PromptGen: Automatically Generate Prompts using Generative Models

Yue Zhang, Hongliang Fei, Dingcheng Li, Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th St, Bellevue, WA 98004, USA
{yuezhang01, hongliangfei, lidingcheng, liping11}@baidu.com

Abstract

Recently, prompt learning has received significant attention, where the downstream tasks are reformulated to the mask-filling task with the help of a textual prompt. The key point of prompt learning is finding the most appropriate prompt. This paper proposes a novel model `PromptGen`, which can automatically generate prompts conditional on the input sentence. `PromptGen` is the first work considering dynamic prompt generation for knowledge probing, based on a pre-trained generative model. To mitigate any label information leaking from the pre-trained generative model, when given a generated prompt, we replace the query input with "None". We pursue that this perturbed context-free prompt cannot trigger the correct label. We evaluate our model on the knowledge probing LAMA benchmark, and show that PromptGen significantly outperforms other baselines.

1 Introduction

Prompt learning (Petroni et al., 2019; Kassner et al., 2021) is a new learning paradigm for utilizing pre-trained language models (LM), where downstream tasks are reformulated as a mask filling task with the help of a textual prompt in the original pre-trained LM. Recently, prompt learning has been used in applications such as knowledge probing (Petroni et al., 2019; Zhong et al., 2021; Jiang et al., 2021), text classification (Gao et al., 2021; Han et al., 2021; Chen et al., 2021; Chai et al., 2020), natural language inference (Shin et al., 2020; Gao et al., 2021). Furthermore, prompt learning has shown its utility in solving few-shot learning problems (Schick and Schütze, 2021; Gao et al., 2021).

The essence of prompt learning is designing the most appropriate prompts to trigger the correct target text for downstream tasks from an LM. The latest methods to construct prompts include: i) handwritten prompts (Petroni et al., 2019), where users manually create intuitive templates based on human introspection, and ii) automatically searched prompts (Shin et al., 2020; Zhong et al., 2021; Gao et al., 2021; Qin and Eisner, 2021), where researchers search over the space of input tokens or embeddings for prompts that elicit correct predictions in the dev set. Although manually written prompts are interpretable, they are limited by the manual effort, and might not be optimal for eliciting correct predictions. The automated approaches (Shin et al., 2020; Zhong et al., 2021; Gao et al., 2021) can overcome the limitations of manual prompts by training a model, but they learn a universal prompt for each task (e.g., factual probing for one relation), regardless of different inputs. But such a setting may result in sub-optimal prompts. For example in factual probing, different subjects might have a different context when describing the same relation in an open-domain corpus. Similarly, for sentiment analysis, different query sentences might have different syntax or semantics.

We hypothesize that learning different prompts conditioned on inputs can benefit the overall masked filling accuracy in prompt learning. Towards that end, we propose a dynamic prompt generation model, named as `promptGen`, to automatically generate prompts based on inputs by leveraging the pre-trained generative model BART (Lewis et al., 2020). Generally, `PromptGen` consists of an encoder and an autoregressive decoder based on Transformer (Vaswani et al., 2017). We show the overall architecture of `PromptGen` applied on factual probing task in Figure 1. A knowledge fact is defined as a triplet: <sub, rel, obj>. The encoder produces a latent representation from input <sub, rel>, and the decoder autoregressively generates prompt in the form of [sub][D$_1$]...[MASK]..., [D$_{m+n}$]. Generated prompts are then passed to a fixed pretrained LM (e.g., BERT) to fill <MASK> as [obj]. A cross-entropy loss will be calculated based on the pre-

Findings of the Association for Computational Linguistics: NAACL 2022, pages 30 - 37
July 10-15, 2022 ©2022 Association for Computational Linguistics

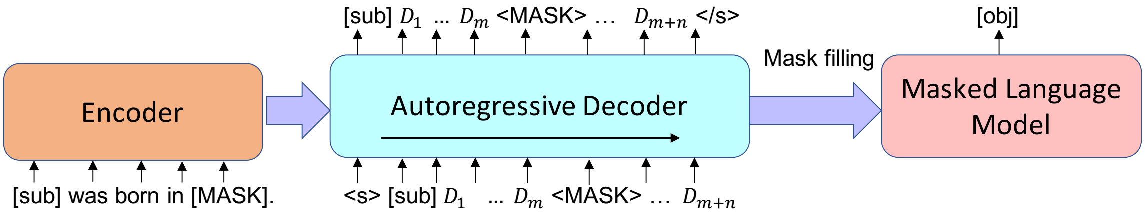
[sub] D_1 ... D_m <MASK> ... D_{m+n} </s>
[obj]
Mask filling
Encoder
Autoregressive Decoder
Masked Language Model
[sub] was born in [MASK].
<s> [sub] D_1 ... D_m <MASK> ... D_{m+n}

Figure 1: The overall architecture of our model `PromptGen`. `PromptGen` consists of an encoder and an autoregressive decoder. The downstream MLM is fixed and without fine-tuning. We will fine-tune the encoder and decoder to generate optimal prompts. Note that [sub] and <MASK> are directly copied in the decoding stage.

dicted [obj] v.s. ground-truth and backpropagated to update BART's weights. Compared to previous search models, although `PromptGen` has a higher computation cost, we find more appropriate and contextualized prompts, which is especially important for knowledge probing.

However, it is nontrivial to adopt a generative model for prompt generation. First, to make our model end-to-end trainable, at each decoding step, our decoder outputs a multinomial distribution over predefined vocabulary. Hence we finally get a sequence of distributions as our prompt, instead of a sequence of tokens. The token embedding of each $[\mathbf{D}_i]$ is a linear combination of the embedding of all tokens in the vocabulary. We then pass the generated prompts into an LM to fill the mask. Moreover, we should avoid any label information leaking from the pre-trained generative model. With pretraining, generative models can store related knowledge regarding input subjects, but we want to generate context-aware (i.e., <sub, rel>) prompts without leaking label information (i.e., object). Without any constraint, after generative model fine-tuning, the generated prompts could be trivial. For example, for input <Obama, place of birth>, the prompts could be "Obama...Hawaii...[MASK]...". It is trivial since it leaks the object label "Hawaii". To mitigate label leaking, we replace [sub] of a generated prompt with "None" and pass the perturbed prompt to LM. We pursue that the perturbed prompt cannot trigger the corresponding [obj] from a downstream MLM. Such a perturbation strategy was previously used for calibration (Zhao et al., 2021) and robustness improvement (Wang et al., 2021), and we are the first to use this strategy for the prompt generation.

Our contributions are as follows: i) We propose the first generative model based prompt generation method for knowledge probing. Meanwhile, we develop effective strategies to make the whole framework end-to-end trainable and avoid label leaking, ii) We evaluate our model on the factual probing benchmark LAMA (Petroni et al., 2019) and show that our model can significantly outperform other baselines. Detailed comparison and analysis justify our modeling choice.

2 Related Work

Factual Probing The factual probing setting was introduced by the LAMA benchmark (Petroni et al., 2019; Jiang et al., 2020; Shin et al., 2020), where given subject and relation, we want to infer the object by querying a pre-trained MLM. In contrast to previous knowledge graph completion models (Zhang et al., 2022b; Huang et al., 2019; Zhang et al., 2020; Liu et al., 2020; Yu et al., 2021) and information extraction models (Zhang et al., 2021, 2022a), where they need to fine-tune a pre-trained MLM. Here, we convert the knowledge graph completion task into a mask filling task, without MLM fine-tuning.

Pre-trained Generative Models. Our work is based on generative models, hence recent pre-trained generative models are related, including GPT-3 (Brown et al., 2020), BART (Lewis et al., 2020) and T5 (Raffel et al., 2020), all of which are capable of filling in missing spans in the input. Among all prompt search methods, Gao et al. (2021) is the most similar to ours since they used T5 to construct prompts. Compared with our work, Gao et al. (2021) uses T5 without fine-tuning, and they learn one prompt for all inputs. In our work, we learn dynamic prompts conditional on the given input and fine-tune on the generative model.

Instance-level Prompt Learning. Concurrently, couple instance-level prompt learning methods are developed, where given different query input, they utilize different prompts. Jin et al. (2022) learns instance-level prompts through calculating the relevance scores between token embedding in a universal prompt and token embedding in a given query, then the relevance scores are used to map the universal prompt into an instance-level prompt. IDPG (Wu et al., 2022) leans a light-weight generator to generate prompts, which are similar to our PromptGen. However, for downstream tasks, IDPG extracts the representation of [CLS] token to make the final predictions. So, IDPG has to fine-tune the pre-trained MLM, while we keep the downstream MLM frozen.

3 Methodology

We elaborate our method on the application of the LAMA task, in which the downstream MLM is BERT (Devlin et al., 2019). Our generative model adopts pre-trained BART (Lewis et al., 2020).

Given a subject s, relation r, a generated prompt $\mathcal{T}_{<r,s>}$, and an MLM, we can identify the word $\hat{o} \in \mathcal{V}$ to which the MLM assigns the highest probability of $P([\text{MASK}] = \hat{o}|\mathcal{T}_{<r,s>})$, where $\mathcal{T}_{<r,s>}$ represents the generated prompt conditional on relation r and subject s; $\mathcal{V}$ represents the predefined vocabulary. If the MLM can fill in the mask with the correct object, we conclude that the MLM encodes information about the fact. In this work, we will fine-tune BART using our novel approach to generate the optimal prompts.

3.1 Conditionally Generate Prompts

3.1.1 Input and Output Format

The input of our generative model is the manual prompt provided by the LAMA dataset. For instances: for relation "place of birth", our input is "[sub] was born in [MASK]"; for relation "occupation", our input is "[sub] is a [MASK] by profession". Here, [sub] will be replaced by a concrete subject name, e.g., "Obama", "Dante".

The prompt is generated from the decoder. Our prompt is in the following form:

$$\texttt{[sub]}\ [\mathrm{D}]_1\ [\mathrm{D}]_2 ... [\mathrm{D}]_\mathrm{m}\ \texttt{[MASK]}\ [\mathrm{D}]_{\mathrm{m+1}} ... [\mathrm{D}]_{\mathrm{m+n}}$$

where m is pre-defined maximal number of triggers between [sub] and [MASK]; n is the maximal number of triggers after [MASK]; each $[D]_i$ represents a multinomial distribution over vocabulary $\mathcal{V}_{\text{common}}$. Since the vocabulary of the generative model and the vocabulary of MLM could be different, we consider the intersection of their vocabularies, which is represented as $\mathcal{V}_{\text{common}}$.

3.1.2 Generating Procedure

Generative models usually are trained under the sequence-to-sequence framework. While, in our work, the target sequence (i.e., prompt) is unknown, our model will generate the optimal target sequence through exploration. Also, in the classic sequence-to-sequence framework, people consider the teacher forcing training strategy, where during training, the model uses the ground truth as decoder input. Since we have no ground truth target sequence, at each decoding step, we use the model output from a prior time as the current input.

At each decoding step t, our decoder computes the current hidden state h_t and current token distribution D_t, based on the current sequence $[\mathrm{D}_1], ..., [\mathrm{D}_{t-1}]$, and the encoding output h_{encode}:

$$h_{\text{encode}} = \texttt{Encoder}(s, r)$$
$$h_t = \texttt{Decoder}(h_{\text{encode}}, [\mathrm{D}_1], ..., [\mathrm{D}_{t-1}])$$
$$\mathrm{D}_t = \mathrm{Softmax}(h_t)$$

where, Encoder and Decoder both adopt Transformer architecture; h_{encode} only needs to be computed once for each input <s, r>; h_t and D_t are calculated recursively from Decoder. In the below section, we will elaborate how to compute word embedding for sequence of distributions $[\mathrm{D}_1], ..., [\mathrm{D}_{m+n}]$ in Transformer Decoder.

Assuming the BART word embedding matrix for tokens in vocabulary $\mathcal{V}_{\text{common}}$ is $\mathcal{E}_V \in \mathcal{R}^{|V| \times d}$, we know that each $[\mathrm{D}_i]$ is a multinomial distribution on $\mathcal{V}_{\text{common}}$, so the embedding vector $\mathcal{E}_{D_i}$ for each $[\mathrm{D}_i]$ is a linear combination on $\mathcal{E}_V$:

$$\mathcal{E}_{D_i} = \mathrm{D}_i^T * \mathcal{E}_V \tag{1}$$

Encoding position embedding for $[\mathrm{D}_i]$ is straightforward, depending on its position in a sequence.

During generating, assuming the current output is D_i, where $i \in [1, m]$, if the highest possibility token is </s> or the sequence reaches the maximal number m, we stop current generation, and start generating $[\mathrm{D}_{m+1}]...[\mathrm{D}_{m+n}]$. The same is for generating $[\mathrm{D}_i]$, where $i \in [m+1, m+n]$.

3.2 Optimization

The generated prompt $\mathcal{T}_{<s,r>}$ is passed forward to a downstream MLM. Following the convention of BERT, we add special tokens [CLS] (or <s>), [SEP] (or </s>) at the first and the last position of the prompt, separately. The calculation of word embedding of $[D_i]$ in the downstream MLM is the same as Equation (1), where $\mathcal{E}_V$ will be from the MLM.

The downstream MLM can be viewed as a black-box, and it is used as a critic to evaluate the quality of our generated prompts. We fine-tune the parameters of the generative model to minimize the negative log-likelihood of a training set $\Pi = \{<s, r, o>\}$:

$$\mathcal{L}_{\Pi} = -\frac{1}{|\Pi|} \sum_{<s,r,o>\in\Pi} \log P([\text{MASK}] = o|\mathcal{T}_{<r,s>}),$$

where we use all the training data from different relations together to train our model.

3.2.1 Label Information Leaking Constraint

The pre-trained generative model has the ability to store open-domain knowledge during pre-training. Without any constraint, the generated prompts could be trivial and leak the label information.

To avoid label leaking, we develop a novel constraint. We replace the [sub] of $\mathcal{T}_{<r,s>}$ with "None", and get a perturbed prompt $\mathcal{T}^{(\text{None})}_{<r,s>}$. We argue that for a non-trivial $\mathcal{T}_{<r,s>}$, its corresponding $\mathcal{T}^{(\text{None})}_{<r,s>}$ has no ability to trigger the correct [obj] from the downstream MLM, since $\mathcal{T}^{(\text{None})}_{<r,s>}$ is a context-free input. For example, assuming we pass "None was born in [MASK]" into an MLM, the possibility of filling the mask with "Hawaii" will be low without knowing the subject of "Obama". We define the second objective function as:

$$\mathcal{L}_{\text{perturb}} = \frac{1}{|\Pi|} \sum_{<s,r,o>\in\Pi} \log P([\text{MASK}] = o|\mathcal{T}^{(\text{None})}_{<r,s>}),$$

through which the log-likelihood of training set is minimized. Finally, the overall objective function becomes $\mathcal{L} = \mathcal{L}_{\Pi} + \alpha * \mathcal{L}_{\text{perturb}}$, where $\alpha \geq 0$ is a hyper-parameter.

4 Experiments

4.1 Experimental setup

Following the same setting of Shin et al. (2020); Zhong et al. (2021), we use the original test set, and the training LAMA dataset contains 1000 facts for each of the 41 relations from T-REx dataset (ElSahar et al., 2018) and Wikidata. Refer to Appendix for implementation details.

We compare our model with the following baselines: 1) manually created prompts (Petroni et al., 2019). 2) LPAQA (Jiang et al., 2020). 3) Gao et al. (2021) [1]. 4) AutoPrompt (Shin et al., 2020), where "* [T]s" means using * token triggers. 5) OptiPrompt (Zhong et al., 2021), where "* [V]s" means using * vector triggers; "manual" means using manually designed prompts as initialization.

4.2 Results

For all our models, we set m=10, n=5. Our results are in Table 1. The LAMA results are broken down by relation category. Relations from each category can refer to Table 4 in Appendix. Overall, `PromptGen` outperforms the previously reported results in terms of top-1 accuracy on the LAMA benchmark. The improvement is consistent across all categories, except for the "1-1" category, which contains two relations, "capital" and its inverse "capital of". We see that the best result in this category is the manual prompt. The intuitive explanation behind this is that the variety of natural language expressions about "capital of" in open-domain knowledge is low, so it's hard for our model outperforms manually designed prompts.

The detailed results on each relation are in Table 4 in the Appendix.

Method	1-1	N-1	N-M	All
Manual	**68.0**	32.4	24.7	31.1
LPAQA	65.0	35.9	27.9	34.1
Gao et al. (2021)	22.5	12.7	8.5	11.4
AutoPrompt (5 [T]s)	58.0	46.5	34.0	42.2
OptiPrompt (5 [V]s)	49.6	53.1	39.4	47.6
OptiPrompt (10 [V]s)	60.7	53.2	39.2	48.1
OptiPrompt (manual)	59.6	54.1	40.1	48.6
Ours ($\alpha = 0.3$)	54.8	**55.3**	**44.0**	**51.0**

Table 1: Micro-averaged results (top-1 accuracy in %) on the LAMA benchmark using the BERT-base-cased model, averaged over relations.

4.2.1 Hyper-parameter Analysis

In this section, we analyze the effect of hyper-parameter α. We set α equals to 0.0, 0.2, 0.3 and 0.4, separately, and the results of variants are reported in Table 2. The best result comes from $\alpha = 0.3$. Although $\alpha = 0.0$ gives us the second best result, we find that when we replace the [sub] in generated prompts into 'None", the top-1 accuracy is still 48.1, which proves that without label information leaking constraint ($\alpha = 0.0$), the generated prompts are trivial. For $\alpha = 0.2, 0.3, 0.4$, their top-1 accuracy using perturbed prompts all equals to 0, which proves the effectiveness of our label information leaking constraint.

[1] We generate one prompt for each relation using T5, given input in the form of "[sub] [extra_id_0] [obj] [extra_id_1]", where [sub] and [obj] are from training set. The filling result of [extra_id_0] and [extra_id_1] will be used as final prompt.

Method	1-1	N-1	N-M	All	"None"
$\alpha = 0.0$	53.9	53.9	43.1	49.7	48.1
$\alpha = 0.2$	53.4	53.5	43.3	49.6	0.0
$\alpha = 0.3$	**54.8**	**55.3**	**44.0**	**51.0**	0.0
$\alpha = 0.4$	39.4	49.3	38.4	44.9	0.0

Table 2: Results of Variants on the LAMA benchmark.

4.2.2 Case Study of Generated Prompts

We show two case studies on relation "instrument" in Table 3 comparing with AutoPrompt, which used a fixed prompt for one relation regardless of input. We report the generated prompts by choosing the highest probability token for each D_i, and the top-1 predictions from BERT. We highlight the [sub] in blue, and wrong predictions in red.

Method	Generated prompt	top-1
AutoPro	Joe Pass playingdrum concertoative electric [MASK].	piano
Ours	Joe Pass and not violin yeah much like majority depending Resources [MASK].	guitar
AutoPro	Marco Benevento playingdrum concertoative electric [MASK].	piano
Ours	Marco Benevento and not violin yeah much like trafficking UNESCO partly [MASK].	piano

Table 3: Case Study on relation "instrument".

We find that AutoPrompt always triggers the MLM to predict the majority label "piano", regardless of the subject. Through dynamic prompts, we bypass this issue.

5 Conclusion

In this work, we propose PromptGen for knowledge probing, which can automatically generate prompts conditional on the given query (i.e., subject, relation). Our PromptGen leverages a pre-trained generative model, e.g., BART. PromptGen is end-to-end trainable, where we fine-tune the parameters of the generative model, while keeping the downstream pre-trained MLM frozen. We evaluate PromptGen on the benchmark LAMA dataset. We observe the significant improvement of the performance on the downstream MLM by finding more appropriate dynamic prompts without label information leaking.

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 *Advances in Neural Information Processing Systems (NeurIPS)*, virtual.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 1371–1382, Virtual Event.

Xiang Chen, Xin Xie, Ningyu Zhang, Jiahuan Yan, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. AdaPrompt: Adaptive prompt-based finetuning for relation extraction. *arXiv preprint arXiv:2104.07650*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186, Minneapolis, MN.

Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, Miyazaki, Japan.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, pages 3816–3830, Virtual Event.

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*.

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the Twelfth ACM*

International Conference on Web Search and Data Mining (WSDM), pages 105–113, Melbourne, Australia.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. 2022. Instance-aware prompt learning for language understanding and generation. *arXiv preprint arXiv:2201.07126*.

Nora Kassner, Philipp Dufter, and Hinrich Schütze. 2021. Multilingual LAMA: investigating knowledge in multilingual pretrained language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 3250–3258, Online.

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 the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7871–7880, Online.

Guiliang Liu, Xu Li, Jiakang Wang, Mingming Sun, and Ping Li. 2020. Extracting knowledge from web text with monte carlo tree search. In *Proceedings of the Web Conference (WWW)*, pages 2585–2591, Taipei.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 5203–5212, Online.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Online.

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 the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, Long Beach, CA.

Zhuoyi Wang, Yu Lin, Yifan Li, Feng Mi, Zachary Tian, Latifur Khan, and Bhavani M. Thuraisingham. 2021. Unsupervised perturbation based self-supervised adversarial training. In *Proceedings of the 7th IEEE International Conference on Big Data Security on Cloud (BigDataSecurity)*, pages 20–25, New York City, NY.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, VG Vydiswaran, and Hao Ma. 2022. IDPG: An instance-dependent prompt generation method. *arXiv preprint arXiv:2204.04497*.

Jinxing Yu, Yunfeng Cai, Mingming Sun, and Ping Li. 2021. Mquade: a unified model for knowledge fact embedding. In *Proceedings of the Web Conference (WWW)*, pages 3442–3452, Virtual Event / Ljubljana, Slovenia.

Jingyuan Zhang, Mingming Sun, Yue Feng, and Ping Li. 2020. Learning interpretable relationships between entities, relations and concepts via bayesian structure learning on open domain facts. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8045–8056, Online.

Yue Zhang, Hongliang Fei, and Ping Li. 2021. ReadsRE: Retrieval-augmented distantly supervised relation extraction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 2257–2262, Virtual Event, Canada.

Yue Zhang, Hongliang Fei, and Ping Li. 2022a. End-to-end distantly information extraction with retrieval augmentation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Yue Zhang, Mingming Sun, Jingyuan Zhang, and Ping Li. 2022b. Explainable concept graph completion by bridging open-domain relations and concepts. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, Alexandria, VT.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 12697–12706, Virtual Event.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 5017–5033, Online.

A Appendix

Relation	Type	Name	Maunal	LPAQA	AutoPro	OptiPro	Ours
P1376	1-1	capital of	**73.8**	67.8	56.2	56.7	61.6
P36	1-1	capital	**62.1**	**62.1**	59.7	61.3	52.2
P103	N-1	native language	72.2	72.2	79.7	86.8	**86.9**
P127	N-1	owned by	34.8	32.5	44.3	49.6	**54.0**
P131	N-1	located in the admin. territorial entity	23.3	22.8	28.9	**41.4**	40.3
P136	N-1	genre	0.8	16.8	55.3	63.6	**68.4**
P138	N-1	named after	61.4	59.5	70.7	73.4	**76.1**
P140	N-1	religion	0.6	59.8	60.5	76.5	**80.9**
P159	N-1	headquarters location	32.4	35.6	35.7	37.4	**37.6**
P17	N-1	country	31.3	39.8	51.0	**57.8**	54.2
P176	N-1	manufacturer	85.5	81.5	87.5	87.3	**91.6**
P19	N-1	place of birth	21.1	21.1	19.5	20.6	**22.8**
P20	N-1	place of death	27.9	27.9	29.8	33.8	**35.8**
P264	N-1	record label	9.6	6.3	4.2	**45.5**	5.6
P276	N-1	location	41.5	41.5	43.0	**47.1**	46.5
P279	N-1	subclass of	30.7	14.7	54.9	64.7	**65.6**
P30	N-1	continent	25.4	16.9	78.6	86.3	**89.1**
P361	N-1	part of	23.6	31.4	37.0	**46.4**	41.1
P364	N-1	original language of film or TV show	44.5	43.9	45.0	51.3	**54.6**
P37	N-1	official language	54.6	56.8	52.7	58.6	**62.9**
P407	N-1	language of work or name	64.2	65.2	68.4	**71.0**	68.2
P413	N-1	position played on team / speciality	0.5	23.7	41.7	44.0	**51.5**
P449	N-1	original network	20.9	9.1	33.1	36.0	**39.8**
P495	N-1	country of origin	28.7	32.2	35.8	**40.8**	37.7
P740	N-1	location of formation	8.9	13.7	13.1	15.0	**17.3**
P1001	N-M	applies to jurisdiction	70.5	72.8	80.5	85.2	**87.0**
P101	N-M	field of work	9.9	5.3	12.1	14.1	**19.4**
P106	N-M	occupation	0.6	0.0	13.6	**35.7**	31.3
P108	N-M	employer	6.8	5.7	7.8	11.2	**12.5**
P1303	N-M	instrument	7.6	18.0	23.1	23.6	**45.8**
P1412	N-M	languages spoken, written or signed	65.0	64.7	71.5	76.1	**77.1**
P178	N-M	developer	62.9	59.4	64.3	67.9	**68.6**
P190	N-M	twinned administrative body	2.2	1.7	2.4	3.1	**3.9**
P27	N-M	country of citizenship	0.0	41.5	45.8	**47.1**	46.5
P31	N-M	instance of	36.7	36.7	53.6	64.9	**68.9**
P39	N-M	position held	8.0	16.1	27.2	42.8	**69.6**
P463	N-M	member of	67.1	57.3	64.0	64.0	**73.8**
P47	N-M	shares border with	13.7	13.7	19.2	**22.2**	21.2
P527	N-M	has part	11.2	10.6	22.1	34.8	**38.7**
P530	N-M	diplomatic relation	2.8	**3.9**	2.8	3.3	2.8
P937	N-M	work location	29.8	39.1	34.4	43.3	**48.2**

Table 4: The accuracy of different prompts on LAMA for each relation using BERT-base-cased.

A.1 Implementation Details

We adopt "BART-large" as our generative module and "BERT-base-cased" as our MLM module, both of which are collected from Huggingface website[2]. We use the Adam optimizer with learning rate $5e-5$, set warm-up ratio to 0.1, and weight decay to 1e-3. We repeat our experiments five times and report the average metrics on the test set.

A.2 Detailed Results

Table 4 shows the per-relation accuracy for each prompting method. We see that our method achieves the best performance for most cases.

[2] https://huggingface.co/models