Cross-Domain Contract Element Extraction with a Bi-directional Feedback Clause-Element Relation Network

Zihan Wang[1]   Hongye Song[2]   Zhaochun Ren[1]*   Pengjie Ren[1]   Zhumin Chen[1]
Xiaozhong Liu[3]   Hongsong Li[2]   Maarten de Rijke[4,5]

[1]Shandong University, Qingdao, China
[2]Alibaba Group, Hangzhou, China
[3]Indiana University Bloomington, Bloomington, United States
[4]University of Amsterdam, Amsterdam, The Netherlands   [5]Ahold Delhaize, Zaandam, The Netherlands
zihanwang.sdu@gmail.com,{hongye.shy,hongsong.lhs}@alibaba-inc.com,{zhaochun.ren,chenzhumin}@sdu.edu.cn
jay.ren@outlook.com,liu237@indiana.edu,m.derijke@uva.nl

ABSTRACT

Contract element extraction (CEE) is the novel task of automatically identifying and extracting legally relevant elements such as contract dates, payments, and legislation references from contracts. Automatic methods for this task view it as a sequence labeling problem and dramatically reduce human labor. However, as contract genres and element types may vary widely, a significant challenge for this sequence labeling task is how to transfer knowledge from one domain to another, i.e., cross-domain CEE. Cross-domain CEE differs from cross-domain named entity recognition (NER) in two important ways. First, contract elements are far more fine-grained than named entities, which hinders the transfer of extractors. Second, the extraction zones for cross-domain CEE are much larger than for cross-domain NER. As a result, the contexts of elements from different domains can be more diverse.

We propose a framework, the *Bi-directional Feedback cLause-Element relaTion network* (Bi-FLEET), for the cross-domain CEE task that addresses the above challenges. Bi-FLEET has three main components: (1) a context encoder, (2) a clause-element relation encoder, and (3) an inference layer. To incorporate invariant knowledge about element and clause types, a clause-element graph is constructed across domains and a hierarchical graph neural network is adopted in the clause-element relation encoder. To reduce the influence of context variations, a multi-task framework with a bi-directional feedback scheme is designed in the inference layer, conducting both clause classification and element extraction. The experimental results over both cross-domain NER and CEE tasks show that Bi-FLEET significantly outperforms state-of-the-art baselines.

CCS CONCEPTS

• **Computing methodologies → Information extraction**; **Transfer learning**; • **Applied computing → Law**.

KEYWORDS

Legal information extraction and retrieval, Cross-domain information extraction, Contract element, Sequence labeling, Transfer learning

1 INTRODUCTION

Extracting information from contracts or other legal agreements is a fundamental task for businesses around the world [4, 6, 29]. Many thousands of contracts, relevant to a large variety of transactions (such as loans, investments, or leases), are drawn up every day. These contracts usually contain legally relevant elements such as termination dates or contract parties. Manually monitoring the legally relevant information in a large number of contracts is time-consuming, labor-intensive, and error-prone, putting a heavy burden on law firms, companies, and government agencies [37]. Automatic *contract element extraction* (CEE) is increasingly attracting interest [27]). As Fig. 1(top) illustrates, given a clause from a contract, the goal of the CEE task is to find the legally relevant elements (such as "within 5 days … fact") in the contract. CEE can bring useful insights into contracts and it can facilitate downstream applications, such as relevant clause retrieval or risk assessment [2].

Modern information extraction methods consider the CEE task as a sequence labeling problem, classifying each word as a (part of a) type of contract element [4, 6, 37]. The CEE task is challenging due to large variations in element mentions. Another important challenge for this sequence labeling problem is how to transfer knowledge from one domain to another [43]. For example, as Fig. 1 shows, compared with individual contracts (top), commercial contracts (bottom) are more formal, giving more precise and complex explanations in the contract clauses. Because of the differences in context genres and element types, transferring an element extractor from the individual domain to the commercial domain (or vice versa) is a challenging problem.

In this paper, we focus on cross-domain contract element extraction (CEE). Cross-domain CEE is similar to the cross-domain named

Clause from Individual Contract.

If the above qualification changes during the execution of the contract, resulting in Party B no longer having the relevant qualification to carry out the business under this contract, Party B shall notify Party A *within 5 days from the date of the occurrence of the fact* .
Element: notification date

Clause from Commercial Contract.

Party B agrees that under any following circumstances, Party A has the right to require Party B to take back the following products and bear the relevant expenses, including but not limited to refund of payment for *goods, labor costs, logistics and other expenses* :
1. Any product that violates laws and regulations,
2. Any product that infringes upon the intellectual property rights or other civil rights of others or any product causing infringement disputes.
Element: refund

Figure 1: Examples of contract element extractions. (Top): A clause from an individual contract. (Bottom): A clause from a commercial contract.

Table 1: Examples of the most frequent elements in clauses from different contract domains. Percentages in brackets indicate ratios of elements appearing in clauses from the individual/commercial contracts.

Clause	The most frequent elements (individual/commercial)
Payment	Payment period (23.8%/25.6%) and rate (12.3%/10.9%)
Deposit	Deposit rate (26.4%/40.6%) and amount (12.4%/14.5%)
Effectiveness	Effective mode (45.5%/31.3%) and time (17.7%/26.7%)

entity recognition (NER) task in many ways [4, 5]. Both tasks aim to identify key information (elements or entities) from texts in different domains and have similar problem formulations (discussed in detail in Sec. 3.1). However, existing cross-domain NER methods are not directly applicable to the cross-domain CEE task and fail to address two important challenges:

Challenge 1: Transfer of fine-grained element types. A generic named entity recognizer typically extracts only several types of entities, such as persons, organizations, locations, dates or amounts. However, in the CEE task, there are far more types of fine-grained contract elements (e.g., more than 70 in our dataset). For example, a typical entity recognizer may identify dates, but cannot distinguish between start, termination or other dates. In the cross-domain setting, the presence of a larger number of fine-grained contract elements makes it more difficult to transfer the extractor. To facilitate the transfer process, invariant knowledge about element types should be incorporated. As Table 1 shows, dependencies between clause and element types are usually shared across domains. E.g., the payment period usually exists in the payment clauses, but not the deposit clauses, which is independent of contract domains.

Challenge 2: Transfer of larger extraction zones. The NER task mainly focuses on locating entities in a single sentence. In contrast, an extractor for the CEE task needs to find contract elements from multiple sentences in a clause. That is, extraction zones (clauses) for contract elements are much larger than the contexts of entities. Sentences in clauses can be organized in various ways for different domains, which brings a new challenge for the cross-domain CEE task. To reduce the influence of sentence organizations, clause types should be decided first. Besides, classifying clauses is of great importance for extracting contract elements. E.g., if we know a clause is a loan clause, it is

easier to figure out that the dates in the clause are loan or repayment dates. Hence, conducting clause classification and element extraction at the same time can enhance the inference process.

To address the challenges identified above, we propose a framework, named the *Bi-directional Feedback cLause-Element relaTion network* (Bi-FLEET), that captures shared knowledge about element types in two domains and interactions between contract element extraction and clause classification. As Fig. 2 shows, Bi-FLEET is composed of three main components: (1) a context encoder, (2) a clause-element relation encoder, and (3) an inference layer. First, the context encoder embeds all sentences in clauses. Then, to capture invariant relations between element and clause types, in the clause-element relation encoder, a clause-element (C-E) graph is constructed across domains and a hierarchical graph neural network (GNN) is implemented to encode the shared clause-element dependencies. Finally, to identify the clause types and facilitate the element extraction process, a multi-task framework with a bi-directional feedback scheme is proposed in the inference layer. Similar to the human annotation process, both clause classification and contract element extraction tasks are conducted. To model interactions between tasks and improve the overall performance, forward and backward information is calculated using representations of contexts and types from the context and clause-element relation encoders, respectively. To evaluate our proposed framework to extract contract elements in cross-domain CEE, we collect both individual and commercial contracts to establish a cross-domain CEE dataset. Experimental results using both cross-domain NER and CEE datasets demonstrate the effectiveness of Bi-FLEET.

The contributions of this paper can be summarized as follows:
- To the best of our knowledge, ours is the first work on the cross-domain CEE task. Compared with cross-domain NER, more fine-grained element types and larger extraction zones bring critical challenges for the transfer process.
- We propose a framework, named the *Bi-directional Feedback cLause-Element relaTion network* (Bi-FLEET), to capture invariant clause-element dependencies and interactions between contract element extraction and clause classification.
- We establish a cross-domain CEE dataset by collecting individual contracts and commercial contracts. To the best of our knowledge, this is the first dataset for cross-domain CEE.
- Experimental results show that the proposed Bi-FLEET model achieves significant improvements over baselines in both cross-domain NER and CEE tasks.

2 RELATED WORK

We survey related work along three dimensions: (1) legal information retrieval and extraction, (2) contract element extraction, and (3) cross-domain named entity recognition.

2.1 Legal information retrieval and extraction

The digitization of legal documents has given rise to the development of legal information retrieval systems [20, 27]. Numerous challenges for legal information retrieval and extraction have been presented [8, 19]. Traditional legal search systems are keyword-based, relying heavily on the professional knowledge of end-users. Recently, to alleviate the dependence on user's expertise and improve

retrieval effectiveness, a significant amount of effort [12, 20, 33, 39] has been devoted to the automatic classification of legal documents and queries. The extraction of key legal concepts can also facilitate the retrieval process [2, 38].

To the best of our knowledge, there is no previous work on legal information retrieval and extraction that focuses on cross-domain contract element extraction (CEE).

2.2 Contract element extraction

The goal of CEE is to recognize essential legal elements, such as execution date, jurisdiction, and amount, in legal documents [9, 16]. Early CEE methods are mainly rule-based or traditional statistical methods. Chalkidis et al. [5] introduce 11 contract element types and extract contract elements based on Logistic Regression and SVM with hand-crafted features. García-Constantino et al. [13] design a system named CLIEL for extracting core information from commercial law documents. Specifically, CLIEL identifies five types of contract elements using rule-based layout detection. Azzopardi et al. [1] propose a hybrid approach based on regular expressions and provide a contract editing tool for lawyers.

Recent CEE methods are developed with deep learning and formulate the CEE task as sequence labeling. Chalkidis and Androutsopoulos [4] explore deep learning methods for the CEE task and employ a BiLSTM without manually written rules. Sun et al. [37] define seven semantic-specific clause categories and introduce a TOI pooling layer for the nested elements. Chalkidis et al. [6] revisit the CEE task and explore how sequence encoders, CRF layers, and input representations affect the extractors.

Existing CEE approaches cannot be applied to the cross-domain scenario because of differences in context genres and element types.

2.3 Cross-domain named entity recognition

The cross-domain NER task aims to identify named entities in a target domain using the shared knowledge from a source domain. Recently, methods based on deep neural networks have been proposed for cross-domain NER. Pan et al. [31] design transfer joint embeddings for this task. Qu et al. [34] model the correlation between source and target entity types with a two-layer neural network. To investigate the transferability of model components, Yang et al. [43] present three sharing architectures for different transfer learning scenarios. Moreover, Jia et al. [17] propose a parameter generation network and incorporate the language modeling (LM) task to deal with zero-shot learning settings. Unlike the above parameter-sharing frameworks, parameter transfer approaches first initialize the target model on the source domain NER or LM [22, 35], and then fine-tune the original model on the labeled NER data from the target domain. Lin and Lu [24] add three neural adaptation layers (word adaptation, sentence adaptation, and output adaptation) to an existing NER method. Jia and Zhang [18] transfer entity type level knowledge using a multi-cell compositional LSTM structure and model each entity type using a separate cell state. For a thorough review of other techniques, please refer to [23].
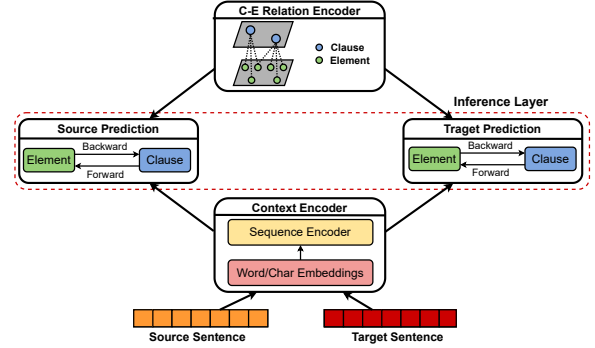

Figure 2: An overview of Bi-FLEET. Bi-FLEET has three components: a context encoder, a clause-element relation encoder, and an inference layer.

We mainly focus on cross-domain contract element extraction (CEE). To the best of our knowledge, ours is the first study to concentrate on this task. The most closely related task is the cross-domain NER task. However, cross-domain contract element extractors need to address adaptations of larger extraction zones and more fine-grained element types. In our proposed model Bi-FLEET, to handle the transfer of fine-grained element types, invariant relations between clauses and elements are captured by constructing a C-E graph across domains (see below). In addition, a multi-task framework with a bi-directional feedback scheme is designed to reduce the impact of more diverse contexts in larger extraction zones.

3 METHOD

In this section, we describe the Bi-FLEET framework, our proposed method for CEE. As illustrated in Fig. 2, the framework has three main components: a context encoder, a clause-element relation encoder, and an inference layer. The context encoder, which includes an input embedding layer and sequence encoder, embeds every word in a sentence from a given clause. The clause-element relation encoder is shared by the source and target domain and calculates representations of clause and element types. The word embeddings and type representations are input to the inference layer for clause classification and contract element extraction across domains.

Next, we formulate the cross-domain CEE task and compare it to the cross-domain NER task (Sec. 3.1). Then, we explain Bi-FLEET's context encoder (Sec. 3.2), the clause-element relation encoder (Sec. 3.3), and the inference layer (Sec. 3.4). Finally, the loss functions and training process (Sec. 3.5) are presented.

3.1 Problem formulation

We write $C = (s_1, s_2, \ldots, s_n)$ for a clause from a contract, where s_i is the i-th sentence in the clause C; $s_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,m})$, where $x_{i,j}$ is the j-th word in sentence s_i. A *contract element* e in the clause C is a sequence of words in one sentence: $e = \{(x_{i,start}, x_{i,start+1}, \ldots, x_{i,end}), l^e\}$, where l^e is the type label of the element e (such as payment period or deposit rate). The aim of the *contract element extraction* (CEE) task is to find the element e in the clause C.

For the cross-domain CEE task, there are N_s labeled clauses in the source domain $\mathcal{S}$, as well as N_t labeled clauses and N_u unlabeled clauses in the target domain $\mathcal{T}$. Our goal is to transfer the contract element extractor to the target domain from the source domain.

Specifically, the extractor is trained on the N_s labeled clauses in the source domain and N_t labeled clauses in the target domain to detect all the elements for the N_u unlabeled clauses in the target domain.

A similar task is cross-domain named entity recognition (NER), which focuses on identifying several types of named entities in only one sentence. In contrast, cross-domain CEE aims at extracting much more fine-grained elements in multiple sentences. As mentioned before, the transfer of larger extraction zones and more element categories brings new challenges to cross-domain CEE.

3.2 Context encoder

Similar to previous CEE and NER methods [4, 6, 18, 43], our context encoder vectorizes words in a sentence from a clause by using two components, an input embedding layer and a sequence encoder.

3.2.1 Input embedding layer. Given an input sentence $\mathbf{s} = \{x_1, x_2, x_3, \ldots, x_m\}$ from a clause $\mathbf{C}$, to capture the word-level and character-level features, each word x_i in the sentence $\mathbf{s}$ is embedded as the concatenation of the word embedding and the output of a char-level CNN:

$$\mathbf{v}_i = \mathbf{E}^w(x_i) \oplus \text{CNN}(\mathbf{E}^c(x_i)), \tag{1}$$

where $\mathbf{E}^w$ denotes a shared word-level embedding lookup table, and $\mathbf{E}^c$ denotes a shared char-level embedding lookup table. $\text{CNN}(\cdot)$ is a standard convolutional neural network operating on the character-level embedding sequence $\mathbf{E}^c(x_i)$ for word x_i in sentence $\mathbf{s}$. And $\oplus$ denotes the vector concatenation operator.

3.2.2 Sequence encoder. To encode sentence-level features, a standard bi-directional LSTM layer [15] is adopted here. Given an input embedding sequence $\mathbf{v} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m]$, at each time step, the Bi-LSTM layer calculates the current hidden vectors based on memory cells. The hidden outputs of the task and domain-specific Bi-LSTM unit can be written as follows:

$$\overrightarrow{\mathbf{h}}_i^{d,t} = \text{LSTM}(\overrightarrow{\mathbf{h}}_{i-1}^{d,t}, \mathbf{v}_i), \quad \overleftarrow{\mathbf{h}}_i^{d,t} = \text{LSTM}(\overleftarrow{\mathbf{h}}_{i+1}^{d,t}, \mathbf{v}_i), \tag{2}$$

where $\mathbf{h}_i^{d,t}$ denotes the output hidden vector for domain $d \in \{S, T\}$ and task $t \in \{CC, CEE\}$ (clause classification and contract element extraction) with the input word x_i; $\rightarrow$ and $\leftarrow$ represent the directions of the LSTM unit. Note that a variety of methods can be adopted as the context encoder. In our experiments, BERT [11] is adopted as the input embedding layer to generate the contextualized word embeddings. Besides, the sequence encoders based on the parameter generation network [17] and multi-cell compositional LSTM [18] are evaluated as well.

3.3 Clause-element relation encoder

To model relations between element and clause classes, we establish the clause-element (C-E) graph and learn representations of the clause and element types with a GNN-based C-E relation encoder.

3.3.1 C-E graph constructions. Before encoding clause and element types with a hierarchical graph neural network, we construct the C-E graph. As mentioned above, contract elements are strongly connected with categories of clauses. For a clause type ct and element type et in a given domain d, to identify the relation between them, we first define the probability $p_{et,ct}$, that the elements of type et exist in clauses of type ct, and $p_{ct,et}$, that clauses of type ct contain
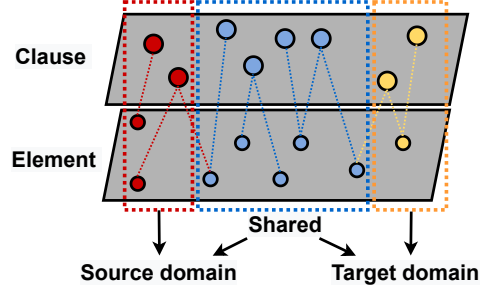

Figure 3: C-E graph construction. Blue nodes are the shared element and clause types, while red and yellow ones are the "not shared" types between the source and target domains.

elements of type et, as follows:

$$p_{et,ct} = f_{et,ct}/f_{et}, \quad p_{ct,et} = f_{et,ct}/f_{ct}, \tag{3}$$

where, in a given domain d, $f_{et,ct}$ is the frequency that elements of type et and clauses of type ct co-occur. Here, f_{et} and f_{ct} are the frequencies that the elements of type et and clauses of type ct appear in the given domain, respectively. Note that we only count once when multiple elements of the same types exist in one clause. When both $p_{et,ct}$ and $p_{ct,et}$ are larger than the preset threshold θ, the element type et and clause type ct are connected in the C-E graph of the given domain d. As Fig. 3 illustrates, we first select the edges between clause and element types (red and blue nodes) to establish the C-E graph of the source domain. And then, for the target domain, we only maintain the shared types (blue nodes) and add the domain-specific types (yellow nodes) into the original graph.

3.3.2 The GNN-based relation encoder. Given a C-E graph G in the domain d, we apply a hierarchical graph neural network to encode the element and clause types. Recent GNN-based frameworks follow the neighbor aggregation strategy [40, ?, 41], updating the embeddings of the central node by iteratively aggregating the information of its neighbors. The l-th layer of a GNN, composed of an aggregation function $\text{AGGREGATE}(\cdot)$ and a combination function $\text{COMBINE}(\cdot)$, can be presented as follows:

$$a_v^{(l)} = \text{AGGREGATE}^{(l)}\left(h_u^{(l-1)} : u \in \mathcal{N}_v\right)$$
$$h_v^{(l)} = \text{COMBINE}^{(l)}\left(h_v^{(l-1)}, a_v^{(l)}\right), \tag{4}$$

where h_v^l is the feature vector of node v at the l-th GNN layer and $\mathcal{N}_v$ is the set of nodes adjacent to the central node v. We set two transformation matrices $\mathbf{W}_{c \rightarrow e}$ and $\mathbf{W}_{e \rightarrow c}$ for the clause-to-element and element-to-clause relations respectively, and the l-th layer of our proposed C-E relation encoder can be formulated:

$$\mathbf{ele}_p^{d,(l)} = \sigma\left(\sum_{u \in \mathcal{N}_p^{d,e}} \alpha_u^{d,c} \mathbf{W}_{c \rightarrow e} \cdot \mathbf{cla}_u^{d,(l-1)} + \mathbf{W}_e \cdot \mathbf{ele}_p^{d,(l-1)}\right)$$

$$\mathbf{cla}_k^{d,(l)} = \sigma\left(\sum_{v \in \mathcal{N}_k^{d,c}} \alpha_v^{d,e} \mathbf{W}_{e \rightarrow c} \cdot \mathbf{ele}_v^{d,(l-1)} + \mathbf{W}_c \cdot \mathbf{cla}_k^{d,(l-1)}\right), \tag{5}$$

where, given a domain d, $\mathbf{ele}_p$ and $\mathbf{cla}_k$ are the representations of the element type et_p and clause type ct_k separately, while α_v^e
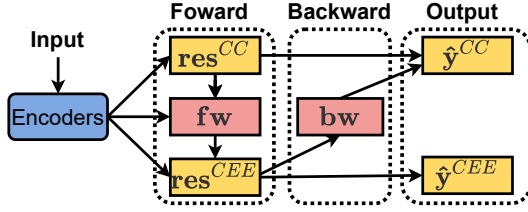
Figure 4: The inference layer with the bi-feedback scheme.

and α_u^c are the trainable weights for the v-th element type and u-th clause, type respectively. Given a C-E graph of domain d, $\mathcal{N}_p^e$ is the set of neighborhood clause types for et_p, and $\mathcal{N}_k^c$ is the set of neighborhood element types for ct_k. $\mathbf{W}_e$ and $\mathbf{W}_c$ denote the matrices for the self-connection. $\sigma(\cdot)$ is the activation function. $\mathbf{W}_{c\rightarrow e}$, $\mathbf{W}_{e\rightarrow c}$, $\mathbf{W}_e$ and $\mathbf{W}_c$ are shared in both source and target domains. If a clause or element types exists in both domains , their representations are also shared across domains.

3.4 Inference layer

Given the sentence-level feature and type representations of elements and clauses, the multi-task inference layer aims to predict the clause category labels (clause classification (CC)) and extract contract elements from clauses (contract element extraction (CEE)). To capture the interactions between the two tasks above, we design a novel bi-feedback scheme in the inference layer. In this section, we demonstrate the bi-feedback scheme first and then explain how to extract elements with forward information and verify the clause classification results using backward information.

3.4.1 The bi-feedback scheme. Given a clause from the contract, a human annotator usually first decides the clause type and then identifies the elements in the clause. In that case, the classification of clauses often facilitates the element extraction process, while the identification of elements can also verify the clause type predictions. Based on this observation, as shown in Fig. 4, we design a multi-task framework with the bi-feedback scheme. Given a training instance for task $t \in \{CC, CEE\}$, assuming that the result distribution vector $\mathbf{res}^t$ is predicted, we combine the result distribution vector with the clause or element representations , and obtain the latent type vector $\mathbf{ltv}^t$ for $\mathbf{res}^t$ as follows:

$$\mathbf{ltv}^{CC} = \sum_k res_k^{CC} \cdot \mathbf{cla}_k, \quad \mathbf{ltv}^{CEE} = \sum_p res_p^{CEE} \cdot \mathbf{ele}_p, \quad (6)$$

where, given a task t, res_k denotes the k-th type of the results. $\mathbf{cla}_k$ and $\mathbf{ele}_p$ are the representations of the k-th clause type or p-th element type. Then, we calculate the forward information $\mathbf{fw}$ and backward information $\mathbf{bw}$ based on the latent type vector $\mathbf{ltv}^t$:

$$\mathbf{fw} = \sigma(\mathbf{W}_f \cdot \mathbf{ltv}^{CC}), \quad \mathbf{bw} = \sigma(\mathbf{W}_b \cdot \mathbf{ltv}^{CEE}), \quad (7)$$

where $\mathbf{W}_f$ and $\mathbf{W}_b$ are the transformation matrices. Note that we set different transformation matrices and biases for source and target domains. In the rest of this section, we explain how to predict the result distribution $\mathbf{res}^t$ and obtain the final predictions, $\hat{\mathbf{y}}^{CC}$ and $\hat{\mathbf{y}}^{CEE}$, using forward and backward information. In our multi-task framework, $\mathbf{fw}$ and $\mathbf{bw}$ serve as gates to filter out the noise and reduce prediction mistakes.

3.4.2 Clause classification (CC). To classify the clause that the given sentence $\mathbf{s}$ belongs to, given a domain d, we use the sentence-level feature $\mathbf{h} = [\overrightarrow{\mathbf{h}}_1 \oplus \overleftarrow{\mathbf{h}}_1, \overrightarrow{\mathbf{h}}_2 \oplus \overleftarrow{\mathbf{h}}_2, \ldots, \overrightarrow{\mathbf{h}}_m \oplus \overleftarrow{\mathbf{h}}_m]$ from the Bi-LSTM. The k-th element of the result distribution vector $\mathbf{res}^{CC}$ is predicted by

$$res_k^{CC} = \sigma(\mathbf{w}_{CC} \cdot [\hat{\mathbf{h}} \oplus \mathbf{cla}_k] + b_{CC}), \quad (8)$$

where $\hat{\mathbf{h}} = \sum_i \alpha_i \cdot \mathbf{h}_i$ and $\sum_i \alpha_i = 1$. $\hat{\mathbf{h}}$ represents the overall feature of the given sentence and α_i denotes the trainable attention weight for word x_i. Considering the different label sets across domains, the parameters, $\mathbf{w}_{CC}$ and b_{CC}, are not shared. Based on the result distribution vector $\mathbf{res}^{CC}$, forward information $\mathbf{fw}_s$ for sentence $\mathbf{s}$ can be calculated by Eq. 6 and 7. The final prediction $\hat{\mathbf{y}}^{CC}$ can be verified by the backward information as follows:

$$\hat{\mathbf{y}}^{CC} = \text{softmax}(\mathbf{bw}_1 \otimes \mathbf{bw}_2 \otimes \cdots \otimes \mathbf{bw}_m \otimes \mathbf{res}^{CC}), \quad (9)$$

where $\mathbf{bw}_i$ denotes the backward information for word x_i in sentence $\mathbf{s}$ (from the contract element extraction task). softmax$(\cdot)$ is the softmax function. $\otimes$ is the element-wise product operator.

3.4.3 The contract element extraction. Similar to the NER task, standard CRFs [26] are adopted in the inference layer for CEE. Given a sentence $\mathbf{s}$ from the contract clause, the sentence-level feature $\mathbf{h} = [\overrightarrow{\mathbf{h}}_1 \oplus \overleftarrow{\mathbf{h}}_1, \overrightarrow{\mathbf{h}}_2 \oplus \overleftarrow{\mathbf{h}}_2, \ldots, \overrightarrow{\mathbf{h}}_m \oplus \overleftarrow{\mathbf{h}}_m]$, where $\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \oplus \overleftarrow{\mathbf{h}}_i$, can be obtained from the context encoder. We merge $\mathbf{fw}_s$ (from the clause classification task for sentence $\mathbf{s}$) with each hidden vector in $\mathbf{h}$:

$$\mathbf{h}^{fw} =$$
$$[\text{norm}(\mathbf{h}_1 \otimes \mathbf{fw}_s), \text{norm}(\mathbf{h}_2 \otimes \mathbf{fw}_s), \ldots, \text{norm}(\mathbf{h}_m \otimes \mathbf{fw}_s)], \quad (10)$$

where norm$(\cdot)$ is the normalization function. $\otimes$ is the element-wise product operator. Then, the output probability $p(\hat{\mathbf{y}}^{CEE}|\mathbf{s})$ of the final prediction sequence $\hat{\mathbf{y}}^{CEE} = [l_1, l_2, \ldots, l_m]$ over the input sentence $\mathbf{s}$ can be calculated by:

$$p(\hat{\mathbf{y}}^{CEE}|\mathbf{s}) = \frac{\exp\{\sum_i (\mathbf{w}_{CRF}^{l_i} \cdot \mathbf{h}_i^{fw} + b_{CRF}^{l_{i-1},l_i})\}}{\sum_{\hat{\mathbf{y}}'} \exp\{\sum_i (\mathbf{w}_{CRF}^{l_i'} \cdot \mathbf{h}_i^{fw} + b_{CRF}^{l_{i-1}',l_i'})\}}, \quad (11)$$

where $\hat{\mathbf{y}}'$ denotes an arbitary label sequence; $\mathbf{w}_{CRF}^{l_i}$ is the model parameter for l_i, and $b_{CRF}^{l_{i-1},l_i}$ is the bias for l_{i-1} and l_i. As Fig. 2 shows, we use CRF(s) and CRF(t) for the different label sets in the source and target domains respectively.

To effectively search the highest scored label sequence, the first-order Viterbi algorithm is used. We consider the result distribution vector $\mathbf{res}_i^{CEE}$ for the word x_i in the given sentence s as a one-hot vector. $res_{i,p}^{CEE} = 1$ only when the word x_i is classified as the p th element type. Then, the backward information $\mathbf{bw}_i$ for word x_i is obtained by Eq. 6 and 7.

3.5 Joint training

Given the labeled dataset $\mathcal{D} = \{(\mathbf{s}_n, \mathbf{y}_n^{CC}, \mathbf{y}_n^{CEE})\}_{n=1}^N$, loss functions for the clause classification task ($\mathcal{L}_{CC}$) and the contract element extraction task ($\mathcal{L}_{CEE}$) are defined over the training dataset. Specifically, for the clause classification task, the cross-entropy loss function is used as follows:

$$\mathcal{L}_{CC} = -\frac{1}{N} \sum_{n=1}^N \mathbf{y}_n^{CC} \log(p(\hat{\mathbf{y}}_n^{CC})). \quad (12)$$

Algorithm 1 Joint Training.

Require: Training data $\mathcal{D}_s$ and $\mathcal{D}_t$; parameter sets for the context encoder θ_{cont}, C-E relation encoder θ_{C-E} (including $\mathbf{W}_{c\to e}$, $\mathbf{W}_{e\to c}$, $\mathbf{W}_c$, $\mathbf{W}_e$, α^e, and α^c), and inference layer θ_{infer} (including $\mathbf{W}_f$, $\mathbf{W}_b$, $\mathbf{w}_{CC}$, b_{CC}, $\mathbf{w}_{CRF}$, and b_{CRF}).
Ensure: Target-domain model;
1: Contruct the C-E graphs $G^{\mathcal{S}}$ and $G^{\mathcal{T}}$ for both domains (Eq. 3);
2: **while** Training process not terminated **do**
3:      **for** d in $\{\mathcal{S}, \mathcal{T}\}$ **do**
4:          Compute sentence-level feature $\mathbf{h}^{d,t} \leftarrow \text{BiLSTM}(\mathbf{v})$ (Eq. 2);
5:          Calculate class representations $\mathbf{ele}^d$, $\mathbf{cla}^d$ (Eq. 5);
6:          Predict clause labels $\mathbf{res}^{d,CC}$ with $\mathbf{h}^{d,CC}$ and $\mathbf{cla}^d$ (Eq. 8);
7:          Compute the forward information $\mathbf{fw}^d$ with $\mathbf{res}^{d,CC}$ (Eq. 7);
8:          Predict element labels $\hat{\mathbf{y}}^{CEE}$ with $\mathbf{h}^{d,CEE}$ and $\mathbf{fw}^d$; (Eq. 10–11);
9:          Compute the backward information $\mathbf{bw}^d$ with $\mathbf{res}^{d,CEE}$ (Eq. 7);
10:          Verify the final prediction $\hat{\mathbf{y}}^{CC}$ using $\mathbf{bw}^d$ (Eq. 9);
11:          Compute the overall loss function $\mathcal{L}$ (Eq. 12–14);
12:      **end for**
13:      Update θ_{cont}, θ_{C-E}, and θ_{infer} based on $\mathcal{L}$.
14: **end while**

For the contract element extraction task, the sentence-level negative log-likelihood loss is adopted for the training process:

$$\mathcal{L}_{CEE} = -\frac{1}{N} \sum_{n=1}^{N} \log(p(\mathbf{y}_n^{CEE}|\mathbf{s}_n)). \tag{13}$$

And the overall loss is defined as follows:

$$\mathcal{L} = \sum_{d\in\{\mathcal{S},\mathcal{T}\}} \lambda^d (\mathcal{L}_{CEE}^d + \lambda^t \mathcal{L}_{CC}^d) + \frac{\lambda}{2}\|\Theta\|^2, \tag{14}$$

where λ^d is the domain weight and λ^t is the task weight. λ is the regularization weight for the parameter set Θ.

As Algorithm 1 shows, we design a cross-domain and multi-task training process. Before training, the C-E graphs are constructed (line 1). In each training time step, we encode the sentence-level feature $\mathbf{H}^{d,t}$ (line 4) and type representations, $\mathbf{ele}^d$ and $\mathbf{cla}^d$ (line 5). Then, the result distribution $\mathbf{res}^{d,CC}$ for clause classification is predicted (line 6) and forward information $\mathbf{fw}^d$ is computed based on the result distribution (line 7). For the contract element extraction, the element labels $\hat{\mathbf{y}}^{CEE}$ are calculated with forward information (line 8). Next, the backward information $\mathbf{bw}^d$ is computed (line 9) and the final predictions $\hat{\mathbf{y}}^{CC}$ for clause classification are verified (line 10). Finally, the overall loss function $\mathcal{L}$ is obtained (line 11) and the model parameters are jointly updated (line 13).

4 EXPERIMENTS

4.1 Research questions

We aim to answer the following research questions: (RQ1) Does Bi-FLEET outperform state-of-the-art methods on the cross-domain CEE taks? (Sec. 5.1) (RQ2) Can Bi-FLEET be generalized to the cross-domain NER task? (Sec. 5.2)

Table 2: Statistics of the CEE datasets.

Dataset	Type		Size	Train./Valid./Test
	Element	Clause		
Individual	70	18	#Pos Sentence	13.6K/1.7K/1.7K
			#Neg Sentence	60.0K/7.5K/7.5K
			#Element	25.7K/3.3K/3.3K
Commercial	79	17	#Pos Sentence	4.8K/0.6K/0.6K
			#Neg Sentence	19.7K/2.5K/2.5K
			#Element	8.7K/1.2K/1.1K

Table 3: Statistics of the NER datasets.

Dataset	Entity Type	Size	Train./Valid./Test
BioNLP13PC	CHEM,CC, GGP	#Sentence #Entity	2.5K/0.9K/1.7K 7.9K/2.7K/5.3K
BioNLP13CG	CHEM,CC, GGP, etc.	#Sentence #Entity	3.0K/1.0K/1.9K 10.8K/3.6K/6.9K
CoNLL-2003	PER, LOC, ORG, MISC	#Sentence #Entity	15.0K/3.5K/3.7K 23.5K/5.9K/5.6K
Broad Twitter	PER, LOC, ORG	#Sentence #Entity	6.3K/1.0K/2.0K 8.8K/1.7K/4.4K
Twitter	PER, LOC, ORG, MISC	#Sentence #Entity	4.3K/1.4K/1.5K 7.5K/2.5K/2.5K

4.2 Datasets

For cross-domain CEE, we collect 340 open individual Chinese contracts from the web and 1,422 business Chinese contracts from partners. For every contract, at least two annotators with a legal background were asked to label the clauses and contract elements. The whole annotation process took about 3 months. To establish the datasets, we first split the contracts sentence by sentence, and filtered out elements that appear no more than 20 times and sentences with more than 100 characters. Then, these sentences were divided into training/validation/test sets with a 8/1/1 ratio, ensuring that all the elements and clauses in the validation and test splits occur at least once in the training set. Detailed statistics are provided in Table 2. Different from NER, some sentences in clauses may not contain any elements. Such sentences are named "Neg" sentences in our experiments, while sentences in which elements do exist are called "Pos" sentences.

For cross-domain NER, we use five public datasets, BioNLP13PC and BioNLP13CG [30], CoNLL-2003 English dataset [36], Broad Twitter dataset [10], and Twitter dataset [25]. Detailed statistics of these datasets are shown in Table 3.

Following Jia and Zhang [18], we reconstruct two cross-domain CEE datasets (I2C and C2I) and three cross-domain NER datasets (BioNLP, Broad Twitter, and Twitter) based on the seven datasets mentioned above. For the I2C (or C2I) dataset, Commercial (or Individual) is used as the target-domain dataset, and the other contract dataset is the source-domain dataset. For the BioNLP dataset, the target-domain dataset is BioNLP13CG, while the source-domain dataset is BioNLP13PC. For the Broad Twitter dataset, Broad Twitter is the target-domain and CoNLL-2003 as the source-domain dataset. For the Twitter dataset, Twitter is used as the target-domain dataset, while CoNLL-2003 is the source-domain dataset.

4.3 Baselines

For cross-domain CEE and NER, we compare Bi-FLEET with recent baselines. These methods can be categorized into three groups:

- Target-domain only methods, BILSTM and MULTI-CELL LSTM [18], are trained without source-domain information.
- For cross-domain settings, we conduct comparisons with the LSTM-based MULTI-TASK (LSTM) [43] and two variants, MULTI-TASK+PGN and the method of Jia et al. [17], both using parameter generation networks (PGN) to generate parameters for source and target domain LSTMs. The multi-cell compositional LSTM based method of Jia and Zhang [18] is considered too.
- In addition, we evaluate language model (LM) pretraining based methods, including BERT [11], BIOBERT [21], and the method of Jia and Zhang (BERT) [18] that leverages the outputs of LM pretraining methods as contextualized word embeddings.

As mentioned in Sec. 3.2, we adopt multiple types of context encoders. Specifically, Bi-FLEET (LSTM) uses the standard LSTM layer as the sequence encoder. Bi-FLEET (PGN) adds an PGN to generate parameters for domain-specific LSTMs. Similar to the MULTI-TASK (LSTM) and MULTI-TASK+PGN, both Bi-FLEET (LSTM) and Bi-FLEET (PGN) employ the max-margin principle [14] for loss functions. Bi-FLEET (MULTI-CELL) and Bi-FLEET (BERT) utilize the multi-cell compositional LSTM and BERT as context encoders.

4.4 Evaluation metrics

Following [17, 18, 36, 37], we employ entity (or element) level Precision (P), Recall (R), F1-score (F1), and Accuracy for evaluation. Precision indicates the percentage of named entities (or elements) extracted by the method that are correct. Recall represents the percentage of entities (or elements) in the datasets that are predicted by the method. To this end, an entity (or element) is correct only when the corresponding entity (or element) in the dataset is exactly matched. The F1-score is the harmonic mean of Precision and Recall. We report P, R, and F1 for the cross-domain CEE task and F1 for the cross-domain NER task. We apply Accuracy to assess the proportion of clause type predictions that is correct.

4.5 Implementation details

Our parameter settings mainly follow Jia and Zhang [18]. We use the NCRF++ toolkit [42] to develop our models. All word embeddings are fine-tuned in the training process, while character embeddings are randomly initialized. The batch size is set to 30, and the initial learning rates for target-domain only, cross-domain, and BERT-based methods are 0.001, 0.0015, and 3e-5 respectively. The number of the layers of the C-E relation encoder is set to 3.

For cross-domain CEE, word embeddings of 100 dimensions are pretrained on the Baidu Encyclopedia and Weibo corpus [3]. For BERT fine-tuning methods, we use the base-sized BERT [11] pretrained on the Chinese Wikipedia corpus.

For cross-domain NER, word embeddings are initialized with PubMed 200 dimension vectors [7] for BioNLP experiments and GloVe 100 dimension vectors [32] for others. In cross-domain NER, "clause" labels do not exist. In that case, we employ the base-sized BERT [11] and BIOBERT [21] models to embed each sentence and then cluster [CLS] vectors by the K-Means algorithm to generate the labels of sentences in an unsupervised manner. Specifically, we
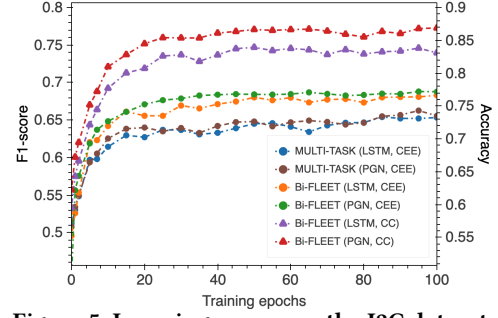

Figure 5: Learning curves on the I2C dataset.

Table 4: Results on two cross-domain CEE datasets. Significant improvements against the best baseline with the same context encoder are marked with $*$ (t-test, $p < 0.05$).

Model	I2C			C2I		
	P	R	F1	P	R	F1
BILSTM (Random)	60.35	64.02	62.13	63.65	68.32	65.90
BILSTM	61.29	65.18	63.18	64.92	69.87	67.30
MULTI-TASK (Random)	62.14	65.89	63.96	65.30	70.36	67.73
MULTI-TASK (LSTM)	63.21	67.50	65.28	65.86	71.61	68.62
MULTI-TASK+PGN	63.67	67.77	65.66	65.99	71.91	68.83
Jia et al. [17]	62.57	66.96	64.69	65.45	70.84	68.04
BERT-BASE	67.57	71.25	69.36	68.72	75.70	72.04
Bi-FLEET (LSTM)	66.07*	70.20*	68.07*	67.98*	75.26*	71.44*
Bi-FLEET (PGN)	66.53*	70.45*	68.43*	68.31*	75.37*	71.69*
Bi-FLEET (BERT)	**70.17***	**73.75***	**71.92***	**70.71***	**78.69***	**74.49***

cluster sentences into 20 groups and represent these groups with the clause-element relation encoder.

5 EXPERIMENTAL RESULTS

Prior to addressing our research questions RQ1 and RQ2, we examine the training process of Bi-FLEET. In particular, we examine whether cross-domain contract element extraction (CEE) and clause classification (CC) conflict with each other during the training process. We use the I2C dataset for this purpose. In Fig. 5, the learning curves of cross-domain CEE and CC against the training epochs are shown. MULTI-TASK (LSTM) and MULTI-TASK+PGN are the cross-domain baselines. For comparison, we also show the learning curves of Bi-FLEET (LSTM) and Bi-FLEET (PGN). We see that:

- F1-scores of all models increase continuously before 60 training epochs, and reach a stable level when the number of training epoch rises to 100.
- Cross-domain CEE and CC share the same improvement trend, and there is no conflict between the two tasks during the training process, indicating that both tasks can adopt the same form of feature space.

5.1 Cross-domain CEE

We turn to RQ1. Table 4 shows the experimental outcomes for the cross-domain CEE on the I2C and C2I datasets. To investigate the influence of pretrained word embeddings, we set up two more baseline models, BILSTM (Random) and MULTI-TASK (Random). In our experiments, the multi-cell compositional LSTM based models [18]

Table 5: Results on three cross-domain NER datasets. Significant improvements against the best baseline with the same context encoder are marked with ∗ (t-test, $p < 0.05$).

Model	Datasets (F1)		
	BioNLP	Broad Twitter	Twitter
BILSTM	79.24	72.98	77.18
MULTI-CELL LSTM	78.76	72.54	77.05
MULTI-TASK (LSTM)	81.06	73.84	79.55
MULTI-TASK+PGN	81.17	73.70	80.07
Jia et al. [17]	79.86	–	–
Jia and Zhang [18]	83.12	74.82	81.37
BERT-BASE	–	77.28	83.77
BIOBERT-BASE	85.72	–	–
Jia and Zhang [18] (BERT)	86.96	78.43	85.80
Bi-FLEET (LSTM)	82.32*	74.91*	80.78*
Bi-FLEET (PGN)	82.46*	75.11*	81.35*
Bi-FLEET (MULTI-CELL)	84.76*	76.02*	82.24*
Bi-FLEET (BERT)	**89.02***	**80.71***	**87.73***

overfit very easily, since their parameter counts increase drastically with element types; for this reason we do not report the results of multi-cell compositional LSTM based methods. Based on Table 4, we can draw the following conclusions:

- Cross-domain CEE is challenging, and for most baselines the F1-score is substantially less than 0.7. In contrast, our proposed framework, Bi-FLEET, can effectively transfer contract element extractors from one domain to another.
- The proposed model Bi-FLEET significantly outperforms the baselines. Compared to the baseline models with the same context encoder, Bi-FLEET (LSTM), Bi-FLEET (PGN) and Bi-FLEET (BERT) can obtain a consistent increase in P, R, and F1-scores, respectively. That is, integrated with the C-E relation encoder and bi-feedback scheme improves the performance across the board.
- In comparison with target-domain only methods, all of the cross-domain methods achieve significant improvements, indicating the importance of source-domain information. Compared to methods with randomly initialized word embeddings, including BILSTM (Random) and MULTI-TASK (Random), BILSTM and MULTI-TASK (LSTM) that use pretrained word embeddings are able to obtain large performance enhancements. Combining with a pretrained LM is also helpful for the cross-domain task, since BERT-based methods perform best among all methods.

In summary, the C-E relation encoder and bi-feedback scheme enhance the performance. Encoding relations between elements and clauses, as well as capturing interactions between clause classification and contract element extraction, are both beneficial for the cross-domain CEE problem.

5.2 Cross-domain NER

To investigate the generality of Bi-FLEET, we turn to RQ2 and report on experiments on the cross-domain NER datasets. F1-scores are adopted here to evaluate the overall performance. As baselines, we use the results reported in [18]. Based on the results in Table 5, we arrive at the following conclusions:

- Even if labels are generated in an unsupervised manner, Bi-FLEET can still effectively transfer named entity recognizers.
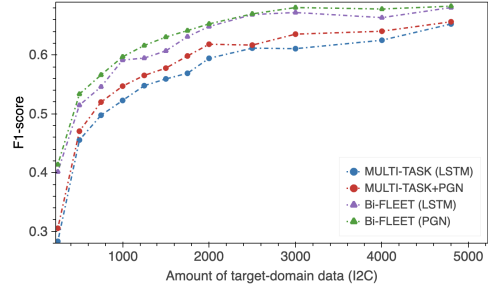

Figure 6: Influence of target-domain data.

- Bi-FLEET outperforms state-of-the-art cross-domain NER methods. Lacking source-domain information, target-domain only methods can hardly handle the cross-domain NER task. Compared to cross-domain and BERT-based NER methods, methods based on the Bi-FLEET framework attain the highest F1-scores on all three datasets.

In summary, the Bi-FLEET framework cannot only effectively address the cross-domain CEE problem, but also achieves state-of-the-art results on the cross-domain NER task.

6 ANALYSIS

Now that we have answered our research questions, we take a closer the look at Bi-FLEET to analyze its performance. We examine how the C-E relation encoder and bi-feedback scheme contribute to its performance, how the amount of target-domain data influences the performance, and how performance varies across element types.

6.1 Ablation studies

We conduct ablation studies on both cross-domain CEE and NER datasets. The results are shown in Table 7. When we only ablate the C-E relation encoder ("- C-E graph"), type representations **cla** and **ele** are randomly initialized. In that case, F1-scores over all of the three datasets and model variants suffer a severe drop. Since the C-E relation encoder cannot be directly incorporated, we cannot remove the bi-feedback scheme alone. When we ablate both the C-E relation encoder and bi-feedback scheme ("- Bi-feedback"), our models obtain a similar performance as baselines (MULTI-TASK (LSTM), MULTI-TASK+PGN and BERT). In short, both the C-E relation encoder and the bi-feedback scheme contribute to the improvements in performance on the cross-domain CEE and NER tasks.

6.2 Influence of target-domain data

Next, we study the influence of target-domain data on the I2C dataset. We compare the F1-scores of the baselines and Bi-FLEET with different amounts of target-domain data in 100 training epochs. We keep "Neg" sentences unchanged and adjust the number of "Pos" sentences, since "Neg" sentences do not contain any type of elements. See Fig. 6 for the results. Initially, Bi-FLEET obtains a large improvement of more than 10% over baselines. The gap between Bi-FLEET and baselines becomes smaller with the increase in target-domain data. Both Bi-FLEET (LSTM) and Bi-FLEET (PGN) outperform the baselines with varying numbers of target-domain data, which demonstrates the effectiveness of our methods.

Table 6: Examples from the I2C dataset. Red and green represent incorrect and correct contract elements, respectively.

Models	Sentences
MULTI-TASK (LSTM)	"Party B shall return the remuneration received and pay 10% (**Payment Rate**) of the total remuneration as liquidated damages." (**Indemnity Clause**) "According to the construction area (**O**), the unit price of the commercial housing is XXX yuan per square meter"
Bi-FLEET (LSTM)	"Party B shall return the remuneration received and pay 10% (**Compensation Ratio**) of the total remuneration as liquidated damages." (**Indemnity Clause**) "According to the construction area (**Calculation Standard**), the unit price of the commercial housing is XXX yuan per square meter"

Table 7: Ablation studies on the BioNLP, I2C and C2I datasets.

Model	Datasets (F1)		
	BioNLP	I2C	C2I
Bi-FLEET (LSTM)	**82.32**	**68.07**	**71.44**
- C-E graph	81.78	66.95	70.03
- Bi-feedback	81.03	65.29	68.59
Bi-FLEET (PGN)	**82.46**	**68.43**	**71.69**
- C-E graph	81.85	67.37	70.15
- Bi-feedback	81.07	65.58	68.72
Bi-FLEET (BERT)	**89.02**	**71.92**	**74.49**
- C-E graph	88.38	70.58	73.31
- Bi-feedback	86.89	69.32	72.06

Table 8: Fine-grained comparisons on the I2C dataset.

Model	Element types (F1)				
	IT	AC	CP	NSM	PP
BILSTM	53.46	60.31	62.75	64.14	67.20
MULTI-TASK (LSTM)	57.92	63.61	63.80	65.43	67.39
MULTI-TASK+PGN	58.05	63.38	64.26	65.65	67.92
Bi-FLEET (LSTM)	62.86	63.94	67.68	68.82	68.15
Bi-FLEET (PGN)	**63.74**	**64.03**	**67.94**	**69.42**	**69.10**

6.3 Fined-grained comparisons

To understand the performance of Bi-FLEET at the element type level, we make fine-grained comparisons on the I2C dataset. As shown in Table 8, we analyze five element types: invoice type (IT), arbitration commission (AC), currency of payment (CP), name of subject matter (NSM), and payment period (PP). Models based on Bi-FLEET accomplish significant F1-score improvements over the baselines, demonstrating that modeling C-E relations and uncovering the bi-feedback scheme are helpful for reducing confusion between element types.

6.4 Case studies

Table 6 provides examples from the I2C dataset. "10%" is a compensation ratio, and "According to the construction area" is a calculation standard. Misled by the keywords "pay" and "total remuneration", MULTI-TASK (LSTM) misjudges "10%" as a payment rate. Due to the data sparsity for the calculation standard, MULTI-TASK (LSTM) is not able to recognize "According to the construction area". By uncovering interactions between the clause classification (CC)

and contract element extraction (CEE) tasks, Bi-FLEET can correctly extract the compensation rate in the indemnity clause. Identifying invariant knowledge about element types makes it much easier to transfer long-tail element types.

7 CONCLUSIONS

In this paper, we have investigated the new problem of cross-domain contract element extraction; solutions to this problem can support clause retrieval and other legal search tasks. Compared to cross-domain named entity recognition, there are two main challenges: more fine-grained element types and larger extraction zones. To address these challenges, we have proposed a framework named Bi-FLEET that captures invariant relations between clauses and elements, and we have designed a bidirectional feedback scheme between the CEE and clause classification tasks. Experimental results on cross-domain CEE and NER datasets show that Bi-FLEET significantly outperforms state-of-the-art baselines, with a high degree of generalizability.

A limitation of Bi-FLEET is that it can only extract elements sentence by sentence. In future work, we intend to consider the clause-level features, and explore relations between elements.

REFERENCES

[1] Shaun Azzopardi, Albert Gatt, and Gordon J Pace. 2016. Integrating natural language and formal analysis for legal documents. In *10th Conference on Language Technologies and Digital Humanities*. 1–4.

[2] Lukasz Borchmann, Dawid Wisniewski, Andrzej Gretkowski, Izabela Kosmala, Dawid Jurkiewicz, Lukasz Szalkiewicz, Gabriela Palka, Karol Kaczmarek, Agnieszka Kaliska, and Filip Gralinski. 2020. Contract discovery: Dataset and a few-shot semantic retrieval challenge with competitive baselines. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*. 4254–4268.

[3] Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, and Shengping Liu. 2018. Adversarial transfer learning for Chinese named entity recognition with self-attention mechanism. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 182–192.

[4] Ilias Chalkidis and Ion Androutsopoulos. 2017. A deep learning approach to contract element extraction. In *Legal Knowledge and Information Systems - JURIX 2017: The Thirtieth Annual Conference*. 155–164.

[5] Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2017. Extracting contract elements. In *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*. 19–28.

[6] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. 2019. Neural contract element extraction revisited. In *Workshop on Document Intelligence at NeurIPS 2019*.

[7] Billy Chiu, Gamal K. O. Crichton, Anna Korhonen, and Sampo Pyysalo. 2016. How to train good word embeddings for biomedical NLP. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*. Association for Computational Linguistics, 166–174.

[8] Heting Chu. 2011. Factors affecting relevance judgment: A report from TREC Legal track. *J. Documentation* 67, 2 (2011), 264–278.

[9] Michael Curtotti and Eric Mccreath. 2010. Corpus based classification of text in Australian contracts. In *Proceedings of the Australasian Language Technology Association Workshop*.

[10] Leon Derczynski, Kalina Bontcheva, and Ian Roberts. 2016. Broad Twitter corpus: A diverse named entity recognition resource. In *COLING 2016, 26th International Conference on Computational Linguistics*. 1169–1179.

[11] 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*. 4171–4186.

[12] 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* (2017).

[] Fuli Feng, Xiangnan He, Jie Tang, and Tat-Seng Chua. 2019. Graph adversarial training: Dynamically regularizing based on graph structure. *IEEE Transactions on Knowledge and Data Engineering* (2019).

[] Fuli Feng, Weiran Huang, Xiangnan He, Xin Xin, Qifan Wang, and Tat-Seng Chua. 2021. Should Graph Convolution Trust Neighbors? A Simple Causal Inference Method. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[13] Matías García-Constantino, Katie Atkinson, Danushka Bollegala, Karl Chapman, Frans Coenen, Claire Roberts, and Katy Robson. 2017. CLIEL: Context-based information extraction from commercial law documents. In *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*. 79–87.

[14] Kevin Gimpel and Noah A. Smith. 2010. Softmax-margin CRFs: Training log-linear models with cost functions. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics*. 733–736.

[15] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks* 18, 5-6 (2005), 602–610.

[16] Kishore Varma Indukuri and P Radha Krishna. 2010. Mining e-contract documents to classify clauses. In *Proceedings of the 3rd Bangalore Annual Compute Conference*. 1–5.

[17] Chen Jia, Liang Xiao, and Yue Zhang. 2019. Cross-domain NER using cross-domain language modeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. 2464–2474.

[18] Chen Jia and Yue Zhang. 2020. Multi-Cell compositional LSTM for NER domain adaptation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 5906–5917.

[19] Yoshinobu Kano, Mi-Young Kim, Randy Goebel, and Ken Satoh. 2017. Overview of COLIEE 2017. In *4th Competition on Legal Information Extraction and Entailment, held in conjunction with the 16th International Conference on Artificial Intelligence and Law*, Vol. 47. 1–8.

[20] Phi Manh Kien, Ha-Thanh Nguyen, Ngo Xuan Bach, Vu Tran, Minh Le Nguyen, and Tu Minh Phuong. 2020. Answering legal questions by learning neural attentive rext representation. In *Proceedings of the 28th International Conference on Computational Linguistics*. 988–998.

[21] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinform*. 36, 4 (2020), 1234–1240.

[22] Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2018. Transfer learning for named-entity recognition with neural networks. In *Proceedings of the 11th International Conference on Language Resources and Evaluation*.

[23] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[24] Bill Yuchen Lin and Wei Lu. 2018. Neural adaptation layers for cross-domain named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2012–2022.

[25] Di Lu, Leonardo Neves, Vitor Carvalho, Ning Zhang, and Heng Ji. 2018. Visual attention model for name tagging in multimodal social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 1990–1999.

[26] Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bidirectional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

[27] K. Tamsin Maxwell and Burkhard Schafer. 2008. Concept and context in legal information retrieval. In *JURIX 2008: The Twenty-First Annual Conference on Legal Knowledge and Information Systems*, Enrico Francesconi, Giovanni Sartor, and Daniela Tiscornia (Eds.), Vol. 189. 63–72.

[29] Zoran Milosevic, Simon Gibson, Peter F Linington, James Cole, and Sachin Kulkarni. 2004. On design and implementation of a contract monitoring facility. In *Proceedings of the 1st IEEE International Workshop on Electronic Contracting*. 62–70.

[30] Claire Nédellec, Robert Bossy, Jin-Dong Kim, Jung-Jae Kim, Tomoko Ohta, Sampo Pyysalo, and Pierre Zweigenbaum. 2013. Overview of BioNLP Shared Task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*. Association for Computational Linguistics, 1–7.

[31] Sinno Jialin Pan, Zhiqiang Toh, and Jian Su. 2013. Transfer joint embedding for cross-domain named entity recognition. *ACM Transactions on Information Systems* 31, 2 (2013), 1–27.

[32] 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*. 1532–1543.

[33] Amal Shehan Perera, Vindula Jayawardana, Dimuthu Lakmal, and Madhavi Perera. 2018. Legal document retrieval using document vector embeddings and deep learning. In *Intelligent Computing: Proceedings of the 2018 Computing Conference*, Vol. 857. 160.

[34] Lizhen Qu, Gabriela Ferraro, Liyuan Zhou, Weiwei Hou, and Timothy Baldwin. 2016. Named entity recognition for novel types by transfer learning. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 899–905.

[35] Devendra Singh Sachan, Pengtao Xie, Mrinmaya Sachan, and Eric P Xing. 2018. Effective use of bidirectional language modeling for transfer learning in biomedical named entity recognition. In *Machine Learning for Healthcare Conference*. PMLR, 383–402.

[36] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the 7th Conference on Natural Language Learning*. 142–147.

[37] Lin Sun, Kai Zhang, Fule Ji, and Zhenhua Yang. 2019. Toi-CNN: A solution of information extraction on Chinese insurance policy. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*. 174–181.

[38] Vu D. Tran, Minh Le Nguyen, and Ken Satoh. 2019. Building Legal Case Retrieval Systems with Lexical Matching and Summarization using A Pre-Trained Phrase Scoring Model. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*. 275–282.

[39] Vu D. Tran, Minh Le Nguyen, Satoshi Tojo, and Ken Satoh. 2020. Encoded summarization: summarizing documents into continuous vector space for legal case retrieval. *Artif. Intell. Law* 28, 4 (2020), 441–467.

[40] Zihan Wang, Zhaochun Ren, Chunyu He, Peng Zhang, and Yue Hu. 2019. Robust embedding with multi-level structures for link prediction. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 5240–5246.

[41] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks?. In *7th International Conference on Learning Representations*.

[42] Jie Yang and Yue Zhang. 2018. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of the 56th Conference of the Association for Computational Linguistics*. 74–79.

[43] Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *5th International Conference on Learning Representations*.