A Cueing Strategy with Prompt Tuning for Relation Extraction

Anonymous ACL submission

Abstract

Prompt tuning shows great potential to support relation extraction because it is effective to take full use of rich knowledge in pretrained language models (PLMs). However, current prompt tuning models are directly implemented on a raw input. It is weak to encode semantic dependencies of a relation instance. In this paper, we designed a cueing strategy which implants task specific cues into the input. It enables PLMs to learn task specific contextual features and semantic dependencies in a relation instance. Experiments on ReTACRED corpus and ACE 2005 corpus show state-of-the-art performance in terms of F1-score.

1 Introduction

Relation extraction (RE) identifies predefined semantic relationships between two named entities in a sentence. Because a sentence usually contains several named entities, which share the same contextual features in a sentence and relation types are asymmetric, it is important to learn semantic dependencies relevant to considered entities. In deep neural networks, many techniques have been developed to do so, for example, position embedding (Zeng et al., 2015), multi-channel (Chen et al., 2020), neuralized feature engineer (Chen et al., 2021) and entity indicators (Qin et al., 2021; Zhou and Chen, 2021). These models are common in that entities relevant features (e.g., entity positions or types) are encoded into a task specific representation, then fed into a deep architecture for classification. They predict confidence scores for every relation instance.

For learning better representations, PLMs like ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018) are widely adopted for embedding tokens into distributed representations. They have achieved great success in relation extraction (Torfi et al., 2020). However, in traditional type classification models (Soares et al., 2019; Li and Tian, 2020; Zhao et al., 2021; Cohen et al., 2020), PLMs are mainly used to support token embedding. The classification only depends on a single representation of the whole input, which undoubtedly results in a serious semantic loss. Furthermore, the process to initialize PLMs is implemented as a masked token prediction task (Devlin et al., 2018). There is a gap between pre-training objectives and fine tuning objectives, which weakens the effectiveness of PLMs.

In prompt tuning, prompts are defined as templates with slots that take values from a verbalized type token set. These prompts are concatenated with an input, then fed into PLMs to predict masked slots, the same as a cloze-style schema (Schick and Schütze, 2020). Because prompt tuning can gap between pre-training objectives and fine tuning objectives, it is effective to take use of knowledge within PLMs. This strategy has been successfully applied in tasks such as text classification and natural language inference (Schick and Schütze, 2020).

In relation extraction, current prompt tuning models are often directly implemented on a raw input concatenated with predefined prompt templates (Han et al., 2021; Shin et al., 2020; Gao et al., 2020; Xiang et al., 2020). Rare work has been done to tune PLMs for learning task specific features about considered entities. Because in relation extraction it is very important to learn semantic dependencies relevant to considered entities. In this paper, we designed a cueing strategy which implants task specific cues into the input. By combining the cueing strategy with prompt tuning, it enables PLMs encoding semantic dependencies between type tokens and contextual words. Furthermore, the predicting process is similar as that of PLMs tuning, it is helpful to bridge the gap between PLMs and relation extraction. Our study shows remarkable improvement. It reveals a meaningful mechanism that is essential for rela-

tion extraction and prompt tuning.

2 Methodology

A relation instance is defined as a 3-tuple $I = \langle r, e_1, e_2 \rangle$, which contains a relation mention r and two named entities e_1 and e_2. Relation mention r is a token sequence $r = [t_1, t_2, \cdots, t_n]$. Entities $e_k = [t_i, \cdots, t_j]$ $(k \in \{1, 2\})$ is a substring of r. Let $\mathbf{Y} = \{y_0, y_1, \cdots, y_M\}$ be a relation type set. It is composed of M positive relation types and one negative relation types y_0. Let $\mathbf{I} = \{I_1, I_2, \cdots\}$ represent a relation instance set. Then, relation extraction is represented as a map between $\mathbf{I}$ and $\mathbf{Y}$, denoted as: $f : \mathbf{I} \rightarrow \mathbf{Y}$, where f is a function which can be a shallow model or a deep neural network.

In a traditional model, a deep architecture (denoted as $\mathcal{N}$) is implemented on the original input r to extract its representation. To encode external knowledge, the network $\mathcal{N}$ can be embedded with a PLM to support token embedding. It is denoted as $\mathcal{N}_\mathcal{M}$. The output of $\mathcal{N}_\mathcal{M}$ is represented as $\mathbf{H} = [H_1, H_2, \cdots, H_n]$. Then, it is fed into a classifier ($\mathcal{C}$) to make a prediction. The process is formalized as:

$$P(\mathbf{Y}|I) = Softmax\Big(\mathcal{C}\Big(\mathcal{N}_\mathcal{M}(r)\Big)\Big) \quad (1)$$

Directly implementing a deep network on r usually cause serious performance degradation, because the network know nothing about the position of considered entities. To handle this problem, task relevant entity cues can be implemented into the input to control the attention of a deep network for learning task specific representation. It is formalized as:

$$\begin{aligned} &Cueing(e_k) = [\langle c_k \rangle, e_k, \langle /c_k \rangle], \\ &Cueing(r) = [\ddot{r}|_{e_k/Cueing(e_k), k=\{1,2\}}]. \end{aligned} \quad (2)$$

where, $\langle c_k \rangle$ and $\langle /c_k \rangle$ are specific tokens representing the start and end boundaries of entity e_k $(k = \{1, 2\})$. They are named as entity cues.

In Equation (2), the first equation concatenates two tokens on both sides of e_k. In the second equation, $e_k/Cueing(e_k)$ denotes to the string replacement operation, where e_k is replaced by $Cueing(e_k)$. Therefore, the function $Cueing(r)$ implant entity cues into both side of the considered entity pair. With this settings, Equation (1) can be revised as:

$$P(\mathbf{Y}|I) = Softmax\Big(\mathcal{C}\Big(\mathcal{N}_\mathcal{M}\big(Cueing(r)\big)\Big)\Big) \quad (3)$$

Entity cues enables the deep network focusing on considered entity pair. Then, the classification is based on a sentence representation relevant to considered entities.

2.1 Prompt Tuning Paradigm

In prompt tuning, class types are verbalized into a token set $\mathbf{V} = \{person, parent, true, \cdots\}$. It is composed of entity types, relation types or category labels (e.g., "true" or "false"). Elements of $\mathbf{V}$ are referred as "type tokens". Then, a prompt is defined as a template with slots can be filled by verbalized type tokens (e.g., "It is a [MASK]"). It is concatenated with a raw input and fed into a deep network for predicting the distribution of type tokens in the position of " [MASK]".

The design of prompt templates heavily depends on the property of a task. At current, it is an art instead of a science. In this paper, we follow the work of Han et al. (Han et al., 2021), where a relation prompt is defined as a template with three slots: "$\mathcal{P}(e_1, e_2)$ = the [MASK]$_1$ e_1 is [MASK]$_2$ to [MASK]$_3$ e_2", where, [MASK] takes values from $\mathbf{V}$. The prompt is concatenated with the input and fed into a deep neural network to learn token representations $\mathbf{H}$. It is represented as:

$$[H_1, \cdots, H_L] = \mathcal{N}_\mathcal{M}\big(Cueing(r) + \mathcal{P}(e_1, e_2)\big) \quad (4)$$

In prompt tuning, instead of outputting a class label based on token representations $[H_1, \cdots, H_L]$, for each slot ([MASK]) in a prompt template, the normalized confidence score that $\mathcal{N}_\mathcal{M}$ assigns a type token $v \in \mathbf{V}$ to [MASK]$_i$ is computed as:

$$\mathcal{S}([MASK] = v|I) = H_v \cdot H_{M_i} \quad (5)$$

where, $H_{M_i} \in \mathbf{H}$ is the representation of a [MASK]$_i$ and H_v is the token type representation of $v \in \mathbf{V}$ in the employed PLMs. Then, given a relation instance I, the distribution of type token v in slot [MASK]$_i$ is computed as:

$$P(v|I) = \frac{\exp\Big(\mathcal{S}\big([\text{MASK}] = v|I\big)\Big)}{\sum_{v' \in \mathbf{V}} \exp\Big(\mathcal{S}\big([\text{MASK}] = v'|I\big)\Big)} \quad (6)$$

In prompt tuning, a right output requires that three slots are correct recognized. Because prompt tuning is effective to use rich knowledge in PLMs, it still shows competitive performance.

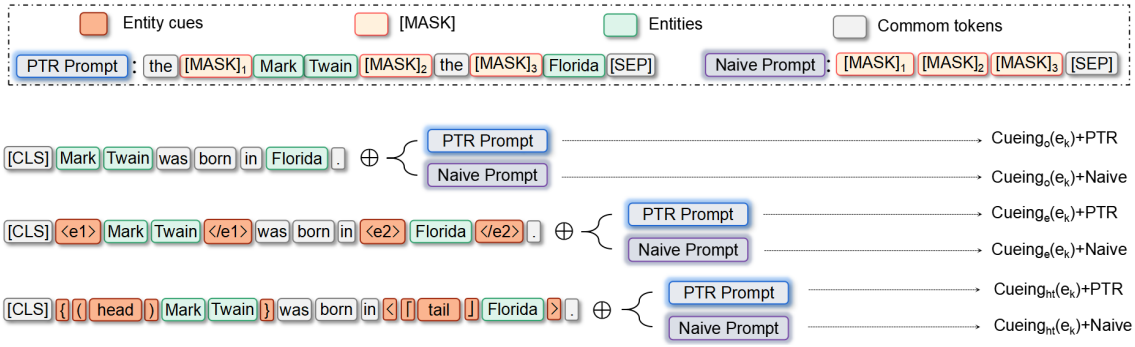

Figure 1: Examples of Cueing Strategies

2.2 Cueing Strategies

In our method, instead of designing new prompt templates, we focus on designing and implanting entity cues for tuning PLMs to support relation extraction. Several cueing strategies have been proposed in this paper. They are listed as follows:

Cueing Types	Demonstration
$Cueing_o(e_k)$	$[e_k]$
$Cueing_e(e_k)$	$[\langle c_k \rangle, e_k, \langle /c_k \rangle]$
$Cueing_{ht}(e_k)$	$[\{(head)\ e_1\}], [\langle \lceil tail \rfloor\ e_2 \rangle]$

Table 1: Cueing Strategies

In Table 1, square brackets is used to indicate that the inner is a token sequence. $Cueing_o$ means that the input relation mention is unchanged. $Cueing_e$ replaces entity e_k ($k \in \{1, 2\}$) with a token sequence "$\langle c_k \rangle, e_k, \langle /c_k \rangle$". Note that all pairs of closed braces and parentheses are also used as tokens to indicate the position of named entities. In $Cueing_{ht}(e_k)$, a "head" token and a "tail" token with different braces are used to distinguish different entities. In this strategy, entity types can also be used as the entity cues.

Entity cues are implanted into the input. Then, the revised input is concatenated with prompt templates to tune PLMs for relation extraction. In Figure 1, we give examples to demonstrate the cueing strategy.

In Figure 1, "PTR prompt" is the prompt template proposed by Han et al. (Han et al., 2021), in which a template has three slots. $[\text{MASK}]_1$ and $[\text{MASK}]_3$ can take values in {"person", "country", $\cdots$ }. They denote to the type of named entities. $[\text{MASK}]_2$ takes values in {"was born in", "was located in", $\cdots$ }. It is used to indicate the relation between named entities. In "Naive prompt", three [MASK] are directly used without any contextual words. It is mainly used for comparison.

The cueing strategies listed in Table 1 are concatenated with both "PTR prompt" and "Naive prompt", where $\oplus$ denotes to the concatenating operation. For example, "$Cueing_{ht}(e_k)$+PTR" means that, given a relation instance $\langle r, e_1, e_2 \rangle$, we first replace e_1 and e_2 in r by two string "$\{(head)\ e_1\}$" and "$\langle \lceil tail \rfloor\ e_2 \rangle$". Then, the revised relation mention ($\ddot{r}|_{e_k/Cueing_{ht}(e_k),k=\{1,2\}}$) is concatenated with the PTR prompt. The output is fed into a PLM to predict type tokens in each [MASK]. If a PLM outputs "person", "is parent of", "person", then a "person:parent" relation is identified between e_1 and e_2.

3 Experiments

Our strategy is evaluated on the ReTACRED corpus (Stoica et al., 2021) and the ACE 2005 English corpus[1]. RoBERTa$_{LARGE}$ (Liu et al., 2019) is adopted as our PLMs[2]. The max length for each input is set as 150. The "Adam" is used as optimizer. Dropout rate is set to 0.1 to avoid the overfitting. Epochs, learning rate and batch size are set as 20, 2e-5, 64, respectively. To compare with related work, we follow experiment settings as Han et al. (2021)[3] in the ReTACRED corpus and Qin et al. (2021) in the ACE corpus.

3.1 Comparing with Related Work

In this experiment, our cueing strategy is compared with **spanBERT** (Joshi et al., 2020), **REBEL** (Cabot and Navigli, 2021), **Typed-marker** (Zhou and Chen, 2021), **PTR** (Han et al., 2021), **KnowPrompt** (Xiang et al., 2020), **Dual PN** (Park and Kim, 2020), **BERT-CNN** (Qin et al.,

[1]https://catalog.ldc.upenn.edu/LDC2006T06
[2]https://huggingface.co/roberta-large
[3]https://github.com/thunlp/PTR

2021) and **SSM** (Yanping et al., 2017). We adopted the $Cueing_{ht}(e_k)$ strategy in Table 1. The revised relation mention is concatenated with the PTR prompt. An example is illustrated in Figure 1. Every concatenated string is fed into a PLM to predict type tokens in masked slots. Table 2 gives the performance of our strategy and related work. All performance is reported in F1 score (%).

Corpus	Tuning	Methods	F1
ReTACRED	Fine	spanBERT	85.3
		REBEL	90.4
		Typed-marker	91.1
	Prompt	KnowPrompt	89.8
		PTR	90.9
		Ours	**92.36**
ACE	Fine	Dual PN	80.8
		BERT-CNN	85.7
		SSM	80.75
	Prompt	**Ours**	**88.92**

Table 2: Comparing with related work

Prompt tuning outputs type tokens based contextual features and semantic dependencies of a sentence, it is effective to take full use of rich knowledge in PLMs and bridge the gap between PLMs training and downstream tasks. As Table 2 showing, compared with fine tuning models, prompt tuning achieves competitive performance. However, prompt tuning is not always better than the fine tuning, because fine tuning models also address the gap between PLMs and integrate external knowledge. Therefore, they are also effective to use knowledge in PLMs.

As cueing strategy is integrated during prompt tuning, the model can better utilize the semantic dependency information between entity pairs. The result shows that, implanting entity cues is valuable to support relation extraction. Our model achieves the state-of-the-art performance in both the ReTACRED and ACE corpora. The conclusion reveals the mechanism of prompt tuning. It is significant to support future studies on both relation extraction and prompt tuning.

3.2 Ablation Study

In order to demonstrate the effectiveness of cueing strategies, we combined them with the Naive prompt and PTR prompt to show the influence of cueing strategies on the performance. The result was listed in Table 3.

ID	Cueing+Prompt	ReTACRED	ACE
(1)	$\text{Cueing}_o(e_k)$+Naive	43.37	72.97
	$\text{Cueing}_o(e_k)$+PTR	90.46	86.07
(2)	$\text{Cueing}_e(e_k)$+Naive	90.62	82.30
	$\text{Cueing}_e(e_k)$+PTR	91.12	87.95
(3)	$\text{Cueing}_{ht}(e_k)$+Naive	90.43	88.92
	$\text{Cueing}_{ht}(e_k)$+PTR	**92.36**	**89.44**

Table 3: Performance with Different Cueing Strategies

(1) In $\text{Cueing}_o(e_k)$+Naive, every original input is directly concatenated with a naive prompt. It is mainly conducted as the baseline of prompt tuning for comparison. $\text{Cueing}_o(e_k)$+PTR is the strategy used in Han et al. (2021). Because the PTR prompt contains contextual words, it considerably improves the performance.

(2) In $\text{Cueing}_e(e_k)$+Naive, entity cues are implanted into the input to indicate the position of entities e_i ($i \in \{1, 2\}$). Comparing with related work in Table 2, it already achieved the state of the art performance. The result indicates that entity cues are every powerful in prompt tuning based models. $\text{Cueing}_e(e_k)$+PTR also outperforms its naive version.

(3) In this cueing strategy, different entity cues are used to make a distinction between entities. In stead of specific tags (e.g., $\langle c_k \rangle$ or $\langle /c_k \rangle$), contextual words are used as entity cues (e.g., entity types or "head" and "tail"). In this cueing strategy, both entity cues and prompts contains contextual words. They are effective to encode contextual features and semantic dependencies of a relation instance.

In all experiments, when both entity cues and prompts are used simultaneously, the relation extraction achieves more robust and superior performance. Compared with the traditional entity cues method which can only mark the location of the entity, our cueing strategy packaged the entity order and location together. This setting achieves the highest performance in our experiments.

4 Future Work

In this paper, we proposed a cueing strategy for relation extraction. It achieves the state of the art performance. In our future work, more studies will be conducted to reveal the mechanism of cueing strategy. Furthermore, the cueing strategy can be extended to support other NLP tasks.

References

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. Rebel: Relation extraction by end-to-end language generation. In *Proceedings of EMNLP*, pages 2370–2381.

Yanping Chen, Kai Wang, Weizhe Yang, Yongbin Qing, Ruizhang Huang, and Ping Chen. 2020. A multi-channel deep neural network for relation extraction. *IEEE Access*, 8:13195–13203.

Yanping Chen, Weizhe Yang, Kai Wang, Yongbin Qin, Ruizhang Huang, and Qinghua Zheng. 2021. A neuralized feature engineering method for entity relation extraction. *Neural Networks*, 141:249–260.

Amir DN Cohen, Shachar Rosenman, and Yoav Goldberg. 2020. Relation classification as two-way span-prediction. *arXiv preprint arXiv:2010.04829*.

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*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Cheng Li and Ye Tian. 2020. Downstream model design of pre-trained language model for relation extraction task. *arXiv preprint arXiv:2004.03786*.

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. *arXiv preprint arXiv:1907.11692*.

Seongsik Park and Harksoo Kim. 2020. Dual pointer network for fast extraction of multiple relations in a sentence. *Applied Sciences*, 10(11):3851.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Yongbin Qin, Weizhe Yang, Kai Wang, Ruizhang Huang, Feng Tian, Shaolin Ao, and Yanping Chen. 2021. Entity relation extraction based on entity indicators. *Symmetry*, 13(4):539.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*.

George Stoica, Emmanouil Antonios Platanios, and Barnabás Póczos. 2021. Re-tacred: Addressing shortcomings of the tacred dataset. *Proceedings of AAAI*, 35(15):13843–13850.

Amirsina Torfi, Rouzbeh A Shirvani, Yaser Keneshloo, Nader Tavaf, and Edward A Fox. 2020. Natural language processing advancements by deep learning: A survey. *arXiv preprint arXiv:2003.01200*.

Chen Xiang, Zhang Ningyu, Xie Xin, Deng Shumin, Yao Yunzhi, Tan Chuanqi, Huang Fei, Si Luo, and Chen Huajun. 2020. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. *arXiv preprint arXiv:2104.07650*.

Yanping, Chen, Qinghua, Zheng, Ping, and Chen. 2017. A set space model for feature calculus. *IEEE Intelligent Systems*, 32(5):36–42.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of EMNLP*, pages 1753–1762.

Kang Zhao, Hua Xu, Yue Cheng, Xiaoteng Li, and Kai Gao. 2021. Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. *Knowledge-Based Systems*, 219:106888.

Wenxuan Zhou and Muhao Chen. 2021. An improved baseline for sentence-level relation extraction. *arXiv preprint arXiv:2102.01373*.