Advances in Computer, Signals and Systems (2024) **DOI: 10.23977/acss.2024.080107**
Clausius Scientific Press, Canada **ISSN 2371-8838 Vol. 8 Num. 1**

TSPPT: Two-Stage Prompt Pre-Train to Promote Few-Shot Learning Performance

Feng Jiang[1], Chengguo Lv[1,*]

[1]Department of Computer Science, Heilongjiang University, Harbin, China
[*]Corresponding author

Keywords: Artificial Intelligence, Natural Language Process, Few-Shot Learning, Prompt Learning

Abstract: The Pretrained-Language Model (PLM) has achieved dominance in the field of Natural Language Processing (NLP), and prompt learning further enhances its impact by aligning the pre-training tasks of the language model with the downstream tasks. However, comparing with traditional fine-tune, prompt learning has some disadvantages such as poor absolute accuracy, low training efficiency and poor robustness, especially in the case of small parameters of the language model itself or insufficient training data. A large number of studies have shown that the main defect of Prompt learning (PL) at the present stage is that the quality of Prompt itself plays an important role in the performance of the model, and the existing initialization method of prompt is often not optimal. Therefore, we propose Two-Stage Prompt Pre-Train (TSPPT): using the special pre-training tasks, obtained by constructing or reforming raw texts and downstream tasks, to pre-train two sub-prompt, Task-oriented sub-Prompt (TSP) and Universal Sub-Prompt (USP), in two advanced stages respectively. By concatenating USP and TSP as the prompt initialization for language model to prompt-tuning on downstream tasks, TSPPT promotes overall performance, such as robustness, accuracy, and generalization. Experiments have shown that TSPPT can achieve or even exceed the performance of traditional fine-tuning while retaining the advantage freezing language model parameters and tuning few parameters only.

1. Introduction

In 2017, Google released Transformer [1] based on self-attention mechanism, which significantly improves the modelling ability of natural language, and various pre-trained language models (PLM) based on its framework emerge in an endless stream, such as BERT [2], T5[3], etc. However, because of the mismatch between the traditional fine-tuning process and the tasks of the pre-training process, the PLMs have to accommodate to downstream tasks, resulting in catastrophic forgetting of PLMs. At the same time, the increasing number of parameters of PLMs makes the traditional fine-tuning cost unacceptable. Based on the above shortcomings, prompt learning began to be applied in the field of NLP. By transforming the format of downstream tasks to the type of tasks used in the language model pre-training stage, prompt tuning bridges the gap between pre-training tasks and various downstream tasks and fully tap the potential of PLMs.

Prompt learning roughly goes through two stages. First, PET [4] groundbreaking proposed to transform downstream tasks into cloze-forms tasks, exactly the same form as tasks using for pre-training masked language model, by manually constructing patterns and verbalizers in 2020, greatly improving the absolute accuracy of BERT models in various classification tasks. GPT-3 [5] provides guidance for language model predictions by adding task descriptions to the input, improving the model without changing the parameters of the model itself. We call this kind of prompt containing human semantic information discrete prompt (a.k.a. hard prompt). Another type of prompt is continuous prompt (a.k.a. soft prompts), which has its own parameters that can be tuned on the training data of downstream tasks. As shown in Figure 1 (c). The main method is prepending a sequence of continuous task-specific vectors, called Prompt, to the input, while keeping the LM parameters frozen, such as Prefix Tuning [6], Prompt Tuning [7].

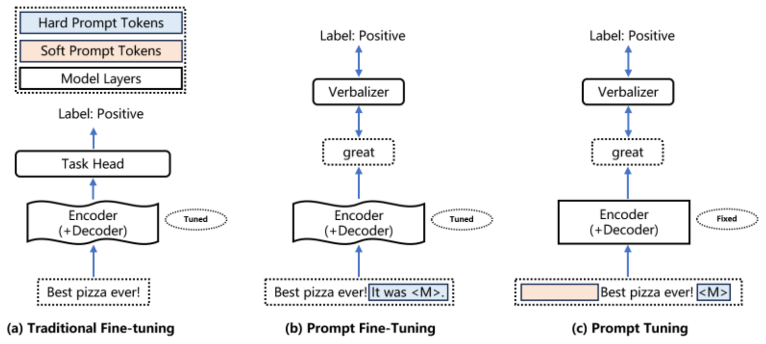

Figure 1: Paradigms of traditional fine-tuning, prompt fine-tuning, and prompt tuning. <M> represents the masked words which have same function in pre-training stage of masked language models. The verbalizer is an injective function that maps each label to real words.

The quality of Prompt itself plays an important role in the performance of the model, and the existing initialization method of prompt is often not optimal. In order to find better Prompt, this paper proposes Two-Stage Prompt Pre-Training (TSPPT): using self-supervised tasks to pre-train two sub-prompts, named Universal Sub-Prompt (USP) and Task-oriented Sub-Prompt (TSP), in two advanced stages respectively. Then we concatenate the two sub-prompts as initialization of prompt for language model to process the downstream tasks. TSPPT further bridges the gap between the pre-training language models and downstream tasks. Not only improving robustness, also retains the feature of allowing the pre-trained language model to process various types of downstream tasks in parallel, as shown in Figure 2, retaining the efficient advantage of prompt learning.

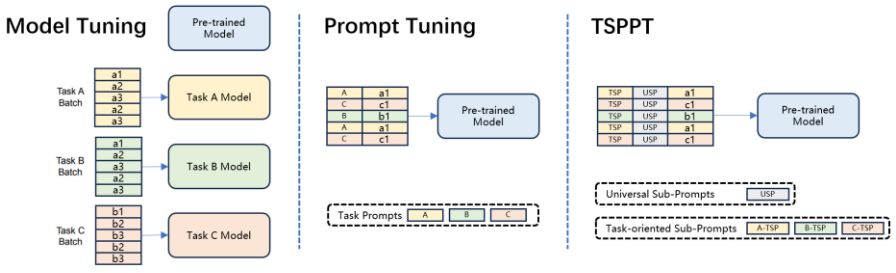

Figure 2: TSTPP retains the efficient advantage of prompt learning.

In this paper, we conduct TSPPT on several datasets based on two PLMs, T5[3] and CPM-2 [8]. Experiments show that TSPPT not only improves few-shot learning performance, but also match or even surpass the traditional fine-tuning on some datasets.

2. Related Work

Prompt- tuning: Most existing PLMs are pre-trained with language modelling objectives, yet the objectives of downstream tasks are quite different. To overcome the gap between pretraining and downstream tasks, prompt-tuning is introduced. In prompt-tuning, downstream tasks are also formalized as language modelling problems by inserting language prompts, and the results of language modelling can correspond to the solutions of downstream tasks. The early prompt learning is mainly based on the prompt contained human semantic information. However, searching [9] or generating [10] prompts in discrete spaces are usually sub-optimal. To overcome the shortcomings of discrete spaces, Prefix Tuning [6]; P-tuning [11]; explore to combine hard prompts and soft prompts. Different from hard prompts using concrete and discrete tokens, soft prompts are composed of several continuous learnable embeddings, and these embeddings are randomly initialized. To step forward, some works propose to only tune soft prompts and fix the entire PLM parameters. When models are large enough, this method can be comparable to full-model tuning.

Prompt Initialization: The prompt initialization has a large impact on the final performance and existing prompt initialization strategies, based on the representations of hard prompt at the embedding layer or random initialization, are sub-optimal and less robustness. So, a new round of research has been launched on how to find better prompts. There are two main approaches to prompt initialization. The first method is to obtain the prompt initialization through transfer learning. Training a prompt from a set of source tasks as the prompt initialization of target tasks. SPoT [12] explores prompt transfer performance by conducting experiments on 186 NLP tasks and predicted the most suitable source tasks for a specific target task based on task similarity method, proving that prompt tuning can benefit from prompt transfer between each other. TPT [13] explored the effects of cross-task transfer and cross-model transfer at the same time and proposed a better approach to judge transfer performance based on overlapping rate of activated neurons. Another is prompt pre-train. Inspired by pre-trained language model, this method pre-trains the soft prompt on self-supervised tasks to get a better prompt and improve the final performance. PPT [14] pretrains Soft Prompt by constructing self-supervised tasks on large-scale unlabelled corpora, improving performance and accelerating the convergence of prompt tuning; ZeroPrompt [15] pre-trains a set of prompts by uniformly modelling different types of tasks, using a smaller set of validations from downstream tasks to pick out the best Prompt.

3. Related Work

In this section, we will introduce the overall framework of TSPPT and demonstrate the pre-training tasks constructed by raw corpus or transformed by existing NLP tasks.

3.1 Overview

According to the current mainstream prompt learning method, we convert various types of downstream tasks into cloze format. Take PET [4] for example, given an input x and its label y, the corresponding mapping function f (x) converts x into a new sequence f (x) by inserting x to a manually constructed Pattern. f (x) contains not only hard prompts, but also mask tokens, which require PLMs to predict. At the same time, PET [4] design a Verbalizer, an injective function, to map label y with label words v (y). Then use f (x) and v (y) to represent the classification task as:

$$\begin{aligned}&\arg\max_{\theta}\sum_{x}\log p(y \mid x;\theta)\\&=\arg\ \max_{\theta}\sum_{x}\log p\left(<M>=v(y)\mid f(x);\theta\right)\end{aligned} \tag{1}$$

The Pattern and the Verbalizer are called PV-Pairs (PVPs, pattern-verbalizer Pairs), and θ represents all adjustable parameters, which are the whole language model parameters in PET [4].

One step further, Prompt Tuning [7] fixed the parameters of the language model and contaminated a set of soft prompts to the beginning of the sequence. Then the model input can be expressed as [P ⊕ f(x)], where "[·⊕ ·]"represents the concatenation operation. By tuning P, Eq (1) is replaced by:

$$\arg\ \max_{P}\sum_{x}\log p\left(<M>=v(y)\mid\left[P\oplus f(x)\right];P\right) \tag{2}$$

The Prompt in TSTPP consists of two sub-prompts. In the first pre-training stage, only the USP is tuned. Similar to Prompt Tuning [7], we can express as:

$$\arg\ \max_{USP}\sum_{x}\log p\left(<M>=v(y)\mid\left[USP\oplus f(x)\right];USP\right) \tag{3}$$

After the first pre-training stage is completed, we get the USP initialization. Now we contaminate TSP before the USP. The second pre-training phase can be expressed as:

$$\arg\ \max_{TSP}\sum_{x}\log p\left(<M>=v(y)\mid\left[TSP\oplus USP\oplus f(x)\right];TSP\right) \tag{4}$$

This paper focus on few-label classification tasks (the number of labels is 2 or 3) and use self-supervised tasks to pre-train prompt to improve the performance of prompt learning. We elaborate the construction and transformation method of pre-training tasks used in the process of prompt pre-training in 3.2 and 3.3 respectively.

3.2 First Stage Prompt Pre-Train

We design two methods to construct pre-train tasks in this stage. One is constructing self-supervised few-label classification tasks on a large-scale unlabelled corpus, and another is transforming non-classification tasks to Few-Label classification tasks.

3.2.1 Construct few-label classification task by raw corpus

For common downstream tasks of NLP, such as natural language inference (NLI) and sentence similarity, the mostly input is sentence pairs.

The mainly manual prompt used in NLI tasks is similar to "determine the content correlation between two sentences.", and we hope that the same entailment will be learned in the process of USP pretraining.

Inspired by the Next Sentence Prediction (NSP) pre-training task in BERT [2], we use unlabelled corpus to construct few-label classification task. Take 3 labels classification task for example, two consecutive sentences in the same document are regarded as a new sample and labelled as 2-coherent. Two discontinuous sentences in the same document are taken as a new sample and labelled 1-similar; Two sentences in different documents are considered as a new sample and labelled as 0-irrelevant. The corresponding PVPs are as follows:

$$f^{pre}(x) = \text{“}S_1\text{<M>}.S_2\text{”}$$

$$v^{pre}(y) = [\text{No, Maybe, Yes}]$$

S_1 and S_2 represent input sentence pairs. When the number of downstream tasks labels is 2, set $v^{pre}(y)$ = [No, Yes].

3.2.2 Transform non-classification tasks to few-label classification tasks

We transform non-classification tasks, such as text summarization, machine translation etc., to few-label classification tasks by prompt learning method. Taking text summarization as an example, the original training sample is usually a text and a corresponding summary sentence. Take the text and its corresponding summary sentence as a new sample, labelled 2 - entailment; Paraphrasing the text then take the new text and its corresponding summary sentence as a new sample, labelled 1-approximate; Taking the text and a random summary sentence as a new sample, labelled 0-contradiction. The paraphrasing method is back translation [16], using a round-trip translation of the sentence into another language then back.

The corresponding PVPs are as follows:

$$f^{pre}(x) = \text{“}T_1\text{?<M>}S_2\text{”}$$

$$v^{pre}(y) = [\text{Wrong, Maybe, Anyhow}]$$

T_1 and S_2 represent text and its corresponding summary sentence respectively. We can set $v^{pre}(y)$ = [Wrong, Anyhow] when the number of downstream tasks labels is 2.

3.3 Second Stage Prompt Pre-Train

In this process, we use classification tasks to pretrain TSP to improve the skill for classification. Compared with the pre-train tasks used in first stage, which are constructed from raw text or transformed from other types of tasks, it is relatively easy to obtain pre-train task. So, the target of this stage is how to improve its classification ability. We use knowledge distillation, a classic method of improving accuracy, to answer this question. Specifically, we use a high-precision large model as the Teacher Model (TM) to get the soft labels and select the more reliable prediction results as the training sample of task TSP. The amount of information brought by training samples for TSP is greater than the traditional training method. Meanwhile, this method provides an opportunity to change the number of labels. Take SST-5 and AG’ NEWs for examples, we show that how to construct few-label tasks by fine-grained sentiment analysis tasks multi-label classification task.

3.3.1 Reconstruct fine-grained classification tasks

We used the TM to classify the sample into three categories (0-Negative, 1-Neutral & 2-Positive) and obtain a Soft-Label TM. Considering that the results of fine-grained task are poor, in order to improve the quality of soft labels, a Simulation-Soft-Label (Table 1) is assigned to each sample as follows:

Table 1: Simulation-Soft-Label of fine-grained classification task

Original Label	Simulation-Soft-Label
0-very negative	0.9 negative, 0.1 neutral, 0 positive
1-negative	0.7 negative, 0.2 neutral, 0.1 positive
2-neutral	0. 1 negative, 0.8 neutral, 0.1 positive
3-positive	0.1 negative, 0.2 neutral, 0.7 positive
4-very positive	0 negative, 0.1 neutral, 0.9 positive

Taking samples with better predictive results, the Training Soft Label is obtained by weighting the Soft-Label TM and Simulation-Soft-Label. This approach not only improves the label quality of the training samples, but also increases the class probability distribution entropy, correspondingly magnifying the information carried by the negative label, playing a role similar to the "temperature" parameters.

The corresponding PVPs are as follows:

$$f^{pre}(x) = \text{“}T_1. \text{<M>.”}$$

$$v^{pre}(y) = [\text{Bad, Neutral, Good}]$$

T_1 represent input sentence. We can set $v^{pre}(y)$ = [bad, good] when the number of downstream tasks labels is 2.

3.3.2 Reconstruct Multi-Label classification task

The labels of AG 'News task are 0-World, 1-Sports, 2-Business, and 3-Sci/Tec. We transform it to three labels task. For example, Remaining samples belong to 0-World and 1-Sports, we combine samples belong to Business and Sci/Tec to a new training batch and assign a new label 2-Others. We can get six pre-train tasks using that method.

This transformation expands the scale of the pre-training tasks, but also brings a problem, that is, the probability language model predicted of 2-Others label will increase. Take a sample SW belongs to 0-World label as an example: For the sentence SW, among the six pre-training tasks, its label is 0-World three times and 2-Others three times too, so the language model tends to output 2-Others label when predicting similar sentence in the downstream tasks. We still use Simulation-Soft-Label to counteract this negative effect. Taking the above transformation as an example, a Simulation-Soft-Label (Table 2) is given to each initial label sample as follows:

Table 2: Simulation-Soft-Label of Multi-Label classification task

Original Label	Simulation-Soft-Label
0-World	0.9 world, 0.1 sports, 0 others
1-Sports	0.1 world, 0.9 sports, 0 others
2-Others	0. 3 world, 0.3 sports, 0.4 others

We use TM to classify the samples into three labels, 0-World, 1-Sports & 2-Others to obtain a Soft-Label TM. The Training Soft Label are also obtained by weighted-sum method. Although this method cannot completely eliminate the negative effects, some of the remaining negative effects can be regarded as noise, which improves the robustness to some extent.

The corresponding PVPs are as follows:

$$f^{pre}(x) = \text{“S1. It is a <M> news.”}$$

$$v^{pre}(y) = [\text{global, sports, business, sci-tech, other}]$$

S1 represent input sentence. For different transformation format, we can take the subset of $v^{pre}(y)$.

Through the task transformation process, we get the pre-train tasks for TSP. Each sample has an original label and a Training Soft Label. TSP needs to learn these two labels separately, adjust the loss weight of the two labels with α, and use p to represent the language model prediction, then the loss function is expressed as follows:

$$\text{Loss} = \text{Cross Entropy (L, p)} + \alpha \text{Cross Entropy (TSL, p)} \qquad (5)$$

4. Experiments

4.1 Set up and Baseline

Concatenating USP and TSP as prompt initialization for downstream tasks, we conduct experiments om both Chinese and English tasks, (see Table 3).

Table 3: Datasets

English		Chinese	
Dataset	Class number	Dataset	Class number
SST-2	2	ChnSent	2
BoolQ	3	LCQMC	3
RTE	3	CMNLI	3
CB	3	OCNLI	3

For English datasets, we conduct TSPPT based on T5-XXL with 11B parameters because previous works have shown that, T5-XXL using Prompt Tuning [7] is comparable with fine-tuning under the full-data setting. For Chinese datasets, we do PT based on a 11B model CPM-2 [8]. We use 150 soft tokens for TSPPT. USP and TSP are 100 and 50 tuneable soft tokens. As a result, the tuneable parameters are only 100 ×4096 = 4.1 × 105 = 410K. Compared with the 11B (1.1 × 1010) parameters of FT, TSPPT only needs to store 3000 times smaller parameters for each task.

For Few-Shot scenario, following the classic method of PET [4] and (Perez et al., 2021) [17], we randomly select 32 samples from the original data set as the training set D_{train}, and select the same number of samples to construct the validation set D_{dev} to tune the hyperparameters. We follow Zhang et al. (2021) [18] and Gao et al. (2021) [10] to use the original validation set as the test set D_{test}, which means $| D_{test} | > | D_{train} | = | D_{dev} |$.

We compare TSPPT with traditional fine-tuning, original Prompt Tuning [7], SPoT [12] and PPT [14]. Considering Hybrid PT, adding hard prompt to prompt tuning, usually improves prompt tuning performance, we take it as a baseline too.

4.2 Main Results

The main results of English and Chinese datasets are shown in Table 4.

Table 4: Classification results.

/	FT	PT	Hybrid PT	PPT	SPoT	TSPPT
SST-2	$91.4_{0.8}$	$70.5_{15.5}$	$87.6_{6.6}$	$\mathbf{93.5}_{0.3}$	$91.2_{0.1}$	$\mathbf{93.5}_{0.2}$
BoolQ	$80.8_{2.4}$	$61.0_{5.3}$	$79.8_{1.5}$	$66.4_{5.7}$	$67.3_{5.0}$	$\mathbf{82.3}_{0.8}$
RTE	$\mathbf{64.1}_{2.0}$	$53.5_{3.5}$	$56.8_{2.6}$	$58.9_{1.6}$	$57.7_{1.8}$	$62.4_{2.1}$
CB	$\mathbf{86.5}_{5.3}$	$50.7_{4.1}$	$66.5_{7.2}$	$71.2_{6.2}$	$70.2_{5.5}$	$74.0_{6.1}$
ChnSent	$86.1_{1.8}$	$62.1_{3.1}$	$79.2_{4.0}$	$\mathbf{90.1}_{0.8}$	/	$89.8_{0.5}$
LCQMC	$58.8_{1.8}$	$51.5_{3.4}$	$54.6_{2.3}$	$59.1_{0.6}$		$\mathbf{64.2}_{0.7}$
CMNLI	$40.7_{1.0}$	$35.4_{0.5}$	$37.1_{0.6}$	$43.0_{0.5}$		$\mathbf{44.0}_{0.4}$
OCNLI	$38.8_{1.5}$	$37.0_{0.5}$	$37.8_{1.4}$	$40.1_{0.4}$		$\mathbf{41.2}_{0.5}$

The experiments are conducted with 32 training samples and 32 validation samples on each dataset. FT means traditional fine-tuning, where the entire model (with about 11B parameters) should be tuned on each dataset. PT means prompt tuning, and its soft prompt initialization is randomly initialized from the normal distribution. We report the mean and the standard deviation over 5 random seeds. The score marked as bold means the best performance among all the methods.

In the experimental results, four important results can be seen. First, compared with FT, TSPPT has achieved better results on all Chinese tasks, and most of the English tasks with less parameters. This means that even though the MLM has been fine-tuned on the downstream task, there is still a gap between the model pre-training and downstream task, and TSTPP bridges the gap to some extent. Second, after adding some hard prompt, Hybrid PT performs better than PT, but there is still a gap with TSTPP. This phenomenon is universal in prompt learning. TSPPT can also improve performance by adding hard prompts. Since we focus on the prompt pre-training process, do not introduce here. Third, few-shot learning is notorious for its instability, which becomes very obvious in Vanilla PT. TSPPTPPT results in lower variances on most of the datasets which means it alleviate this problem to some extent. Finally, comparing with PPT [14] and SPoT [12], TSPPT still perform better on most of datasets. It proves the conclusion that two kinds of skill and knowledge distillation are helpful for prompt tuning.

5. Conclusion and Future Work

In this paper, we propose two-stage Prompt pre-training (TSTPP) to improve prompt tuning performance in the field of Few-Labels classification (few-shot and Full-data). We designed the pre-training tasks in a variety of ways, that is, constructing from raw corpus, transforming non-classification tasks and multi-labels tasks to few-label classification tasks. Then use them to pre-train sub-prompts USP and TSP in two stages respectively. Finally concatenate USP and TSP as prompt initialization used in downstream tasks. TSPPT also incorporates knowledge distillation and data enhancement techniques. We conducted experiments both on Chinese and English datasets, showing that TSTPP outperforms other prompt tuning methods, and achieves even surpass the performance of traditional fine-tune with fewer tuned parameters.

In the future, it will be interesting to see how to get rid of the label number constraint by structuring the pre-training process more cleverly. In addition, TSPPT uses generative tasks as the source tasks for classification. How to apply TSPPT to generative tasks is also worth exploring.

References

[1] 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, pages 5998–6008.

[2] 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-HLT.

[3] 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. JMLR.

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

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, et al. 2020. Language models are few-shot learners. In Proceedings of NeurIPS.

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

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

[8] Zhengyan Zhang, Yuxian Gu, Xu Han, Shengqi Chen, et al. 2022. CPM-2: Large-scale cost-effective pre-trained language models. AI Open.

[9] 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 Empirical Methods in Natural Language Processing (EMNLP).

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

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

[12] Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2021. Spot: Better frozen model adaptation through soft prompt transfer. CoRR, abs/2110.07904.

[13] Yusheng Su and Xiaozhi Wang. On Transferability of Prompt Tuning for Natural Language Understanding. arXiv

preprint arxiv:2111.06719.
[14] Yuxian Gu, Xu Han, Zhiyuan Liu and Minlie Huang. PPT: Pre-trained Prompt Tuning for Few-shot Learning. In Proceedings of ACL.
[15] Hanwei Xu, Yujun Chen and Yulun Du. Zero Prompt: Scaling Prompt-Based Pretraining to 1,000 Tasks Improves Zero-Shot Generalization. arXiv preprint arxiv:2201.06910.
[16] Xie, Q., Dai, Z., Hovy, E.H., Luong, T., Le, Q., 2020. Unsupervised data augmentation for consistency training. December 6-12, 2020. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020. virtual.
[17] Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In Proceedings of NeurIPS.
[18] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. 2021. Revisiting few-sample bert fine-tuning. In Proceedings of ICLR.