Answering Legal Questions by Learning Neural Attentive Text Representation

Phi Manh Kien[*†] **Ha-Thanh Nguyen**[*‡] **Ngo Xuan Bach**[†]
Vu Tran[‡] **Minh Le Nguyen**[‡] **Tu Minh Phuong**[†]
[†]Department of Computer Science,
Posts and Telecommunications Institute of Technology, Hanoi, Vietnam
[‡]Japan Advanced Institute of Science and Technology
kienpm2205@gmail.com {bachnx,phuongtm}@ptit.edu.vn
{nguyenhathanh,vu.tran,nguyenml}@jaist.ac.jp

Abstract

Text representation plays a vital role in retrieval-based question answering, especially in the legal domain where documents are usually long and complicated. The better the question and the legal documents are represented, the more accurate they are matched. In this paper, we focus on the task of answering legal questions at the article level. Given a legal question, the goal is to retrieve all the correct and valid legal articles, that can be used as the basic to answer the question. We present a retrieval-based model for the task by learning neural attentive text representation. Our text representation method first leverages convolutional neural networks to extract important information in a question and legal articles. Attention mechanisms are then used to represent the question and articles and select appropriate information to align them in a matching process. Experimental results on an annotated corpus consisting of 5,922 Vietnamese legal questions show that our model outperforms state-of-the-art retrieval-based methods for question answering by large margins in terms of both recall and NDCG.

1 Introduction

Question answering in the legal domain is a difficult task due to the complication and topic variety of legal document systems. Giving an accurate answer to a legal question usually requires domain expert knowledge, making this task more challenging even with human beings. Retrieval-based approaches, which return relevant legal documents respect to the input question, are often feasible and more appropriate. Legal documents, however, are usually long and complicated, which takes time to read and locate exact information. Relevant legal articles are, therefore, the expected result of retrieval-based legal question answering systems, where users are easier to find a useful answer than the whole documents.

Similar to an information retrieval system, a typical retrieval-based question answering one usually consists of three main steps: question[1] representation, document representation, and matching, which judges the relevance between the question and documents (Jurafsky and Martin, 2009). Text representation, i.e., question and document representation, therefore, plays a crucial role in such systems. Good text representation methods are expected to be able to extract important information from the input question and documents and be able to select appropriate information to align them in the matching step.

Recently, many successful text representation methods have been proposed thanks to the advancement of deep neural network models. The most popular deep neural architectures for text representation include convolutional neural networks (CNNs) (Kim, 2014; Shen et al., 2014; Severyn and Moschitti, 2015; Vaswani et al., 2017), recurrent neural networks (RNNs) (Mikolov et al., 2011), and their variations such as long short-term memories (LSTMs) (Wang et al., 2016; Palangi et al., 2016; Mueller and Thyagarajan, 2016; Chen et al., 2017; Bach et al., 2019a; Bach et al., 2019b) and gated recurrent units (GRUs) (Tang et al., 2015), attention mechanisms (Vaswani et al., 2017), and pre-trained models like BERT (Devlin et al., 2019), among the others. The main advantage of deep neural networks for text

[*]The first two authors have equal contribution.

[1]In this paper, "question" and "query" are used interchangeably.

Proceedings of the 28th International Conference on Computational Linguistics, pages 988–998
Barcelona, Spain (Online), December 8-13, 2020

representation, as well as for other tasks, is that they can capture various types of information at different levels by integrating multiple neural architectures in a multi-layer model.

In this paper, we deal with the task of retrieval-based legal question answering at the article level, which poses several challenges compared with the task at the document level. The number of legal articles is much more than the number of documents. Moreover, differentiating articles in the same document is very challenging because they usually focus on the same topic and share the vocabulary. Table 1 demonstrates a sample legal question and the expected answer, Article 651 from the 2015 Code of Civil law of Vietnam. By investigating legal questions and articles, we found that just a few sentences in an answer article contain relevant information, and only some phrases in such sentences are matched to the question. To capture such properties, we present a neural attentive text representation method based on convolutional neural networks and attention mechanisms. While the former components are designed to extract important information of the input question and legal articles, the later ones determine the important parts to preresent and match each other.

Table 1: A sample in the dataset with highlighted parts

Question	Do stepchildren have rights of inheritance from the deceased father where there is no will?
Answer	Article 651 from the Code of Civil law of Vietnam (2015).
Article content	Article 651. Heirs at law 1. Heirs at law are categorized in the following order of priority: a) The first level of heirs comprises: spouses, biological parents, adoptive parents, offspring and adopted children of the deceased; b) The second level of heirs comprises: grandparents and siblings of the deceased; and biological grandchildren of the deceased; c) The third level of heirs comprises: biological great-grandparents of the deceased, biological uncles and aunts of the deceased and biological nephews and nieces of the deceased. 2. Heirs at the same level shall be entitled to equal shares of the estate. 3. Heirs at a lower level shall be entitled to inherit where there are no heirs at a higher level because such heirs have died, or because they are not entitled to inherit, have been deprived of the right to inherit or have disclaimed the right to inherit.

The contributions of this paper are two-fold: First, we introduce a retrieval-based legal question answering model by learning neural attentive representations of the input question and legal articles. Second, we show the effectiveness of the proposed model by introducing an annotated corpus and conducting a series of experiments to compare our model with state-of-the-art methods in the field. In the following, we first describe related work in Section 2. Section 3 presents our text representation method and retrieval-based question answering model. Section 4 describes our datasets and experimental setup. Experimental results are presented in Section 5. Finally, we conclude the paper and discuss future work in Section 6.

2 Related Work

Several approaches have been developed to overcome the challenge of the difference between semantics and query morphology in which lexical overlapping may not mean semantical relevance.

Non-neural Approaches These methods, without using neural networks, mainly determine relevance based on the occurrence and frequency of the terms in the query and the documents. They are old-fashioned approaches for text processing, but the ideas in these methods are essential for current state-of-the-art models. The most basic models for text retrieval are logical models (Cooper, 1971). It requires

users to understand the corpus deeply to create their query for exact logical matching. The first idea for overcoming the problem of exact matching in logical models, Luhn et al. (1957), proposed a statistical matching process, which is the origin of the vector space approaches for textual retrieval, for example, term weighting (tf-idf (Salton and Buckley, 1988)).

Neural Approaches Lexical mismatching and ambiguity because of the richness of natural language challenge the non-neural approaches to achieve state-of-the-art results in retrieval tasks. Systems, therefore, need to better understand semantics. Instead of labor features, authors propose effective neural architectures as well as training methods, and let models extract features of different abstraction levels of semantics from data by itself, which expands system capability by data availability.

Before BERT, most of systems are built based on feeding word embeddings pretrained (Word2Vec (Mikolov et al., 2013), or GloVe (Pennington et al., 2014)) or randomized to a neural network (CNN, LSTM, etc.). Palangi et al. (2016) used an LSTM network to construct the semantic vector of a sentence via sentence-pair similarity modeling. Shen et al. (2014) proposed a contextual semantic model based on CNN using both local and global contexts to compresses a word sequence into a low-dimensional vector. Pang et al. (2017) claimed that neural retrieval systems like DSSM (Huang et al., 2013) don't truly understand relevance. The authors, then, proposed DeepRank to mimic the process of humans in assessing relevance.

Adopting BERT, a heavy text-encoding model pretrained on a huge amount of texts, Yilmaz et al. (2019) proposed an ad-hoc retrieval system that can handle document-level retrieval. The system, also, combines lexical matching and BERT scores for better performance. The system, however, requires costly computation resource.

Legal Text Ad-hoc Retrieval Digitization of legal documents has created the need for the invention of a more efficient search system. For legal text, using a keyphrase-based search system is a challenge because of contingent on the end-users' expertise in the subject area, such as being aware of complex legal terms, the classification of case laws and statutes.

In order to reduce the dependence of system users on legal professionals to interpret results as well as the amount of time taken in retrieving information, attempts have been made to invent systems that can automatically classify legal text and queries. However, the majority of problems identified in legal information retrieval systems is that lengthy texts have to be scrutinized before making a meaningful conclusion. Sugathadasa et al. (2018) used neural networks for legal text retrieval. They conducted experiments on a dataset of over 2500 legal cases from various online resources. Using deep learning, the authors propose a system with a TF-IDF page ranking network to construct document embedding. Tran et al. (2020) proposed a deep learning phrase scoring framework to summarize a document into a continuous vector space. Base on their experiments, the authors concluded that the summary features extracted by their model could represent selective essential information from a lengthy legal case. Do et al. (2017) used SVM to rank the candidates. In terms of input features for the model, the author used TF-IDF, Euclidean distance, Manhattan distance, Jaccard distance, Latent semantic indexing, and Latent Dirichlet allocation.

3 Method

3.1 General Approach

The general idea of our approach contains two folds. At first, we encode all articles and queries into vectors. To represent queries and articles, we introduce a sentence encoder and a paragraph encoder, which will be described in the next sections. After that, we use dot product as a similarity function to estimate the relevance between them. Legal articles are often lengthy and contain legal phrases. The related tokens could lie next to each other or have a very long distance in the text. Understanding that characteristic, we use CNN architecture combining with the attention mechanism in our models to capture both local and global context to construct representation vectors.

We train the models using the negative sampling paradigm. We label the article that is related to a query is positive, and the article that is not related is negative. For each positive article of each query,

we choose K samples as negative samples. Then, the models learn to classify these $K + 1$ articles into positive and negative. We use two strategies to select K negative samples. The first strategy is based on Elastic Search, and the second one is randomly selecting.

3.2 Sentence Encoder

Figure 1 shows the architecture of our sentence encoder, which converts a sentence into a representative vector in three layers: word embedding, convolution, and attention.

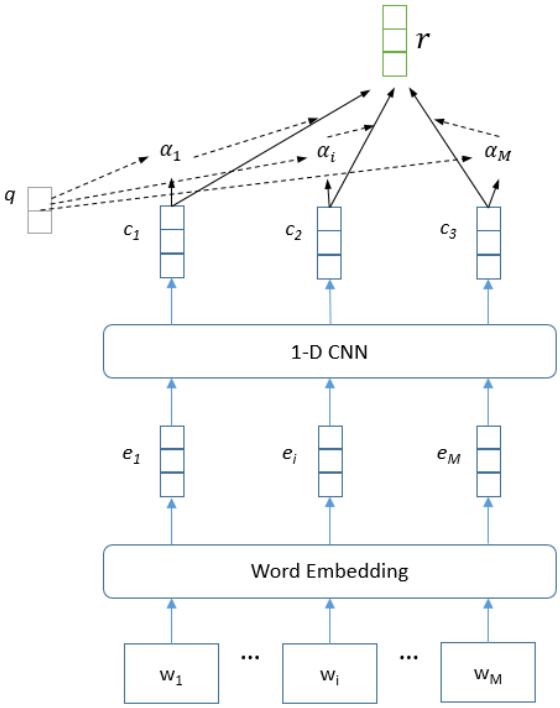

Figure 1: Sentence encoder architecture

Word embedding convert words from the input sequence into a vector via a mapping matrix. Word vectors after the training procedure can reflect the semantic relationship between the words. Let the input sequence is $(w_1, w_2, ..., w_M)$ with M is the length of the sequence, word embedding layer outputs a vector sequence $(e_1, e_2, ..., e_M)$.

The convolutional layer is to make use of local context around the words, which are important to learn the representation of the whole sentence. For example, in the sentence "Xbox One on sale this week," to understand the word "One" correctly as a gaming device, its context words "Xbox" and "on sale" are crucial. The formula calculates the context c_i of word i is as following:

$$c_i = \text{ReLU}\left(F \times e_{(i-K):(i+K)}\right) + b_t \tag{1}$$

in which:

- $e_{(i-K):(i+K)}$ be the vector at the positions from $(i - K)$ to $(i + K)$
- $F \in \mathbb{R}^{N_f \times (2K+1)D}$ and $b_t \in \mathbb{R}^{N_f}$ is kernel and bias of convolutional layer, N_f is number of filters, $2K + 1$ is the window size.

In a sequence, each word contributes differently to the whole meaning. Based on that observation, the attention layer is to calculate how important a token is. Let α_i be the weight of the word i, q be the attention query vector, α_i is calculated by the formulas (2) and (3).

$$a_i = q^T \tanh\left(V \times c_i + v\right) \tag{2}$$

$$\alpha_i = \frac{\exp(a_i)}{\sum_{j=1}^{M} \exp(a_j)} \tag{3}$$

The final representation vector is the weighted sum of c_i, calculated by the formula:

$$r = \sum_{i=1}^{M} \alpha_i c_i \tag{4}$$

3.3 Paragraph encoder

This architecture is to represent paragraphs like articles in legal documents. Figure 2 represents the components of this architecture. Instead of treating an article as a sequence of words, we consider them as a paragraph containing sentences.

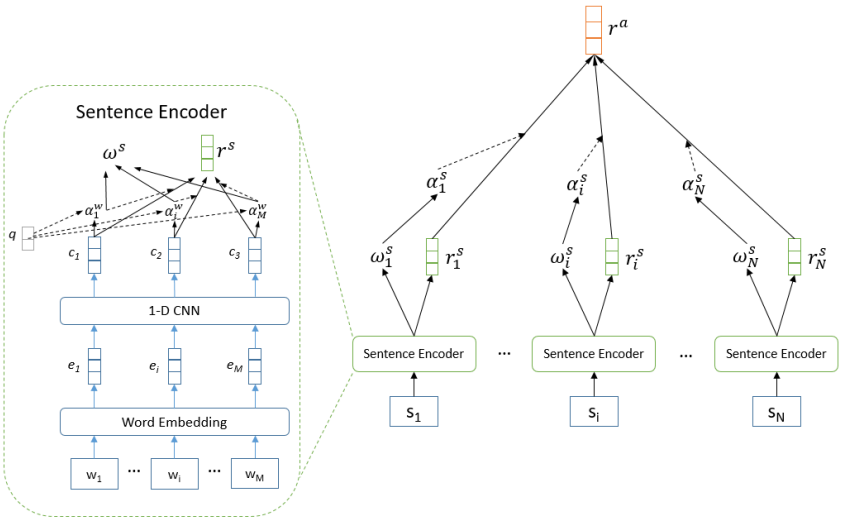

Figure 2: Paragraph encoder architecture

We calculate the attention weight of a sentence by averaging the attention weights of the words belonging to the sentence. Base on the observation that not every sentence contributes to the paragraph meaning, we replace softmax with sparsemax (Martins and Astudillo, 2016). Representation vector r^a in this model is calculated by the Formulas 5, 6, and 7.

$$\omega^s = \frac{\sum_i a_i^w}{|s|} \tag{5}$$

$$\alpha_i^S = \text{sparsemax}(\omega_i^s)^* \tag{6}$$

$$r^a = \sum_{j=1}^{N} \alpha_j^s r_j^s \tag{7}$$

In the example given in Table 1, we can easily see that the highlighted sentences are essential for answering the question while other sentences do not contain such kind of information. With the architecture proposed, the system can learn to focus on the important part and ignore other irrelevant ones. Besides, with the ability to highlight the important sentences in a lengthy article, the system can benefit the real user experience in its application.

4 Datasets and Experimental Setup

4.1 Datasets

To conduct experiments, we built two datasets: 1) the legal document corpus, which contains Vietnamese legal documents; and 2) the QA dataset, which contains a set of legal questions (queries) and a list of relevant articles for each question. The raw legal documents were first crawled from the official online sites [2,3]. The queries were collected from the legal advice websites [4,5,6].

Originally, each query contains both the title and content. After analyzing, we found that the content is often lengthy and confusing. Therefore, we only kept the good titles in the dataset, rewrote the uninformative titles, and filtered out content parts. The raw data for legal document corpus is the whole set of Vietnamese legal documents which contains multiple different versions of each law and regulation. We filtered out the redundant old versions and only kept and mapped the answers to the currently effective articles. This process was done with the supports of lawyers. Finally, we obtained the legal document corpus containing 8,586 documents with 117,545 articles, and the query dataset containing 5,922 queries coming along with their relevant articles. Table 2 shows the statistics on the query dataset. On average, each query contains 12.5 words (17.3 syllables) and has 1.6 relevant articles.

Table 2: Statistics about the query dataset

	Min	**Max**	**Average**
Query length in words	2	36	12.5
Query length in tokens	4	45	17.3
Number of relevant documents for each query	1	4	1.19
Number of relevant articles for each query	1	11	1.6

4.2 Experimental Setup

In our experiments, we used 90% of the query set for training, and the rest 10% for evaluation. We trained the model as a binary classifier. We normalize the probability that article i is related to the query by the following formula:

$$p_i = \frac{\exp\left(\hat{y}_i^+\right)}{\exp\left(\hat{y}_i^+\right) + \sum_{j=1}^{K} \exp\left(\hat{y}_{i,j}^-\right)} \tag{8}$$

where $\hat{y}_i^+$ and $\hat{y}_{i,j}^-$ are the probabilities that article i and article j, which belongs to the negative set of article i, is related to the query respectively.

We conducted experiments with two versions of our model. In the first one *(i.e. System I)*, we only used sentence encoder to find the representation vectors for both articles and queries. In the second one *(i.e. System II)*, we use sentence encoder for queries and paragraph encoder for articles. The two systems share the same value of parameters as listed in Table 3. In both systems, the maximum number of tokens in a query is 40. In terms of the article, the maximum number of tokens in the first system is 600; and in the second system, each item contains at most 30 sentences, which does not exceed 25 words.

From 117,545 articles available in the dataset, for each query, we first selected top N articles using ElasticSearch. By doing so, we can filter out clearly unrelated articles. After that, we reranked the selected articles by our method, as described in Section 3. We trained the model by forcing it to distinguish the positive articles from negative samples, which are selected by ElasticSearch scores *(i.e. ES-neg)* or randomly *(i.e. random-neg)*.

[2]http://vbpl.vn/tw/pages/home.aspx

[3]https://thuvienphapluat.vn

[4]https://hdpl.moj.gov.vn/Pages/home.aspx

[5]http://hethongphapluat.com/hoi-dap-phap-luat.html

[6]https://hoidapphapluat.net/

Table 3: Value of parameters in the models

Parameter	Meaning	Value
INPUT_VOCAB_SIZE	Size of the vocabulary	31450
D_EMBEDDING	Size of Word Embedding layer	512
D_CNN	Number of CNN filter	512
D_Q	Size of attention query vector	200
DROPOUT_RATE	Dropout rate	0.2

Two evaluation metrics were used to measure the performance of retrieval systems: (macro) $recall@k$ and $NDCG@k$ (Järvelin and Kekäläinen, 2002), where k is the number of the top selected articles.

5 Experimental Results

5.1 Results of Retrieval Models

At first, we conducted experiments to compare our proposed models with several strong baselines. Models to compare were as follows.

- **ElasticSearch** (BM25): Using ElasticSearch[7] with BM25 as the similarity measure.
- **ElasticSearch** (TF-IDF): Similar to the previous one but using TF-IDF instead of BM25.
- **Birch** (256 first words): Adapting Birch (Yilmaz et al., 2019), a BERT based system for document retrieval, which achieves state-of-the-art results on TREC (Lin et al., 2014). For each article, we used only the first 256 words.
- **Birch** (title): Another baseline using Birch with the title of each article.
- **Our System I**: The first version using flat architecture with the sentence encoder, which considers each article as a "very long" sentence.
- **Our System II**: The second version using hierarchical architecture with the paragraph encoder.

For both Birch and our systems, we used the same results from ElasticSearch as the input of the reranking models to ensure the fairness of the comparison. All the systems ranked the articles as follows.

$$S_f = w \cdot S_{doc} + (1 - w) \cdot S_{DL} \tag{9}$$

where S_{doc} is the score given by ElasticSearch, S_{DL} is the semantic score evaluated by the deep learning model, $w \in [0, 1]$ is the hyperparameter determining the weights of the two scores. We tuned w for each system to get the optimal value.

Experimental results of retrieval models are shown in Table 4, where N was set to be 1000. We used the same number for *ES-neg* and *random-neg*, and conducted experiments with two different values: 30 and 50. It means that for each query and a relevant article, we randomly selected 30 (or 50) negative articles and selected the same number of negative articles from ElasticSearch. Our first observation is that both Birch and our systems performed much better than ElasticSearch. This showed the effectiveness of the reranking strategies used in those systems. The next observation is that both our systems with flat (System I) and hierarchical (System II) architectures outperformed both versions of Birch, a state-of-the-art document retrieval method. Among two version of Birch, the one that used titles got better results. This is reasonable because the title often contains the most important information of each article. Using titles, moreover, can reduce noise in representing articles. Our best system with the paragraph encoder model achieved 0.825 $Recall@20$ and 0.688 in $NDCG@20$, which improved 0.042 and 0.097, respectively, compared with the best results of Birch. The results also showed the superior of the hierarchical architecture in our system, compared to the flat one.

[7] https://www.elastic.co/

Table 4: Performance of the retrieval systems

System	Recall@20	NDCG@20
ElasticSearch (BM25)	0.357	0.334
ElasticSearch (TF-IDF)	0.478	0.351
Birch (256 first words)	0.763	0.542
Birch (title)	0.783	0.591
Our System I (1000,30,30)	0.798	0.641
Our System I (1000,50,50)	0.811	0.669
Our System II (1000,30,30)	0.801	0.665
Our System II (1000,50,50)	**0.825**	**0.688**

5.2 Ablation Study

In the next experiments, we show the reasonableness of the design of our model. We conducted an ablation study by modifying or removing a component of our System II:

- *Removing sentence level attention layer*: the purpose of this experiment is to evaluate the impact of the attention layer at the sentence level in the paragraph encoder.
- *Removing word and sentence level attention layers*: the purpose is to investigate the importance of the attention layers.
- *Using softmax instead of sparsemax*: the purpose is to prove the effectiveness of using sparsemax.
- *Using normal attention*: in this experiment, we used attention in Formula 2 instead of calculating ω^s in Formula 5. The purpose is to show the necessity of our attention design.

Experimental results in Table 5 showed the reasonableness of our model. Each time we modified or removed a component, the performance of the system degraded. For $Recall@20$, while the full model got 0.825, the score reduced to 0.816 and 0.776, when we removed the attention layers at the sentence level and at both levels, respectively. The score was only 0.801 when using softmax and 0.782 when using normal attention. For $NDCG@20$, we got a similar situation.

Table 5: Ablation study

Architecture	Recall@20	NDCG@20
Full model	**0.825**	**0.688**
Removing sentence level attention	0.816	0.666
Removing word and sentence level attention	0.776	0.577
Using softmax instead of sparsemax	0.801	0.666
Using normal attention	0.782	0.638

5.3 Model Behaviour

To see the behaviour of our model in representing queries and articles, we visualized the weights of the attention vectors. Figure 3 demonstrates the attention weight visualization of the example in Table 1. For the query, the attention vector and the query have the same length, where each weight corresponds to a word. The larger weight (the more imporant word) is, the darker color is used to display. The visualization shows that our model can focus on important words like “inheritance”, “stepchildren”, “father”, “rights”, and “will”. For the article, each weight of the attention vector corresponds to a sentence in the article. The visualization also shows that important sentences can be captured by our model like sentences with numbers 3, 4, 5, and 1.

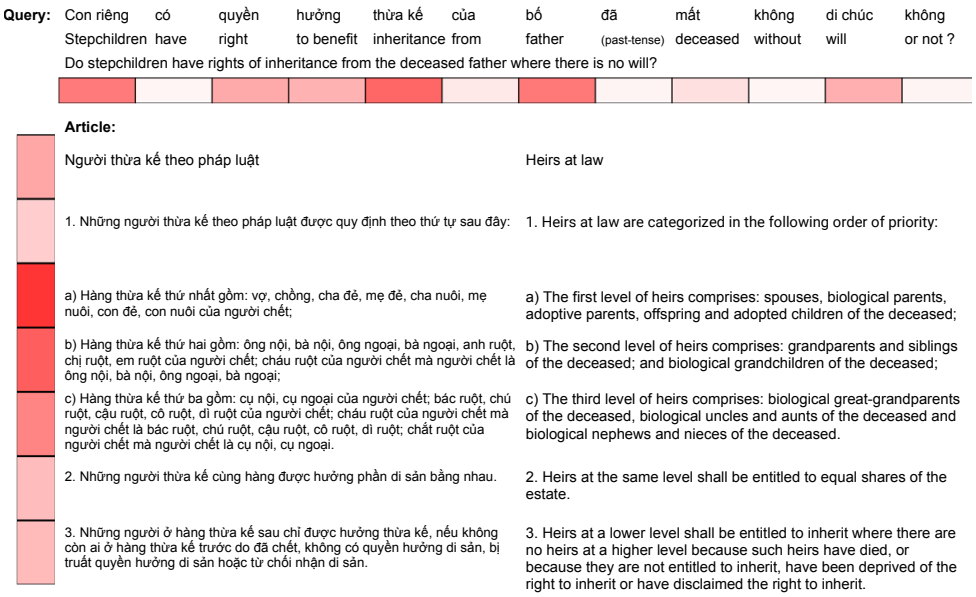

Figure 3: Weight visualization of the example in Table 1.

5.4 Results with Different Values of N

In the above experiments, we fixed the value of N at 1000. To investigate the impact and determine the suitable range of N, we have varied the value of N in the range [300, 400, 500, 1000, 1500, 2000]. Table 6 shows experimental results of our System II with different values of N. The system performed gradually better when N increased from 300 to 1000 and achieved the best result at 1500. The performance, however, degraded slightly when N increased to 2000. The experimental results suggested that the suitable range of N can be from 1000 to 2000. Using a small value of N may miss the correct articles, while using a large value of N may add noise to the reranking step.

Table 6: The impact of N to *System II* performance

N	300	400	500	1000	1500	2000
Recall@20	0.810	0.811	0.812	0.825	**0.830**	0.829
NDCG@20	0.678	0.678	0.682	0.688	**0.690**	0.689

6 Conclusion

In this paper, we have proposed a method for learning neural attentive text representation for legal text and applied to a retrieval-based question answering system. We choose the appropriate architecture based on the understanding of the characteristic of the legal text. Our proposed architecture captures both local dependencies using CNN and long-range context using attention mechanism. With a hierarchical architecture, the models learn the vector representation at different abstract levels of a query and an article and the relevance between the two. Our models can make a big gap comparing to Elastic Search and outperform the BERT-based model.

Acknowledgement

This work was supported by Ministry of Science and Technology of Vietnam under Grant No. KC.01.23/16–20, and in part by the Asian Office of Aerospace R&D (AOARD), Air Force Office of Scientific Research (Grant no. FA2386-19-1-4041).

References

Ngo Xuan Bach, Trieu Khuong Duy, and Tu Minh Phuong. 2019a. A POS tagging model for Vietnamese social media text using BiLSTM-CRF with rich features. In *Proceedings of the 16th Pacific Rim International Conference on Artificial Intelligence (PRICAI), Part III*, pages 206–219.

Ngo Xuan Bach, Nguyen Thi Thanh Thuy, Dang Bao Chien, Trieu Khuong Duy, To Minh Hien, and Tu Minh Phuong. 2019b. Reference extraction from Vietnamese legal documents. In *Proceedings of the 10th International Symposium on Information and Communication Technology (SoICT)*, pages 486–493.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

William S Cooper. 1971. A definition of relevance for information retrieval. *Information storage and retrieval*, 7(1):19–37.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Phong-Khac Do, Huy-Tien Nguyen, Chien-Xuan Tran, Minh-Tien Nguyen, and Minh-Le Nguyen. 2017. Legal question answering using ranking SVM and deep convolutional neural network. *arXiv preprint arXiv:1703.05320*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing, Chapter 23 (2nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Jimmy Lin, Miles Efron, Yulu Wang, and Garrick Sherman. 2014. Overview of the trec-2014 microblog track. Technical report, MARYLAND UNIV COLLEGE PARK.

Hans Peter Luhn. 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4):309–317.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pages 1614–1623.

Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5528–5531. IEEE.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *thirtieth AAAI conference on artificial intelligence*.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. 2017. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 257–266.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 373–382.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, pages 101–110.

Keet Sugathadasa, Buddhi Ayesha, Nisansa de Silva, Amal Shehan Perera, Vindula Jayawardana, Dimuthu Lakmal, and Madhavi Perera. 2018. Legal document retrieval using document vector embeddings and deep learning. In *Science and Information Conference*, pages 160–175. Springer.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432.

Vu Tran, Minh Le Nguyen, Satoshi Tojo, and Ken Satoh. 2020. Encoded summarization: summarizing documents into continuous vector space for legal case retrieval. *Artificial Intelligence and Law*, pages 1–27.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615.

Zeynep Akkalyoncu Yilmaz, Shengjin Wang, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Applying BERT to document retrieval with birch. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 19–24.