Generative Prompt Tuning for Relation Classification

Jiale Han[1], **Shuai Zhao**[1], **Bo Cheng**[1], **Shengkun Ma**[1] and **Wei Lu**[2]
[1]State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications
[2]StatNLP Research Group, Singapore University of Technology and Design
{hanjl,zhaoshuaiby,chengbo,mashengkun}@bupt.edu.cn, luwei@sutd.edu.sg

Abstract

Using prompts to explore the knowledge contained within pre-trained language models for downstream tasks has now become an active topic. Current prompt tuning methods mostly convert the downstream tasks to masked language modeling problems by adding cloze-style phrases and mapping all labels to verbalizations with fixed length, which has proven effective for tasks with simple label spaces. However, when applied to relation classification exhibiting complex label spaces, vanilla prompt tuning methods may struggle with label verbalizations with arbitrary lengths due to rigid prompt restrictions. Inspired by the text infilling task for pre-training generative models that can flexibly predict missing spans, we propose a novel generative prompt tuning method to reformulate relation classification as an infilling problem, which frees our approach from limitations of current prompt based approaches and thus fully exploits rich semantics of entity and relation types. In addition, we design entity-guided decoding and discriminative relation scoring to generate and align relations effectively and efficiently during inference. Extensive experiments under fully supervised settings and low-resource settings demonstrate the effectiveness of our approach.

1 Introduction

Relation classification (RC) is a fundamental task in natural language processing (NLP), aiming to detect the relations between the entities contained in a sentence. With the rise of a series of pre-trained language models (PLMs) (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020; Raffel et al., 2020), fine-tuning PLMs has become a dominating approach to RC (Joshi et al., 2020; Xue et al., 2021; Zhou and Chen, 2021). However, the significant objective gap between pre-training and fine-tuning may hinder the full potential of pre-trained knowledge for such a downstream task.

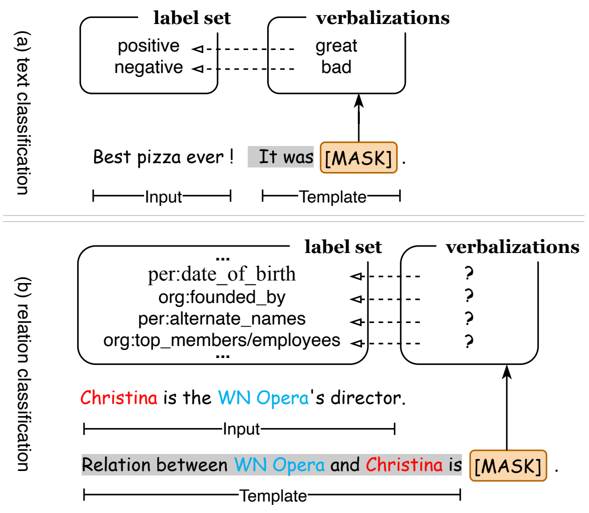


Figure 1: Examples of prompt tuning applied to (a) text classification and (b) relation classification. Existing prompt-based approaches are effective when the label space is simple, but struggle in cases where labels require more complex and elaborate descriptions. We can see from Figure (b) that different classes own label tokens of arbitrary lengths, and it may not always be easy to map them to verbalizations of the same length without losing semantic information.

To this end, prompt tuning (Brown et al., 2020; Schick and Schütze, 2021a,b; Liu et al., 2021a) has been recently proposed and proven effective especially for low-resource scenarios (Gao et al., 2021; Scao and Rush, 2021). The core idea is to convert the objective of downstream tasks to be closer to that of the pre-training tasks, which designs a template to reformulate input examples as cloze-style phrases and a verbalizer to map labels to candidate words. By predicting the mask token, we can determine the label of the input example.

One disadvantage of prompt tuning is the rigid template restriction, in which the number and position of masked tokens are typically fixed. As presented in Figure 1, when the label space is simple, downstream tasks can easily adapt to this paradigm (Hambardzumyan et al., 2021; Lester et al., 2021), which predicts one verbalization token at one masked position. However, when applying prompt tuning to RC with complex label space that conveys rich semantic information, vanilla prompt tuning methods may struggle with handling complex label verbalizations with varying lengths. As an attempt to resolve this issue, Han et al. (2021c) abridge different labels into verbalizations of fixed length, which, however, may lead to loss of important semantic information. Sainz et al. (2021) convert RC to an entailment problem with hand-crafted verbalizations as hypothesis. Such an approach requires expert efforts, making it difficult to adapt to new datasets and tasks.

We argue that the fundamental reason for this limitation is that the existing prompt tuning methods imitate masked language modeling (MLM), which predicts only one token at one masked position. Different from MLM, the text infilling task (Zhu et al., 2019) for pre-training generative models (Lewis et al., 2020; Raffel et al., 2020) appears to be more compatible with RC. The task drops consecutive spans of tokens and learns to predict not only which but also how many tokens are missing from each snippet. Following this paradigm allows the model to generate an arbitrary number of tokens at multiple prediction slots.

This paper proposes a novel *Gen*erative *P*rompt *T*uning method (GenPT), which reformulates RC as a text infilling task to eliminate the rigid prompt restrictions and thus fully exploit the label semantics. Entity type information is further injected thanks to the flexible task format, which is crucial for RC (Zhou and Chen, 2021). Specifically, we construct a multi-slot continuous prompt, in which the template converts input sentences to infilling style phrases by leveraging three sentinel tokens as placeholders and desires to fill in the label verbalizations of head entity type, tail entity type, and relation, respectively. Trainable continuous prompt embeddings are employed to avoid manual prompt engineering. In addition, how to efficiently determine the final class label is a practical problem when applying generative models to discriminative tasks. We design entity-guided decoding and relation scoring strategies to align the generated sequences with the pre-defined set of labels, making the prediction process more effective and efficient.

Extensive experiments are conducted on four widely used relation classification datasets under fully supervised and low-resource settings. Compared to a series of strong discriminative and generative baselines, our method achieves better performance, especially in cases where relations are rarely seen during training, demonstrating the effectiveness of our approach. Our main contributions are summarized as follows:[1]

- We reformulate RC as a text infilling task and propose a novel generative prompt tuning method, which eliminates the rigid prompt restrictions and makes full use of semantic information of entity types and relation labels.
- We design entity-guided decoding and discriminative relation scoring strategies to predict relations effectively and efficiently.
- Experiments on four datasets demonstrate the effectiveness of our model in both fully supervised and low-resource settings.

2 Background

2.1 MLM and Text Infilling

Masked language modeling (Taylor, 1953) is widely adopted as a pre-training task to obtain a bidirectional pre-trained model (Devlin et al., 2019; Liu et al., 2019; Conneau and Lample, 2019). Generally speaking, a masked language model (MLM) randomly masks out some tokens from the input sentences. Each `[MASK]` corresponds to one token. The objective is to predict the masked word based on the rest of the tokens (see Figure 2 (a)). Different from MLM which only predicts one token for one `[MASK]`, the text infilling task for pre-training seq2seq models (Raffel et al., 2020; Lewis et al., 2020) can flexibly recover spans of different lengths. As shown in Figure 2 (b), the text infilling task samples a number of text spans with different lengths from the original sentence. Then each span is replaced with a single sentinel token. The encoder is fed with the corrupted sequence, and the decoder sequentially produces the consecutive tokens of dropped-out spans delimited by sentinel tokens. This task is more flexible and can be more compatible with some complex downstream tasks, but is now significantly overlooked.

2.2 Prompt-Tuning of PLMs

For standard fine-tuning of classification, the input instance x is converted to a token sequence

[1]Our code is available at `https://github.com/hanjiale/GenPT`.

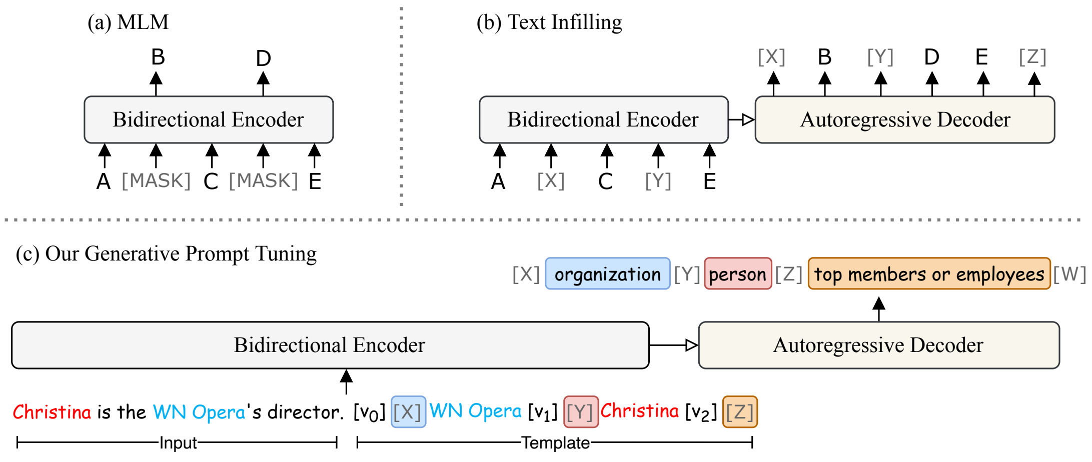


Figure 2: An illustration of (a) MLM pre-training, (b) text infilling pre-training, and (c) our proposed generative prompt tuning approach for RC.

$\widetilde{\boldsymbol{x}}$ = [CLS] $\boldsymbol{x}$ [SEP]. The model predicts the output classes by feeding the input sequence into PLMs and adding a classifier on top of the [CLS] representations, which introduces extra parameters and makes it hard to generalize well, especially for low-resource setting. To this end, prompt-tuning is proposed to make the downstream task consistent with the pre-training task. Current prompt-tuning approaches mainly cast tasks to cloze-style questions to imitate MLM. Formally, a prompt consists of two key components (Schick and Schütze, 2021a), a template and a verbalizer. The template $T(\cdot)$ reformulates the original input $\boldsymbol{x}$ as a cloze-style phrase $T(\boldsymbol{x})$ by adding a set of additional tokens and one [MASK] token. The verbalizer $\phi : \mathcal{R} \to \mathcal{V}$ maps task labels $\mathcal{R}$ to textual tokens $\mathcal{V}$, where $\mathcal{V}$ refers to a set of label words in the vocabulary of a language model $\mathcal{M}$. In this way, a classification task is transformed into an MLM task:

$$\mathrm{P}(r \in \mathcal{R}|\boldsymbol{x}) = \mathrm{P}(\texttt{[MASK]} = \phi(r)|T(\boldsymbol{x}))$$

Most prompt tuning methods include one mask token in the template and map each label to one verbalization token to predict classes. Although effective, it is hard to handle tasks with complex label spaces involving labels of varying lengths.

3 Approach

As presented in Figure 2 (c), this paper considers relation classification as a text infilling style task, which takes the sequence $T(\boldsymbol{x})$ processed by the template as source input and outputs a target sequence $\boldsymbol{y}$ to predict relations. The problem definition is formally given in Section 3.1. We first introduce how to construct entity-oriented prompts in Section 3.2, and then present the model and training objective in Section 3.3. The inference details including entity-guided decoding and relation scoring are discussed in Section 3.4.

3.1 Problem Definition

Formally, given an instance $\boldsymbol{x} = [x_1, x_2, ..., x_{|\boldsymbol{x}|}]$ with head and tail entity mentions $\boldsymbol{e}_h$ and $\boldsymbol{e}_t$ spanning several tokens in the sequence, as well as entity types $\boldsymbol{t}_h$ and $\boldsymbol{t}_t$, relation classification task is required to predict the relation $r \in \mathcal{R}$ between the entites, where $\mathcal{R}$ is the set of possible relations. $\boldsymbol{r}$ represents the corresponding label verbalization. Take a sentence $\boldsymbol{x}$ = "*Christina is the Washington National Opera's director*" with relation r = "*org:top_members/employees*" as an example, $\boldsymbol{e}_h$ and $\boldsymbol{e}_t$ are "*Christina*" and "*Washington National Opera*", and their entity types are "*organization*" and "*person*" respectively. The relation label verbalization $\boldsymbol{r}$ = "*top members or employees*" [2].

3.2 Entity-Oriented Prompt Construction

We design an entity-oriented continuous template $T(\cdot)$ combining entity mentions and type information, which uses a series of learnable continuous tokens (Liu et al., 2021b) as prompts rather than handcrafted token phrases. Specifically, for an input sentence $\boldsymbol{x}$ with two marked entities $\boldsymbol{e}_h$ and $\boldsymbol{e}_t$, instead of utilizing a template with discrete tokens like "$\boldsymbol{x}$. *The relation between* [X] $\boldsymbol{e}_h$ *and* [Y] $\boldsymbol{e}_t$ *is* [Z]." which is hand-crafted and it is hard to find

[2]The relation label verbalization $\boldsymbol{r}$ is derived from label r, which involves removing attribute words "*org:*", discarding symbols of "_", and replacing "/" with "*or*".

the optimal prompt, we leverage a few learnable continuous tokens to serve as prompts that can be optimized by gradient descent.

$$T(\boldsymbol{x}) = \boldsymbol{x}.\ [v_{0:n_0-1}]\ \texttt{[X]}\ \boldsymbol{e}_h\ [v_{n_0:n_1-1}]$$
$$\texttt{[Y]}\ \boldsymbol{e}_t\ [v_{n_1:n2-1}]\ \texttt{[Z]}.$$

where $[v_i] \in \mathbb{R}^d$ refers to the i-th continuous token in the template, and n_0, $n_1 - n_0$, and $n_2 - n_1$ are the lengths of these token phrases. We add three sentinel tokens in the template, where `[X]` and `[Y]` in front of entity mentions denote type information of head and tail entities, and `[Z]` is used to represent relation label tokens. The target sequence then consists of head and tail entity types and label verbalizations, delimited by the sentinel tokens used in the input plus a final sentinel token `[W]`.

$$\boldsymbol{y} = \texttt{[X]}\ \boldsymbol{t}_h\ \texttt{[Y]}\ \boldsymbol{t}_t\ \texttt{[Z]}\ \boldsymbol{r}\ \texttt{[W]}$$

Our prompt is flexible to handle predicted tokens with arbitrary lengths at arbitrary positions, benefiting from the generative text infilling form.

3.3 Model and Training

Given a PLM $\mathcal{M}$ and a template $T(\boldsymbol{x})$ as input, we map $T(\boldsymbol{x})$ into embeddings in which the continuous tokens are mapped to a sequence of continuous vectors,

$$\mathrm{e}(\boldsymbol{x}), h_0, ..., h_{n_0-1}, \mathrm{e}(\texttt{[X]}), \mathrm{e}(\boldsymbol{e}_h), h_{n_0}, ...,$$
$$h_{n_1-1}, \mathrm{e}(\texttt{[Y]}), \mathrm{e}(\boldsymbol{e}_t), h_{n_1}, ..., h_{n_2-1}, \mathrm{e}(\texttt{[Z]})$$

where $\mathrm{e}(\cdot)$ is the embedding layer of $\mathcal{M}$, $h_i \in \mathbb{R}^d$ are trainable embedding tensors with random initialization, d is the embedding dimension of $\mathcal{M}$, and $0 \le i < n_2$. We feed the input embeddings to the encoder of the model, and obtain hidden representations $\mathbf{h}$ of the sentence:

$$\mathbf{h} = \mathrm{Enc}(T(\boldsymbol{x}))$$

At the j-th step of the decoder, the model attends to previously generated tokens $y_{<j}$ and the encoder output $\mathbf{h}$, and then predicts the probability of the next token:

$$p(y_j | y_{<j}, T(\boldsymbol{x})) = \mathrm{Dec}(y_{<j}, \mathbf{h})$$

We train our model by minimizing the negative log-likelihood of label text $\boldsymbol{y}$ tokens given $T(\boldsymbol{x})$ as input:

$$\mathcal{L}_{gen} = -\sum_{j=1}^{|\boldsymbol{y}|} \log p(y_j | y_{<j}, T(\boldsymbol{x}))$$

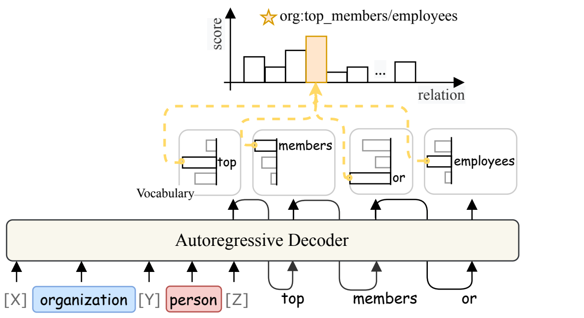


Figure 3: Entity-guided decoding and relation scoring.

3.4 Entity-Guided Decoding and Scoring

We propose a simple yet effective entity-guided decoding strategy, which exploits entity type information to implicitly influence the choice of possible candidate relations. As shown in Figure 3, at the beginning of decoding, instead of only inputting the start-of-sequence token <s> to the decoder, we also append the entity type tokens. With $\hat{\boldsymbol{y}} = \texttt{<s>}\ \texttt{[X]}\ \boldsymbol{t}_h\ \texttt{[Y]}\ \boldsymbol{t}_t\ \texttt{[Z]}$ as initial decoder inputs that serves as "preamble", the model iteratively predicts the subsequent tokens:

$$p_{y_j} = p(y_j | \hat{\boldsymbol{y}}, y_{<j}, T(\boldsymbol{x}))$$

where p_{y_j} is the probability of token y_j at the j-th prediction step. We sum the predicted probabilities of tokens in a label verbalization and normalize it by verbalization length to get the prediction score of the corresponding relation. Formally, for each relation $r \in \mathcal{R}$ with its label verbalization $\boldsymbol{r}$, the prediction score s_r is calculated as follows:

$$s_r = \frac{1}{|\boldsymbol{r}|} \sum_{j=1}^{|\boldsymbol{r}|} p_{\boldsymbol{r}_j}$$

where $p_{\boldsymbol{r}_j}$ represents the probability of token $\boldsymbol{r}_j$ at the j-th step of decoding. In this simple way, we can easily align the generated sequences with the label set. Following the work of Sainz et al. (2021), we discard relations that do not match the entity types of instances. The sentence is classified into the relation with the highest score.

4 Experimental Setup

4.1 Datasets and Setups

Following the work of Han et al. (2021c); Chen et al. (2022); Zhou and Chen (2021), we conduct experiments on popular RC datasets TACRED (Zhang et al., 2017), TACREV (Alt et al., 2020), and Re-TACRED (Stoica et al., 2021). Wiki80 (Han et al.,

	Model	*Extra Data*	*Entity Type*	*Label Semantics*	*Encoder*	TACRED	TACREV	Re-TACRED	Wiki80
Fine-tuning	SpanBERT (Joshi et al., 2020)	✓	✓	✗	SpanBERT	70.8	$78.0^{\diamondsuit}$	85.3¶	$88.1^{\star}$
	LUKE (Yamada et al., 2020)	✓	✗	✗	LUKE	72.7	$80.6^{\ddagger}$	$90.3^{\ddagger}$	$89.2^{\star}$
	GDPNet (Xue et al., 2021)	✓	✓	✗	SpanBERT	70.5	80.2	–	–
	TANL (Paolini et al., 2021)	✗	✗	✓	T5	$72.1^{\star}$	$81.2^{\star}$	$90.8^{\star}$	$89.1^{\star}$
	NLI (Sainz et al., 2021)	✓	✓	✓	RoBERTa-mnli	71.0	–	–	–
	TYP Marker (Zhou and Chen, 2021)	✗	✓	✗	RoBERTa	74.6	83.2	91.1	$89.9^{\star}$
Prompt Tuning	PTR (Han et al., 2021c)	✗	✓	✓	RoBERTa	72.4	81.4	90.9	–
	KnowPrompt (Chen et al., 2022)	✗	✓	✓	RoBERTa	72.4	82.4	**91.3**	$89.0^{\star}$
	GenPT (BART)	✗	✓	✓	BART	74.6	82.9	91.0	**90.8**
	GenPT (T5)	✗	✓	✓	T5	**75.3**	**84.0**	91.0	90.6
	GenPT (RoBERTa)	✗	✓	✓	RoBERTa	74.7	83.4	91.1	90.7

Table 1: Fully supervised results of micro F_1 (%) on four datasets. ◇ are reported by Alt et al. (2020), ¶ are reported by Stoica et al. (2021), ‡ are reported by Zhou and Chen (2021), † are reported by Chen et al. (2022), ⋆ indicates we rerun original code, and others are from the original papers. **Best** numbers are highlighted in each column.

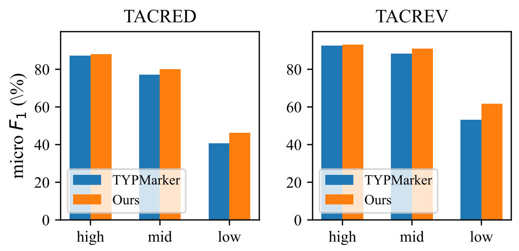


Figure 4: Comparison of F_1 (%) with different frequency relations on TACRED and TACREV. See Appendix E.2 for detailed scores.

2019) is also adopted to evaluate our proposed model. More details on datasets can be found in Appendix A. We evaluate our model under the fully supervised setting and low-resource setting. For the fully supervised setting, all training and validation data are available. For the low-resource setting, we randomly sample K instances per relation for training and validation, with K being 8, 16, and 32, respectively. Following the work of Gao et al. (2021), we measure the average performance across five different randomly sampled data points using a fixed set of seeds for each experiment.

4.2 Implementation Details

The approach is based on Pytorch (Paszke et al., 2019) and the Transformer library of Huggingface (Wolf et al., 2020). We implement our method on pre-trained generative models T5-large (Raffel et al., 2020) and BART-large (Lewis et al., 2020). For a fair comparison, we further adopt the same pre-trained model RoBERTa-large (Liu et al., 2019) as in prior work. The implementation details for different pre-trained models are in Appendix B. The training procedures and hyperparameter selection are in Appendix C. We conduct ablation experiments and performance analysis based on the BART version for time efficiency.

4.3 Baselines

We compare our model with some recent efforts. ***Discriminative fine-tuning methods:*** 1) SpanBERT (Joshi et al., 2020), a span-based pre-trained model, 2) LUKE (Yamada et al., 2020), pre-trained contextualized representations of words and entities based on Transformer, 3) GDPNet (Xue et al., 2021), constructing a latent multi-view graph to find indicative words from long sequences for RC, 4) TYP Marker (Zhou and Chen, 2021), incorporating entity representations with typed markers, 5) NLI (Sainz et al., 2021), reformulating RC as an entailment problem with hand-crafted verbalizations.

Generative fine-tuning methods: 6) TANL (Paolini et al., 2021), a generative model framing structured prediction as a translation task. We rerun the original code based on T5-large.

Discriminative prompt-tuning methods: 7) PTR (Han et al., 2021c), a prompt tuning method applying logic rules to construct prompts with several sub-prompts, and 8) KnowPrompt (Chen et al., 2022), a knowledge-aware prompt tuning approach that injects knowledge into template design and answer construction.

Among these works, SpanBERT, LUKE, GDPNet, and NLI adopt extra data for pre-training. SpanBERT, GDPNet, NLI, PTR, KnowPrompt, and TYP Marker utilize entity type information in their methods. TANL, NLI, PTR, and KnowPrompt exploit label semantics. This information is shown in Table 1.

	Model	TACRED			TACREV			Re-TACRED			Wiki80		
		K=8	K=16	K=32	K=8	K=16	K=32	K=8	K=16	K=32	K=8	K=16	K=32
Fine-tuning	SpanBERT*	8.4	17.5	17.9	5.2	5.7	18.6	14.2	29.3	43.9	40.2	70.2	73.6
	LUKE*	9.5	21.5	28.7	9.8	22.0	29.3	14.1	37.5	52.3	53.9	71.6	81.2
	GDPNet†	11.8	22.5	28.8	8.3	20.8	28.1	18.8	48.0	54.8	45.7	61.2	72.3
	TANL*	18.1	27.6	32.1	18.6	28.8	32.2	26.7	50.4	59.2	68.5	77.9	82.2
	NLI*	31.8	35.6	36.7	–	–	–	–	–	–	–	–	–
	TYP Marker‡	28.9	32.0	32.4	27.6	31.2	32.0	44.8	54.1	60.0	70.6*	77.6*	82.2*
Prompt Tuning	PTR ‡	28.1	30.7	32.1	28.7	31.4	32.4	51.5	56.2	62.1	–	–	–
	KnowPrompt †	32.0	35.4	36.5	32.1	33.1	34.7	55.3	**63.3**	65.0	78.5*	82.2*	84.5*
	GenPT (BART)	32.6 (±0.5)	35.6 (±1.8)	37.1 (±1.0)	31.1 (±0.8)	34.3 (±1.9)	**36.6** (±1.1)	52.9 (±1.5)	58.1 (±1.8)	63.9 (±1.5)	77.8 (±0.5)	82.4 (±0.7)	85.1 (±0.4)
	GenPT (T5)	31.8 (±0.8)	33.3 (±1.3)	36.0 (±1.1)	31.3 (±1.7)	32.6 (±1.5)	34.6 (±1.2)	54.7 (±2.3)	58.7 (±2.4)	62.5 (±1.0)	77.1 (±0.9)	82.0 (±0.6)	84.3 (±0.4)
	GenPT (RoBERTa)	**35.7** (±1.1)	**36.6** (±1.5)	**37.4** (±1.8)	**34.4** (±0.8)	**34.6** (±1.6)	36.2 (±1.8)	**57.1** (±2.3)	60.4 (±1.3)	**65.2** (±2.0)	**78.6** (±0.4)	**83.3** (±0.3)	**85.7** (±0.4)

Table 2: Low-resource results on four datasets. We report the mean and standard deviation performance of micro F_1 (%) over 5 different splits. Results marked with † are reported by Chen et al. (2022), ‡ are reported by Han et al. (2021c), and $\star$ indicates we rerun original code under low-resource settings. **Best** numbers are highlighted in each column.

5 Results and Discussion

In this section, we conduct extensive experiments and answer the following questions: 1) How does our proposed method perform? We demonstrate the superiority of our approach under extensive datasets and settings. 2) What is the impact of prompt design? We conduct extensive experiments to investigate the influence of different prompts. 3) How much benefit can label semantics bring to the RC task? We carry out experiments to illustrate the impact of label semantics on the performance of RE. 4) What is the effect of the proposed decoding strategy? We perform ablation and case studies to prove the effectiveness and efficiency of decoding strategies.

5.1 Comparison to Baselines

Results of fully supervised RC As shown in Table 1, we evaluate our model in the fully supervised setting. Specifically, our model outperforms the state-of-the-art discriminative fine-tuning model TYP Marker and prompt tuning methods PTR and KnowPrompt. Compared to the generative approach TANL which converts RC to augmented natural language translation, GenPT-T5 improves 3.2 points in terms of F_1 on TACRED, suggesting the format of text infilling can better exploit pre-trained knowledge in language models.

Impact of training relation frequency To further explore the impact of relation frequencies in training data, we split the test set into three subsets according to the class frequency in training. Specifically, we regard the relations with more than 300 training instances to form a high frequency subset, those with 50-300 training instances form a middle frequency subset, and the rest form a low frequency subset. Detailed statistics and data splits are in Appendix E.2. As shown in Figure 4, we evaluate our model and TYP Marker on the three subsets of the test data. Our model outperforms TYP Marker on all three subsets, especially on the low frequency set, proving its effectiveness when the class rarely appears in the training data.

Results of low-resource RC Table 2 presents the results of micro F_1 under the low-resource setting. Our model achieves the best performance in comparison to existing approaches, proving that our method can better handle extremely few-shot classification tasks. The performance gain mainly comes from reformulating RC to a text infilling task to eliminate the rigid prompt restrictions and thus fully leveraging the label semantics. In addition, we notice our method achieves strong performance based on BART and T5, while yielding the best results in low-resource settings when using the same backbone RoBERTa as the discriminative baselines, which on the one hand fairly demonstrates the effectiveness of our method, on the other hand suggests RoBERTa is more data-efficient.

5.2 The Effect of Prompt

The impact of prompt format Extensive experiments with various templates are conducted to illustrate the effect of prompt construction, as shown in Table 3. We choose to put the relation at the end

No.	Inputs	Targets	TACRED K=8	TACRED K=16	TACRED K=32
1	$\boldsymbol{x}$. $[v_{0:n_0-1}]$ [MASK] $\boldsymbol{e}_h$ $[v_{n_0:n_1-1}]$ [MASK] $\boldsymbol{e}_t$ $[v_{n_1:n_2-1}]$ [MASK].	[MASK] $\boldsymbol{t}_h$ [MASK] $\boldsymbol{t}_t$ [MASK] $\boldsymbol{r}$ [MASK]	**32.6**	**35.6**	**37.1**
2	$\boldsymbol{x}$. *The relation between* [MASK] $\boldsymbol{e}_h$ *and* [MASK] $\boldsymbol{e}_t$ *is* [MASK].	[MASK] $\boldsymbol{t}_h$ [MASK] $\boldsymbol{t}_t$ [MASK] $\boldsymbol{r}$ [MASK]	32.3	34.7	37.0
3	$\boldsymbol{x}$. $[v_{0:n_0-1}]$ $\boldsymbol{t}_h$ $\boldsymbol{e}_h$ $[v_{n_0:n_1-1}]$ $\boldsymbol{t}_t$ $\boldsymbol{e}_t$ $[v_{n_1:n_2-1}]$ [MASK].	[MASK] $\boldsymbol{r}$ [MASK]	30.0	33.8	35.6
4	$\boldsymbol{x}$. $[v_{0:n_0-1}]$ $\boldsymbol{e}_h$ $[v_{n_0:n_1-1}]$ $\boldsymbol{e}_t$ $[v_{n_1:n_2-1}]$ [MASK].	[MASK] $\boldsymbol{r}$ [MASK]	30.4	34.1	35.9
5	$\boldsymbol{x}$. $[v_{0:n_2-1}]$ [MASK].	[MASK] $\boldsymbol{r}$ [MASK]	23.4	25.2	26.4
6	$\boldsymbol{x}$.	$\boldsymbol{r}$	22.7	25.0	26.1

Table 3: Ablation study on TACRED showing micro F_1 (%) to illustrate the impact of prompt designs.

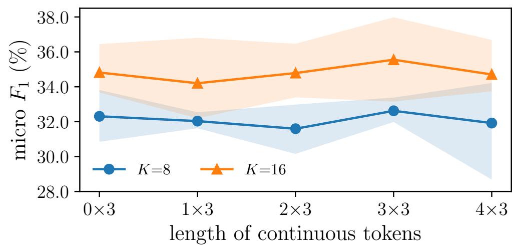


Figure 5: Micro F_1 (%) with different numbers of continuous tokens on TACRED.

Model	TACRED K=8	TACRED K=16	TACRED K=32
Ours (full-semantic)	**32.6**	**35.6**	**37.1**
Handmade verbalization (partial-semantic)	29.8	31.5	33.2
Numeric relation id (non-semantic)	16.2	24.4	30.8

Table 4: Analysis of verbalizations with original label tokens, handcrafted tokens with fixed length, and numeric relation ids.

of the template due to its agreement with autoregressive generation, allowing utilizing entity types as guidance during inference. Row #2 displays the results of manually designed template with discrete tokens. The results indicate the learnable continuous prompt achieves better performance and frees us from prompt engineering, i.e., picking the optimal discrete template. As depicted in row #3, we add entity types to the input sequence instead of predicting them in the targets. Row #4 represents predicting the relation labels without entity types. Row #5 further removes entity mentions in the template. The F_1 scores of these implementations all degrade, suggesting the importance of entity types and mentions for RE. Moreover, we compare our model to the vanilla seq2seq fine-tuning method (row #6). Our model outperforms the vanilla fine-tuning by a large margin, revealing the effectiveness of our entity-oriented prompt design and tuning.

Discussion of continuous token length Here we discuss the effect of continuous token lengths. The length of continuous tokens in the template $T(\cdot)$ is set to $n \times 3$, where $n_0 = n_1 - n_0 = n_2 - n_1 = n$. The results are shown in Figure 5. The F_1 increases when n increases from 0 to 3, and then decreases slightly. In our experiment, we fix n to 3 to achieve effective performance, that is totally 9 continuous tokens in the template. Interestingly, we observe that the model gains valid performance when n is 0, which suggests our method can be adapted to zero-shot setting by selecting n as 0. We show the experimental results under zero-shot setting in Appendix F.

5.3 Analysis of Label Semantics

Our approach makes full use of label semantics by learning to predict entire label verbalizations with different lengths. To verify the benefits coming from the label semantics, we compare our method with predicting partial-semantic hand-crafted verbalization and non-semantic numeric relation id. The results are presented in Table 4. Specifically, predicting hand-crafted verbalization means mapping each label to manually crafted label verbalization with fixed length, which may lose some semantic information. Here we adopt the hand-crafted verbalizations by the work of Han et al. (2021c). For example, relation "*per:country_of_birth*" is mapped to [*person, was, born, in, country*]. See Table 12 in Appendix E.3 for all manually crafted label verbalizations. For predicting non-semantic numeric relation id, we assign a relation id (e.g., "`<Rel_0>`"), a special text token to a specific relation, and make the model learn to generate the correct relation id. Our model obtains the best results by leveraging full label semantics, proving that label semantics is crucial for the RC task.

5.4 Analysis of Decoding Strategy

The effect of relation scoring When re-running TANL on TACRED, we notice that it takes a long time (86.62 hours) to perform inference on the test

Model	TACRED	
	K=8	K=16
Ours	32.6	35.6
Likelihood-based Prediction (LP)	32.4	35.5

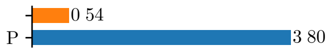


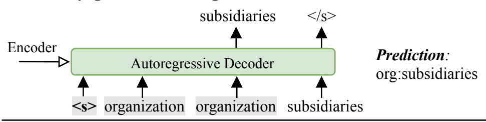
Table 5: Micro F_1 (%) and inference time (hours) on the test set with relation scoring and likelihood-based prediction (LP), respectively.

Model	TACRED			
	K=8	K=16	K=32	full
Ours	32.6	35.6	37.1	74.6
w/o entity-guided decoding	32.1	35.0	36.7	73.5

Case
However , the government let the deal expire in December 2001 amid protests from local politicians and workers at a Semen Gresik unit, Semen Padang in West Sumatra .

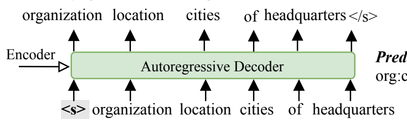
Gold org:subsidiaries *Entity Types* <organization, organization>

with entity-guided decoding



***Prediction**:* org:subsidiaries

w/o entity-guided decoding



***Prediction**:* org:city_of_headquarters

Table 6: Ablation and case studies to illustrate the effect of entity-guided decoding.

set. To illustrate the efficiency of our approach, we compare our relation scoring strategy with likelihood-based prediction (Nogueira dos Santos et al., 2020; Paolini et al., 2021), which feeds each candidate sequence into decoder and uses output token likelihoods as the corresponding class scores. More details are in Appendix E.4. As shown in Table 5, our method achieves promising performance with less inference time. In addition, we note all generative methods are slower than their discriminative counterparts due to the autoregressive nature of the process (see Section 7).

The effect of entity-guided decoding As shown in Table 6, by removing the entity-guided decoding, micro F_1 drops from 32.6 to 32.1 with K=8, proving its effectiveness. We further carry out a detailed case study. A real test instance from TACRED with its entity type information is also given in Table 6. When there is no entity type guidance, the decoder generates the sequence "*organization location cities of headquarters*", and incorrectly classifies the instance as relation "*org:city_of_headquarters*". Our model equipped with entity-guided decoding correctly predicts the relation as "*org:subsidiaries*". This strategy implicitly restricts the generated candidates, gaining performance improvement.

6 Related Work

6.1 Language Model Prompting

Language model prompting has emerged with the introduction of the GPT series (Radford et al., 2018, 2019; Brown et al., 2020). PET (Schick and Schütze, 2021a,b) reformulates input examples as cloze-style phrases to help language models understand given tasks. ADAPET (Tam et al., 2021) modifies PET's objective to provide denser supervision. However, these methods require manually designed patterns and label verbalizers. To avoid labor-intensive prompt design, automatic prompt search (Shin et al., 2020; Schick et al., 2020) has been extensively explored. LM-BFF (Gao et al., 2021) adopts T5 to generate prompt candidates and verify their effectiveness through prompt tuning. We would like to highlight the differences between LM-BFF and ours: they adopt T5 for generating templates but still use a single token for each label, whereas we employ T5 to capture variable-length label descriptions. Continuous prompt learning (Li and Liang, 2021; Qin and Eisner, 2021; Liu et al., 2021b, 2022; Zhang et al., 2022a) is further proposed, which directly uses learnable continuous embeddings as prompts rather than discrete token phrases. A few related methods on tasks such as knowledge probing (Zhang et al., 2022b) and NER (Chen et al., 2021) use generative PLMs to generate prompts, or directly generate labels into a sequence without any template, but none of them employs the generative PLMs to fill in the blanks contained in the template as we do.

6.2 Relation Classification

Fine-tuning PLMs for RC (Joshi et al., 2020; Yamada et al., 2020; Xue et al., 2021; Lyu and Chen, 2021) has achieved promising performance. Zhou and Chen (2021) achieves state-of-the-art results by incorporating entity type information into entity markers. Another interesting line is converting information extraction into generation form. Zeng et al. (2018) and Nayak and Ng (2020) propose seq2seq models to extract relational facts. Huang et al. (2021) present a generative framework for document-level entity-based extraction tasks.

There are other works converting RC to translation framework (Paolini et al., 2021; Wang et al., 2021) and entailment task (Sainz et al., 2021). When applying prompt tuning to RC, labels of variable lengths in RC is a major obstacle. To address this issue, existing works (Han et al., 2021c; Chen et al., 2022) manually abridge labels into fixed-length tokens, or use one virtual token to represent one relation. Unlike them, GenPT frees prompt tuning from fixed-length verbalizations and predicts complete label tokens. In addition, our method does not need any manual effort, which is more practical when adapted to other datasets or similar tasks.

Few-shot relation classification has attracted recent attention, which can be roughly divided into two directions according to the few-shot setting. One direction is meta-learning based few-shot relation classification (Han et al., 2018a, 2021a,b). Given large-scale labeled base class training data and novel class data with scarce labeled instances, the task is required to learn generalized representations from base classes and then adapt these representations to novel classes. The other is generalized few-shot relation classification (Sainz et al., 2021; Han et al., 2021c; Chen et al., 2022), which only relies on a small amount of labeled data to train the model. In our work, we conduct experiments following the generalized few-shot setting, which is more practical and challenging.

7 Conclusion

This paper presents a novel generative prompt tuning method for RC. Unlike vanilla prompt tuning that converts a specific task into an MLM problem, we reformulate RC as a text infilling task, which can predict label verbalizations with varying lengths at multiple predicted positions and thus better exploit the semantics of entity and relation types. In addition, we design a simple yet effective entity-guided decoding and discriminative scoring strategy to efficiently decode predicted relations, making our generative model more practical. Qualitative and quantitative experiments on four benchmarks prove the effectiveness of our approach.

Future work includes applying the proposed method to other related tasks where the label space is large with rich semantic information. In addition, we believe GenPT can be adapted to support multiple labels for the same class. This is analogous to the typical NMT model involving many-to-many mapping (where the target may not be unique). How to exactly introduce multiple valid, semantically equivalent label descriptions (including design of label verbalizations, learning process, etc.) is a research topic deserving a separate study.

Limitations

We argue that the main limitation of our work is the time efficiency of the generative approach compared to discriminative methods, even though we are more efficient than other generative models (see Section 5.4 for a detailed discussion). The decreased time efficiency is mainly due to the autoregressive nature of the generation process. Specifically, compared to the discriminative model PTR based on RoBERTa-large which takes 0.11 hours to evaluate on TACRED test data (batch size is 32), our GenPT-BART takes 0.54 hours with the same inference batch size. GenPT-T5 and GenPT-RoBERTa take more inference time due to larger model parameters and 2-dimensional attention masks respectively, which are 1.97 hours and 1.29 hours.

Acknowledgements

We would like to thank the anonymous reviewers, our meta-reviewer and senior area chairs for their thoughtful comments and support on this work. We would also like to Shengqiang Zhang for helpful discussions. The first author was a visiting PhD student at SUTD when conducting this work. This research/project is supported by Beijing Nova Program of Science and Technology (Grant No. Z191100001119031), National Natural Science Foundation of China (Grant No.U21A20468, 52071312, 61972043, 61921003), Guangxi Key Laboratory of Cryptography and Information Security (Grant No. GCIS202111), the Open Program of Zhejiang Lab (Grant No.2021PD0AB02), the Fundamental Research Funds for the Central Universities under grant 2020XD-A07-1, China Scholarship Council Foundation, and the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Program (AISG Award No: AISG2-RP-2020-016).

References

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. 2020. TACRED revisited: A thorough evaluation of the TACRED relation extraction task. In *Proceedings of ACL*.

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 *Proceedings of NeurIPS*.

Xiang Chen, Lei Li, Shumin Deng, Chuanqi Tan, Changliang Xu, Fei Huang, Luo Si, Huajun Chen, and Ningyu Zhang. 2021. Lightner: A lightweight tuning paradigm for low-resource NER via pluggable prompting. *arXiv preprint arXiv:2109.00720*.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of WWW*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Proceedings of NeurIPS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of ACL*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: word-level adversarial reprogramming. In *Proceedings of ACL*.

Jiale Han, Bo Cheng, and Wei Lu. 2021a. Exploring task difficulty for few-shot relation extraction. In *Proceedings of EMNLP*.

Jiale Han, Bo Cheng, and Guoshun Nan. 2021b. Learning discriminative and unbiased representations for few-shot relation extraction. In *Proceedings of CIKM*.

Xu Han, Tianyu Gao, Yuan Yao, Deming Ye, Zhiyuan Liu, and Maosong Sun. 2019. OpenNRE: An open and extensible toolkit for neural relation extraction. In *Proceedings of EMNLP*.

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

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018a. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of EMNLP*.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018b. FewRel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of EMNLP*.

Kung-Hsiang Huang, Sam Tang, and Nanyun Peng. 2021. Document-level entity-based extraction as template generation. In *Proceedings of EMNLP*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *TACL*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of EMNLP*.

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 *Proceedings of ACL*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of ACL*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. GPT understands, too. *arXiv preprint arXiv:2103.10385*.

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*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of ICLR*.

Shengfei Lyu and Huanhuan Chen. 2021. Relation classification with entity type restriction. In *Proceedings of Findings of ACL*.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *Proceedings of AAAI*.

Cicero Nogueira dos Santos, Xiaofei Ma, Ramesh Nallapati, Zhiheng Huang, and Bing Xiang. 2020. Beyond [CLS] through ranking by generation. In *Proceedings of EMNLP*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *Proceedings of ICLR*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of NeurIPS*.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of NAACL*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *Technical report, OpenAI*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *Technical report, OpenAI*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Oscar Sainz, Oier Lopez de Lacalle, Gorka Labaka, Ander Barrena, and Eneko Agirre. 2021. Label verbalization and entailment for effective zero and few-shot relation extraction. In *Proceedings of EMNLP*.

Teven Le Scao and Alexander M. Rush. 2021. How many data points is a prompt worth? In *Proceedings of NAACL*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *Proceedings of COLING*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of EACL*.

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

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 *Proceedings of EMNLP*.

George Stoica, Emmanouil Antonios Platanios, and Barnabás Póczos. 2021. Re-tacred: Addressing shortcomings of the TACRED dataset. In *Proceedings of AAAI*.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of EMNLP*.

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*.

Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*.

Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. 2021. Zero-shot information extraction as a unified text-to-triple translation. In *Proceedings of EMNLP*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP*.

Fuzhao Xue, Aixin Sun, Hao Zhang, and Eng Siong Chng. 2021. Gdpnet: Refining latent multi-view graph for relation extraction. In *Proceedings of AAAI*.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of EMNLP*.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of ACL*.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022a. Differentiable prompt makes pre-trained language models better few-shot learners. In *Proceedings of ICLR*.

Yue Zhang, Hongliang Fei, Dingcheng Li, and Ping Li. 2022b. Promptgen: Automatically generate prompts using generative models. In *Proceedings of Findings of NAACL*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of EMNLP*.

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

Wanrong Zhu, Zhiting Hu, and Eric P. Xing. 2019. Text infilling. *arXiv preprint arXiv:1901.00158*.

A Datasets

We conduct experiments on four RC datasets, which are TACRED[3] (Zhang et al., 2017), TACREV[4] (Alt et al., 2020), Re-TACRED[5] (Stoica et al., 2021), and Wiki80[6] (Han et al., 2019). Table 7 summarizes key statistics of the four datasets.

TACRED is one of the most widely used RC datasets. Each instance includes a natural sentence sequence, the types and spans of the entity mentions, and the relation held between the entities or *no_relation* label if no relation was found.

TACREV is a dataset revised from TACRED, which has the same training data as the original TACRED and extensively relabeled development and test sets.

Re-TACRED is another completely re-annotated version of TACRED dataset through an improved crowdsourcing strategy. They re-define relation labels to make them more clear and intuitive and re-annotate the full TACRED dataset.

Wiki80 is a relation classification dataset derived from FewRel (Han et al., 2018b), a large scale few-shot RC dataset. The original data only has a training/development split, so we randomly resplit the training set into a training set and a development set, and treat the original development set as test set, following the split used in Chen et al. (2022). The entity type information is obtained from Wikidata (Vrandecic and Krötzsch, 2014).

B Implementation Details

T5 Version The approach based on T5 is described in Section 3. We use `<extra_id_0>`, `<extra_id_1>`, `<extra_id_2>`, and `<extra_id_3>` as sentinel tokens `[X]`, `[Y]`, `[Z]`, and `[W]`.

BART Version The model based on BART is basically the same as the T5 version, except that the sentinel tokens in the template are replaced with `[MASK]` tokens, following the pre-training task format of BART, and the target sequence is composed of entity types and label verbalizations.

[3]https://nlp.stanford.edu/projects/tacred/
[4]https://github.com/DFKI-NLP/tacrev
[5]https://github.com/gstoica27/Re-TACRED
[6]https://github.com/thunlp/OpenNRE

Dataset	#train	#dev	#test	#rel
TACRED	68,124	22,631	15,509	42
TACREV	68,124	22,631	15,509	42
Re-TACRED	58,465	19,584	13,418	40
Wiki80	44,800	5,600	5,600	80

Table 7: Statistics of datasets used.

RoBERTa Version In this section, we detail the implementation based on RoBERTa. For the RoBERTa version, we concatenate the source and target as inputs together and use partial causal masking to distinguish the encoder-decoder representations, as shown in Figure 6. Specifically, we utilize one RoBERTa model as both the source and target embeddings by concatenating the source sequence $T(\boldsymbol{x})$ and target $\boldsymbol{y}$ as inputs together. We provide a partial causal attention mask to distinguish the source / target representations. The attention mask – 2-dimensional matrix denotes as $\mathbf{M}$. Specifically, for the source tokens, the mask allows full source self-attention, but mask out all target tokens. For $i = [0, 1, ..., |T(\boldsymbol{x})| - 1]$, $|\cdot|$ represents the length of sequence,

$$M_{ij} = \begin{cases} 1, & 0 \le j < |T(\boldsymbol{x})|, \\ 0, & otherwise \end{cases}$$

To guarantee that the decoder is autoregressive, we enforce the target tokens to only attend to previous tokens and not attend to future tokens. For $i = [0, 1, ..., |\boldsymbol{y}| - 1] + |T(\boldsymbol{x})|$,

$$M_{ij} = \begin{cases} 1, & 0 \le j < |T(\boldsymbol{x})|, \\ 1, & |T(\boldsymbol{x})| \le j \le i, \\ 0, & otherwise \end{cases}$$

C Setup and Training Details

The model is implemented with Pytorch[7] and Huggingface transformer library[8]. T5-large[9], BART-large[10], and RoBERTa-large[11] are adopted as the backbone to validate our method, which have 770 million, 406 million, and 355 million parameters, respectively. We utilize 1 NVIDIA Tesla V100 GPU with 32 GB memory to run all experiments. For our implementation based on BART, the training times of TACRED under $K = 16$ and fully

[7]https://pytorch.org/
[8]https://github.com/huggingface/transformers
[9]https://huggingface.co/t5-large
[10]https://huggingface.co/facebook/bart-large
[11]https://huggingface.co/roberta-large

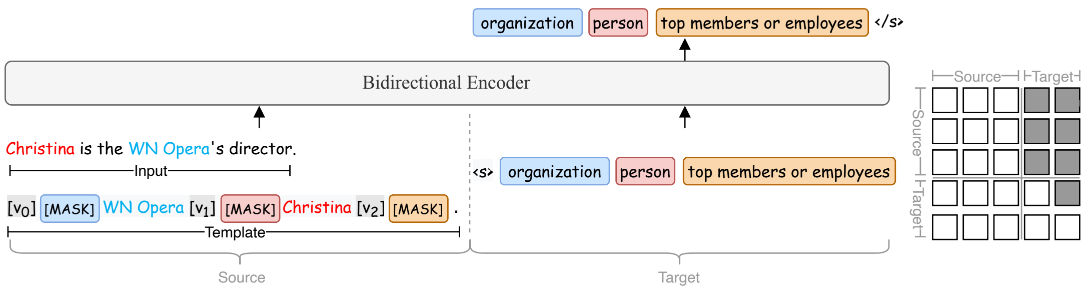

organization
person
top members or employees
</s>
Bidirectional Encoder
Christina is the WN Opera's director.
Input
[V0] [MASK] WN Opera [V1] [MASK] Christina [V2] [MASK] .
Template
<s> organization person top members or employees
Source
Target

Figure 6: Our proposed generative prompt tuning approach based on RoBERTa. The right part is the partially causal masking strategy (white cell: unmasked; grey cell: masked).

supervised settings are 0.36 hours and 10.1 hours, respectively, and testing time is 0.54 hours. The maximum generation length L depends on the maximum length of label verbalizations. The length of continuous tokens in the template $T(\cdot)$ is set to $n \times 3$, where $n_0 = n_1 - n_0 = n_2 - n_1 = n$. n is 3 in our implementation, and detailed discussion is in Section 5.2. We use a batch size of 4 for T5 and 16 for BART, which are chosen for practical consideration to fit into GPU memory. During training, our model is optimized with AdamW (Loshchilov and Hutter, 2019). The hyper-parameters are manually adjusted based on the performance on the development set.

C.1 Low-Resource Setting

For low-resource setting, only a small amount of labeled data is available, i.e. K labeled instances per relation for training and validation sets. K is chosen from 8, 16, and 32. The model is finally evaluated on the whole test set. To validate the performance under the low-resource setting, we report the average performance across five different randomly sampled training and development data using a fixed set of seeds $S_{seed} = \{13, 21, 42, 87, 100\}$.

The hyper-parameters we chose are shown as follows:

- maximum length of the input sequence: 512
- learning rate of optimization for pre-trained language model parameters: $3e-5$
- learning rate of optimization for the learnable continuous token in the template: $1e-5$
- epochs: 10 for TACRED, TACREV, Re-TACRED, and 20 for Wiki80

C.2 Fully Supervised Setting

Under this setting, whole training, development, and test data are all available. The hyper-parameters we chose are shown as follows:

- maximum length of the input sequence: 512
- learning rate of optimization for pre-trained language model parameters: $3e-5$
- learning rate of optimization for the learnable continuous token in the template: $1e-5$
- epochs: 5 for TACRED, TACREV, Re-TACRED, and 10 for Wiki80

D Metrics and Validation Performance

We use the micro F_1 score[12] for test metrics to evaluate models, which is calculated globally by counting the total true positives, false negatives and false positives. The performance of our method on the validation set is shown in Table 8. Note that the validation data under low-resource setting is also obtained by randomly sampling K instances. We report the average micro F_1 and standard deviation over 5 different splits.

Dataset	K=8	K=16	K=32	full
TACRED	80.1 (±2.0)	82.7 (±1.9)	84.3 (±0.8)	74.3
TACREV	82.4 (±1.1)	85.6 (±1.0)	88.9 (±0.8)	82.1
Re-TACRED	85.6 (±1.8)	89.4 (±1.2)	91.0 (±0.4)	90.8
Wiki80	82.1 (±0.8)	87.1 (±0.6)	89.1 (±0.2)	93.5

Table 8: Micro F_1 of GenPT-RoBERTa on the validation set.

E Experimental Details

E.1 Implementation Details of Baselines

The rest of baseline systems were trained using the hyperparameters reported by the original papers

[12]https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

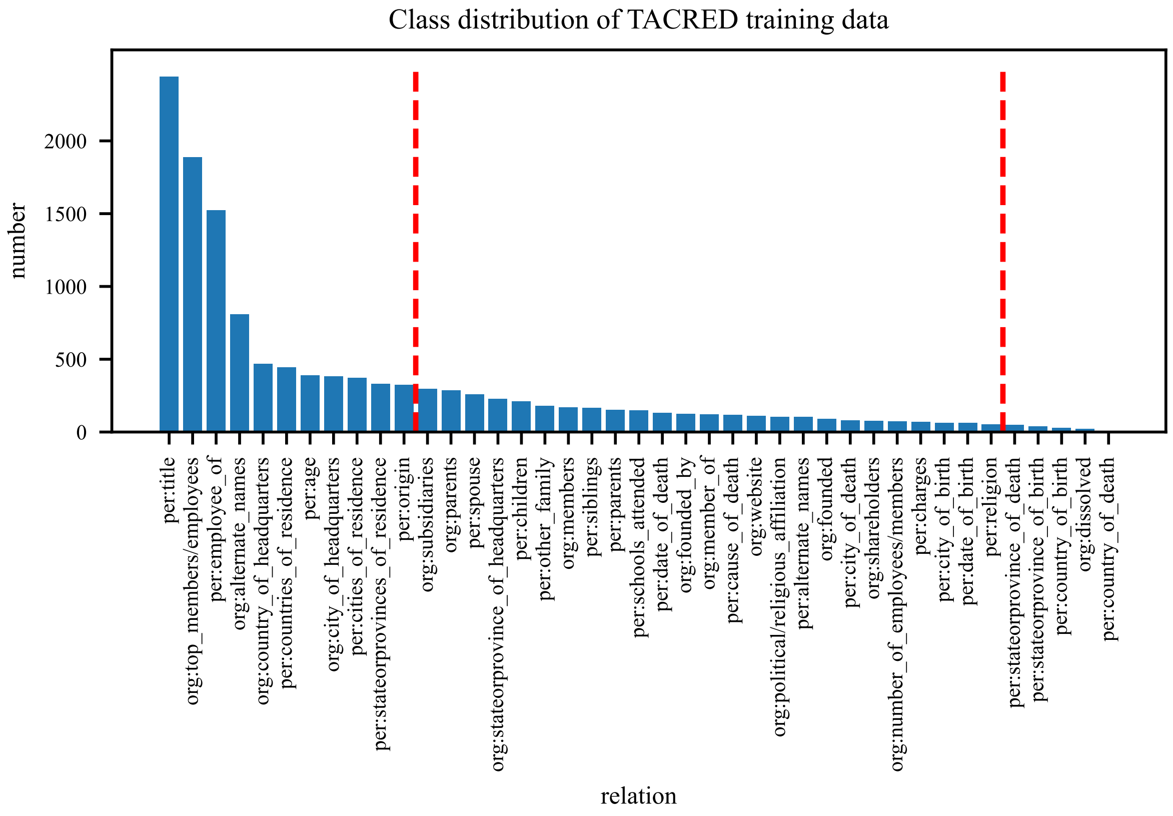


Figure 7: Class distribution of TACRED training data.

(SpanBERT[13], LUKE[14], TANL[15], NLI[16], and TYP Marker[17]). For NLI, we rerun the original code based on RoBERTa-large-mnli[18]. Since they need manual designed template for each relation, we only rerun on the TACRED dataset they use. We notice that although NLI achieves high average performance, the standard deviation on 5 different training and dev sets is also quite high, as shown in Table 9. The reason is that they use a threshold-based detection method to detect *no_relation* class. Performance is highly correlated with the choice of threshold. In addition, NLI takes 0.81 hours during inference, which makes it less efficient than other discriminative models.

One thing we need to notice is we adopt different few-shot settings compared to the original settings in NLI. Concretely, we follow the work of Gao et al. (2021); Han et al. (2021c); Chen et al. (2022), which randomly samples a small set of training and development data, that is, K for each class including *no_relation*. However, NLI collects data from positive and negative in equal ratio, e.g., under its 5% split setting, there are around 16 examples per positive relation and 2,756 examples for negative *no_relation*. The main difference of settings between NLI and us is that they adopt more negative examples during training.

Model	TACRED		
	K=8	K=16	K=32
NLI	31.8 ($\pm$2.0)	35.6 ($\pm$5.2)	36.7 ($\pm$3.8)

Table 9: Experimental results of NLI under low-resource setting.

E.2 Training Relation Frequency

As shown in Figure 7, we count the number of training data for each relaiton in TACRED, which follows a long-tailed distribution. To figure out the performance on classes of different frequencies, we split the test data into three subsets according to the class frequency in training. The relations with more than 300 training instances forms a high

[13]https://github.com/facebookresearch/SpanBERT
[14]https://github.com/studio-ousia/luke
[15]https://github.com/amazon-research/tanl
[16]https://github.com/osainz59/Ask2Transformers
[17]https://github.com/wzhouad/RE_improved_baseline
[18]https://huggingface.co/roberta-large-mnli

frequency subset (except for "*no_relation*") , those with 50-300 training instances form a middle frequency subset, and the rest form a low frequency subset. The high, middle, and low frequency subsets consist of 11, 25, and 5 relations, with each containing 2,263, 1,024, and 38 instances on TACRED and 2,180, 912, and 31 on TACREV. The specific scores of Figure 4 are shown in Table 10.

Model	TACRED			TACREV		
	high	mid	low	high	mid	low
TYP Marker	87.0	77.1	40.7	92.4	88.1	53.1
GenPT	87.9	79.8	46.2	93.1	90.9	61.5

Table 10: Detailed F_1 (%) scores of different frequency relations on TACRED and TACREV.

E.3 Details of Hand-Crafted Label Words

As shown in Table 12, we list all hand-crafted label words with fixed length (5) for relations in TACRED, which is from PTR[19]. In our experiments, we use non-semantic numeric relation id, partial-semantic handmade verbalization, and full-semantic complete label words to prove the significance of label semantic information for the relation classification task.

E.4 Details of Likelihood-based Prediction

We compare our proposed decoding relation scoring strategy with likelihood-based prediction, which performs prediction using sequence likelihoods as class scores. Specifically, given a predefined set of relations $\mathcal{R}$, likelihood-based prediction calculates score for each relation r by feeding the corresponding target sequence $\boldsymbol{y}$ = `[X]` $\boldsymbol{t}_h$ `[Y]` $\boldsymbol{t}_t$ `[Z]` $\boldsymbol{r}$ `[W]` to the decoder and taking target sequence likelihoods as class score s_r,

$$s_r = \sum_{j=1}^{|\boldsymbol{y}|} \log P(y_j | y_{<j}, T(\boldsymbol{x}))$$

where $\boldsymbol{x}$ is the instance to be predicted, $\boldsymbol{t}_h$ and $\boldsymbol{t}_t$ denote the subject entity type and object entity type sequences of $\boldsymbol{x}$, and $\boldsymbol{r}$ represents the token verbalizations of relation r. The relation with the highest score is regarded as the predicted class.

F Results in Zero-Shot Setting

We conduct experiments with manual template,

$$T(\boldsymbol{x}) = \boldsymbol{x}. \textit{ The relation between } \texttt{[X]}\ \boldsymbol{e}_h \textit{ and } \texttt{[Y]}\ \boldsymbol{e}_t \textit{ is } \texttt{[Z]}.$$

[19] `https://github.com/thunlp/PTR`

GenPT	TACRED	TACREV	Re-TACRED
manual	13.1	10.3	19.2
continuous	15.7	12.7	18.6

Table 11: Micro F_1 (%) of GenPT (BART) under zero-shot setting.

and continuous prompt (the continuous token length is 0),

$$T(\boldsymbol{x}) = \boldsymbol{x}.\ \texttt{[X]}\ \boldsymbol{e}_h\ \texttt{[Y]}\ \boldsymbol{e}_t\ \texttt{[Z]}.$$

The experiment results in zero-shot setting are shown in Table 11.

Class Label	Hand-made Verbalization	Relation id
per:charges	person, was, charged, with, event	<Rel_0>
per:date_of_death	person, was, died, on, date	<Rel_1>
per:country_of_death	person, was, died, in, country	<Rel_2>
per:cause_of_death	person, was, died, of, event	<Rel_3>
org:founded_by	organization, was, founded, by, person	<Rel_4>
org:founded	organization, was, founded, in, date	<Rel_5>
per:city_of_death	person, was, died, in, city	<Rel_6>
per:stateorprovince_of_death	person, was, died, in, state	<Rel_7>
per:date_of_birth	person, was, born, in, date	<Rel_8>
per:stateorprovince_of_birth	person, was, born, in, state	<Rel_9>
per:country_of_birth	person, was, born, in, country	<Rel_10>
per:city_of_birth	person, was, born, in, city	<Rel_11>
org:shareholders	organization, was, invested, by, person	<Rel_12>
per:other_family	person, 's, relative, is, person	<Rel_13>
per:title	person, 's, title, is, title	<Rel_14>
org:dissolved	organization, was, dissolved, in, date	<Rel_15>
org:stateorprovince_of_headquarters	organization, was, located, in, state	<Rel_16>
org:country_of_headquarters	organization, was, located, in, country	<Rel_17>
org:city_of_headquarters	organization, was, located, in, city	<Rel_18>
per:countries_of_residence	person, was, living, in, country	<Rel_19>
per:stateorprovinces_of_residence	person, was, living, in, state	<Rel_20>
per:cities_of_residence	person, was, living, in, city	<Rel_21>
org:member_of	organization, was, member, of, organization	<Rel_22>
per:religion	person, was, member, of, religion	<Rel_23>
org:political/religious_affiliation	organization, was, member, of, religion	<Rel_24>
org:top_members/employees	organization, 's, employer, was, person	<Rel_25>
org:number_of_employees/members	organization, 's, employer, has, number	<Rel_26>
per:schools_attended	person, 's, school, was, organization	<Rel_27>
per:employee_of	person, 's, employee, was, organization	<Rel_28>
per:siblings	person, 's, sibling, was ,person	<Rel_29>
per:spouse	person, 's, spouse, was, person	<Rel_30>
per:parents	person, 's, parent, was, person	<Rel_31>
per:children	person, 's, child, was, person	<Rel_32>
per:alternate_names	person, 's, alias, was, person	<Rel_33>
org:alternate_names	organization, 's, alias, was, organization	<Rel_34>
org:members	organization, 's, member, was, organization	<Rel_35>
org:parents	organization, 's, parent, was, organization	<Rel_36>
org:subsidiaries	organization, 's, subsidiary, was, organization	<Rel_37>
per:origin	person, 's, nationality, was, country	<Rel_38>
org:website	organization, 's, website, was, url	<Rel_39>
per:age	person, 's, age, was, number	<Rel_40>
no_relation	entity, is, irrelevant, to, entity	<Rel_41>

Table 12: Hand-crafted label words with fixed length (5) for relations in TACRED.