $\boldsymbol{d}_{qk}$ is passed through a fully-connected layer followed by a softmax function to make prediction as follows:

$$\hat{\boldsymbol{y}}_{qk} = \mathrm{softmax}(\boldsymbol{W}_p \cdot \boldsymbol{d}_{qk} + \boldsymbol{b}_p), \quad (6)$$

where $\boldsymbol{W}_p \in \mathbb{R}^{|R| \times H_R}$, $\boldsymbol{b}_p \in \mathbb{R}^{|R|}$, and R denotes the set of relevance labels, e.g., $R = \{0, 1\}$ and $|R| = 2$. During the training procedure, we optimize the following cross-entropy loss:

$$\mathcal{L}_{qk}\left(\hat{\mathbf{y}}_{qk}, \mathbf{y}_{qk}\right) = -\sum_{r=1}^{|R|} \mathbf{y}_{qkr} \log\left(\hat{\mathbf{y}}_{qkr}\right), \quad (7)$$

In the practice of the legal case retrieval task, BERT-PLI is combined with the two stages mentioned above. To be specific, the candidate documents at the input of BERT-PLI are those selected by BM25 in Stage 1. After Stage 2, we consider that the BERT is able to well represent the paragraph-level semantic relationships in legal case documents. Hence, we utilize the fine-tuned parameters directly without updating them when training BERT-PLI. The multi-stage operations make the training process easy and affordable. As for testing, we return the ones that are predicted as relevant cases by BERT-PLI corresponding to a given query case.

4 Experiments

4.1 Datasets and Evaluation Metrics

Our experiments are conducted based on the COLIEE 2019 datasets [Rabelo *et al.*, 2019]. Task 1 is a legal case retrieval task, which involves reading a new case Q and extracting supporting cases $S_1, S_2, \ldots, S_n$ for the decision of Q from the case law corpus. The supporting cases are considered as "relevant cases" or "noticed cases" for the query case Q. Task 2 is a legal case entailment task to identify paragraphs that entail the given decision paragraph of a query case from a given relevant case. Data in both tasks are sampled from a database of predominantly Federal Court of Canada case laws. Table 1 gives a statistical summary of the raw datasets in these two tasks. Task 1 is the main focus of our work, while the data of Task 2 are used to fine-tune BERT in Stage 2.

Task 1	Train	Test
# query case	285	61
# candidate cases / query	200	200
# noticed cases / query	5.21	5.41
Task 2	**Train**	**Test**
# query case	181	44
# candidate paragraphs / query	32.12	32.91
# entailing paragraphs / query	1.12	1.02

Table 1: Summary of datasets in COLIEE 2019 task 1 and task 2.

We follow the evaluation metrics in the competition. Micro-average of precision, recall, and F1 are used.

4.2 Baseline Methods

We compare our model with the following three types of baselines.

- **Traditional bag-of-words retrieval models**, including VSM [Salton and Buckley, 1988], BM25 [Robertson and Walker, 1994], and LMIR [Song and Croft, 1999].
- **Deep retrieval models**. Prior work shows that matching function learning methods usually outperform the representation learning ones [Xu *et al.*, 2018], so we consider two matching function learning models, ARC-II [Hu *et al.*, 2014] and MatchPyramid [Palangi *et al.*, 2016]. Both of them utilize the CNN structure, which is faster than RNN-based models, especially when dealing with long texts. We do not include the state-of-art neural models that involve rich behavioral signals (e.g., click) since we focus on text-based retrieval here.
- **Methods in the competition**. We also compare with the methods of the top 2 teams [Tran *et al.*, 2019; Rossi and Kanoulas, 2019] in the competition. The champion team (named as "JNLP") trained a supervised summarization model based on COLIEE 2018's dataset and applied the model to encoding the case document into a continuous vector. They combined such the summary embeddings with lexical matching features, calculated by ROUGE [Lin, 2004], and learned the document rankings via RankSVM. Another team ranked following JNLP (named as "ILPS") generated summaries by the TextRank algorithm [Mihalcea and Tarau, 2004] first and assessed pairwise relevance by a carefully fine-tuned BERT model, combined with oversampling strategies.

4.3 Experimental Settings

In the raw legal case documents, along with the text body of the case (known as "facts"), some meta information is also provided, such as the court, the date, the head note and so on. However, the types of metadata vary with documents and have a high missing rate, in which case, we build our model based on the text body of a case. Some cases contain both

English and French versions of description, and only the English one is considered in our experiments. As illustrated in Figure 1 (Stage 1), we select candidates according to BM25 scores. We set $K = 50$ (top 50 candidates for each query), considering both the recall and the effectiveness in the following stages. Note that only the cases that happened before the current case could be noticed according to the problem definition, those invalid cases in terms of time are dismissed, which would otherwise be noisy in the candidates. The recall of Stage 1 on the training and testing set are 0.9159 and 0.9273, respectively. The relative high recall suggests that setting $K = 50$ in Stage 1 brings little harm to the overall retrieval performance.

In Stage 2, paragraph pairs are constructed using the decision and candidate paragraphs in Task 2. The paragraphs have no more than 100 words on average and we truncate the words symmetrically if the pair exceeds the maximum input length of BERT. We use the uncased base version of BERT.[1] At first, it is fine-tuned on the training data and tested on the remaining testing data. We use the Adam optimizer and set the learning rate as 10^{-5}. The fine-tuning procedure can coverage after three epochs and $F1$ on the test set reaches 0.6526, which is better than the results of BM25 in this task. After that, we utilize the same hyperparameter settings but merge the training and testing data to fine-tune BERT for 3 epochs from scratch.

In Stage 3, the paragraph segmentation given by the original documents is adopted. Similarly, if the paragraph pair exceeds the maximum input length, we simply truncate the texts. We set $N = 54$ and $M = 40$, which can cover 3/4 of paragraphs in most query and candidate cases. In BERT-PLI, $H_B = 768$, which is determined by the size of the BERT hidden vector. As for RNN, H_R is set as 256 and only one hidden layer is used for both LSTM and GRU. The training set is split into two parts. 20% queries from the training set as well as all of their candidates are treated as the validation set. We train the model on the training data left for no more than 60 epochs and select the best model in the training process according to the F1 measure on the validation set. During the training process, we use the Adam optimizer and set the start learning rate as 10^{-4} with a weight decay of 10^{-6}.

As for the baseline methods, we use a bigram language model with linear smoothing in LMIR while VSM and BM25 are calculated based on the standard scoring functions. ARC-II and MatchPyramid are implemented by MatchZoo [Guo *et al.*, 2019]. We train ARC-II and MatchPyramid in a pairwise way since pairwise training usually outperforms pointwise training. The input length is also restricted due to memory and time limit. We truncate both the query and candidate document to 256 words, which is consistent with the restrictions of BERT. Meanwhile, we use TextRank[2] to generate a summary with a 256-words length limit for each case. JNLP group also provides the summary encoding and the lexical features, and we further conduct experiments based on the provided features using RankSVM.[3]

[1]https://github.com/google-research/bert

[2]https://radimrehurek.com/gensim/

[3]https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

Model	Precision	Recall	F1
BM25	0.5100	0.4636	0.4867
VSM	0.4833	0.4394	0.4603
LMIR	0.5467	0.4970	0.5306
ARC-II	0.1967	0.1788	0.1873
MP	0.1933	0.1758	0.1841
ARC-II$_{sum}$	0.1900	0.1727	0.1810
MP$_{sum}$	0.2233	0.2030	0.2127
BERT-PLI (LSTM)	0.5931	**0.5697**	0.5812
BERT-PLI (GRU)	**0.6026**	**0.5697**	**0.5857**
BERT-PLI$_{org}$ (LSTM)	0.5278	0.4606	0.4919
BERT-PLI$_{org}$ (GRU)	0.4958	0.5364	0.5153

Table 2: Performance of traditional retrieval models, deep retrieval models, and BERT-PLI on the test set, measured by micro-average of precision, recall and F1. MP is the abbreviation for MatchPyramid. sum denotes the model that uses the generated summary as the model input. org denotes the model without Stage 2, which uses the original BERT parameters.

4.4 Results and Analysis

All of the models employ the cascade framework except BM25. In other words, these models conduct further ranking or classification based on the dataset D_1 selected in Stage 1. In particular, all of the deep learning models are trained on the same training data and selected according to F1 on the validation data. For the ranking models, we consider the top 5 results as the relevant ones[4], while for the classification models, we simply use the label given by the model.

Table 2 shows the performance on the whole test set. In the BERT-PLI model, we use LSTM and GRU as the RNN layer respectively. The LSTM and GRU versions achieve similar performance here and both outperform the baseline methods, including the traditional retrieval models and the deep learning ones, by a large margin. The structure of BERT-PLI is able to take the whole case document into consideration. At the same time, it has a better semantic understanding ability than the bag-of-words IR models with the help of BERT and sequence modeling components.

Among the baseline retrieval methods, deep learning retrieval models perform much worse than the traditional retrieval models in this task. Since these deep learning models are mostly designed for the ad-hoc scenario, it is hard for them to handle long text retrieval. The length of the input is restricted in these models. The first 256 words are not enough to represent the document well, so it is not surprising that they perform poorly in this task. In addition to truncation, we also attempt to utilize automatic summarization techniques to shorten the input length. However, it does not result in stable improvements. Even taking the generated summary as the input, the deep models still underperform. Assuming that the legal documents contain plenty of information and complex logic, it is hard to express them well in a much

[4]The threshold 5 is widely used by teams in the competition and is reasonable considering there are about 5 relevant cases per query on average.

Team / Method	Precision	Recall	F1
JNLP	0.6000	0.5545	0.5764
ILPS	**0.6810**	0.4333	0.5296
BERT-PLI (LSTM)	0.5931	**0.5697**	0.5812
BERT-PLI (GRU)	0.6026	**0.5697**	**0.5857**

Table 3: Comparison with the best runs of the top 2 teams in the competition. "JNLP" and "ILPS" are the team names.

shorter summary. Meanwhile, the unsupervised summarization techniques might cause additional information loss and noise. The results emphasize the importance of considering the full document information in the legal case retrieval task. On the other hand, traditional retrieval models give relatively good results. They take advantage of the whole document though they are weak in semantic understanding.

Ablation study. We further investigate the effects of the fine-tuning stage. We use the original parameters of the pre-trained BERT model rather than the fine-tuned one to infer the paragraph interactions. Then the remaining parts of BERT-PLI are trained and evaluated under the same experimental settings. The model without Stage 2 is denoted as BERT-PLI$_{org}$. As shown in Table 2, there is a big drop in the performance for both the LSTM and GRU versions compared with the results of BERT-PLI. Without Stage 2, the model has a similar performance with traditional retrieval models. Recall that we do not update the parameters of BERT during training BERT-PLI. Therefore, fine-tuning is essential for the BERT model to well represent the semantic relationships between paragraphs. In conclusion, the experimental results suggest that it is useful and effective to adapt BERT to a paragraph-level modeling task in the legal domain.

Comparison with results in the competition. Table 3 shows the best results of the top two teams in the competition leaderboard. In terms of the final evaluation results on the test set, our methods (BERT-PLI (GRU/LSTM)) achieve a better recall and F1 score, even though Stage 1 hurts the recall of our approach a bit. The results show that our method can reach a better balance in precision and recall. Further, with the summary encoding and lexical features provided by the JNLP group, we conduct an additional experiment, which combines the outputs of BERT-PLI with their features. The experiment is two-fold. In order to combine the two approaches, we first arrange their model in the cascade framework. Similarly, we conduct the operations of Stage 1 to select candidate cases and get the dataset D_1, which is the same as the one for BERT-PLI. Then, a RankSVM model is trained and the top 5 documents are used as the noticed cases. Since their model combines two types of features, the representation-based features (denoted as "EM") and the term-matching features (denoted as "ROUGE"), we conduct experiments on each type of features separately. As shown in Table 4, the performance of all kinds of features goes down, which might result from that we do not use the same training set (we only use D_1 instead of the whole corpus to train the RankSVM). We further append the probabilities of relevance predicted by BERT-PLI to their features and then apply the RankSVM algorithm. The GRU and LSTM versions achieve similar performance but GRU gives a slightly better result in the prior experiments (Table 2), so we use the probabilities given by BERT-PLI (GRU) as the features. The results are given in the second part of Table 4, which show that combination with our model can lead to improvements in all types of their features. Compared among different types of features, our model can improve the representation-based features by a larger amount. We assume that the embedding features are generated by a summarization model while our method considers the whole document, and these two aspects might be complementary.

Features	Precision	Recall	F1
JNLP's features			
EM	0.4700	0.4273	0.4476
ROUGE	0.4800	0.4364	0.4571
EM-ROUGE	0.4933	0.4485	0.4698
Combined with BERT-PLI outputs			
EM	0.5833	0.5303	0.5566
ROUGE	0.5367	0.4879	0.5111
EM-ROUGE	0.5967	0.5424	0.5683

Table 4: Performances of combining the features of JNLP with the outputs of BERT-PLI (GRU) on the test set.

5 Conclusions

In this paper, we propose to address the problem of legal case retrieval. To tackle the challenge raised by the long and complex legal documents, we introduce a novel model, BERT-PLI[5], which models the paragraph-level interactions of case documents via BERT and then aggregate these interactions to infer the document relevance via a sequential modeling mechanism. We propose to arrange BERT-PLI in a multi-stage pipeline in the practice of legal case retrieval. To be specific, we prune the candidate set according to BM25 rankings in the first stage. In order to enhance the ability to model the semantic relationships between legal paragraphs, we fine-tune the BERT model with an accessible entailment dataset in the legal domain before applying it to BERT-PLI. The ablation study also supports the effectiveness of the fine-tuning stage. Finally, BERT-PLI is employed to further identify the relevant cases with respect to a query case. We conduct extensive experiments on the datasets of COLIEE 2019. The experimental results demonstrate that our approach is effective in legal case retrieval and the combination with BERT-PLI can further improve other models for this task.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (2018YFC0831700), Natural Science Foundation of China (Grant No. 61732008, 61532011, 61902209), Beijing Academy of Artificial Intelligence (BAAI), JST CREST Grant Number JPMJCR1513 and JSPS KAKENHI Grant Number JP17H06103.

[5]https://github.com/ThuYShao/BERT-PLI-IJCAI2020. The implementation has been available.

References

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

[Bhattacharya *et al.*, 2019] Paheli Bhattacharya, Kripabandhu Ghosh, Saptarshi Ghosh, Arindam Pal, Parth Mehta, Arnab Bhattacharya, and Prasenjit Majumder. Overview of the FIRE 2019 AILA track: Artificial intelligence for legal assistance. *FIRE'19*, pages 12–15, 2019.

[Dai and Callan, 2019] Zhuyun Dai and Jamie Callan. Deeper text understanding for IR with contextual neural language modeling. *arXiv:1905.09217*, 2019.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*, 2018.

[Guo *et al.*, 2019] Jiafeng Guo, Yixing Fan, Xiang Ji, and Xueqi Cheng. MatchZoo: A learning, practicing, and developing system for neural text matching. In *SIGIR'19*, pages 1297–1300, New York, NY, USA, 2019. ACM.

[Hu *et al.*, 2014] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *NeurIPS'14*, pages 2042–2050, 2014.

[Lin, 2004] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, 2004.

[Liu and others, 2009] Tie-Yan Liu et al. Learning to rank for information retrieval. *FnTIR*, 3(3):225–331, 2009.

[Maxwell and Schafer, 2008] K Tamsin Maxwell and Burkhard Schafer. Concept and context in legal information retrieval. In *JURIX*, pages 63–72, 2008.

[Mihalcea and Tarau, 2004] Rada Mihalcea and Paul Tarau. TextRank: Bringing order into text. In *EMNLP'04*, pages 404–411, 2004.

[Monroy *et al.*, 2013] Alfredo López Monroy, Hiram Calvo, Alexander Gelbukh, and Georgina García Pacheco. Link analysis for representing and retrieving legal information. In *CICLing'13*, pages 380–393. Springer, 2013.

[Oard *et al.*, 2013] Douglas W Oard, William Webber, et al. Information retrieval for e-discovery. *FnTIR*, 7(2–3):99–237, 2013.

[Palangi *et al.*, 2016] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *TASLP'16*, 24(4):694–707, 2016.

[Pang *et al.*, 2016] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xueqi Cheng. Text matching as image recognition. In *AAAI'16*, 2016.

[Rabelo *et al.*, 2019] Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. COLIEE 2019 overview. In *COLIEE'19*, pages 1–9, 2019.

[Robertson and Walker, 1994] Stephen E Robertson and Steve Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94*, pages 232–241. Springer, 1994.

[Rossi and Kanoulas, 2019] Juline Rossi and Evangelos Kanoulas. Legal information retrieval with generalized language models. In *COLIEE'19*, pages 16–19, 2019.

[Salton and Buckley, 1988] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *IP&M*, 24(5):513–523, 1988.

[Saravanan *et al.*, 2009] M Saravanan, Balaraman Ravindran, and S Raman. Improving legal information retrieval using an ontological framework. *AI & Law*, 17(2):101–124, 2009.

[Shen *et al.*, 2014] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. A latent semantic model with convolutional-pooling structure for information retrieval. In *CIKM'14*, pages 101–110. ACM, 2014.

[Song and Croft, 1999] Fei Song and W Bruce Croft. A general language model for information retrieval. In *CIKM'99*, pages 316–321. ACM, 1999.

[Tran *et al.*, 2019] Vu Tran, Minh Le Nguyen, and Ken Satoh. Building legal case retrieval systems with lexical matching and summarization using a pre-trained phrase scoring model. In *ICAIL'19*, pages 275–282, 2019.

[Turtle, 1995] Howard Turtle. Text retrieval in the legal world. *AI & Law*, 3(1-2):5–54, 1995.

[Van Opijnen and Santos, 2017] Marc Van Opijnen and Cristiana Santos. On the concept of relevance in legal information retrieval. *AI & Law*, 25(1):65–87, 2017.

[Wan *et al.*, 2016] Shengxian Wan, Yanyan Lan, Jun Xu, Jiafeng Guo, Liang Pang, and Xueqi Cheng. Match-SRNN: Modeling the recursive matching structure with spatial RNN. *arXiv:1604.04378*, 2016.

[Xu *et al.*, 2018] Jun Xu, Xiangnan He, and Hang Li. Deep learning for matching in search and recommendation. In *SIGIR'18*, pages 1365–1368. ACM, 2018.

[Yilmaz *et al.*, 2019] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. Cross-domain modeling of sentence-level evidence for document retrieval. In *EMNLP-IJCNLP'19*, pages 3481–3487, 2019.

[Zeng *et al.*, 2005] Yiming Zeng, Ruili Wang, John Zeleznikow, and Elizabeth Kemp. Knowledge representation for the intelligent legal case retrieval. In *KES'05*, pages 339–345. Springer, 2005.