sciendo

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **23**, No 2
Sofia • 2023 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2023-0014

Joint Reference and Relation Extraction from Legal Documents with Enhanced Decoder Input

Nguyen Thi Thanh Thuy, Nguyen Ngoc Diep, Ngo Xuan Bach, Tu Minh Phuong

Department of Computer Science, Posts and Telecommunications Institute of Technology, Hanoi, Vietnam
E-mails: thuyntt@ptit.edu.vn diepnguyenngoc@ptit.edu.vn bachnx@ptit.edu.vn phuongtm@ptit.edu.vn

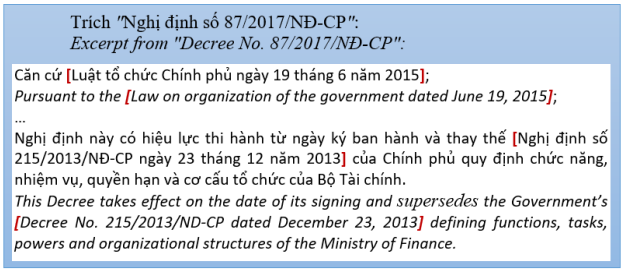
***Abstract*:** *This paper deals with an important task in legal text processing, namely reference and relation extraction from legal documents, which includes two subtasks:* 1) *reference extraction;* 2) *relation determination. Motivated by the fact that two subtasks are related and share common information, we propose a joint learning model that solves simultaneously both subtasks. Our model employs a Transformer-based encoder-decoder architecture with non-autoregressive decoding that allows relaxing the sequentiality of traditional seq2seq models and extracting references and relations in one inference step. We also propose a method to enrich the decoder input with learnable meaningful information and therefore, improve the model accuracy. Experimental results on a dataset consisting of 5031 legal documents in Vietnamese with 61,446 references show that our proposed model performs better results than several strong baselines and achieves an F_1 score of 99.4% for the joint reference and relation extraction task.*

***Keywords*:** *Reference extraction; relation extraction; legal documents; transformer; joint models.*

1. Introduction

With the goal of assisting people in accessing, retrieving, and gathering necessary legal information, Legal Text Processing (LTP) has attracted researchers worldwide over the past few decades. Legal documents have some specific characteristics compared to other common types of documents. One of the most distinctive characteristics is that legal documents usually contain references to other legal documents, provisions in other legal documents, or within the same document. The identification of these references can assist individuals in understanding the legal document and aid in other tasks and applications within LTP too. In addition, for each reference, there is a relation between the current document and the one being referenced. Fig. 1 shows an excerpt from a Vietnamese decree (we regard as the current document) and its English version. The decree contains several references,

including “Law on organization of the government dated 19 June 2015” (reference type is “law”) and “Decree No 215/2013/ND-CP dated 23 December 2013” (reference type is “decree”). The relations between the current document and two references are “Is pursuant to” and “Supersedes”, respectively. Determining these relations is also crucial for readers or a legal text processing system to comprehend the role and meaning of the referenced documents.



References	Reference Type	Relation Type
Luật tổ chức Chính phủ ngày 19 tháng 6 năm 2015 *Law on organizing the government dated June 19, 2015*	Luật *Law*	Căn cứ *Is pursuant to*
Nghị định số 215/2013/NĐ-CP ngày 23 tháng 12 năm 2013 *Decree No. 215/2013/ND-CP dated December 23, 2013*	Nghị định *Decree*	Thay thế *Supersedes*

Fig. 1. Examples of references in a legal document

There are a number of studies on extracting references from legal documents [1, 6, 16, 18]. However, to the best of our knowledge, there has been no previous research on determining the relations of references. In this paper, we consider the complete task consisting of two subtasks: extract references from a given legal document (Subtask 1) and determine the relationship between the document and each extracted reference (Subtask 2). An important property of this problem is that the two subtasks are closely related in nature. For example, a decree usually supersedes another decree, not a law; and decrees are usually pursuant to laws, but the reverse is not true. Therefore, a joint model that can solve the two subtasks simultaneously would improve the accuracy of the system.

Recently, numerous joint learning models have been introduced and achieved state-of-the-art performance on different Natural Language Processing (NLP) and LTP problems [2, 5, 7-9, 12, 17, 20, 22, 23]. In this study, we introduce a joint model that leverages recent advancements in deep learning research to address both subtasks at the same time.

We make three main contributions as follows.

1. We propose a joint model that employs a Transformer-based encoder-decoder architecture with non-autoregressive parallel decoding to extract simultaneously references and relations from legal documents. The advantage of the model is not only that it is independent of the order of references but also can extract nested or overlapping references. In order to enhance the performance of the joint extraction model, we also introduce a decoder input enhancement method by learning important clues of references.

2. We present a large, annotated dataset for the task consisting of 5031 Vietnamese legal documents with 61,446 references of 9 reference types. The dataset is available at: https://github.com/mlalab/VNLegalText.

3. We conduct experiments on the legal dataset to compare the proposed model with several strong, advanced baselines. The experimental results validate the effectiveness of our proposed model. The detailed experimental analyses also indicate that the model performs well on complex sentences with multiple references.

2. Related work

2.1. Information extraction in the legal domain

Information extraction in the legal domain is a critical but challenging research area, primarily because legal documents contain intricate logical meanings and numerous specialized terms. In recent years, various studies in this field have been conducted on different legal systems [2, 9, 13, 24]. They apply supervised learning techniques to datasets built from legal documents to extract essential information. Experimental results and analyses, however, are relatively limited due to the shortage of large annotated legal document datasets.

A fundamental and typical task of legal information extraction is extracting references from legal documents, which have been investigated in various languages. Tran et al. [18] introduce a system that resolves references at sub-document level while other studies only detect and extract references at document level. From the algorithmic perspective, Palmirani, Brighi and Massini [16] and Gonzalez, Fuente and Vicente [6] employ rule-based approaches; Tran et al. [18] use a hybrid method, which combines rules with machine learning algorithms; Bach et al. [1] utilize a deep learning model with rich features. Unlike previous studies, we tackle a more intricate task that involves extracting both references and relations from legal texts.

2.2. Joint entity and relation extraction using deep learning models

Entity and relation extraction is perhaps the NLP task most closely related to ours. Advanced approaches using joint models have been introduced to deal with this task. Initial studies such as [28] propose shared parameter based joint models to reduce error propagation and obtain the interaction between the two subtasks. However, these models do not conduct joint decoding and instead send the discovered entity pair to a relation classifier to determine the relation. The authors in [2, 22, 23, 27] propose a new approach to achieve joint decoding. In these studies, the joint task has been transformed into the problem of string tagging, involving simultaneous entity/relation extraction.

More prominently, several recent studies have proposed and employed extraction models based on encoder-decoder seq2seq architecture, such as [14, 21, 25, 26]. These methods not only improve the accuracy of the joint extraction but also increase processing speed. Zeng, Zhang and Liu [25] introduce CopyMTL, a method based on multi-task learning and copy mechanism. Nayak and Ng [14] also employ encoder-decoder architecture with a proposed pointer network-based

decoder, where all triplets from a source sentence are discovered as a set at every time step. W a n g et al. [21] transform the joint task into a table-filling problem with only one label space, where each entry in the input table represents the interaction between two individual words.

However, these models use autoregressive decoders, which are complicated and time-consuming because the output of entity and relation pairs in the input sentence must be in sequence. To tackle this challenge, S u i et al. [17] propose a more efficient encoder-decoder model which can simultaneously decode entities and relations. The authors treat the joint task as a set predicted problem, regardless of the order of triplets. Despite its success, this method lacks the use of explicit context information. We address this issue by encoding prior knowledge to help construct a better decoder input. The idea is to improve decoder inputs with prior knowledge of the reference's starting position in the input sentence. This makes the proposed model more efficient.

3. Proposed model

3.1. Model architecture

Given a legal document *x*, our model aims to extract: 1) references, i.e., word segments along with reference labels, in *x*; and 2) a relation between *x* and each extracted reference. Motivated by the fact that a reference usually fits within a sentence and the relation can be determined using only the context of the sentence, our model processes each sentence *s* (represented as a sequence of *n* tokens $s = t_1 t_2 \dots t_n$) in *x* one by one. The model's output consists of *m* (unordered) triples, each of which corresponds to a reference in the form (r_{start}, r_{end}, rel), where r_{start} and r_{end} denote the starting and the ending positions of the reference in the input sentence, and rel is a label concatenated by a reference type and a relation type with a slash in the middle of "reference type/relation type".

Our proposed model is illustrated in Fig. 2, which consists of four main components: sentence encoder, input enhancer, decoder, and predictor.

- **Sentence encoder.** The sentence encoder employs a pre-trained model to generate contextualized token representations.
- **Input enhancer.** As shown in previous studies, the quality of the decoder input greatly affects the model accuracy. Therefore, instead of using the same randomized queries for all input sentences, our input enhancer learns *N* queries, and each of them contains important clues of a possible reference in the input sentence. Here, *N* is a hyper-parameter, which is larger than the maximum number of references in a legal sentence.
- **Decoder.** The decoder receives the outputs of the sentence encoder and input enhancer as the input and produces *N* output embeddings, which are then fed into the predictor to extract references and relations.
- **Predictor.** The predictor employs feed-forward networks to identify a reference and a relation (or no reference) from each output embedding of the decoder.

We next describe the model's components in detail.

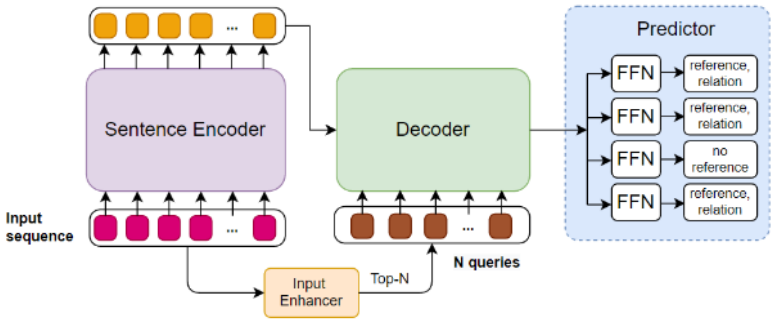


Fig. 2. Illustration of our model's architecture

3.2. Sentence encoder

Our sentence encoder uses a BERT-based language model [4], to produce contextualized token embeddings from the static embeddings and positional embeddings:

$$\mathbf{c}_i = \text{PretrainedLM}(t_{1:n}, i), \quad (1)$$

where $t_{1:n}$ denotes the input sequence with n tokens and $\mathbf{c}_i \in \mathbb{R}^{1\times d}$ $(1 \le i \le n)$ is the contextualized token embedding of the i-th token, i.e., t_i, and d denotes the embedding size.

3.3. Input enhancer

The identification of starting phrase of a reference (called starting phrases) plays a vital role in solving the task. This is because starting phrases not only aid in extracting the reference but also assist in determining the reference and relation type.

Motivated by the fact that some phrases tend to occur at the beginning of references more often than others, we propose a decoder input enhancement method that examines phrases in the input sentence and estimates the likelihood of each phrase being the first phrase of a reference. The most likely starting phrases are used to provide important hints to the decoder for extracting references and relations.

Several techniques can be employed to implement the input enhancer. A simple and straightforward method is to build a dictionary that contains the most frequent starting phrases of each reference type. The input enhancer looks for phrases in the input sentence that appear in the dictionary and considers them as starting phrases of potential references. Apart from the dictionary-based approach, we also propose a more flexible method based on classification, which considers the context of the input sentence. Specifically, the input enhancer takes a phrase consisting of m consecutive tokens along with its context, i.e., surrounding tokens, as the input and returns the possibility of it being a starting phrase.

Our input enhancer is illustrated in Fig. 3. The input consists of three components: a phrase (m consecutive tokens) and the left/right context (q tokens on the left-hand side/right-hand side). So, the input can be considered as a sequence of $m + 2q$ tokens where m and q are hyper-parameters. The input is fed into an embedding layer and then a fully connected layer with sigmoid function to produce a score. The higher the score is, the more likely starting phrase is.

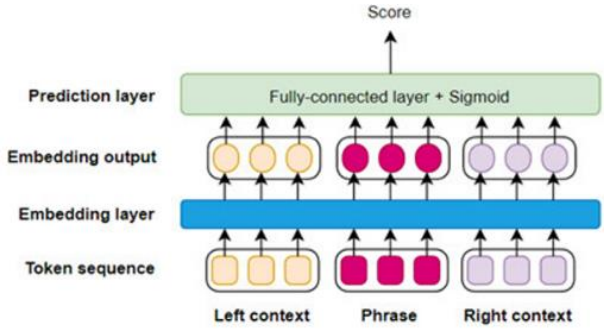


Fig. 3. Illustration of our input enhancer

The input enhancer is trained independently of the joint model. The training dataset consists of positive samples and negative samples as follows.

- **Positive samples.** Each positive sample is created by using the starting phrase of a reference.
- **Negative samples.** In theory, all phrases except the starting ones can be used to generate negative samples. To deal with the imbalance data problem, however, we propose a negative sampling strategy in which we randomly select only two negative samples according to each positive one. Between the two negative samples, one is inside and the other one is outside the reference. Note that the phrase's relative position (inside/outside) to the reference is determined based on the first token of the phrase.

The trained input enhancer is then employed in both the training and inference stages of the joint model. For each input legal sentence, the input enhancer conducts the following steps:

- Extracts all possible phrase candidates in the input sentence. Each candidate is a sub-sequence of m consecutive tokens.
- Ranks phrase candidates and selects top N phrases with the highest probability of being starting phrases.
- For each selected phrase, retrieves the first token.
- Returns the positional embeddings of N retrieved tokens: $\mathbf{P} = \{p_i\}_{i=1}^{N}$, $\mathbf{P} \in \mathbb{R}^{N \times d}$, where $p_i \in \mathbb{R}^{1 \times d}$ is the positional embedding of the first token of the i-th selected phrase.

The positional embeddings of the first tokens of selected phrases provide important hints to the decoder in reference and relation extraction.

3.4. Decoder

Our decoder employs the standard Transformer architecture [19] with K identical Transformer layers. By using a parallel non-autoregressive decoding mechanism, our model decodes a set of N triples for both references and relations simultaneously at each decoding layer instead of sequence decoding. Recall that N is a hyper-parameter that is greater than the maximum number of references in a legal sentence. Unlike previous encoder-decoder models [17] which use a fixed query for all inputs, we

consider the output of the input enhancer as flexible queries that changes according to the input sentence, and provides hints of references.

The decoder takes the sequence of n token embeddings $\mathbf{c}_i$ $(1 \le i \le n)$ and N positional embeddings $\mathbf{p}_i$ $(1 \le i \le N)$ as the input, transforms them, and produces N output embeddings. Those output embeddings are then fed into a predictor in which each of them is decoded into a triple representing a reference and its relation,

(2) $$\mathbf{H} = \text{Decoder}(\mathbf{c}_{1:n}, \mathbf{p}_{1:N}),$$

where $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{N}$, $\mathbf{H} \in \mathbb{R}^{N \times d}$, and $\mathbf{h}_i \in \mathbb{R}^{1 \times d}$ denotes the i-th output embedding of the decoder.

3.5. Predictor

The predictor takes the output $\mathbf{H}$ of the decoder and the output $\mathbf{C} = \{\mathbf{c}_i\}_{i=1}^{n}$, $\mathbf{C} \in \mathbb{R}^{n \times d}$, of the sentence encoder as the input and predicts the set of N triples (r_{start}, r_{end}, rel). Each of them corresponds to an output vector h_i and represents a reference and its relation type. Recall that a rel label contains information about both the reference type and the relation type. Please, note also, that we have a special relation for "no reference"; the actual number of extracted references may be less than N.

Three Feed-Forward Networks (FFN) are used to detect starting positions of references (r_{start}), ending positions of references (r_{end}), and type of reference and relation (rel) as follows:

(3) $$\mathbf{p}_i^{\text{start}} = \text{softmax}(\mathbf{v}_1 \tanh(\mathbf{W}_1 \mathbf{H}_i^{\text{T}} + \mathbf{W}_2 \mathbf{C}^{\text{T}})),$$

(4) $$\mathbf{p}_i^{\text{end}} = \text{softmax}(\mathbf{v}_2 \tanh(\mathbf{W}_3 \mathbf{H}_i^{\text{T}} + \mathbf{W}_4 \mathbf{C}^{\text{T}})),$$

(5) $$\mathbf{p}_i^{\text{rel}} = \text{softmax}(\mathbf{h}_i \mathbf{W}_r^{\text{T}}),$$

where $\mathbf{W}_* \in \mathbb{R}^{d \times d}$, $\mathbf{v}_* \in \mathbb{R}^{1 \times d}$, and $\mathbf{W}_r \in \mathbb{R}^{t \times d}$ are learnable parameters and t denotes the number of labels of references and their relation types (including the special label for "no reference" cases), and $\mathbf{H}_i \in \mathbb{R}^{n \times d}$ is a matrix with n rows, each row equals to $\mathbf{h}_i$. Then r_{start}, r_{end} and rel are determined as indices of maximum in vectors of probabilities $\mathbf{p}_i^{\text{start}}$, $\mathbf{p}_i^{\text{end}}$ and $\mathbf{p}_i^{\text{rel}}$, respectively:

(6) $$r_{\text{start}} = \arg\max_{1 \le j \le n} \mathbf{p}_i^{\text{start}}(j),$$

(7) $$r_{\text{end}} = \arg\max_{1 \le j \le n} \mathbf{p}_i^{\text{end}}(j),$$

(8) $$\text{rel} = \arg\max_{1 \le j \le t} \mathbf{p}_i^{\text{rel}}(t).$$

3.6. Joint training

Let $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{N}$ denote the set of ground truths where $\mathbf{y}_i$ are triples in the form (e_i^{start}, e_i^{end}, r_i). Here e_i^{start} and e_i^{end} are starting and ending positions of the i-th reference and r_i denotes the reference type. Let $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_i\}_{i=1}^{N}$ be the output of the joint model where $\hat{\mathbf{y}}_i$ are triples in the form ($\mathbf{p}_i^{\text{start}}$, $\mathbf{p}_i^{\text{end}}$, $\mathbf{p}_i^{\text{rel}}$). In case the number of references in ground truths is less than N, we add dummy ($\emptyset$) references. We first define a pair-wise matching loss between a pair of ground truth/prediction $(\mathbf{y}_i, \hat{\mathbf{y}}_i)$ as follows:

(9) $$\mathcal{C}_{\text{match}}(\mathbf{y}_i, \hat{\mathbf{y}}_i) = -\mathbb{1}_{\{r_i \neq \emptyset\}}[\mathbf{p}_i^{\text{start}}(e_i^{\text{start}}) + \mathbf{p}_i^{\text{end}}(e_i^{\text{end}}) + \mathbf{p}_i^{\text{rel}}(r_i)].$$

Let π^* be the optimal matching between two sets $\mathbf{Y}$ and $\hat{\mathbf{Y}}$:

(10) $$\pi^{\star} = \arg\min_{\pi \in \Pi(N)} \sum_{i=1}^{N} \mathcal{C}_{\text{match}}\left(\mathbf{y}_i, \hat{\mathbf{y}}_{\pi(i)}\right),$$

where $\Pi(N)$ is the set of $N!$ permutations of $\{1, 2, \ldots, N\}$. The optimal assignment π^{*} can be found by using the Hungarian algorithm [10].

Finally, a loss function is defined to train the joint model:

(11) $$\mathcal{L}(\mathbf{Y}, \hat{\mathbf{Y}}) = \sum_{i=1}^{N} \{-\log \mathbf{p}_{\pi^{\star}(i)}^{\text{rel}}(r_i) + \mathbb{1}_{\{r_i \neq \emptyset\}}[-\log \mathbf{p}_{\pi^{\star}(i)}^{\text{start}}(e_i^{\text{start}}) - \log \mathbf{p}_{\pi^{\star}(i)}^{\text{end}}(e_i^{\text{end}})]\}.$$

4. Dataset

4.1. Data collection

We retrieved legal documents from **http://vbpl.vn**, which is the largest repository of legal documents in Vietnam. Specifically, we retrieved laws, decrees, and circulars, which are the most numerous and important in the legal system of Vietnam. A subset of the documents has been randomly selected to create the dataset. Before labelling, we have pre-processed the data with the following steps:

- Remove extraneous text like headers and footers.
- Separate faulty syllables that stick together.
- Standardize accents (tones).
- Split sentences, and words in sentences using Pyvi tool (**https://github.com/trungtv/pyvi**).

At the end of this stage, we have obtained a dataset of 5,031 pre-processed Vietnamese legal documents.

4.2. Automatic labelling

Automatic labelling allows us to speed up the annotation procedure by using several rules with dictionaries and regular expressions based on our observations of legal documents.

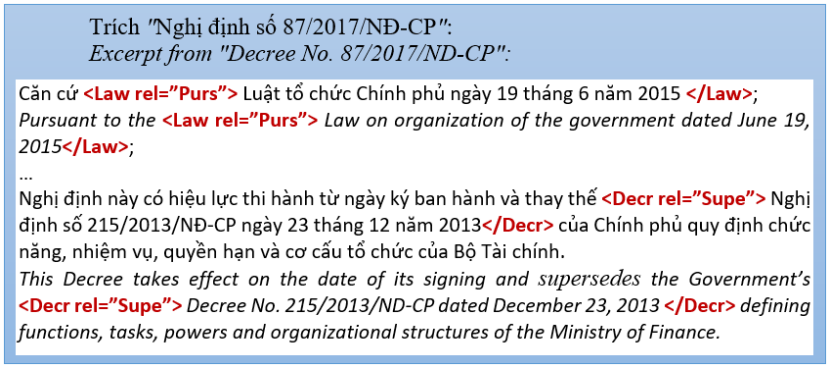
- **Beginning positions.** References usually start with specific words related to the type of the legal documents being referenced. Thus, it is possible to build a dictionary of keywords to detect the beginning positions of references.
- **Ending positions.** References often end in one of the following formats: year (e.g., *2015*), date (e.g., *December 23, 2013*), or legal document number (e.g., *215/2013/ND-CP*). We, therefore, construct regular expressions corresponding to those formats to detect the ending positions of references.
- **Reference types.** Reference types also often relate to the first word used in a reference. Therefore, we have created a similar dictionary to automatically classify them.
- **Relation types.** Relation types are usually described by some keywords or key phrases surrounding the references like "*pursuant to*" and "*supersedes*". We also have constructed a dictionary of such keywords/phrases to identify relation types.

4.3. Manual labelling

Two annotators, who are students of our department, with some background in information extraction, legal text processing, and machine learning, have corrected four kinds of information: starting positions of references, ending positions of

references, types of references, and types of relations. In case of disagreement, we have asked a third annotator, a graduate student from a law university, to exam, discuss, and make the final decision. To evaluate the level of agreement between the two annotators, we computed the Cohen's kappa score. The scores were 0.92 and 0.94 for references and relations, respectively, indicating the perfect agreement [3].

Fig. 4 shows an example of annotated data in the XML format from an excerpt of circular *No. 87/2017/ND-CP*. The example contains two references with the types "*Law*" (the referenced document is a law) and "*Decree*" (the referenced document is a decree). The relation types are "*Is pursuant to*" (the current document is pursuant to the referenced document) and "*Supersedes*" (the current document supersedes the referenced document), respectively.


Trích "Nghị định số 87/2017/NĐ-CP":
Excerpt from "Decree No. 87/2017/ND-CP":

Căn cứ **<Law rel="Purs">** Luật tổ chức Chính phủ ngày 19 tháng 6 năm 2015 **</Law>**;
Pursuant to the **<Law rel="Purs">** *Law on organization of the government dated June 19, 2015* **</Law>**;
...
Nghị định này có hiệu lực thi hành từ ngày ký ban hành và thay thế **<Decr rel="Supe">** Nghị định số 215/2013/NĐ-CP ngày 23 tháng 12 năm 2013 **</Decr>** của Chính phủ quy định chức năng, nhiệm vụ, quyền hạn và cơ cấu tổ chức của Bộ Tài chính.
This Decree takes effect on the date of its signing and supersedes the Government's **<Decr rel="Supe">** *Decree No. 215/2013/ND-CP dated December 23, 2013* **</Decr>** *defining functions, tasks, powers and organizational structures of the Ministry of Finance.*

Fig. 4. Examples of annotated data

4.4. Corpus statistics

Tables 1 and 2 show statistical information on references and relations along with their types. Totally, we have 61,446 references belonging to nine types: Constitution (103), Code (960), Law (21,157), Decree (22,917), Circular (7033), Joint Circular (424), Decision (4036), Ordinance (3926), and Resolution (890). References have one of the following seven types of relations: Is pursuant to (18,540), Refers (27,783), Expires (1618), Supersedes (1765), Amends or Supplements (1203), Guides (320), and No Relationship (10,217).

Table 1. Statistical information of reference types

No	Reference type	Label	Number of references
1	Constitution	Cons	103
2	Code	Code	960
3	Law	Law	21,157
4	Decree	Decr	22,917
5	Circular	Circ	7033
6	Joint circular	JCir	424
7	Decision	Deci	4036
8	Ordinance	Ordi	3926
9	Resolution	Reso	890
	Total		**61,446**

Table 2. Statistical information of relation types

No	Reference type	Description	Label	Number of references
1	Is pursuant to	The current document is pursuant to the referenced document	Purs	18,540
2	Refers	The current document refers to the referenced document	Refe	27,783
3	Expires	The referenced document has been expired	Expi	1618
4	Supersedes	The current document supersedes the referenced document	Supe	1765
5	Amends or supplements	The current document amends or supplements the referenced document	Amen	1203
6	Guides	The current document guides the referenced document	Guid	320
7	No relationship	No relationship between the current document and the referenced document	None	10,217
	Total			**61,446**

5. Experiments

5.1. Experimental setting

To conduct experiments, we have randomly divided the corpus into training/validation/test sets with a ratio of 70/10/20, respectively, which we have used for training extraction models, tuning hyper-parameters, and testing the models respectively.

To evaluate the performance of extraction models we have used precision, recall, and the F_1 score for references only, and for both – references and relations. For references only, a reference is correctly extracted if the system correctly detects the starting position, the ending position, and the reference type. For both references and relations, the system must correctly identify the relation type in addition to the three kinds of information.

5.2. Models to compare

We have compared our joint model with the following baselines, which are also deep learning-based joint models.

- **CasRel [22].** A model utilizing a cascade binary tagging framework for extracting relational triples. Additionally, it employs a Transformer encoder to encode the input sentence.
- **SPERT [5].** A span-based joint model with attention mechanisms and BERT embeddings.
- **JointER [23].** A joint model based on a decomposition strategy.
- **SPN [17].** A joint model using set prediction networks.

5.3. Network training

We have implemented our model in PyTorch using HuggingFace (**https://huggingface.co/**) and we have used PhoBERT-base [15] as the pre-trained language model.

Table 3. Hyper-parameters of our model

Model	Hyper-parameter	Value
Joint model	Number of decoding layers	3
	Batch size	8
	Learning rate	2×10^{-4}
	Weight decay	1×10^{-8}
	Dropout rate	0.1
Input enhancer	Batch size	32
	Length of phrases (m)	2
	Length of left/right context (q)	3
	Learning rate	1×10^{-3}

We set N, the maximum number of expected output references, to 15, which is larger than the maximum number of references observed in one sentence in the dataset. We have trained our model using the AdamW optimizer [11] with the epsilon and weight decay set to the default value in PyTorch, i.e., 1×10^{-8}. We have varied the numbers of decoding layers, the learning rate, and the batch size in {1, 2, 3, 4, 5}, {1×10^{-4}, 2×10^{-4}, 3×10^{-4}, 4×10^{-4}, 5×10^{-4}}, and {4, 8, 16, 32}, respectively. We have used grid search to find the best combination of these hyper-parameters.

To mitigate overfitting, we have used a dropout rate of 0.1 for each hidden layer in our model. For the input enhancer, we set the length of the left/right context (q) to 3. The batch size, the length of phrases (m), and the learning rate have been tuned in {8, 16, 32, 64}, {1, 2, 3, 4}, and {1×10^{-3}, 2×10^{-3}, 3×10^{-3}}, respectively. In each experiment, we have trained the model for 100 epochs and calculated precision, recall, and the F_1 score after each epoch on the validation set. The version with the highest F_1 score on the joint extraction task has been selected to apply to the test set. Our model's hyper-parameters are summarized in Table 3.

To ensure a fair comparison, we have also used PhoBERT-base [15] as the BERT encoder for all other methods, except for JointER which does not rely on BERT for encoding. The same dataset and evaluation method are used to assess the performance of all models. The experiments have been conducted on a consistent hardware platform, which utilizes Linux operating system, Intel E5v4 processor, 64 GB of RAM, and two NVIDIA GTX 2080 Ti GPU cards.

5.4. Experimental results

First, we compare the performance of our model with baselines. Table 4 shows the experimental results of reference and relation extraction models. Our model outperforms all the baselines in both scenarios, i.e., references only and both references and relations. For references only, we got 99.7% in the F_1 score, which improves 0.4% (57% error rate reduction) compared with the second-best model SPN [17]. For both references and relations, we have achieved 99.4% in the F_1 score, which improveds 1.1% (65% error rate reduction) compared with SPN [17].

Table 4. Performance of reference and relation extraction models (in precision, recall, and F1)

Model	References only			Both references and relations		
	Precision	Recall	F1	Precision	Recall	F1
CasRel	98.7	91.3	94.8	94.8	87.7	91.1
SPERT	98.3	91.7	94.8	96.9	90.4	93.5
JointER	98.4	98.0	98.2	97.2	96.9	97.1
SPN	99.8	98.7	99.3	98.8	97.7	98.3
Our model	**99.8**	**99.6**	**99.7**	**99.5**	**99.3**	**99.4**

Table 5. Number of parameters and training time of reference and relation extraction models

Model	Number of parameters (in millions)	Training time
CasRel	135.10	4 h 20 min
SPERT	135.62	4 h 30 min
JointER	13.17	33 min
SPN	165.77	4 h 45 min
Our model	165.80	4 h 50 min

In addition, we evaluate the complexity of the proposed model in relation to other joint extraction models by examining both the number of parameters and the training time of each model. The results in Table 5 show that JointER model has the smallest number of parameters, and it also has the shortest training time. The proposed model has a comparable number of parameters and training time to other models, but delivers the best results. The difference in terms of parameters and training time across the models is acceptable, but given their similar training time, accuracy remains the more crucial factor.

Next, we have investigated the performance of reference and relation extraction models according to the complexity of the input legal sentences. We have divided the test set into five distinct subsets as follows and measured the F_1 score of each subset separately. Subset i ($1 \leq i \leq 4$) contains sentences with exact i references in ground truth. Subset 5 consists of other sentences with at least fifth references in ground truth. The extraction models' performance on the five subsets is illustrated in Table 6. We have made the following observations: 1) the performance of all the models decrease with an increase in the complexity of the input sentences; 2) our model consistently outperforms all baseline models on all five subsets; 3) the performance gap between our model and the second-best model increaseds with the complexity of the input sentences. These results demonstrate the superior performance of our model in handling complex cases.

Table 6. Performance of extraction models according to the complexity of the input legal sentences

Model	F1 (both references and relations)				
	Subset1	Subset2	Subset3	Subset4	Subset5
CasRel	97.7	90.1	80.6	75.1	56.0
SPERT	96.4	94.3	90.3	87.5	82.3
JointER	99.8	97.4	91.1	90.3	90.0
SPN	99.4	98.7	98.2	96.4	91.9
Our model	**99.9**	**99.6**	**99.1**	**98.9**	**96.8**

Table 7. The effect of the input enhancer

Model (Variant)	References only			Both references and relations		
	Precision	Recall	F1	Precision	Recall	F1
Without	99.5	98.7	99.1	98.7	97.6	98.2
Dictionary-based	99.5	99.1	99.3	98.6	98.1	98.4
Classification-based CNN	99.9	99.7	99.8	99.5	99.3	99.4
Classification-based MLP	99.8	99.6	99.7	99.5	99.3	99.4

We have also conducted experiments to evaluate the effect of our input enhancer on the performance of the reference and relation extraction model. We consider the following model variants with different input enhancers:

- **Without.** This variant does not contain the input enhancer.
- **Dictionary-based.** This variant uses the dictionary-based input enhancer.
- **Classification-based MLP.** Our model with classification-based input enhancer using multi-layer perceptron.
- **Classification-based CNN.** This variant is like the previous one except that we employ a Convolutional Neural Network (CNN) for classification. We have used a simple CNN architecture: two 1D convolutional layers, a hidden fully connected layer, and an output layer with a sigmoid activation function. Similar to the MLP variant, we also have employed the binary cross entropy loss with Adam optimizer. The number of filters and the filter size have been tuned in {64, 128, 256} and {3, 4, 5}, respectively, using the validation set.

As shown in Table 7, three variants with an input enhancer perform better than the *Without* one. This confirms the effectiveness of our decoder input enhancement method. Experimental results also point out that the classification-based approach outperforms the simple dictionary-based approach. Moreover, the two classification-based variants yield similar results, indicating the stability of the proposed enhancement method.

In addition, we have carried out an experiment to assess how the number of decoding layers affects the model's performance by testing the proposed model with varying numbers of decoding layers. The results in Table 8 show that increasing the number of decoding layers can lead to better results, but the highest performance is achieved with three decoding layers. The best outcome is achieved when the number of decoding layers is set to three, with an F1 score of 99.4%. When the number of decoding layers is increased to four or five, there is no significant improvement in results. The main reason behind this is as the decoding layers become deeper, more multi-head self-attention and inter-attention modules are available. This allows for a more comprehensive integration of sentence information into queries. However, when the number of decoding layers increases, this advantage becomes less apparent.

Table 8. The impact of the number of decoding layers on the performance of the proposed model

Number of decoding layers	Precision	Recall	F1
1	99.2	99.0	99.1
2	99.3	99.2	99.3
3	99.5	99.3	99.4
4	99.4	99.3	99.4
5	99.4	99.3	99.4

6. Conclusion and future work

We introduce in this paper a study on extracting references and relations from legal documents, which is a complete and important problem in legal text processing. To solve the task, we propose a Transformer-based joint learning model that can extract both references and relations at the same time. We have verified the effectiveness of our proposed model by introducing a large annotated legal corpus for the task and conducting experiments to compare our model with strong, advanced baselines. Experimental results show that: 1) our model can extract both references and relations accurately with an F1 score of 99.4%; 2) our model gives better results than other baselines, especially on complex legal sentences with multiple references; 3) the input enhancer is stable in the sense that it is not affected much by the chosen machine learning algorithm.

There are different ways to continue this work. First, the extracted references and relations from legal documents can be useful for other legal text processing tasks such as legal information retrieval, legal text summarization, and question answering in the legal domain. Second, it is interesting to build legal text processing applications that help people in reading, understanding, and retrieving necessary information from legal documents. We plan to investigate and conduct such extensions in future work.

Acknowledgments: This research is supported by Posts and Telecommunications Institute of Technology, Hanoi, Vietnam.

References

1. Bach, N. X., N. T. T. Thuy, D. B. Chien, T. K. Duy, T. M. Hien, T. M. Phuong. Reference Extraction from Vietnamese Legal Documents. – In: Proc. of 10th International Symposium on Information and Communication Technology, 2019, pp. 486-493.
2. Chen, Y., Y. Sun, Z. Yang, H. Lin. Joint Entity and Relation Extraction for Legal Documents with Legal Feature Enhancement. – In: Proc. of 28th International Conference on Computational Linguistics, 2020, pp. 1561-1571.
3. Cohen, J. A Coefficient of Agreement for Nominal Scales. – Educational and Psychological Measurement, Vol. **20**, 1960, No 1, pp. 37-46.
4. Devlin, J., M.-W. Chang, K. Lee, K. Toutanova. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. – In: Proc. of North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2019, pp. 4171-4186.
5. Eberts, M., A. Ulges. Span-Based Joint Entity and Relation Extraction with Transformer Pre-Training. – In: Proc. of 24th European Conference on Artificial Intelligence, 2020.
6. Gonzalez, M. M., P. de la Fuente, D. J. Vicente. Reference Extraction and Resolution for Legal Texts. – In: Proc. of International Conference on Pattern Recognition and Machine Intelligence, 2005, pp. 218-221.
7. Hui, Y., J. Wang, N. Cheng, F. Yu, T. Wu, J. Xiao. Joint Intent Detection and Slot Filling Based on Continual Learning Model. – In: Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'21), 2021, pp. 7643-7647.
8. Ji, D., J. Gao, H. Fei, C. Teng, Y. Ren. A Deep Neural Network Model for Speakers Coreference Resolution in Legal Texts. – Information Processing & Management, Vol. **57**, 2020, No 6, p. 102365.
9. Judith Jeyafreeda Andrew, X. T. Automatic Extraction of Entities and Relation from Legal Documents. – In: Proc. of 7th Named Entities Workshop, ACL, 2018, pp. 1-8.
10. Kuhn, H. W. The Hungarian Method for the Assignment Problem. – Naval Research Logistics Quarterly, Vol. **2**, 1955, pp. 83-97.

11. L o s h c h i l o v, I., F. H u t t e r. Decoupled Weight Decay Regularization. – In: Proc. of International Conference on Learning Representations, 2019.
12. M a r t i n s, P. H., Z. M a r i n h o, A. F. T. M a r t i n s. Joint Learning of Named Entity Recognition and Entity Linking. – In: Proc. of 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, 2019, pp. 190-196.
13. M i s t i c a, M., G. Z. Z h a n g, H. C h i a, K. M. S h r e s t h a, R. K. G u p t a, S. K h a n d e l w a l et al. Information Extraction from Legal Documents: A Study in the Context of Common Law Court Judgements. – In: Proc. of ALTA, 2020.
14. N a y a k, T., H. T. N g. Effective Modeling of Encoder-Decoder Architecture for Joint Entity and Relation Extraction. – In: Proc. of 32th AAAI Conference on Artificial Intelligence, 2020, pp. 8528-8535.
15. N g u y e n, D. Q., A. T. N g u y e n. PhoBERT: Pre-Trained Language Models for Vietnamese. – ArXiv Preprint ArXiv:2003.00744, 2020.
16. P a l m i r a n i, M., R. B r i g h i, M. M a s s i n i. Automated Extraction of Normative References in Legal Texts. – In: Proc. of 9th International Conference on Artificial Intelligence and Law, 2003, pp. 105-106.
17. S u i, D., Y. C h e n, K. L i u, J. Z h a o, X. Z e n g, S. L i u. Joint Entity and Relation Extraction with Set Prediction Networks. – CoRR, Vol. **abs/2011.01675**, 2020 (online). **https://arxiv.org/abs/2011.01675**
18. T r a n, O. T., B. X. N g o, M. L e N g u y e n, A. S h i m a z u. Automated Reference Resolution in Legal Texts. – Artificial Intelligence and Law, Vol. **22**, 2014, No 1, pp. 29-60.
19. V a s w a n i, A., N. S h a z e e r, N. P a r m a r, J. U s z k o r e i t, L. J o n e s, A. N. G o m e z et al. Attention Is All You Need. – Advances in Neural Information Processing Systems, 2017, pp. 5998-6008.
20. W a n g, J., W. L u. Two Are Better than One: Joint Entity and Relation Extraction with Table-Sequence Encoders. – In: Proc. of Conference on Empirical Methods in Natural Language Processing (EMNLP'20), Association for Computational Linguistics, 2020, pp. 1706-1721.
21. W a n g, Y., C. S u n, Y. W u, H. Z h o u, L. L i, J. Y a n. UniRE: A Unified Label Space for Entity Relation Extraction. – In: Proc. of 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing. Volume **1**. Long Papers. 2021, pp. 220-231.
22. W e i, Z., J. S u, Y. W a n g, Y. T i a n, Y. C h a n g. A Novel Cascade Binary Tagging Framework for Relational Triple Extraction. – In: Proc. of 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 1476-1488.
23. Y u, B., Z. Z h a n g, X. S h u, T. L i u, Y. W a n g, B. W a n g et al. Joint Extraction of Entities and Relations Based on a Novel Decomposition Strategy. – In: Proc. of 24th European Conference on Artificial Intelligence (ECAI'20), 2020.
24. Z a d g a o n k a r, A. V., A. J. A g r a w a l. An Overview of Information Extraction Techniques for Legal Document Analysis and Processing. – International Journal of Electrical and Computer Engineering (IJECE), Vol. **11**, 2021, No 6, pp. 5450-5457.
25. Z e n g, D., H. Z h a n g, Q. L i u. CopyMTL: Copy Mechanism for Joint Extraction of Entities and Relations with Multi-Task Learning. – In: Proc. of 34th AAAI Conference on Artificial Intelligence (AAAI'20), 2020, pp. 9507-9514.
26. Z e n g, X., D. Z e n g, S. H e, K. L i u, J. Z h a o. Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism. – In: Proc. of 56th Annual Meeting of the Association for Computational Linguistics. Volume **1**. Long Papers. 2018, pp. 506-514.
27. Z h e n g, H., R. W e n, X. C h e n, Y. Y a n g, Y. Z h a n g, Z. Z h a n g et al. PRGC: Potential Relation and Global Correspondence Based Joint Relational Triple Extraction. – In: Proc. of 59th Annual Meeting of the Association for Computational Linguistics and 11th International Joint Conference on Natural Language Processing. Volume **1**. Long Papers. 2021, pp. 6225-6235.
28. Z h e n g, S., Y. H a o, D. L u, H. B a o, J. X u, H. H a o et al. Joint Entity and Relation Extraction Based on a Hybrid Neural Network. – Neurocomputing, Vol. **257**, 2017, pp. 59-66.

Received: 17.02.2023; Second Version: 24.04.2023; Accepted: 03.05.2023 (fast track)