Enhance Performance of Ad-hoc Search via Prompt Learning

Shenghao Yang, Yiqun Liu(✉), Xiaohui Xie, Min Zhang, and Shaoping Ma

Department of Computer Science and Technology, Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China
ysh21@mails.tsinghua.edu.cn, {yiqunliu,z-m,msp}@tsinghua.edu.cn, xiexiaohui@mail.tsinghua.edu.cn

Abstract. Recently, pre-trained language models (PTM) have achieved great success on ad hoc search. However, the performance decline in low-resource scenarios demonstrates the capability of PTM has not been inspired fully. As a novel paradigm to apply PTM to downstream tasks, prompt learning is a feasible scheme to boost PTM's performance by aligning the pre-training task and downstream task. This paper investigates the effectiveness of the standard prompt learning paradigm on the ad hoc search task. Based on various PTMs, two types of prompts are tailored for the ad hoc search task. Overall experimental results on the MS Marco dataset show the credible better performance of our prompt learning method than fine-tuning based methods and another previous prompt learning based model. Experiments conducted in various resource scenarios show the stability of prompt learning. RoBERTa and T5 deliver better results compared to BM25 using 100 training queries utilizing prompt learning, while fine-tuning based methods need more data. Further analysis shows the significance of the uniformity of tasks' format and adding continuous tokens into training in our prompt learning method.

Keywords: Ad hoc search · Prompt learning · Pre-trained language model

1 Introduction

In recent years, large-scale pre-trained language models (PTM), e.g. BERT [1], have boosted numerous downstream tasks' performance. In practice, the "pre-train and fine-tuning" paradigm is widely adopted. For ad hoc search, especially in the second stage (i.e. re-ranking), the query and document are usually concatenated and fed into the encoder of PTM in fine-tuning [2]. The credible relevance score is obtained utilizing the efficient self-attention mechanism. However, it has been pointed out that this paradigm cannot fully inspire the capabilities of PTM. The main reason is the gap in the form of pre-training and downstream tasks [3]. We argue that this problem also limited the performance of PTM on the ad hoc search task. Especially in the low-resource scenarios, the performance decline obviously [4].

Y. Chang and X. Zhu (Eds.): CCIR 2022, LNCS 13819, pp. 28–39, 2023.
https://doi.org/10.1007/978-3-031-24755-2_3

To tackle this problem, a potentially feasible method is adopting a novel paradigm, namely prompt learning, on the ad hoc search task instead of the "pre-train and fine-tuning" paradigm. This new paradigm is describe as "pre-train, prompt, and predict" [3]. As distinguished from fine-tuning, it is called prompt-tuning. It boosts the performance of PTM by aligning the form of pre-training and fine-tuning. For example, the Masked language model (MLM) is a popular pre-training task and aims to accomplish a cloze style task. Intuitively, converting the form of the downstream task into cloze style can contribute to the performance. Utilizing the powerful natural language understanding ability of MLM, cloze style prompt-tuning is commonly adopted in many tasks and has achieved convincing performance [5–8].

Research on prompt learning for the ad hoc search task is not sufficient. MonoT5 [9] converts the re-ranking task into a text generation task and utilizes a Seq2Seq model (i.e. T5 [10]) to perform task. P^3 ranker [11] follows monoT5 and proposes a pre-fine-tuning scheme to boost performance. It first conducts prompt-tuning on a natural language inference (NLI) dataset MNLI [12] and further experiments on ad hoc search datasets.

Compared with existing research, we adopt the standard prompt learning (i.e. "pre-train, prompt, and predict" paradigm) without additional training data and training stage. We investigate the prompt-tuning performance of lightweight models (i.e. BERT and RoBERTa [13]). In the experiments based on a popular ad hoc search benchmark dataset (i.e. MS Marco [14]), RoBERTa with prompt-tuning achieves better performance compared to the BERT and RoBERTa with fine-tuning, especially in the low-resource setting. The experimental results demonstrate the effectiveness of prompt learning and it is more convincing since the uniformity of model architecture. We further apply our method to T5 and achieve further performance improvement.

In summary, the contributions of our work are as follows:

- We investigate the effectiveness of the standard prompt learning paradigm on the ad hoc search task, especially in low-resource scenarios.
- We tailor the prompt-tuning method for the ad hoc search task and summarize the performance of using various PTMs as the backbone and utilizing various prompt types.
- Experimental results on MS Marco demonstrate the effectiveness of our prompt learning method regardless in low-resource and full-resource scenarios. Further analysis demonstrates that aligning the task format and adding the continuous tokens contribute to our method's performance.

2 Related Work

2.1 Ad Hoc Search with PTM

Utilizing the outstanding natural language model ability of PTM, the performance of ad hoc search has gained significant improvements. In the retrieval stage of ad hoc search, differing from some sparse retrieval models, e.g. BM25 [15],

DPR [16] applies PTM to obtain the dense representations of query and document. The relevance score is the dot product between the representations of query and document. Then the approximate nearest neighbor (ANN) algorithm is adopted to perform retrieval. The following researches [17,18] mainly focus on the hard negative sampling strategy based on the framework of DPR. In the re-ranking stage of ad hoc search, [2] proposed to obtain the relevance score with a vanilla BERT attached with a MLP head. It concatenates query and candidate passage retrieved in the first stage with a special token (i.e. [SEP] token) as the input of BERT. This scheme is widely adopted in the following researches [19–21]. However, it has been witnessed an obviously performance decline when training data is insufficient [4]. In this paper, we focus on alleviating the performance decline in the re-ranking stage. We speculate the aforementioned scheme may not attain the best potential performance and we are concerned with a different scheme, called prompt learning.

2.2 Prompt Learning

Prompt learning has achieved remarkable development and it has been extensive study and applications in recent years. In GPT3 [6], tasks such as translation can be accomplished by adding contextual prompts. This scheme accomplishes the aligning of the pre-training task and downstream task and achieves performance boosting.

At the early development phase of prompt learning, prompts were generally designed manually [5,6]. However, subtle differences in the manual design will cause a significant impact on performance [7]. In that regard, many automatic approaches have been proposed to find suitable prompts. For prompt in natural language format, suitable template words are mined in the discrete space [22]. Further, some approaches [7,8,23,24] get rid of the natural language format and adopt the continuous format, i.e., not using a specific word but embedding. The advantage of using continuous prompt is that the embedding can participate in the training as a trainable parameter.

Prompt learning has been applied in various NLP tasks, yet the research on ad hoc search is not enough. P^3 ranker [11] first brought the concept of prompt learning into ad hoc search, it follows the prompt of monoT5 [9] and adds a pre-fine-tuning stage to warm up the training process with MNLI [12] dataset. MNLI is a sentence pair classification task that is similar to the ad hoc search task. Thus, the pre-fine-tuning stage contributes to the performance. Above works all conduct prompt learning with discrete prompts. In this paper, we follow the standard prompt learning scheme and conduct a comprehensive experiment on both discrete and continuous prompts.

3 Preliminary

3.1 Ad hoc Search

In this section, we will briefly introduce the basic knowledge of the ad hoc search. Generally, the ad hoc search task is performed in two stages. Given queries,

candidate documents are obtained using a retrieval system (e.g. BM25 [15]) in the first stage. The second stage, called re-ranking, calculates the relevance score of the query to each candidate document using more sophisticated methods (e.g. learning to rank algorithms and neural ranking models).

In this paper, we focus on the re-ranking task. It aims to build a ranking function f_r which can measure the relevance score $s_{i,j}$ for each query q_i and all its candidate documents $d_{i,j}$. The optimal ranking function f_r^* is obtained by minimizing the ranking loss L with supervised learning, which can be formalized as:

$$f_r^* = argmin\ L(f_r; Q, D, Y), \tag{1}$$

where Q, D, Y is the set of queries, candidate documents, and relevance labels, respectively. In the experiments of this paper, the ranking loss is chosen as MarginRankingLoss:

$$L(f_r; Q, D, Y) = \sum_{q_i \in Q} \sum_{d_j, d_k \in D} max(0, -y_{i,j,k}(r_{i,j,k} + m)) \tag{2}$$

where $r_{i,j,k} = f_r(q_i, d_j) - f_r(q_i, d_k)$, and $f_r(q, d)$ represents the relevance score calculated by ranking function f_r. Given q_i, $\begin{cases} y_{i,j,k} = 1, d_{i,j} \succ d_{i,k} \\ y_{i,j,k} = -1, d_{i,k} \succ d_{i,j} \end{cases}$, $\succ$ represents more relevant, m is set to 1.

3.2 Prompt Learning

In this section, we will briefly introduce the basic concepts of prompt learning. We will take the sentiment analysis task as an example.

Template. Consider a sentence "$[X]$, it is ___". This sentence with a slot is called the template in prompt learning. $[X]$ is a placeholder for input text. We formally defined a template function f_t. It outputs $x^{'}$ after inserting the input x into the template, i.e. $x^{'} = f_t(x)$. $x^{'}$ will be fed into PTM to obtain the probability of the token at the slot position.

Verbalizer. For the sentence above, if we define the word "good" corresponds to the positive label, the prediction probability of "good" at the slot position can be regarded as the positive probability. The word is called a label word. The process of mapping words to labels is called verbalizer and can be defined as a function f_v. Given a PTM, the probability of label word w_i is formalized as $P(s = w_i | x^{'}, \theta_m)$, where θ_m is the parameters of the PTM and s is the token at slot position. In summary, given the input x, we obtain the probability of the label y by Eq. 3.

$$P(y|x; \theta_m) = \frac{1}{N_y} \sum_{w_i \in f_v(y)} P(s = w_i | f_t(x); \theta_m), \tag{3}$$

where $f_v(y)$ mapping label y to label words and N_y is the number of label words of label y.

4 Methodology

In this section, we will present our method to apply prompt learning to the re-ranking task. For this task, the input x contains two parts, i.e. query and document. Thus, two placeholders need to be set in the template, namely [q] and [d] corresponding to the query and document, respectively. The slot position is represented with a special token [mask]. For the verbalizer, we assign one label word per label. Specifically, the prompt tailored for the re-ranking task in various PTM settings is shown in Table 1. We adopt these prompts after a few trials. In other words, these prompts achieve better performance in the experiment.

Table 1. Templates and verbalizers tailored for re-ranking task

PTM	Template	Verbalizer (pos/neg)
BERT/RoBERTa	[q] and [d] are [mask]	Relevant/irrelevant
	[q] [soft]×3 [mask] [soft]×3 [d]	Yes/but
T5	Query: [q] Document: [d] Relevant: [mask]	True/false
	[soft] [q] [soft] [d] [soft] [mask]	

We summarize our method into two type: *hard prompt* and *soft prompt*. Specifically, *hard prompt* is designed entirely in natural language and *soft prompt* may contain continuous embedding. In the training of *hard prompt* based method, only the PTM's parameters need to be optimized. The label probability is predicted using Eq. 3. This process is shown as Fig. 1(a).

For *soft prompt*, the template contains [soft] placeholder as shown in Table 1. It will be initialized with word embedding and optimized as parameters in training. For BERT/RoBERTa, the word embedding is initialized randomly. For T5, the three [soft] placeholder will be initialized by "Query:", "Document:" and "Relevant:", respectively. Specifically, the raw token is fed to the word embedding layer of the PTM, while the soft token is fed to a custom soft embedding layer. This scheme is followed by [7]. The outputs of both embedding layers are then combined and fed to the following layers.

In the verbalizer process of *soft prompt*, we follow the setting in [24]. Unlike utilizing the probability of label words, the last hidden state of the [mask] token is fed into a fully-connected (FC) layer to obtain label probability. The weight of the FC layer is initialized with label words' embedding. The framework of *soft prompt* is shown in Fig. 1(b). We set two separate optimizers for the soft embedding layer and the FC layer.

In the case of *soft prompt*, Eq. 3 will be take the form of Eq. 4:

$$P(y|x;\theta_m,\theta_v,\theta_t) = \frac{exp\theta_v^y f_m(f_t(x);\theta_m,\theta_t)}{\sum_{i\in C} exp\theta_v^{y_i} f_m(f_t(x);\theta_m,\theta_t)}, \tag{4}$$

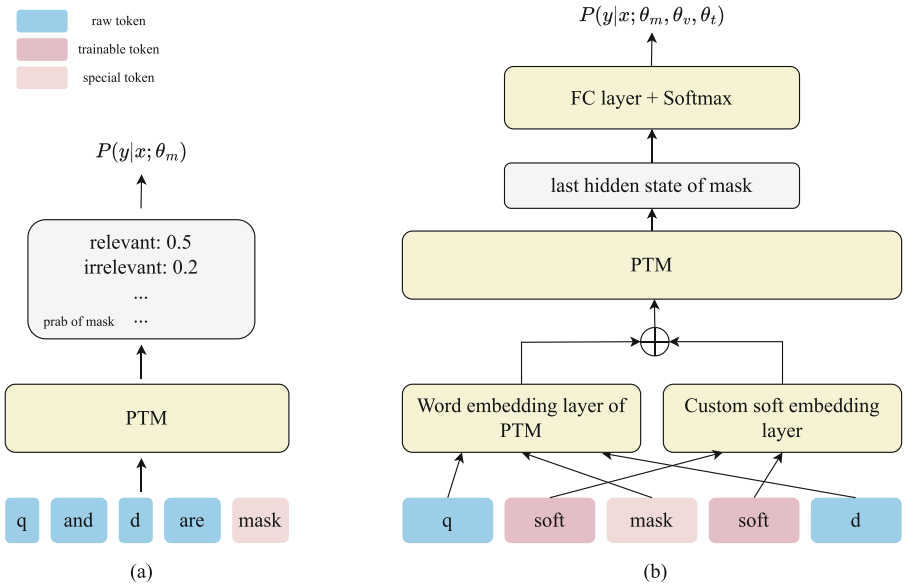

Fig. 1. The prompt learning framework tailored for the re-ranking task: (a) *hard prompt*, (b) *soft prompt*. Note that this figure shows the template and verbalizer of BERT/RoBERTa. The framework with T5 is broadly consistent.

where $f_m(\cdot)$ is the last hidden state of [mask] token output by PTM, θ_v^y is the label word's embedding corresponding to label y, and C is the number of labels. Compared to *hard prompt*, *soft prompt* put template and verbalizer into training parameters, which benefit to alleviate the bottleneck of manual design.

Finally, to obtain the relevance scores of the query-document pair in the input x, we subtract the probability of negative label from the probability of positive label, i.e. $f_r(x) = P(y = 1|x;\theta) - P(y = 0|x;\theta)$, where θ represents all possible parameters. $f_r(x)$ can be involved in the calculation of the ranking loss as $f_r(q, d)$ in Eq. 2 and optimized by Eq. 1.

5 Experiments

In this section, we will introduce the experimental settings and results in detail.

5.1 Dataset and Metric

We experiment on the MS MARCO passage ranking dataset [14]. It contains approximately 530 k queries in the training set and 6,980 queries in the development set. Each query has one relevant document on average.

To investigate the performance of prompt learning in various resource scenarios, we sample queries to construct the training sets. Specifically, the training

Table 2. Overall performance of our prompt learning method on MS MARCO passage ranking dataset. †‡ represent significant performance improvement using pairwise t-test with $p < 0.05$ than BERT with fine-tuning and RoBERTa with fine-tuning, respectively.

Model	50	100	500	1 k	5 k	10 k	530 k
BM25	.1874						
Fine-tuning based models							
BERT	**.1462**‡	.1677‡	.2125‡	.2309	.2760	.2792	.3109
RoBERTa	.0322	.0855	.2031	.2524†	.2874†	.2831	.3248†
Prompt-tuning based models							
BERT (hard)	.0316	.0372	.1514	.2208	.2733	.2776	.3038
BERT (soft)	.0454‡	.1087‡	.2203†‡	.2283	.2726	.2826	.3060
RoBERTa (hard)	.0408‡	.0511	.2404†‡	.2581†	.2925†	.2968†‡	.3133†
RoBERTa (soft)	.1121‡	.1936†‡	.2336†‡	.2617†‡	.2932†‡	**.3028**†‡	.3182†
T5 (hard)	.1121‡	**.2142**†‡	**.2730**†‡	**.2833**†‡	.3029†‡	.3000†‡	**.3334**†‡
T5 (soft)	.0269	.1400‡	.2650†‡	.2763†‡	**.3038**†‡	.2997†‡	.3219†

sets contain 50/100/500/1 k/5 k/10 k/530 k (all) queries, respectively. We don't construct training sets with too few queries (e.g. 5/10 queries) considering the performance will depend heavily on the sample queries and more randomness. To conduct pairwise training, each query in the training set is paired with a relevant document and an irrelevant document. The irrelevant document is sampled from hard negative documents retrieved by BM25. For each training set, we sample queries from the development set as the validation set. In the 50/100/500 scenarios, we sample equally queries in the validation set, and we all sample 500 queries in 1 k/5 k/10 k/530 k scenarios.

When we evaluate our method on the validation set and development set, we first retrieved the top 100 documents for each query using BM25 and then re-rank the candidate documents. The evaluation metric is MRR@10.

5.2 Experimental Setup

We take three types of models as the baseline in our experiment. A lexical model, BM25. Two fine-tuning based models, BERT and RoBERTa with fine-tuning. A prompt-tuning based model, P^3 ranker [11].

Our experimental framework is implemented based on Openprompt [25] and transformer [26] library. We chose BERT/RoBERTa/T5 as the backbone of our models. We adopt the base version of these models. The prompts tailored for these models have been presented in Sect. 4.

In the experiment, we train our models for 100 epochs and set the batch size to 10 and gradient accumulation steps to 2. The max length of the input is set to 256. We use ADAM [22] with the initial learning rate set to 3e-5, β1=0.9, β2=0.999, epsilon=1e-8, L2 weight decay of 0.01, learning rate warmup over the first 10% steps, and linear decay of the learning rate.

Table 3. Performance comparison between our prompt learning method and P^{3}ranker [11] on MS MARCO Passage Ranking.

Model	50	1 k	530 k
P3 ranker	0.0949	0.2027	0.3311
BERT (hard)	0.0316	0.2208	0.3038
BERT (soft)	0.0454	0.2283	0.3060
RoBERTa (hard)	0.0408	0.2581	0.3133
RoBERTa (soft)	0.1120	0.2617	0.3182
T5 (hard)	**0.1121**	**0.2833**	**0.3334**
T5 (soft)	0.0269	0.2763	0.3219

5.3 Result and Analysis

The overall performance of our prompt learning method is shown in Table 2. From the results, it is clear that the prompt-tuning models outperform fine-tuning and lexical methods except in the 50 queries scenario. We will analyze this exception in 5.3.1. T5 and RoBERTa with prompt-tuning achieve better results than BM25 using 100 training queries while fine-tuning method needs more training queries. As shown in Table 3, compared to another prompt learning based method, i.e. P^3 ranker, superior results are seen when using our method.

5.3.1 Prompt-Tuning vs Fine-Tuning

From experimental results, we find that T5 with prompt-tuning can achieve overall better performance than fine-tuning methods. This is in line with our expectations considering T5 has more parameters and a more sophisticated pre-training scheme. One exceptional case is found in the 50 queries setting. The BERT with fine-tuning performs well in that case.

We will explain the experimental results of four methods with the same model architecture from the perspective of format aligning. Table 4 compares the format of BERT and RoBERTa in various tasks. Combined with the experimental results, we have several findings:

- BERT with prompt-tuning underperforms BERT with fine-tuning. It can be seen in Fig. 2(b). We speculate the reason is that BERT retains the next sentence prediction (NSP) task in pre-training. Considering the NSP task and re-ranking task are both text pair classification tasks and take the same input format. That can be seen as a uniformity between the pre-training task and downstream task and contribute to the performance. This may also be the reason why BERT with fine-tuning performs well in the 50 queries setting.
- However, The good trend of BERT with fine-tuning is not maintained when the number of training queries increases while RoBERTa's performance increases rapidly with more training queries. It can be shown in Fig. 2(a). The result is reasonable since RoBERTa is an optimal version of BERT.

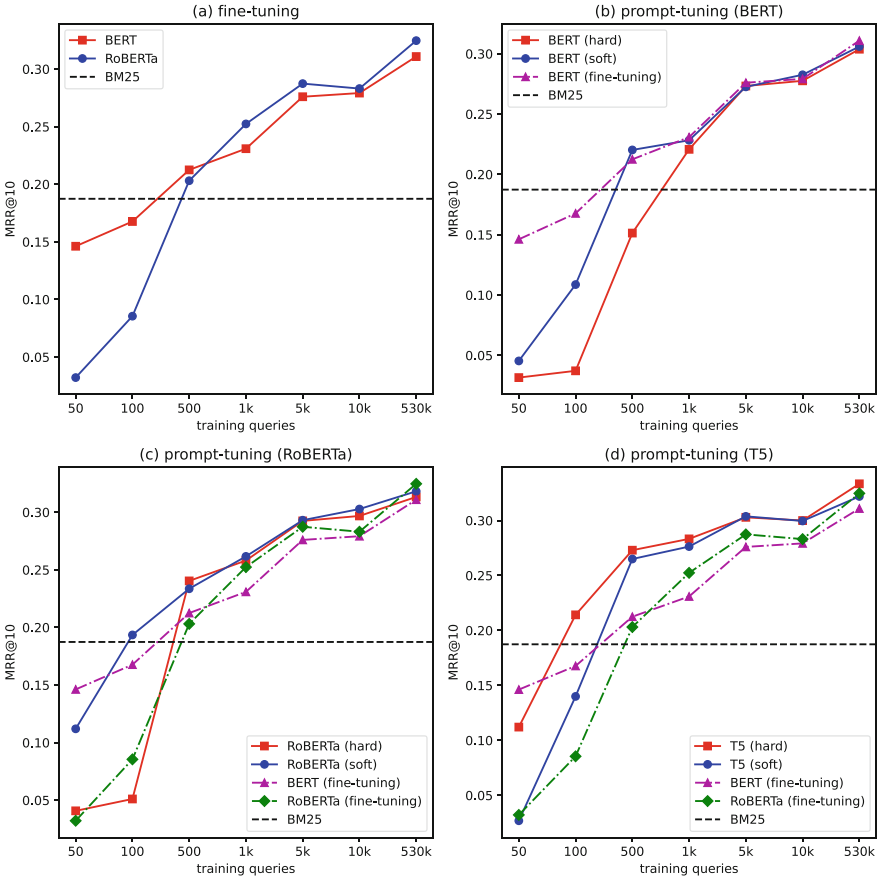

Fig. 2. The performance with various PTM backbones and various prompts in each resource scenario.

- RoBERTa with prompt-tuning gives clearly better results than BERT with prompt-tuning. This demonstrates that RoBERTa is more appropriate for the cloze style prompt learning method since RoBERTa only retains the MLM task in pre-training and conducts more training iterations. Superior results are seen for RoBERTa with prompt-tuning compared with two fine-tuning methods. It can be seen in Fig. 2(c). Specifically, prompt-tuning boosts the performance of RoBERTa in very low-resource scenarios and this demonstrates the advantage of prompt learning to inspire the capacity of PTM.

5.3.2 Soft Prompt vs Hard Prompt

In our prompt learning method, each PTM includes two types of prompts (detail in Sect. 4). For BERT/RoBERTa, they deliver significantly better results due to soft prompt in low-resource scenarios (i.e. 50/100 setting). In particular,

Table 4. Comparison of the format in different types of tasks using BERT and RoBERTa.

Task	PTM	Task type	Input format
MLM	BERT, RoBERTa	Pre-training	$[t_1]$ $[t_2]$ ... $[mask]$ ... $[t_n]$
NSP	BERT	Pre-training	$[s_1]$ $[sep]$ $[s_2]$
Re-ranking	BERT, RoBERTa	Fine-tuning	$[q]$ $[sep]$ $[d]$
		Prompt-tuning (hard)	$[q]$ and $[d]$ are $[mask]$
		Prompt-tuning (soft)	$[q]$ $[soft]$ $[mask]$ $[soft]$ $[d]$

RoBERTa can give a better result than BM25 with soft prompt-tuning using 100 training queries. It can be seen in Fig. 2(c). In the setting with more training queries, the two types of prompts show similar performance. However, the advantage of the soft prompt doesn't appear for T5. The performance even declines in several scenarios and the trend intensifies when using fewer training queries. We speculate that it might be due to the soft prompt we design for T5 is not potentially optimal. Even so, T5 with hard prompt-tuning has achieved superior results compared with other methods and we believe that a more sophisticated soft prompt for T5 will further boost its performance.

5.4 Case Study

In this section, we will investigate some cases in which the prompt-tuning based model significantly improves the performance compared to the fine-tuning based model. Specifically, we choose RoBERTa with hard prompt-tuning and RoBERTa with fine-tuning in the 10 k queries setting for the comparison. Intuitively, prompt-tuning is a token classification task that focuses on the semantic match at the token level. While as a text pair classification task, fine-tuning can capture the text level relevance signal but it may neglect the lexical match.

In the case shown in Table 5, fine-tuning based model ranks an irrelevant document at the first position since it is a guide for one job in texas. However, the job in query is a broker rather than a notary. Fine-tuning based model only considers the coarse-grained semantic match while neglecting the keyword "broker". Prompt-tuning based model can capture the keyword information in the relevant document and ranks it at the first position while fine-tuning based model ranks it at the 13th position.

Table 5. Case study on query 393696. Document 759874 (second row) is a relevant document and document 1455664 (third row) is an irrelevant document for this query, respectively.

Query: in texas what requirements are needed to apply to become a broker
The Texas State Securities Board has no established educational requirements for becoming a stockbroker, but many broker-dealer firms will require that applicants for sponsorship hold a bachelor's degree. Additionally, if you choose to pursue professional designations during your career, a bachelor's degree, at minimum, is usually required. **Rank result:** RoBERTa with hard prompt-tuning (1), RoBERTa with fine-tuning (15).
Here is a step-by-step guide to become a notary in Texas. Notaries.com handles all of these requirements through our easy application, so rest assured you have everything you need to become a notary! In order to become a notary in the State of Texas, you must: Be at least 18 years old. **Rank result:** RoBERTa with hard prompt-tuning (60), RoBERTa with fine-tuning (1)

6 Conclusion

In this paper, we investigate the performance of prompt learning in the ad hoc search task. We first introduce the formulaic representation of the ad hoc search task and prompt learning. Then we design two types of prompts, i.e. hard and soft prompts, for three PTM, i.e. BERT, RoBERTa and T5. With a prompt learning framework tailored for the ad hoc search task, we conduct experiments on MS Marco dataset. The experimental results in full-resource and low-resource scenarios show a convincing performance of our prompt learning method compared with baseline methods. We further analyze the experimental results and show that aligning the task format and utilizing soft prompt can contribute to better results in our method. In the case study, we note that prompt-tuning and fine-tuning have different relevance signal preferences. This inspires us to further analyze how prompt learning methods enhance the performance of the ad hoc search in the future.

Acknowledgments. This work is supported by the Natural Science Foundation of China (Grant No. 61732008) and Tsinghua University Guoqiang Research Institute.

References

1. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
2. Nogueira, R., Cho, K.: Passage re-ranking with BERT. arXiv preprint arXiv:1901.04085 (2019)
3. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. arXiv preprint arXiv:2107.13586 (2021)

4. Zhang, X., Yates, A., Lin, J.: A little bit is worse than none: ranking with limited training data. In: Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing, pp. 107–112 (2020)
5. Petroni, F., et al.: Language models as knowledge bases? arXiv preprint arXiv:1909.01066 (2019)
6. Brown, T., et al.: Language models are few-shot learners. Adv. Neural. Inf. Process. Syst. **33**, 1877–1901 (2020)
7. Liu, X., et al.: GPT understands, too. arXiv preprint arXiv:2103.10385 (2021)
8. Han, X., Zhao, W., Ding, N., Liu, Z., Sun, M.: PTR: prompt tuning with rules for text classification. arXiv preprint arXiv:2105.11259 (2021)
9. Nogueira, R., Jiang, Z., Lin, J.: Document ranking with a pretrained sequence-to-sequence model. arXiv preprint arXiv:2003.06713 (2020)
10. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
11. Hu, X., Yu, S., Xiong, C., Liu, Z., Liu, Z., Yu, G.: P ^{3} ranker: mitigating the gaps between pre-training and ranking fine-tuning with prompt-based learning and pre-finetuning. arXiv preprint arXiv:2205.01886 (2022)
12. Williams, A., Nangia, N., Bowman, S.R.: A broad-coverage challenge corpus for sentence understanding through inference. arXiv preprint arXiv:1704.05426 (2017)
13. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
14. Nguyen, T., et al.: MS MARCO: a human generated machine reading comprehension dataset. In: CoCo@ NIPS (2016)
15. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M., et al.: Okapi at TREC-3. NIST Spec. Publ. **109**, 109 (1995)
16. Karpukhin, V., et al.: Dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2004.04906 (2020)
17. Xiong, L., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020)
18. Qu, Y., et al.: RocketQA: an optimized training approach to dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2010.08191 (2020)
19. Li, C., Yates, A., MacAvaney, S., He, B., Sun, Y.: PARADE: passage representation aggregation for document reranking. arXiv preprint arXiv:2008.09093 (2020)
20. Dai, Z., Callan, J.: Deeper text understanding for IR with contextual neural language modeling. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 985–988 (2019)
21. Nogueira, R., Yang, W., Cho, K., Lin, J.: Multi-stage document ranking with BERT. arXiv preprint arXiv:1910.14424 (2019)
22. Shin, T., Razeghi, Y., Logan IV, R.L., Wallace, E., Singh, S.: Autoprompt: eliciting knowledge from language models with automatically generated prompts. arXiv preprint arXiv:2010.15980 (2020)
23. Li, X.L., Liang, P.: Prefix-tuning: optimizing continuous prompts for generation. arXiv preprint arXiv:2101.00190 (2021)
24. Hambardzumyan, K., Khachatrian, H., May, J.: WARP: word-level adversarial reprogramming. arXiv preprint arXiv:2101.00121 (2021)
25. Ding, N., et al.: OpenPrompt: an open-source framework for prompt-learning. arXiv preprint arXiv:2111.01998 (2021)
26. Wolf, T., et al.: HuggingFace's transformers: state-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019)