ParaBART: A Prompt-based Method with Parabiotic Decoder for Few-shot Named Entity Recognition

Anonymous ACL submission

Abstract

Prompt-based methods have been widely used in few-shot named entity recognition (NER). We first conduct a preliminary experiment and observe that what really affects prompt-based NER models is the ability to detect entity boundaries. However, previous prompt-based NER models neglect to enhance the ability of entity boundary detection. To solve the issue, we propose a novel method, ParaBART, which consists of a BART encoder and the Parabiotic[1] Decoder we design. Parabiotic Decoder includes two BART decoders and a conjoint module. The two decoders are responsible for entity boundary detection and entity type classification respectively and share the well-learned knowledge through the conjoint module, which replaces unimportant tokens' embeddings in one decoder with the average embedding of all tokens in the other decoder. Moreover, we propose a novel boundary expansion strategy to enhance the ability of entity type classification. Experimental results show that ParaBART can achieve significant performance gains over previous state-of-the-art methods. For reproducibility, all datasets and codes are provided in the supplementary materials.

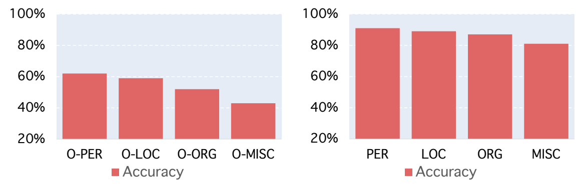
Republicans controlled [the [White House]$_{\text{ORG}}$]$_{\text{O-ORG}}$ ...

[The [Congress]$_{\text{ORG}}$]$_{\text{O-ORG}}$ hold that ...

[Mr. [Adel Ibrahim]$_{\text{PER}}$]$_{\text{O-PER}}$ has asked ...

... by [[John Doe]$_{\text{PER}}$, Jr.]$_{\text{O-PER}}$

Figure 1: Examples of `O-entity` on CoNLL03 dataset. An `O-entity` span means the span is not an entity but it is very similar to a certain entity span.



(a) (b)

Figure 2: Results of the preliminary experiment on CoNLL03 dataset. (a) More than half of `O-entity` spans are predicted incorrectly. (b) High accuracy for entity type classification when entity boundaries are known.

1 Introduction

Named entity recognition (NER) is a fundamental task in Natural Language Processing (NLP), which aims to identify and categorize spans of text into a set of pre-defined entity types, such as `people`, `organization`, and `location`. While a considerable number of approaches (Li et al., 2020; Yadav and Bethard, 2019) based on deep neural networks have shown remarkable success in NER, they generally require massive labeled data as training set. Unfortunately, in some specific domains, named entities that need professional knowledge to understand are difficult to be manually annotated in a large scale.

To address the issue, few-shot NER has been proposed, which aims to improve the performance of NER models on the few-shot scenario. Recently, prompt-based methods achieve impressive results and show promising prospects for few-shot NER (Cui et al., 2021; Ma et al., 2021; Hou et al., 2022). Instead of adapting Pre-trained Language Models (PLMs) to downstream tasks directly, prompt-based methods reformulate downstream tasks to keep pace with those solved during the original PLMs pre-training by resorting to a textual prompt. For example, when recognizing named entities in the sentence, "*ACL will be held in Toronto*", we may continue with a prompt "*`<candidate_span>` is a ____ entity*". Specifically, the `<candidate_span>` can be replaced by all possible textual spans (e.g. *"Toronto"*) in

[1]*Parabiotic*, a biological term, means combining two living organisms which are joined together surgically to develop a single, shared physiological system.

the original sentence. After that, we will ask the PLM to fill the blank with an entity type (e.g. *"location"*).

NER can be further decomposed into two subtasks: entity boundary detection and entity type classification. We conduct a preliminary experiment[2] in the 10-shot setting on the CoNLL03 dataset to determine what exactly affects the performance of prompt-based methods on few-shot NER. On the one hand, we choose the text spans[3] that are not entities but very similar to entity spans and label the type of spans as `O-entity`. For example, the type of *"The White House"* is labeled as `O-Organization` (as shown in Figure 1). Then we make predictions for `O-entity` spans. On the other hand, we assume that the entity boundaries are known, and only classify the entity spans. The results are shown in Figure 2. The accuracy for `O-entity` spans is very low. We observe that most of them are confused into the entity types they are similar to. For example, *"The White House"*, which belongs to `Other(O)` class, is predicted incorrectly to `Organization`. The type of error is caused by the model's insufficient ability to detect entity boundaries. Conversely, the prompt-based model[2] has great accuracy for entity type classification. The results show that **what really affects prompt-based models is the ability of entity boundary detection, rather than entity type classification**. Although previous prompt-based methods have achieved good performance, but all of them neglect to enhance the ability of entity boundary detection.

Therefore, in this paper, we propose a BART-based model with Parabiotic[1] Decoder, namely, **ParaBART**, to enhance the ability of entity boundary detection. ParaBART consists of a BART encoder and the Parabiotic Decoder we propose. Parabiotic Decoder includes two BART decoders and a conjoint module. The two decoders are responsible for entity boundary detection and entity type classification respectively. The conjoint module aims to share the well-learned knowledge between the two decoders, which replaces unimportant tokens' embeddings in one decoder with the average embedding of all tokens in the other decoder. The two decoders are like a parabiotic system so we name it Parabiotic Decoder. In addition, inspired by label smoothing (Szegedy et al., 2016; Müller et al., 2019), we propose a novel boundary expansion strategy to improve the ability of entity type classification. We summarize our main contributions in this paper as follows.

- We propose a BART-based model with Parabiotic Decoder (ParaBART) to enhance the ability to detect entity boundaries for few-shot NER.
- We design a novel boundary expansion strategy for improving the ability to classify entity types.
- We perform extensive experiments to show the superiority of ParaBART over other competitors.

2 Related Work

2.1 Prompt-based learning

Despite the success of Pretrained Language Models (PLMs) (Devlin et al., 2018; Liu et al., 2019; Yang et al., 2019) in massive NLP tasks, most of them are hard to fine-tune in low-resource scenarios due to the gap between pre-training and downstream tasks. Inspired by GPT-3 (Brown et al., 2020), stimulating model knowledge with a few prompts has recently received much attention. In prompt-based learning, instead of adapting PLMs to downstream tasks via objective engineering, downstream tasks are reformulated to keep pace with those solved during the original LM training with the help of a textual prompt. Early attempts (Schick and Schütze, 2021a,b) introduce manual prompts to text classification tasks. Building manual prompts requires the knowledge of domain experts, limiting the application of prompt-based methods in real-world scenarios. To solve this problem, automatically searching discrete prompts methods are proposed such as AUTOPROMPT (Shin et al., 2020) and LM-BFF (Gao et al., 2021). Meanwhile, generating continuous prompts through neural networks for both text classification and generation tasks (Han et al., 2021; Li and Liang, 2021) have been proposed. Although prompt-based methods are proved to be useful in sentence-level tasks, they are very complicated for NER task, which will be introduced in Section 2.2.

2.2 Few-shot NER

Few-shot NER has recently received much attention (Huang et al., 2020; Hou et al., 2020; Das et al.,

[2]The model used in the preliminary experiment is TemplateBART (Cui et al., 2021).

[3]We construct `O-entity` spans by adding entity spans to its previous or subsequent word in practice.

2021). The current mainstream methods for few-shot NER can be grouped into two main categories:

Meta-learning-based methods Fritzler et al. (2019a) combine PROTO (Snell et al., 2017) with conditional random field for few-shot NER. Inspired by the nearest neighbor inference (Wiseman and Stratos, 2019), StructShot (Yang and Katiyar, 2020) employs structured nearest neighbor learning and Viterbi algorithm to further improve PROTO. MUCO (Tong et al., 2021) trains a binary classifier to learn multiple prototype vectors for representing miscellaneous semantics of `O`-class. CONTaiNER (Das et al., 2021) proposes a contrastive learning method that optimizes the inter-token distribution distance for few-shot NER. ESD (Wang et al., 2021) uses various types of attention based on PROTO to improve the model performance. Ma et al. (2022) addresses few-shot NER by sequentially tackling few-shot span detection and few-shot entity typing using meta-learning. However, most of these methods assume a resource-rich source domain. In the few-shot setting without a data-rich source domain, the performance of these methods is limited.

Prompt-based methods Cui et al. (2021) uses BART (Lewis et al., 2020) as the backbone and constructs templates by dividing sentences into spans for few-shot NER. EntLM (Ma et al., 2021) proposes a template-free approach through replacing entity spans with verbalizers. LightNER (Chen et al., 2021) generates a index of an entity span in the input as well as a label word. ProtoVerb (Cui et al., 2022) combines PROTO (Snell et al., 2017) and prompt-based learning by generating prototype vectors as verbalizers for few-shot NER. QaNER (Liu et al., 2022) proposes a refined strategy for converting NER problem into the Question Answering (QA) formulation and generates templates for QA models. Hou et al. (2022) improves model prediction efficiency by introducing an inverse paradigm. Although previous prompt-based methods have achieved good performance, but all of them neglect to enhance the ability of entity boundary detection.

3 Problem Definition

In this work, we focus on few-shot NER task. Specifically, a training set $\mathcal{D}_{train}$ consists of word sequences and their label sequences. Given a word sequence $X = \{x_1, ..., x_n\}$, we denote $L = \{l_1, ..., l_n\}$ as its corresponding label sequence. Here, we assume only K training examples (K-shot) for each of N classes (N-way) in the training set $\mathcal{D}_{train}$. Our goal is to develop a model that learns from these few-shot training samples then makes predictions on the test set $\mathcal{D}_{test}$. Different from previous works that assume a resource-rich source domain and available support sets during testing, we follow the few-shot setting of Gao et al. (2021), which supposes that only a small number of examples are used for fine-tuning. Such setting makes minimal assumptions about available resources and is more practical.

4 Method

We propose a prompt-based method with Parabiotic Decoder (ParaBART) to improve the ability of entity boundary detection. We first give an overview of ParaBART, which is illustrated in Figure 3. ParaBART consists of a BART encoder and the Parabiotic Decoder we propose. Parabiotic Decoder, includes two BART decoders and a conjoint module. The two decoders solve the tasks of entity boundary detection and entity type classification respectively and share the well-learned knowledge between the two decoders through the conjoint module, which replaces unimportant tokens' embeddings in one decoder with the average embedding of all tokens in the other decoder. Additionally, we design a boundary expansion strategy to enhance the ability of entity type classification. Next, we describe main components of ParaBART.

4.1 Parabiotic Decoder

Parabiotic Decoder includes two BART decoders and a conjoint module. One decoder is responsible for entity boundary detection, called EBD decoder. The other decoder is for entity type classification, named ETC decoder. The conjoint module is used for sharing the well-learned knowledge between the two decoders.

Firstly, we manually create the templates for two decoders respectively. For ETC decoder, the template has one slot for `candidate_span` and the other slot for label words. We set a one to one mapping function to transfer the label set $\mathbf{L} = \{l_1, ..., l_{|\mathbf{L}|}\}$ (e.g., l_k =*"LOC"*) to a natural word set $\mathbf{Y} = \{y_1, ..., y_{|\mathbf{L}|}\}$ (e.g. y_k =*"location"*), and use words to define templates $\mathbf{T}_{ETC}^{y_k}$ (e.g. *<candidate_span> belongs to location category.*) In this way,

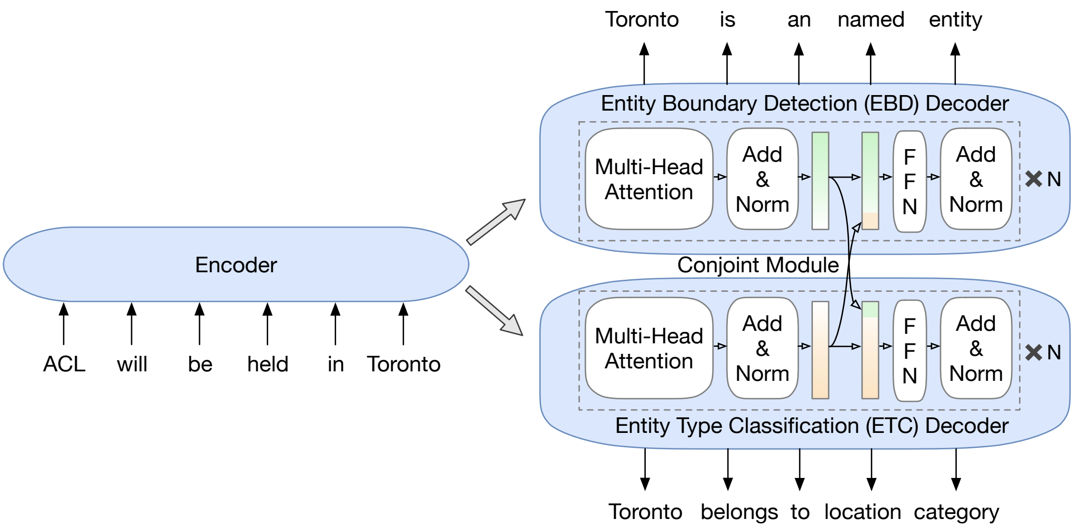


Figure 3: The overall architecture of ParaBART.

we can obtain a list of templates $\mathbf{T}_{ETC} = [\mathbf{T}_{ETC}^{y_1}, ..., \mathbf{T}_{ETC}^{y_{|\mathbf{L}|}}]$. For EBD decoder, we create an entity template $\mathbf{T}_{EBD}^{+}$ for all of the named entity spans (e.g., `<candidate_span>` *is a named entity.*) and a non-entity template $\mathbf{T}_{EBD}^{-}$ for non-entity spans (e.g., `<candidate_span>` *is not a named entity.*). We can obtain a list of template $\mathbf{T}_{EBD} = [\mathbf{T}_{EBD}^{+}, \mathbf{T}_{EBD}^{-}]$. The training procedure is detailed in Section 4.3.

After that, we propose a conjoint module to share the well-learned knowledge between the two decoders. Inspired by Caron et al. (2021); Liang et al. (2022), we select tokens in one decoder with a proportion of the smallest attention scores to *cls*[4] to filter the less important tokens. After that, the selected tokens' embeddings are replaced with the average embedding of all tokens in the other decoder. Further, we employ residual connection (He et al., 2016) to reduce the information loss caused by the replacement. The overall procedure of the conjoint module at layer μ is summarized in Algorithm 1. In particular, inspired by Pu et al. (2022), we only add the module on the shallow layers (i.e. $\mu \in [1, 2, 3]$) of the decoders, to share the general perceptions.

Algorithm 1 Conjoint Procedure

Input: The embedding matrices $\mathbf{E}_1$, $\mathbf{E}_2$ and the cls attention vectors $\mathbf{A}_1$, $\mathbf{A}_2$ of two decoders; conjoint proportion θ;
$\mathbf{E}_1, \mathbf{E}_2 \in \mathbb{R}^{seq\_len \times emb\_dim}$
$\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{R}^{1 \times seq\_len}$

Obtain the positions P_1, P_2, whose attention scores are smaller than the θ-quantile of $\mathbf{A}_1$, $\mathbf{A}_2$, respectively;
$n \leftarrow$ the size of $\mathbf{A}_1$;
for $p \in P_1$ **do**
    $\mathbf{E}_1[p] \leftarrow \mathbf{E}_1[p] + \frac{1}{n}\sum_{i=1}^{n} \mathbf{E}_2[i]$;
end for
for $p \in P_2$ **do**
    $\mathbf{E}_2[p] \leftarrow \mathbf{E}_2[p] + \frac{1}{n}\sum_{i=1}^{n} \mathbf{E}_1[i]$;
end for

4.2 Boundary Expansion

Zhu and Li (2022) hold that the annotated spans are scarce and assigned with full probability to be an entity, whereas all other spans are assigned with zero probability. This creates noticeable sharpness between the classification targets of adjacent spans, and may thus plague the trainability of neural networks. Inspired by label smoothing (Szegedy et al., 2016; Müller et al., 2019), we design a boundary expansion strategy to solve the problem. Specifically, given the sentence *"ACL will be held in Toronto"*, where *"Toronto"* has a gold label *"location"*. For a span (e.g. *"in Toronto"*) that includes an entity span and its previous or subsequent token, we change its label from `O`-class to the entity type corresponding to the entity span. It is noted that we only implement entity boundary expansion for ETC decoder. After boundary expansion, $\mathbf{T}_{ETC} = \mathbf{T}_{ETC}^{y_i}$ (e.g. *"<u>in Toronto</u> belongs to location category"*) and $\mathbf{T}_{EBD} = \mathbf{T}_{EBD}^{-}$

[4] The special token in Transformer that can be used to derive the sentence-level embedding.

(e.g. *"in Toronto is not a named entity"*). Since ETC decoder is only responsible for predicting the category of entities, the expansion of entity boundaries to ETC decoder does not affect the model's ability to detect entity boundaries. Meanwhile, the strategy effectively improves the performance of ETC decoder to classify entity types.

4.3 Training

Gold entities are used to create template during training. Suppose that the entity type of a text span x_i is y_k. We fill the text span x_i and the entity type y_k into $\mathbf{T}_{ETC}$ and $\mathbf{T}_{EBD}$ to create two target sentences $\mathbf{T}_{ETC}^{y_k,x_i}$ and $\mathbf{T}_{EBD}^{+}$. If x_j is a non-entity span, we only need the target sentence $\mathbf{T}_{EBD}^{-}$. We use all gold entities in the training set to construct positive samples $(\mathbf{X}, \mathbf{T}_{ETC}, \mathbf{T}_{EBD}^{+})$ and create negative samples $(\mathbf{X}, \mathbf{T}_{EBD}^{-})$ by randomly sampling non-entity text spans. Through the proposed boundary expansion, we can create some expanded samples $(\mathbf{X}, \mathbf{T}_{ETC}, \mathbf{T}_{EBD}^{-})$. The ratio of the number of positive, negative and expanded samples is 1:1:1.

Given a positive sample $(\mathbf{X}, \mathbf{T}_{ETC}, \mathbf{T}_{EBD}^{+})$ or an expanded sample $(\mathbf{X}, \mathbf{T}_{ETC}, \mathbf{T}_{EBD}^{-})$, we feed the input $\mathbf{X}$ to the encoder of the BART, and then we obtain hidden representations of the sentence:

$$\mathbf{h}^{enc} = \texttt{Encoder}(X) \tag{1}$$

For each decoder[5] , at the c-th step, $\mathbf{h}^{enc}$ and previous output tokens $t_{1:c-1}$ are then as inputs, yielding a representation using attention (Vaswani et al., 2017):

$$\mathbf{h}_c^{dec} = \texttt{Decoder}(\mathbf{h}^{enc}, t_{1:c-1}) \tag{2}$$

The conditional probability of the word t_c is defined as:

$$p(t_c|t_{1:c-1}, \mathbf{X}) = \texttt{Softmax}(\mathbf{h}_c^{dec}\mathbf{W}_{lm} + \mathbf{b}_{lm}) \tag{3}$$

where $\mathbf{W}_{lm} \in \mathbb{R}^{d_h \times |\mathcal{V}|}$ and $\mathbf{b}_{lm} \in \mathcal{R}^{|\mathcal{V}|}$. $|\mathcal{V}|$ represents the vocab size of pre-trained BART. The cross-entropy between each decoder's output and the corresponding target template is used as the loss function:

$$\mathcal{L} = -\sum_{c=1}^{m} \log p(t_c|t_{1:c-1}, \mathbf{X}) \tag{4}$$

The ETC and EBD decoders get $\mathcal{L}_{ETC}$ and $\mathcal{L}_{EBD}$ respectively by Equation 4. $\mathcal{L}_{ETC}$ and $\mathcal{L}_{EBD}$ update their corresponding decoder and jointly update the encoder.

[5]The decoder represents BART decoder with our proposed conjoint module.

Given a negative sample pair $(\mathbf{X}, \mathbf{T}_{EBD}^{-})$, we only feed the encoder output $\mathbf{h}^{enc}$ to the EBD decoder and obtain $\mathcal{L}_{EBD}$ to update the encoder and EBD decoder.

4.4 Inference

We first enumerate all possible spans in the sentence $\{x_1, ..., x_n\}$ and fill them in the prepared templates. Following Cui et al. (2021), we restrict the number of n-grams for a span from one to eight for efficiency. Then, we use the fine-tuned pre-trained generative language model to assign a score for each template, formulated as

$$f(\mathbf{T}) = -\sum_{c=1}^{m} \log p(t_c|t_{1:c-1}, \mathbf{X}) \tag{5}$$

We first calculate scores $f(\mathbf{T}_{EBD}^{+})$ and $f(\mathbf{T}_{EBD}^{-})$ for each candidate spans through the EBD decoder. If $f(\mathbf{T}_{EBD}^{-}) > f(\mathbf{T}_{EBD}^{+})$, we predict the text span is not an entity. Otherwise, we calculate scores $f(\mathbf{T}_{ETC}^{y_k})$ for each entity type through the ETC decoder. Then we assign the entity type with the largest score to the text span.

5 Experiments

We compare our proposed method with several baselines on two classic few-shot scenarios: (1) few-shot setting, where all training data are only a few labeled data. (2) resource-rich setting, where some additional data-rich source domains are available for pretraining.

Implements Following Cui et al. (2021), we use the pre-trained `bart-large` model for all the datasets. Besides, we set the learning rate as $4e-5$ and batch size as 2 for few-shot training. Following Hou et al. (2022), we fine-tune the model only on few-shot training set for 2 epochs (4 on 10/20 shots settings) with the AdamW optimizer and linear decaying scheduler for all our experiments. We use the templates *"<candidate_span> is a named entity"* and *"<candidate_span> is not a named entity"* for EBD decoder and *"<candidate_span> belongs to <entity_type> category"* for ETC decoder. The impact of different choice of templates are detailed in Appendix B. Since there is no development set, all hyperparameters are roughly set based on experience without tuning. All baseline results except QaNER (Liu et al., 2022) are recorded in Hou et al. (2022). For QaNER, we

Method	MIT-Restaurant						
	10-shot	**20-shot**	**50-shot**	**100-shot**	**200-shot**	**500-shot**	**Average**
ExampleNER + PT	27.6	29.5	31.2	33.7	34.5	34.6	31.9
Multi-Proto + PT	46.1	48.2	49.6	50.0	50.1	-	-
Sequence Labeling BART + PT	8.8	11.1	42.7	45.3	47.8	58.2	35.7
Sequence Labeling BERT + PT	27.2	40.9	56.3	57.4	58.6	75.3	52.6
Template-based BART + PT	53.1	60.3	64.1	67.3	72.2	75.7	65.5
Sequence Labeling BERT	21.8	39.4	52.7	53.5	57.4	61.3	47.7
Template-based BART	46.0	57.1	58.7	60.1	62.8	65.0	58.3
QaNER	55.3	63.9	67.1	69.8	71.3	73.2	66.8
Inverse Prompt	52.1	61.5	66.8	71.0	74.0	76.4	67.0
ParaBART (ours)	**59.71**	**67.45**	**71.22**	**74.58**	**76.14**	**78.94**	**71.34**

Method	MIT-Movie-Hard						
	10-shot	**20-shot**	**50-shot**	**100-shot**	**200-shot**	**500-shot**	**Average**
ExampleNER + PT	40.1	39.5	40.2	40.0	40.0	39.5	39.9
Multi-Proto + PT	36.4	36.8	38.0	38.2	35.4	38.3	37.2
Sequence Labeling BART + PT	13.6	30.4	47.8	49.1	55.8	66.9	43.9
Sequence Labeling BERT + PT	28.3	45.2	50.0	52.4	60.7	76.8	52.2
Template-based BART + PT	42.4	54.2	59.6	65.3	69.6	80.3	61.9
Sequence Labeling BERT	25.2	42.2	49.6	50.7	59.3	74.4	50.2
Template-based BART	37.3	48.5	52.2	56.3	62.0	74.9	55.2
QaNER	56.5	62.3	66.1	68.7	70.2	72.4	66.0
Inverse Prompt	53.3	60.2	66.1	69.6	72.5	74.8	66.1
ParaBART (ours)	**61.34**	**64.79**	**70.33**	**72.81**	**74.58**	**76.17**	**70.00**

Method	MIT-Movie						
	10-shot	**20-shot**	**50-shot**	**100-shot**	**200-shot**	**500-shot**	**Average**
Sequence Labeling BERT	50.6	59.3	71.3	-	-	-	-
NNShot	50.5	59.0	71.2	-	-	-	-
StructShot	53.2	61.4	72.1	-	-	-	-
Template-based BART	49.3	59.1	65.1	-	-	-	-
EntLM	57.3	62.4	71.9	-	-	-	-
QaNER	62.5	67.0	71.1	75.8	78.3	81.2	72.7
Inverse Prompt	59.7	70.1	77.6	80.6	82.6	84.5	75.9
ParaBART (ours)	**70.34**	**75.28**	**81.91**	**83.52**	**84.35**	**86.17**	**80.26**

Table 1: F1 scores (%) of 10, 20, 50, 100, 200, 500-shot problems over MIT-Restaurant, MIT-Movie-Hard and MIT-Movie datasets. `+PT` denotes the model is pre-trained on additional datasets. We highlight the best results in bold.

use the original codes released by their authors and keep the experimental setup consistent with other baselines. We run all the experiments on a single NVIDIA v100 GPU.

5.1 Few-Shot Setting

Datasets Following Hou et al. (2022), we conduct experiments on three few-shot datasets with only in-domain data: MIT-Restaurant Review (Liu et al., 2013), MIT-Movie Review (Liu et al., 2013) and MIT-Movie-Hard Review[6]. We conduct experiments with $K \in \{10, 20, 50, 100, 200, 500\}$ shots settings to fully evaluate the performance of our method in all three datasets. To overcome the randomness associated with training set selection, we sample 10 different training sets for each K-shot setting and report averaged results. All baselines are trained and tested with the same data.

[6]MIT-Movie Review has two datasets: a simple one and a complex one. We denote the simple one as MIT-Movie and combine both as MIT-Movie-Hard.

Baselines In our experiments, we compare with some competitive baselines which can be grouped into three categories: (1) *conventional sequence labeling methods*: ExampleNER (Ziyadi et al., 2020), Sequence Labeling BERT (Devlin et al., 2018) and Sequence Labeling BART (Lewis et al., 2020); (2) *metric-based methods*: Multi-Proto (Huang et al., 2020), NNShot and StructShot (Yang and Katiyar,

Method	5-shot SNIPS							
	We	Mu	Pl	Bo	Se	Re	Cr	Average
Bi-LSTM	25.44	39.69	45.36	73.58	55.03	40.30	40.49	45.70
SimBERT	53.46	54.13	42.81	75.54	57.10	55.30	32.38	52.96
TransferBERT	56.01	43.85	50.65	14.19	23.89	36.99	14.29	34.27
MN	38.80	37.98	51.97	70.61	37.24	34.29	72.34	49.03
WPZ+BERT	69.06	57.97	44.44	71.97	74.62	51.01	69.22	62.61
TapNet+CDT	67.83	68.72	73.74	86.94	72.12	69.19	66.54	72.15
L-WPZ+CDT	**78.23**	62.36	59.74	76.19	83.66	69.69	71.51	71.62
L-TapNet+CDT	69.58	64.09	74.93	85.37	83.76	69.89	73.80	74.49
Inverse Prompt	70.63	71.97	78.73	87.34	81.95	72.07	74.44	76.73
ConVEx*	71.50	**77.60**	79.00	84.50	84.00	73.80	67.40	76.80
ParaBART (ours)	72.19	74.58	**80.41**	**89.58**	**84.13**	**75.62**	**76.95**	**79.07**

Table 2: F1 scores (%) on 5-shot SNIPS dataset. We highlight the best results in bold.

2020); (3) *prompt-based methods*: Template-based BART (Cui et al., 2021), EntLM (Ma et al., 2021), QaNER (Liu et al., 2022) and Inverse Prompt (Hou et al., 2022). Among them, Template-based BART is a prompt-based method that query BART-(Lewis et al., 2020) every possible span in a sentence if it belongs to a certain entity type. QaNER proposes a refined strategy for converting NER problem into the Question Answering (QA) formulation and generates templates for QA models. Inverse Prompt introduces an inverse paradigm for prompting and an iterative prediction strategy to improve the efficiency of prompt-based methods. For more details of other baselines, see Appendix A.1.

Results The results of few-shot settings on MIT-Restaurant, MIT-Movie-Hard and MIT-Movie datasets are shown in Table 1. From the table, ParaBART consistently outperforms all the baselines by a large margin. For example, compared with Inverse Prompt, ParaBART achieves 7.6% improvements in 10-shot setting on MIT-Restaurant dataset. When compared against Template-based BART, ParaBART leads by 14.8% in the average F1 score on MIT-Movie-Hard dataset, which clearly demonstrates that our model is very effective in improving BART-based model. All these results show that ParaBART can leverage information from limited labeled data more efficiently.

5.2 Resource-Rich Setting

Datasets We also evaluate the ability of transferring from data-rich source domains to unseen few-shot domains and conduct experiments on SNIPS (Coucke et al., 2018) dataset. We use 5-shot SNIPS datasets provided by Hou et al. (2022). The few-shot SNIPS dataset consists of 7 domains with different label sets: GetWeather (We), Music (Mu), PlayList (Pl), RateBook (Bo), SearchScreenEvent (Se), BookRestaurant (Re), and SearchCreativeWork (Cr). Each domain contains 100 few-shot episodes, and each episode consists of a support set and a query.

Baselines We provide competitive strong baselines including: (1) *traditional finetune-based methods*: Bi-LSTM (Schuster and Paliwal, 1997), SimBERT (Su, 2020), TransferBERT and ConVEx (Henderson and Vulić, 2020); (2) *few-shot learning methods*: Matching Network (MN) (Vinyals et al., 2016), WPZ (Fritzler et al., 2019b), TapNet+CDT, L-TapNet+CDT, L-WPZ+CDT (Hou et al., 2020) and Inverse Prompt (Hou et al., 2022). TapNet+CDT, L-TapNet+CDT and L-WPZ+CDT are metric-based few-shot learning methods designed for slot tagging, which introduces a CRF-based framework to consider the relation between different slots. ConVEx is a finetuning-based method that models slot tagging as a cloze task. The method pre-trained on Reddit data and fine-tuned on few-shot slot tagging data. It is noted that the Reddit data is not used by our method and other baselines during the experiments. For more details of other baselines, see Appendix A.2.

Results The results of cross-domain settings on 5-shot SNIPS dataset are shown in Table 2. From the table, we see that our method outperforms all the baselines on the average F1 score including ConVEx which uses extra Reddit data in the cross-domain 5-shot setting. Compared with Inverse Prompt, ParaBART achieves 2.34% improvements on the average F1 score. All these results clearly show the generalizability of our model on cross-domain few-shot NER task.

5.3 Ablation Study

	Method	MIT-R	MIT-MM	MIT-M
10-shot	ParaBART	**59.71**	**61.34**	**70.34**
	w/o CM	57.32	60.18	68.94
	w/o BE	55.47	57.32	68.17
20-shot	ParaBART	**67.45**	**64.79**	**75.28**
	w/o CM	65.33	61.87	73.19
	w/o BE	62.97	60.13	73.65
50-shot	ParaBART	**71.22**	**70.33**	**81.91**
	w/o CM	69.01	68.05	79.23
	w/o BE	68.39	66.58	79.88
100-shot	ParaBART	**74.58**	**72.81**	**83.52**
	w/o CM	72.78	69.32	81.01
	w/o BE	72.11	69.98	81.14
200-shot	ParaBART	**76.14**	**74.58**	**84.35**
	w/o CM	74.20	71.19	82.87
	w/o BE	74.36	72.32	82.91
500-shot	ParaBART	**78.94**	**76.17**	**86.17**
	w/o CM	76.59	74.82	84.18
	w/o BE	77.54	74.77	85.12

Table 3: Ablation study: F1 scores (%) of 10, 20, 50, 100, 200, 500-shot problems over MIT-Restaurant (MIT-R), MIT-Movie-Hard (MIT-MM) and MIT-Movie (MIT-M) datasets. **w/o CM** denotes removing conjoint module and **w/o BE** denotes removing boundary expansion.

We conduct an ablation study to understand the characteristics of the main components of ParaBART. As shown in Table 3, the conjoint module brings consistent improvement across all the datasets. This shows that the the conjoint module can effectively improve the model performance. When removing boundary expansion, ParaBART has a significant decline in all the datasets, especially in low-resource settings. For example, ParaBART drops 4.24% in 10-shot setting on MIT-Restaurant dataset, which demonstrates that our proposed boundary expansion strategy is highly effective in few-shot settings.

5.4 Analysis

To verify the ability of our model to detect entity boundaries, we conduct an experiment following the experimental setup of the preliminary experiment in the Section 1. From the Figure 4, we can

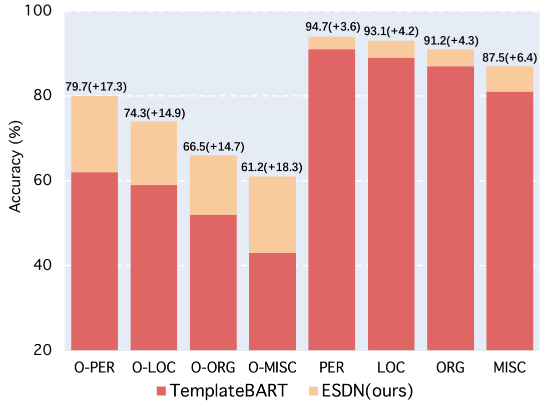


Figure 4: Results of the preliminary experiment introduced in Section 1. Our model outperforms TemplateBART (Cui et al., 2021) by a large margin, especially the prediction of `O-entity` spans, which illustrates the superiority of our model in entity boundary detection.

see that our model achieves a significant improvement (about 16.3% on average) on the accuracy for `O-entity` spans, which clearly demonstrates that our model has a huge advantage in entity boundary detection. Moreover, when entity boundaries are known, the accuracy of our model on entity type classification also increases by 4.6% on average. All the results show that ParaBART can perform reasonably well.

6 Conclusion

In this paper, we first conducted a preliminary experiment and found that what really affects prompt-based NER models is the ability to detect entity boundaries. Based on the observation, we proposed ParaBART to improve the performance of prompt-based methods on entity boundary detection. ParaBART consists of a BART encoder and the Parabiotic Decoder we proposed. Parabiotic Decoder includes two BART decoders and a conjoint module. The two decoders are responsible for entity boundary detection and entity type classification respectively and share the general knowledge through the conjoint module, which replaces unimportant tokens' embeddings in one decoder with the average embedding of all tokens in the other decoder. Moreover, we design a novel boundary expansion strategy to enhance the ability of entity type classification. Experimental results show that ParaBART can achieve significant performance gains over other state-of-the-art methods.

Ethics Statement

The proposed method has no obvious potential risks. All the scientific artifacts used/created are properly cited/licensed, and the usage is consistent with their intended use. Also, we open up our codes and hyperparameters to facilitate future reproduction without repeated energy cost.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NIPS*.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *CVPR*, pages 9650–9660.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner. *arXiv preprint arXiv:2109.00720*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Ganqu Cui, Shengding Hu, Ning Ding, Longtao Huang, and Zhiyuan Liu. 2022. Prototypical verbalizer for prompt-based few-shot tuning. *arXiv preprint arXiv:2203.09770*.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using bart. In *Findings of ACL*, pages 1835–1845.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca J Passonneau, and Rui Zhang. 2021. Container: Few-shot named entity recognition via contrastive learning. *arXiv preprint arXiv:2109.07589*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019a. Few-shot classification in named entity recognition task. In *SAC*, pages 993–1000.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019b. Few-shot classification in named entity recognition task. In *SAC*, pages 993–1000.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL*, pages 3816–3830.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. PTR: prompt tuning with rules for text classification. *CoRR*, abs/2105.11259.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Matthew Henderson and Ivan Vulić. 2020. Convex: Data-efficient and few-shot slot labeling. *arXiv preprint arXiv:2010.11791*.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *ACL*, pages 1381–1393.

Yutai Hou, Cheng Chen, Xianzhen Luo, Bohan Li, and Wanxiang Che. 2022. Inverse is better! fast and accurate prompt for few-shot slot tagging. In *Findings of ACL*, pages 637–647.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2020. Few-shot named entity recognition: A comprehensive study. *arXiv preprint arXiv:2012.14978*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880.

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*, 34(1):50–70.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*, pages 4582–4597.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. 2022. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*.

Andy T Liu, Wei Xiao, Henghui Zhu, Dejiao Zhang, Shang-Wen Li, and Andrew Arnold. 2022. Qaner: Prompting question answering models for few-shot named entity recognition. *arXiv preprint arXiv:2203.01543*.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and Jim Glass. 2013. Query understanding enhanced by hierarchical parsing structures. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 72–77.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Tingting Ma, Huiqiang Jiang, Qianhui Wu, Tiejun Zhao, and Chin-Yew Lin. 2022. Decomposed meta-learning for few-shot named entity recognition. *arXiv preprint arXiv:2204.05751*.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *NIPS*, 32.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Mengyang Pu, Yaping Huang, Yuming Liu, Qingji Guan, and Haibin Ling. 2022. Edter: Edge detection with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1402–1412.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *EACL*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *NAACL*, pages 2339–2352.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*, pages 4222–4235.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *NIPS*, pages 4077–4087.

Jianlin Su. 2020. Simbert: Integrating retrieval and generation into bert. Technical report.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition. *arXiv preprint arXiv:2106.15167*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NIPS*, 30.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. *arXiv preprint arXiv:1606.04080*.

Peiyi Wang, Runxin Xu, Tianyu Liu, Qingyu Zhou, Yunbo Cao, Baobao Chang, and Zhifang Sui. 2021. An enhanced span-based decomposition method for few-shot sequence labeling. *CoRR*, abs/2109.13023.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *ACL*, pages 5363–5369.

Vikas Yadav and Steven Bethard. 2019. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *EMNLP*, pages 6365–6375.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NIPS*, pages 5754–5764.

Enwei Zhu and Jinpeng Li. 2022. Boundary smoothing for named entity recognition. In *ACL 2022*, pages 7096–7108.

Morteza Ziyadi, Yuting Sun, Abhishek Goswami, Jade Huang, and Weizhu Chen. 2020. Example-based named entity recognition. *arXiv preprint arXiv:2008.10570*.

A Baselines

In our experiments, we compare with competitive baselines including both conventional methods and recent prompt-based methods.

A.1 Few-Shot Setting

- **ExampleNER** (Ziyadi et al., 2020) uses large open-domain NER datasets to train an entity-agnostic model to further capture the correlation between support examples and a query. Meanwhile, ExampleNER applies a sentence-level attention to choose the most related examples as support examples to identify new entities.

- **Multi-Proto** (Huang et al., 2020) proposes multiple prototypes for each entity type and pre-trained the model with the task of randomly masked token prediction on massive corpora.

- **Sequence Labeling BERT** (Devlin et al., 2018) can be seen as a BERT-based sequence labeling baseline which fine-tunes the BERT model with a token-level linear classifier.

- **Template-based BART** (Cui et al., 2021) is a prompt-based method that query BART-based LM (Lewis et al., 2020) every possible span in sentence if it belongs to a certain category.

- **NNShot and StructShot** (Yang and Katiyar, 2020) are two metric-based few-shot learning approaches for slot tagging and NER. NNShot is an instance-level nearest neighbor classifier for few-shot prediction, and StructShot promotes NNShot with a Viterbi algorithm during decoding.

- **EntLM** (Ma et al., 2021) is a prompt-based method that leverage substitution between words of the same type to achieve one pass prediction.

- **QaNER** (Liu et al., 2022) proposes a refined strategy for converting NER problem into the Question Answering (QA) formulation and generates templates for QA models.

- **Inverse Prompt** (Hou et al., 2022) introduces an inverse paradigm for prompting and an iterative prediction strategy to improve the efficiency of prompt-based methods.

A.2 Resource-Rich Setting

- **Bi-LSTM** (Schuster and Paliwal, 1997) uses GLoVe (Pennington et al., 2014) embedding for slot tagging and is trained on the support sets.

- **SimBERT** (Su, 2020) is a metric-based method using cosine similarity of BERT-based embedding to label tokens with the most similar token's label.

- **Matching Network (MN)** (Vinyals et al., 2016) is a few-shot sequence labeling model based on the matching network and uses BERT embedding.

- **TransferBERT** is a domain transfer-based conventional NER model using BERT, which is first pre-trained on source domains and then fine-tuned on the target domain support set.

- **WPZ** (Fritzler et al., 2019b) is a metric-based few-shot slot tagging method similar to MN, but is based on the prototypical network (Snell et al., 2017).

- **TapNet+CDT, L-TapNet+CDT, L-WPZ+CDT** (Hou et al., 2020) are metric-based few-shot learning methods designed for slot tagging, which introduces a CRF-based framework to consider the relation between different slots.

- **ConVEx** (Henderson and Vulić, 2020) is a finetuning-based method that models slot tagging as a cloze task and is first pre-trained on Reddit data then fine-tuned on few-shot slot tagging data. Note that the Reddit data is not used by our method and other baselines during the experiments.

B Template Influence

There can be different templates for expressing the same meaning. For instance, *“<candidate_span> is a* person *entity”* can also be expressed by *“<candidate_span> belongs to the* person *category”*. We investigate the impact of manual templates using MIT-Movie dataset on 10-shot setting. Table 4 shows the performance impact of different choice of templates. We observe: (1) When $\mathbf{T}_{EBD}$ is fixed, Different choice of $\mathbf{T}_{ETC}$ has little effect on the performance of the model. (2) When $\mathbf{T}_{ETC}$ is fixed, Different choice of $\mathbf{T}_{ETC}$

$\mathbf{T}_{EBD}$	$\mathbf{T}_{ETC}$	F1(%)
<candidate_span> is a named entity <candidate_span> is not a named entity	<candidate_span> is a <entity_type> entity	69.71
	<candidate_span> belongs to <entity_type> category	70.34
	The entity type of <candidate_span> is <entity_type>	72.11
	<candidate_span> should be tagged as <entity_type>	70.89
<candidate_span> belongs to named entity <candidate_span> belongs to none entity	<candidate_span> is a <entity_type> entity	66.11
	<candidate_span> belongs to <entity_type> category	62.32
	The entity type of <candidate_span> is <entity_type>	64.51
	<candidate_span> should be tagged as <entity_type>	62.29

Table 4: The results of using different templates in 10-shot setting on MIT-Movie dataset.

has a great impact on the model. For instance, when $\mathbf{T}_{ETC}$ is *"<candidate_span> belongs to <entity_type> category"*, the two $\mathbf{T}_{EBD}$ give 70.34% and 62.32% F1 score respectively, which indicates the templates for entity boundary detection is a key factor that influences the final performance. Since we assume that there is no development set, we randomly choose templates in our main experiments.