Multitask Pre-training of Modular Prompt for Chinese Few-Shot Learning

Tianxiang Sun* **Zhengfu He*** **Qin Zhu** **Xipeng Qiu†** **Xuanjing Huang**
School of Computer Science, Fudan University
Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
{txsun19,zfhe19,xpqiu,xjhuang}@fudan.edu.cn zhuq22@m.fudan.edu.cn

Abstract

Prompt tuning is a parameter-efficient approach to adapting pre-trained language models to downstream tasks. Although prompt tuning has been shown to match the performance of full model tuning when training data is sufficient, it tends to struggle in few-shot learning settings. In this paper, we present **M**ulti-task **P**re-trained **M**odular **P**rompt (**MP²**) to boost prompt tuning for few-shot learning. MP² is a set of combinable prompts pre-trained on 38 Chinese tasks. On downstream tasks, the pre-trained prompts are selectively activated and combined, leading to strong compositional generalization to unseen tasks. To bridge the gap between pre-training and fine-tuning, we formulate upstream and downstream tasks into a unified machine reading comprehension task. Extensive experiments under two learning paradigms, i.e., gradient descent and black-box tuning, show that MP² significantly outperforms prompt tuning, full model tuning, and prior prompt pre-training methods in few-shot settings. In addition, we demonstrate that MP² can achieve surprisingly fast and strong adaptation to downstream tasks by merely learning 8 parameters to combine the pre-trained modular prompts.

1 Introduction

Pre-trained models (PTMs) (Devlin et al., 2019; Lewis et al., 2020; Raffel et al., 2020; Qiu et al., 2020) with prompt-based learning have achieved remarkable progress in few-shot learning. A major reason behind their success is the closed gap between upstream pre-training and downstream fine-tuning (Liu et al., 2021a; Sun et al., 2022b). Since the downstream tasks are reformulated into a unified (masked) language modeling ((M)LM for short) task, one can reuse the pre-trained (M)LM head instead of training a randomly initialized classification head to solve tasks with limited data.

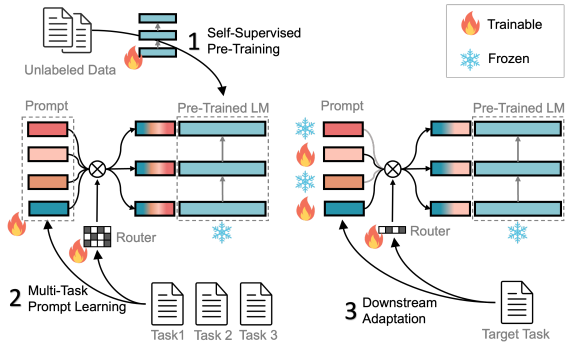


Figure 1: MP² achieves fast adaptation to downstream tasks through three steps: (1) Self-supervised pre-training on large-scale unlabeled data. (2) Pre-training modular prompts and the corresponding router with multi-task learning. (3) A subset of prompts is activated and tuned for adaptation to downstream tasks.

However, prompt-based learning (e.g., PET (Schick and Schütze, 2021) and LM-BFF (Gao et al., 2021)) usually fine-tunes all the parameters of the PTM for each downstream task, which can be computationally expensive and deployment-inefficient, especially for large PTMs such as GPT-3 (Brown et al., 2020).

Recently, much effort has been devoted to parameter-efficient prompt tuning (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2021c; Sun et al., 2022c), which only learns a small number of soft prompt parameters while keeping the main body of the PTM untouched. In contrast to full model tuning, prompt tuning can get specialized models for specific tasks by simply attaching task-specific prompts, and therefore is highly efficient for serving different tasks. Though it has been demonstrated that prompt tuning can match the performance of full model tuning when training data is sufficient (Lester et al., 2021), the soft prompt cannot be well trained from scratch in few-shot learning settings (Gu et al., 2021) because the randomly initialized soft prompt introduces a new gap between pre-training and fine-tuning.

* Equal contribution.
† Corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers, pages 11156–11172
July 9-14, 2023

To bridge the gap between pre-training and fine-tuning for prompt tuning, we present **M**ulti-task **P**re-trained **M**odular **P**rompt (**MP**^{2}). As illustrated in Figure 1, we insert a second pre-training procedure before downstream fine-tuning, in which we pre-train a set of modular prompts with multi-task learning. The modular prompts are selectively activated and combined by a trainable router for specific tasks. By this, we can achieve fast adaptation to downstream tasks by learning to combine and reuse the pre-trained modular prompts. Drawing inspiration from the success of deep prompt tuning (Li and Liang, 2021; Liu et al., 2021b), we inject soft prompt into every layer of the PTM. Further, considering that a variety of tasks cannot be reformulated into a (M)LM task, we instead recast upstream and downstream tasks into a unified machine reading comprehension (MRC) task, which has shown great potential to unify various NLP tasks (McCann et al., 2018; Sun et al., 2022b).

We pre-train MP^2 on 38 Chinese NLP tasks and evaluate on 14 downstream tasks including sentiment analysis, topic classification, natural language inference, question answering, multiple choice classification, and keyword extraction. Experimental results in few-shot learning settings demonstrate that MP^2 outperforms prompt tuning, full model tuning, and previous prompt pre-training methods (Gu et al., 2021; Vu et al., 2022) by a large margin. We also evaluate the compatibility of MP^2 with black-box tuning (BBT) (Sun et al., 2022c) and BBTv2 (Sun et al., 2022a), which are gradient-free prompt tuning methods. As a result, MP^2 achieves significant improvement over BBT and BBTv2. Besides, we demonstrate that MP^2 can achieve surprisingly fast adaptation to target tasks by merely tuning the router (only 8 parameters) while freezing the PTM and all the prompts.[1]

2 Related Work

This work lies in the line of parameter-efficient tuning (PET) (He et al., 2021; Ding et al., 2022), which trains a small portion of parameters to adapt PTMs to downstream tasks. The small tunable parameters can be lightweight neural adapters between PTM layers (Houlsby et al., 2019), or soft prompt attached to the input examples (Lester et al., 2021) or hidden states (Li and Liang, 2021), or bias terms in the PTM parameters (Zaken et al., 2022), or low-rank matrices to be added to attention weights (Hu et al., 2021). Especially, this work is closely related to two prior works on prompt tuning, namely PPT (Gu et al., 2021) and SPoT (Vu et al., 2022).

Method	Params.	Data Size	Data/Param.
PPT	410K	10 GB	24.39 GB/M
MP^2 (Ours)	307M	15 GB	0.05 GB/M
BERT	335M	16 GB	0.05 GB/M
XLNet	335M	158 GB	0.47 GB/M
RoBERTa	355M	160 GB	0.48 GB/M
BART	406M	160 GB	0.39 GB/M
T5	11B	745 GB	0.07 GB/M

Table 1: Comparison of model size and data size for various pre-training methods. In contrast to conventional PTMs, there is a mismatch between the number of learnable parameters and the volume of training data for PPT.

Comparison with PPT. A prior work with the similar motivation is Pre-trained Prompt Tuning (PPT) (Gu et al., 2021), which pre-trains soft prompt prepended to the input embedding on large-scale unlabeled corpora with an objective of next sentence prediction (NSP). Different from the NSP in BERT (Devlin et al., 2019), PPT recasts the NSP task into a multiple choice classification (MCC) format. For downstream tasks, PPT formulates three types of tasks, namely single-sentence, sentence-pair, and multiple choice classification, into a unified MCC format such that the gap between the pre-training task and downstream tasks can be filled. Despite their success, we argue that PPT has three possible defects: **(1) Complexity Mismatch**: The number of learnable parameters and the volume of training data are mismatched. PPT trains 410K parameters with 10 GB training data. By contrast, conventional PTMs have much smaller data-parameter ratios (see Table 1). Hence, the limited number of parameters can hardly contain the rich knowledge in the large corpora. **(2) Simple Objective**: The pre-training objective of PPT, i.e., NSP, is not difficult enough. It has been shown that the impact of the NSP objective is unreliable (Yang et al., 2019b; Liu et al., 2019). As formulated by Lan et al. (2020), NSP can be accomplished through two subtasks, *topic prediction* and *coherence prediction*. Nevertheless, topic prediction is easier to learn than coherence prediction, and therefore can dominate learning and makes NSP a rather simple task. **(3) Limited Task**: The downstream tasks handled by PPT are limited. PPT cannot address

[1] Code and data are publicly available at `https://github.com/Hzfinfdu/MPMP`.

tasks that cannot be reformulated into a MCC task, such as question answering. Besides, when pre-training with the MCC format, PPT supports up to 16 options (A-P), which means it only promises to adapt to tasks with no more than 16 labels. In this work, the above issues are well addressed by MP^2. **First**, MP^2 increases capacity of prompt in two dimensions, i.e., depth (deep prompt) and width (modular prompt), to match the complexity of training data. **Second**, MP^2 is pre-trained on 38 real-world Chinese tasks with multi-task learning, instead of pre-training in a self-supervised fashion with the NSP loss. **Third**, MP^2 recasts upstream and downstream tasks into a unified MRC task to support a wider range of downstream tasks.

Comparison with SPoT. Another work that is similar to ours is Soft Prompt Transfer (SPoT) (Vu et al., 2022), which also explored training soft prompt with multi-task learning and then using it to initialize the prompt for a target task. By comparison, our proposed MP^2 has three main differences from SPoT: (1) We pre-train a set of modular prompts that are selectively combined and attached to every layer of the PTM rather than training a single prompt to be prepended merely to the input layer. (2) We formulate upstream and downstream tasks into a unified MRC task instead of unifying tasks into a text-to-text format (Raffel et al., 2020) where the output label words cannot be shared between upstream and downstream tasks.[2] (3) Unlike SPoT that is mainly evaluated in full data settings, MP^2 is dedicated to few-shot learning.

3 Methods

We first introduce the MRC format used to unify different tasks in §3.1, and then describe the deep modular prompt in §3.2, and finally we detail the procedure of multi-task pre-training and downstream fine-tuning in §3.3 and §3.4, respectively.

3.1 Unifying Tasks with MRC

Bridging the gap between upstream and downstream tasks is crucial for few-shot learning. Prompt-based learning (Liu et al., 2021a) formulates downstream tasks into a (M)LM task, which, however, cannot cover a wide range of tasks. Besides, the label words (a.k.a. verbalizer) can be different across tasks. Therefore, the soft prompt pre-trained with a certain set of label words can be less effective to be used in a target task with a different set of label words. To that end, PPT (Gu et al., 2021) recasts upstream and downstream tasks into a MCC task such that different tasks can share the same set of label words, i.e., 16 option indicators (A-P). As a result, there is still a gap between pre-training and fine-tuning when performing classification with more than 16 labels. In addition, the task types supported by MCC can still be limited.

In MP^2, we adopt a more general format, machine reading comprehension (MRC), to unify upstream and downstream tasks. MRC has achieved great success in unifying a variety of NLP tasks (Sun et al., 2022b). The input of MRC is comprised of a *passage* (also referred to as *context*) and a *query*, and the output is the *answer* of the query, which is a span of text in the input. Typically, the prediction of the answer is achieved by two binary classification heads on each token of the input, one for predicting the start position and one for predicting the end position (Xiong et al., 2017; Seo et al., 2017).

For classification tasks, we use the original sample as the *context* and construct a *query* consisting of all possible labels. In contrast to PPT that pre-defines a set of option indicators, MP^2 directly extracts the answer from the query, and therefore can generalize across tasks with different numbers of labels. Appendix C contains some examples of converting tasks into the MRC format.

3.2 Deep Modular Prompt

To increase the capacity of the soft prompt such that it can match the complexity of training data, we extend soft prompt in two dimensions, depth and width. Figure 2 provides an overview of the deep modular prompt.

Deep Prompt. Inspired by the success of deep prompt tuning (Li and Liang, 2021; Qin and Eisner, 2021; Liu et al., 2021b), we inject soft prompt to every layer of the PTM instead of the mere input layer. The incorporation of deep prompt increases the number of learnable parameters and so as the adaptation ability to hard tasks.

Modular Prompt. For the soft prompt attached to each layer of the PTM, we extend the single static prompt to a set of modular prompts. Formally, we pre-train K soft prompts $\{\mathbf{p}_1^{(l)}, \ldots, \mathbf{p}_K^{(l)}\}$ for

[2] A shared set of label words in prompt pre-training can be crucial to few-shot leaning. For example, PPT recasts tasks into the MCC format such that the label words are constrained to option words, i.e., {A, B, C, ...}.

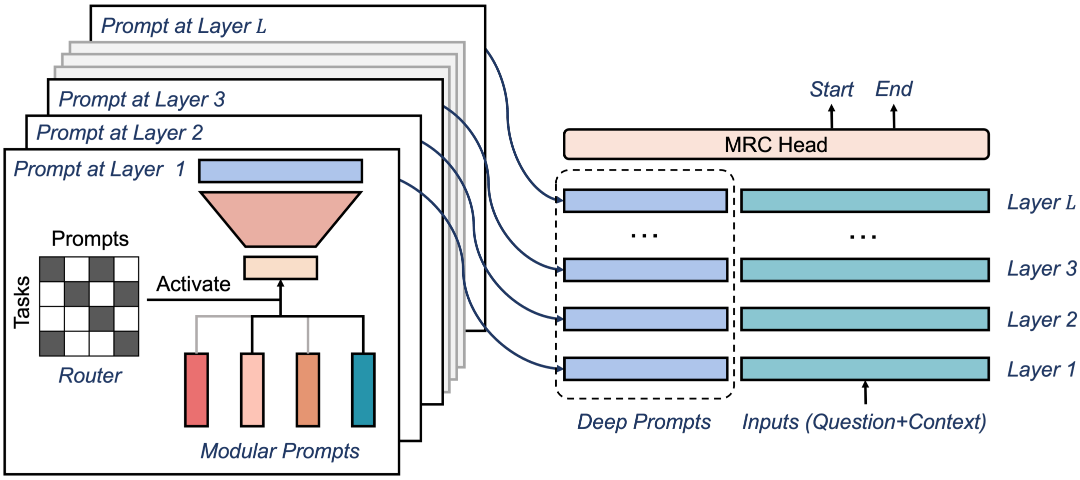


Figure 2: An illustration of the deep modular prompt during pre-training.

each layer l. For a certain task, the prompt at layer l is the weighted mean of the set of soft prompts,

$$\mathbf{p}^{(l)} = \frac{1}{K}\sum_{k=1}^{K} w_k^{(l)} \mathbf{p}_k^{(l)}, \tag{1}$$

where $\mathbf{w}^{(l)} = \{w_1^{(l)}, \ldots, w_K^{(l)}\}$ are layer- and task-specific learnable parameters called *router*. To pursue compositional generalization, we encourage the prompts to be sparsely activated and combined. Thus, the router $\mathbf{w}^{(l)}$ should be binary-valued, i.e., $\mathbf{w}^{(l)} \in \{0, 1\}^K$. Each single prompt can be viewed as some fundamental skill, and a task can be solved by combining such modular skills. Different tasks tend to require different subsets of the skills. Though similar ideas have been proposed in other names and contexts (Sun et al., 2020b; Zhang et al., 2022a; Ponti et al., 2022), this is the first work that implements the skills with soft prompts to drive pre-trained language models.

Relaxed Bernoulli Distribution. A challenge is that the discrete router $\mathbf{w}$ [3] is not differentiable and therefore cannot be optimized by gradient descent in an end-to-end fashion. To that end, we keep $\mathbf{w} \in \mathbb{R}^K$ as free parameters to parameterize a relaxed Bernoulli (or binary concrete) distribution (Maddison et al., 2017), which can be considered as a continuous relaxation of the Bernoulli distribution. From the relaxed Bernoulli distribution, we sample $\hat{\mathbf{w}}$ to weight the modular prompts, i.e., $\mathbf{p} = \sum_{k=1}^{K} \hat{w}_k \mathbf{p}_k / K$. By using the reparameterization trick (Kingma and Welling, 2014), the router can be learned via gradient descent while maintaining some degree of stochasticity. Formally, the sampling procedure for $\hat{w}_k \sim \texttt{RelaxedBernoulli}(\alpha, \tau)$ is as follows,

$$u \sim \texttt{Uniform}(0, 1), \tag{2}$$

$$v = \log(\alpha) + \log(u) - \log(1 - u), \tag{3}$$

$$\hat{w}_k = \sigma(v/\tau), \tag{4}$$

where $\alpha \in (0, \infty)$ is the location parameter, σ is the Sigmoid function, and $\tau \in (0, \infty)$ is the temperature to control the degree of approximation. Note that w_k can be negative during training and therefore cannot be used directly as the location parameter α. To ensure that $\alpha \in (0, \infty)$, we set α as follows,

$$\alpha = \frac{\sigma(w_k)}{1 - \sigma(w_k)}. \tag{5}$$

During inference, we simply set $\hat{w}_k = 1$ if $w_k > 0$, otherwise $\hat{w}_k = 0$.

Intrinsic Reparameterization. Recent studies (Sun et al., 2022c; Diao et al., 2022) have demonstrated that prompt tuning can be achieved in a much lower dimensional *intrinsic subspace* through gradient-free optimization. To benefit tuning in the intrinsic subspace, we perform *intrinsic reparameterization*, which is to decompose the original modular prompt $\mathbf{p}_k \in \mathbb{R}^D$ into an intrinsic prompt $\mathbf{z}_k \in \mathbb{R}^d$ and a projection matrix $\mathbf{A} \in \mathbb{R}^{D\times d}$. Note that $\mathbf{A}$ is shared by the modular prompts $\{\mathbf{p}_k\}_{k=1}^K$ at the same layer. During multi-task pre-training, both $\mathbf{z}_k$ and $\mathbf{A}$ are updated. On downstream tasks, black-box tuning (BBT) (Sun et al., 2022c) can be enabled by only tuning the intrinsic prompt $\mathbf{z}_k$ while keeping $\mathbf{A}$ frozen.

[3] For simplicity, we omit the superscript (l) without causing confusion.

3.3 Multi-Task Pre-Training

Multi-task learning has been shown to boost the performance of prompt tuning in a variety of tasks (Vu et al., 2022). Following their success, we pre-train the deep modular prompts on a mixture of 38 Chinese NLP tasks with varying types, domains, and sizes. To handle the unbalanced data sizes, for each forward computation, we first randomly sample a task ID from 1 to 38 and then fetch a batch of training data corresponding to the sampled task, such that the number of learning steps for each task is expected to be identical.

Fast and Slow Learning. For the pre-training of the routers and the prompts, we intuitively encourage fast learning for the routers to reuse existing modular prompts to adapt to the current task, and slow learning for the task-specific prompts. In particular, we adopt a higher learning rate for the routers $\mathbf{z}$ to change quickly, and adopt a lower learning rate for the modular prompts $\mathbf{p}$ to change slowly and stably. Similar ideas are also explored by Madan et al. (2021); Ponti et al. (2022).

3.4 Downstream Fine-Tuning

For fast adaptation to downstream tasks, we propose the *two-stage tuning*. **In stage I**, we allocate a random router for each layer to a new target task and train the routers to selectively reuse pre-trained modular prompts to solve the target task while keeping all other parameters frozen. **In stage II**, we freeze the routers and only tune the selected prompts. The PTM parameters are unchanged throughout the entire fine-tuning process.

We explore fine-tuning MP^2 under two learning paradigms, namely *gradient descent* and *black-box tuning*. For gradient descent, we use an Adam (Kingma and Ba, 2015) optimizer to perform two-stage tuning. For black-box tuning, we adopt the Bayesian optimization (BO) (Mockus, 1974) in stage I to optimize the routers, and adopt the CMA-ES (Hansen and Ostermeier, 2001) to optimize the selected intrinsic prompts $\mathbf{z}_k$ while freezing the projection matrices $\mathbf{A}$. See Appendix A for detailed description of fine-tuning.

4 Experiments

4.1 Datasets and Tasks

Pre-training Tasks. We collect 38 public Chinese NLP tasks ranging from different task types, domains, and data sizes as upstream tasks for pre-training. The total size of the pre-training data is 15GB. Appendix D contains full details of the pre-training tasks.

Setting	Dataset	Task	|Test|	|Labels|
UNSEEN DATA	Amazon	TC	5789	5
	THUCNews	TC	5000	10
	BQ	NLI	10000	2
	CMNLI	NLI	12545	3
	CMRC-2018	MRC	2886	N/A
	CCPM	MCQA	2720	4
	COTE-MFW	KE	8251	N/A
UNSEEN TASK	ChnSent	TC	1200	2
	TNews	TC	10000	15
	OCNLI	NLI	2950	3
	LCQMC	NLI	8802	2
	DRCD	MRC	1238	N/A
	C^3	MCQA	1991	[2, 4]
	COTE-BD	KE	1706	N/A

Table 2: Statistics of downstream tasks. TC: text classification. NLI: natural language inference. MRC: machine reading comprehension. MCQA: multiple choice question answering. KE: keyword extraction.

Downstream Tasks. We divide 14 downstream tasks into two tracks: UNSEEN DATA and UNSEEN TASK. The 7 tasks in the UNSEEN DATA track are a subset of upstream tasks, for which we retain a small portion of training data from the pre-training corpora to ensure that the downstream samples are unseen to MP^2. The UNSEEN TASK track is comprised of 7 tasks that are completely held-out tasks. Table 2 contains statistics of the downstream tasks. The sources of the tasks are in Appendix D.

True Few-Shot Setting. For downstream tasks, we follow the same procedure as Gu et al. (2021) to form the true few-shot learning settings (Perez et al., 2021). In particular, we randomly draw 32 samples from the original training set to construct a few-shot training set $\mathcal{D}_{\text{train}}$, and construct a development set $\mathcal{D}_{\text{dev}}$ by randomly selecting another 32 samples from the original training set. We ensure that the number of labels is balanced for both training and development set. For tasks with more than 5 labels, we randomly select 8 samples for each label. We use the original development sets as the test sets. For datasets without development sets, we use the original test sets.

4.2 Backbones and Baselines

We choose CPT-large (Shao et al., 2021) as our backbone model, which is a competitive Chinese

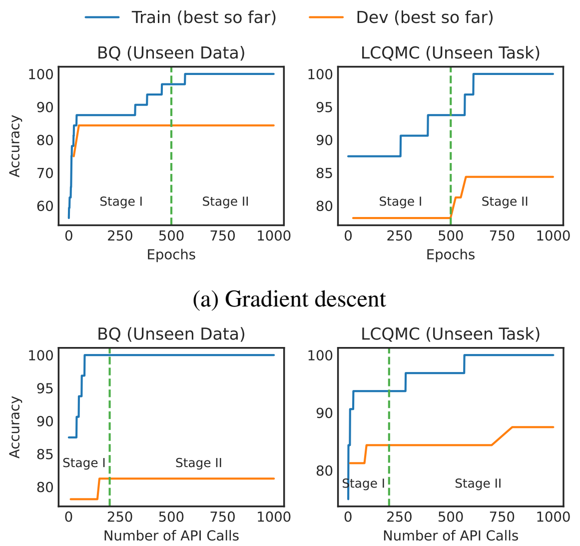
Unseen Data											
Paradigm	**Backbone**	**Methods**	**Tunable Params**	**Amazon** Acc.	**THUCNews** Acc.	**BQ** Acc.	**CMNLI** Acc.	**CMRC-2018** F1	**CCPM** Acc.	**COTE-MFW** F1	**Avg.**
Gradient Descent	CPM-2 (11B)	Model Tuning	11B	$42.5_{2.0}$	-	-	$40.7_{1.0}$	-	$81.8_{1.6}$	-	-
		Prompt Tuning	410K	$30.3_{4.8}$	-	-	$35.4_{0.5}$	-	$31.0_{9.7}$	-	-
		PPT	410K	$44.6_{1.1}$	-	-	$40.6_{0.4}$	-	$83.4_{0.9}$	-	-
	CPT (393M)	Model Tuning	393M	$47.3_{5.3}$	$93.5_{0.3}$	$57.3_{1.7}$	$34.7_{0.1}$	$37.5_{7.4}$	$76.1_{2.4}$	$81.7_{1.3}$	61.2
		Prompt Tuning	50K	$32.9_{2.4}$	$68.6_{4.2}$	$51.3_{0.7}$	$33.8_{0.4}$	$3.5_{0.4}$	$27.3_{1.9}$	$57.7_{1.0}$	39.3
		P-Tuning v2	1.2M	$47.7_{2.3}$	$90.4_{0.6}$	$54.6_{1.6}$	$34.5_{0.2}$	$34.4_{10.4}$	$76.3_{2.0}$	$81.8_{2.0}$	60.0
		PPT	50K	$49.7_{2.3}$	$87.9_{1.3}$	$53.3_{0.9}$	$34.2_{0.6}$	$6.1_{0.6}$	$83.1_{1.2}$	$74.0_{4.1}$	55.5
		SPoT	50K	$55.2_{2.9}$	$89.4_{0.9}$	$61.1_{1.5}$	$39.0_{0.5}$	$56.6_{1.7}$	$85.2_{0.5}$	$86.5_{0.7}$	67.6
		Shallow MP^2	50K$\sim$400K	$62.3_{1.0}$	$91.2_{1.6}$	$71.8_{2.0}$	$66.5_{2.3}$	$68.6_{2.3}$	$85.3_{1.8}$	$87.4_{1.2}$	76.2
		Deep MP^2	1.2M$\sim$9.6M	**65.3**$_{1.7}$	**95.2**$_{0.2}$	**81.4**$_{1.3}$	**76.3**$_{0.8}$	**82.8**$_{1.0}$	**92.4**$_{0.3}$	**90.5**$_{0.2}$	**83.4**
Black-Box Tuning	CPT (393M)	BBT	300	$44.5_{1.5}$	$49.2_{6.0}$	$51.7_{0.5}$	$35.4_{0.7}$	-	$26.4_{0.5}$	-	-
		BBTv2	7.2K	$47.7_{1.7}$	$84.0_{0.8}$	$53.5_{0.8}$	$37.8_{0.4}$	-	$26.9_{1.5}$	-	-
		Shallow MP^2	308	$58.5_{5.1}$	$92.4_{0.4}$	$75.2_{0.8}$	$66.4_{1.4}$	$75.6_{1.9}$	$90.6_{0.2}$	$88.1_{1.1}$	78.1
		- Router-only	8	$62.5_{3.9}$	$92.6_{0.5}$	$75.6_{0.8}$	$63.4_{3.3}$	$77.7_{0.6}$	$90.3_{0.7}$	$89.2_{0.6}$	78.7
		Deep MP^2	7.4K	$66.0_{1.0}$	**94.6**$_{0.2}$	**80.9**$_{0.8}$	**76.3**$_{2.1}$	$83.9_{0.8}$	**92.4**$_{0.7}$	$90.1_{0.2}$	**83.5**
		- Router-only	192	**66.1**$_{0.5}$	**94.6**$_{0.2}$	**80.9**$_{0.8}$	$74.2_{2.2}$	**84.0**$_{0.9}$	$91.8_{0.7}$	**90.2**$_{0.2}$	83.1

Unseen Task											
Paradigm	**Backbone**	**Methods**	**Tunable Params**	**ChnSent** Acc.	**TNews** Acc.	**OCNLI** Acc.	**LCQMC** Acc.	**DRCD** F1	**C^3** Acc.	**COTE-BD** F1	**Avg.**
Gradient Descent	CPM-2 (11B)	Model Tuning	11B	$86.1_{1.8}$	-	$38.5_{1.5}$	$58.8_{1.8}$	-	$38.4_{3.7}$	-	-
		Prompt Tuning	410K	$62.1_{3.1}$	-	$37.0_{0.5}$	$51.5_{3.4}$	-	$28.2_{0.4}$	-	-
		PPT	410K	$90.7_{0.2}$	-	$41.5_{1.5}$	$55.0_{0.4}$	-	$50.2_{0.6}$	-	-
	CPT (393M)	Model Tuning	393M	$76.8_{2.9}$	$47.8_{0.8}$	$35.6_{1.6}$	$55.3_{2.1}$	$29.0_{9.7}$	$30.0_{2.5}$	$85.2_{1.4}$	51.4
		Prompt Tuning	50K	$60.6_{2.9}$	$27.0_{0.9}$	$33.0_{1.8}$	$49.2_{2.6}$	$2.9_{0.2}$	$25.5_{0.8}$	$61.9_{1.2}$	37.2
		P-Tuning v2	1.2M	$75.9_{2.3}$	$46.9_{0.8}$	$33.7_{0.2}$	$49.7_{2.2}$	$17.8_{7.9}$	$28.0_{3.7}$	$86.2_{2.1}$	48.3
		PPT	50K	$64.1_{3.4}$	$44.8_{0.9}$	$34.2_{0.7}$	$51.4_{2.1}$	$5.0_{1.4}$	$36.8_{2.4}$	$77.5_{1.0}$	44.8
		SPoT	50K	$87.0_{0.9}$	$48.2_{1.2}$	$38.7_{1.0}$	$60.9_{2.1}$	$57.8_{1.2}$	**44.9**$_{0.8}$	$88.1_{0.3}$	60.8
		Shallow MP^2	50K$\sim$400K	$90.5_{0.2}$	$51.4_{1.1}$	$53.4_{5.0}$	$72.5_{1.9}$	$67.2_{3.0}$	$44.1_{1.6}$	$88.8_{0.7}$	66.8
		Deep MP^2	1.2M$\sim$9.6M	**92.0**$_{0.1}$	**54.7**$_{0.3}$	**64.1**$_{2.3}$	**83.5**$_{1.0}$	**80.6**$_{0.9}$	$35.4_{0.9}$	**91.8**$_{0.3}$	**71.7**
Black-Box Tuning	CPT (393M)	BBT	300	$84.7_{1.7}$	$35.5_{1.7}$	$32.6_{0.4}$	$50.7_{4.0}$	-	$28.7_{1.1}$	-	-
		BBTv2	7.2K	$85.8_{0.8}$	$47.2_{1.2}$	$36.0_{1.0}$	$56.6_{2.2}$	-	$29.3_{0.4}$	-	-
		Shallow MP^2	308	$90.2_{0.4}$	$52.4_{1.0}$	$54.0_{2.7}$	$77.1_{1.8}$	$73.4_{1.1}$	$42.7_{0.9}$	$89.7_{0.4}$	68.6
		- Router-only	8	$90.4_{0.3}$	$49.9_{3.2}$	$53.3_{3.8}$	$72.6_{0.9}$	$71.5_{0.8}$	**43.7**$_{2.1}$	$88.3_{0.9}$	67.1
		Deep MP^2	7.4K	**91.7**$_{0.4}$	**55.1**$_{0.9}$	**65.7**$_{1.9}$	**84.6**$_{0.9}$	$79.2_{0.8}$	$36.0_{0.5}$	$91.5_{0.2}$	**72.0**
		- Router-only	192	**91.7**$_{0.4}$	$54.3_{0.7}$	$65.3_{2.3}$	$83.4_{1.7}$	**79.8**$_{1.2}$	$36.8_{0.9}$	**91.6**$_{0.1}$	71.8

Table 3: Main results on downstream tasks. Results on CPM-2 are taken from Gu et al. (2021) since our experimental settings are consistent. For PPT on CPM-2, we take the results of the "Unified PPT" reported in the original paper.

PTM consisting of a 20-layered shared encoder, a 4-layered understanding decoder and a 4-layered generation decoder. In our experiment, we use the encoder and the understanding decoder to compose a 24-layered PTM. We attach soft prompt to the input layer and all intermediate layers except the last layer, which has no effect on the output. Therefore, we pre-trained 24 sets of modular prompts, each corresponding to one layer of CPT. In addition to the pre-trained **Deep MP^2**, we also pre-trained a set of modular prompts that are merely attached to the input layer, denoted as **Shallow MP^2**.

We evaluate MP^2 under two learning paradigms: *gradient descent* and *black-box tuning*. For gradient descent, we consider **(1) Model Tuning**, which fine-tunes all parameters of the PTM; **(2) Prompt Tuning** (Lester et al., 2021), which prepends a sequence of soft prompt tokens to the input and only tunes the soft prompt for adaptation; **(3) P-Tuning v2** (Liu et al., 2021b): which incorporates and tunes soft prompt at every layer of the PTM. Prompt tuning and p-tuning v2 can be seen as the baselines to Shallow MP^2 and Deep MP^2, respectively. Besides, we compare with two previous prompt pre-training methods: **(4) PPT** (Gu et al., 2021), which pre-trains soft prompt on large-scale unlabeled data with self-supervised learning; and **(5) SPoT** (Vu et al., 2022), which pre-trains soft prompt with multi-task learning. For fair comparison, we reimplement PPT and SPoT with the same backbone model, i.e., CPT-large. For PPT, we pre-trained the "Unified PPT" on the same pre-training corpora as in the original paper, i.e., 10GB WuDaoCorpora (Yuan et al., 2021). For SPoT, we pre-trained a single soft prompt with the same 38 Chinese NLP tasks as used by MP^2. Therefore, experiments of SPoT can be seen as an ablation study on the effect of the modular prompt. For black-box tuning,



(a) Gradient descent

(b) Black-box tuning

Figure 3: Two-stage tuning with shallow MP^2 for initialization under two learning paradigms. The green dashed lines indicate the boundary between the two stages.

Stage	Unseen Data		Unseen Task	
	THUCNews	BQ	TNews	LCQMC
Shallow MP^2 with Black-Box Tuning				
Only Stage I	1.26	1.10	1.61	1.11
Two-Stage	14.46	7.74	25.20	6.70
Deep MP^2 with Black-Box Tuning				
Only Stage I	2.62	2.90	8.20	2.28
Two-Stage	7.88	5.57	17.44	4.51

Table 4: Comparison of training time (in minutes) between two tuning stages.

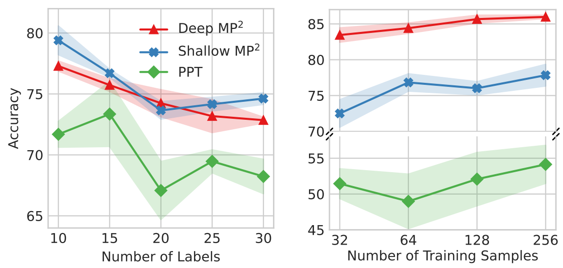
we consider two baselines: **(1) BBT** (Sun et al., 2022c), which adopts a gradient-free optimizer to tune a low-dimensional intrinsic prompt, and then randomly embeds it into the original prompt space to be concatenated with the input embedding; and **(2) BBTv2** (Sun et al., 2022a), which extends BBT by incorporating soft prompt into every layer of the PTM and uses a divide-and-conquer algorithm to alternately optimize the soft prompt at each layer.

The prompt length is set to 50 for both shallow MP^2 and deep MP^2. Each set of modular prompts is consisting of $K = 8$ soft prompts, and therefore the pre-trained routers are in the shape of 38×8. Shallow MP^2 has only one router while deep MP^2 contains 24 routers corresponding to 24 layers. Hyper-parameters and more implementation details are provided in Appendix A.

4.3 Results

Main Results. Main results on 14 downstream tasks are listed in Table 3. We report mean and standard deviation of performance over 5 runs with different random seeds. Overall, MP^2 outperforms all baselines by a large margin. By further comparison, we have the following findings: *(1) Deep Prompt vs. Shallow Prompt*: Deep prompt methods (i.e., p-tuning v2, BBTv2, and deep MP^2) significantly outperform their corresponding shallow versions (i.e., prompt tuning, BBT, and shallow MP^2). *(2) Modular Prompt vs. Single Prompt*: Shallow MP^2 achieves better performance than SPoT on 13/14 tasks, demonstrating the strong compositional generalization of the modular prompts. *(3) MRC vs. MCC*: PPT lags far behind MP^2 (and even p-tuning v2) on two MRC tasks, namely CMRC-2018 and DRCD, demonstrating the limitation of the MCC format. *(4) Pre-trained Prompt Tuning vs. Prompt Tuning From Scratch*: Pre-trained prompt tuning (i.e., PPT, SPoT, and MP^2) performs consistently better than tuning randomly initialized prompt with the same number of tunable parameters. *(5) Gradient Descent vs. Black-Box Tuning*: Without MP^2 for initialization, BBT and BBTv2 achieve better performance than prompt tuning and p-tuning v2, respectively, on most tasks but much worse performance on a few tasks such as CCPM. By using MP^2 for initialization, the gap between gradient descent and black-box tuning on these tasks are closed, and in average, BBT and BBTv2 outperform their gradient-based counterparts, showing the superiority of gradient-free optimization in few-shot learning settings.

Two-Stage Tuning. As demonstrated in Table 3, by only tuning the router (only stage I), which contains merely 8 parameters for shallow MP^2 or $8 \times 24 = 192$ parameters for deep MP^2, we can achieve surprisingly strong performance that can be comparable to two-stage tuning. For shallow MP^2, only tuning the router even outperforms two-stage tuning in average on Unseen Data tasks. To take a closer look, we demonstrate the process of two-stage tuning with shallow MP^2 for initialization in Figure 3. For both learning paradigms, we find that the best performance on the development set of the Unseen Data task (here is the BQ task) can be observed in stage I, where we only tune the router to reuse pre-trained prompts. On Unseen Task (here is the LCQMC task), we observe improve-



(a) On number of labels (b) On sample efficiency

Figure 4: Comparison of MP^2 and PPT with varying numbers of labels and training samples.

ment of performance during stage II. In Table 4, we compare the training time of the two stages to show the high efficiency of stage I when using black-box tuning. Results suggest that learning to combine instead of tuning the prompts is a promising way to achieve fast adaptation to downstream tasks.

On Many-Label Classification Tasks. In contrast to PPT that is pre-trained to perform up to 16-label classification, our proposed MP^2 unifies tasks into the MRC format such that it can generalize to downstream tasks with varying numbers of labels. To simulate tasks with different numbers of labels, we extract subsets with 10/15/20/25/30 labels from the IFLYTEK dataset, which contains 119 labels in total. We follow the same procedure (§4.1) to generate train/dev/test splits from the extracted subsets. As shown in Figure 4(a), there is a sharp decline in the accuracy of PPT when the number of labels exceeds 16. By contrast, the performance of MP^2 is decreasing more slowly and steadily as the number of labels increases, demonstrating the superiority of the MRC format.

On Sample Efficiency. We compare MP^2 and PPT with different numbers of training samples on the LCQMC task. As shown in Figure 4(b), increasing training samples generally confers improved performance for both MP^2 and PPT while MP^2 consistently outperforms PPT under varying numbers of training samples. In addition, the gap between MP^2 and PPT cannot be easily filled with enlarged training set.

Task Partitions Induced From the Router. We take a closer look at the learned router and find that non-trivial task partitions can be induced from it. For simplicity, we focus on the shallow MP^2, which has only one router. There are totally 8 modular prompts corresponding to $2^8 = 256$ possible combinations. We perform a hierarchical clustering on the router learned on 38 upstream tasks and visualize the task partitions in Figure 5. The 38 upstream tasks can be partitioned into 8 groups. For instance, group A is mainly comprised of topic classification tasks; group D contains all the sentiment analysis tasks; group C and E are all comprised of NLI tasks, among which group E covers all the "Zhidao" tasks, which are question-answer matching tasks.

Figure 5: Task partitions induced from the router. Similar tasks are assigned similar subsets of prompts.

5 Conclusion

This work aims to bridge the gap between pre-training and fine-tuning of soft prompt tuning for few-shot learning. To achieve this, we extend the soft prompt in two dimensions, depth and width. The extended prompt, named deep modular prompt, is pre-trained on a mixture of 38 public Chinese NLP tasks, which are reformulated into the MRC format. For adaptation to downstream tasks, we propose the two-stage tuning, where we first learn to combine and reuse pre-trained prompts and then tune the selected prompts with gradient descent or black-box optimization. Extensive experiments on 14 downstream tasks demonstrate that, the **Multi**-task **Pre**-trained **M**odular **P**rompt (**MP^2**) significantly outperforms prompt tuning, full model tuning, and previous prompt pre-training methods, namely PPT and SPoT. Surprisingly, we demonstrate that MP^2 can achieve extremely fast adaptation to downstream tasks by only learning to combine pre-trained prompts.

Limitations

In this work, we demonstrate the effectiveness of the proposed MP^2 with the backbone PTM of CPT-large on a set of Chinese NLP tasks. Due to the expensive pre-training cost, we did not explore MP^2 on other PTMs with varying sizes, pre-training objectives and architectures. Besides, it is also unknown how does the number of pre-training tasks affect the performance of MP^2. For resource-rich languages such as English and Chinese, it would be promising for MP^2 to be well-performed since one can easily collect sufficient public upstream tasks. Nevertheless, for low-resource languages or domains, the effect of MP^2 is still under-explored.

Ethics Statement

The proposed MP^2 is a parameter-efficient approach for few-shot learning. In addition, we demonstrate that MP^2 can achieve highly efficient adaptation to a target task by only tuning a few parameters. Therefore, this work helps reduce computation costs and carbon emissions, and can facilitate the adaptation of PTMs to low-resource downstream tasks. Though all the datasets used in our experiments are publicly available and have not been reported to carry social bias against any sensitive attributes, and the proposed approach would not explicitly introduce new negative societal impacts, more work is still needed to investigate the potential unfairness in these datasets.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 62236004 and No. 62022027) and CAAI-Huawei MindSpore Open Fund.

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 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu, and Buzhou Tang. 2018. The BQ corpus: A large-scale domain-specific Chinese corpus for sentence semantic equivalence identification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4946–4951, Brussels, Belgium. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Shizhe Diao, Xuechun Li, Yong Lin, Zhichao Huang, and Tong Zhang. 2022. Black-box prompt learning for pre-trained language models. *CoRR*, abs/2201.08531.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *CoRR*, abs/2203.06904.

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 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3816–3830. Association for Computational Linguistics.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. PPT: pre-trained prompt tuning for few-shot learning. *CoRR*, abs/2109.04332.

Nikolaus Hansen and Andreas Ostermeier. 2001. Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.*, 9(2):159–195.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *CoRR*, abs/2110.04366.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence S. Moss. 2020. OCNLI: original chinese natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 3512–3526. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

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 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Yanzeng Li, Tingwen Liu, Diying Li, Quangang Li, Jinqiao Shi, and Yanqiu Wang. 2018. Character-based bilstm-crf incorporating pos and dictionaries for chinese opinion target extraction. In *Asian Conference on Machine Learning*, pages 518–533. PMLR.

Ziran Li, Ning Ding, Zhiyuan Liu, Haitao Zheng, and Ying Shen. 2019. Chinese relation extraction with multi-grained information and external linguistic knowledge. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4377–4386, Florence, Italy. Association for Computational Linguistics.

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. *CoRR*, abs/2107.13586.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021c. GPT understands, too. *CoRR*, abs/2103.10385.

Xin Liu, Qingcai Chen, Chong Deng, Huajun Zeng, Jing Chen, Dongfang Li, and Buzhou Tang. 2018. LCQMC: A large-scale chinese question matching corpus. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1952–1962. Association for Computational Linguistics.

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. *CoRR*, abs/1907.11692.

Kanika Madan, Nan Rosemary Ke, Anirudh Goyal, Bernhard Schölkopf, and Yoshua Bengio. 2021. Fast and slow learning of recurrent independent mechanisms. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

Jonas Mockus. 1974. On bayesian methods for seeking the extremum. In *Optimization Techniques, IFIP Technical Conference, Novosibirsk, USSR, July 1-7, 1974*, volume 27 of *Lecture Notes in Computer Science*, pages 400–404. Springer.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *CoRR*, abs/2105.11447.

Edoardo Maria Ponti, Alessandro Sordoni, and Siva Reddy. 2022. Combining modular skills in multitask learning. *CoRR*, abs/2202.13914.

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 2021, Online, June 6-11, 2021*, pages 5203–5212. Association for Computational Linguistics.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *SCIENCE CHINA Technological Sciences*, 63:1872–1897.

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: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 255–269. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Chih-Chieh Shao, Trois Liu, Yuting Lai, Yiying Tseng, and Sam Tsai. 2018. DRCD: a chinese machine reading comprehension dataset. *CoRR*, abs/1806.00920.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. CPT: A pre-trained unbalanced transformer for both chinese language understanding and generation. *CoRR*, abs/2109.05729.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2020a. Investigating prior knowledge for challenging chinese machine reading comprehension. *Trans. Assoc. Comput. Linguistics*, 8:141–155.

Tianxiang Sun, Zhengfu He, Hong Qian, Yunhua Zhou, Xuanjing Huang, and Xipeng Qiu. 2022a. Bbtv2: Towards a gradient-free future with large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 3916–3930. Association for Computational Linguistics.

Tianxiang Sun, Xiangyang Liu, Xipeng Qiu, and Xuanjing Huang. 2022b. Paradigm shift in natural language processing. *Machine Intelligence Research*, 19:169–183.

Tianxiang Sun, Yunfan Shao, Xiaonan Li, Pengfei Liu, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020b. Learning sparse sharing architectures for multiple tasks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8936–8943. AAAI Press.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022c. Black-box tuning for language-model-as-a-service. In *Proceedings of the 39th International Conference on Machine Learning, ICML 2022, Baltimore, Maryland, USA*.

Hongxuan Tang, Hongyu Li, Jing Liu, Yu Hong, Hua Wu, and Haifeng Wang. 2021. Dureader_robust: A chinese dataset towards evaluating robustness and generalization of machine reading comprehension in real-world applications. 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 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 955–963. Association for Computational Linguistics.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. Spot: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5039–5059. Association for Computational Linguistics.

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Ke Xu, Julian McAuley, and Furu Wei. 2021. Blow the dog whistle: A chinese dataset for cant understanding with common sense and world knowledge. *arXiv preprint arXiv:2104.02704*.

Jingjing Xu, Ji Wen, Xu Sun, and Qi Su. 2017. A discourse-level named entity recognition and relation extraction dataset for chinese literature text. volume abs/1711.07010.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, Yin Tian, Qianqian Dong, Weitang Liu, Bo Shi, Yiming Cui, Junyi Li, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Zhe Zhao, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, Kyle Richardson, and Zhenzhong Lan. 2020. CLUE: A Chinese language understanding evaluation benchmark. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4762–4772, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019a. Paws-x: A cross-lingual adversarial dataset for paraphrase identification. *arXiv preprint arXiv:1908.11828*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019b. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1–9. Association for Computational Linguistics.

Fan Zhang, Duyu Tang, Yong Dai, Cong Zhou, Shuangzhi Wu, and Shuming Shi. 2022a. Skillnet-nlu: A sparsely activated model for general-purpose natural language understanding.

Ningyu Zhang, Mosha Chen, Zhen Bi, Xiaozhuan Liang, Lei Li, Xin Shang, Kangping Yin, Chuanqi Tan, Jian Xu, Fei Huang, Luo Si, Yuan Ni, Guotong Xie, Zhifang Sui, Baobao Chang, Hui Zong, Zheng Yuan, Linfeng Li, Jun Yan, Hongying Zan, Kunli Zhang, Buzhou Tang, and Qingcai Chen. 2022b. CBLUE: A chinese biomedical language understanding evaluation benchmark. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 7888–7915. Association for Computational Linguistics.

Hui Zong, Jinxuan Yang, Zeyu Zhang, Zuofeng Li, and Xiaoyan Zhang. 2021. Semantic categorization of chinese eligibility criteria in clinical trials using machine learning methods. *BMC Medical Informatics Decis. Mak.*, 21(1):128.

A Implementation Details

A.1 Upstream Pre-training

MP^2. MP^2 is pre-trained on 38 upstream tasks using an Adam optimizer with batch size of 32 for 1M steps. During each forward computation, we first randomly select a task and then fetch a batch of training data corresponding to the selected task. By this, the number of learning steps on each task is expected to be identical. As demonstrated in Table 5, the fast and slow learning (FSL) can be beneficial to deep MP^2, and therefore we use two-speed learning rate for pre-training the routers and the prompts of deep MP^2. In particular, the learning rate of the routers is 5e-4, and the learning rate of the prompts is 1e-4. For shallow MP^2, we use a single learning rate of 1e-3 for the router and the modular prompts. The prompt length is set to 50 for both shallow MP^2 and deep MP^2. For shallow MP^2 and each layer of the deep MP^2 , we allocate $K = 8$ modular prompts and one router to combine them. In addition to the routers and the prompts, we also train the randomly initialized MRC head on the top of the PTM. The original parameters of the PTM are frozen during pre-training. We run pre-training on NVIDIA A100 GPUs.

Baselines. For fair comparison, we also reimplement PPT and SPoT with the same backbone model as MP^2, i.e., CPT-large. For pre-training PPT, we implement the "Unified PPT" variant, which is to formulate tasks into a unified MCC format, to support a variety of downstream tasks. We follow the experimental setup in the original paper and use 10GB data sampled from the WuDaoCorpora for pre-training. We train for 400K steps using an Adam optimizer with batch size of 32 and learning rate of 3e-2. For SPoT, we pre-trained a single soft prompt on the same 38 upstream tasks as used by MP^2 using an Adam optimizer with batch size of 32 and learning rate of 3e-3 for 650K steps. Though the numbers of training steps for PPT and SPoT are less than MP^2, they are sufficient for convergence due to their limited numbers of parameters. To be consistent with MP^2, we set prompt length to 50 for PPT and SPoT.

A.2 Downstream Fine-tuning

We use the two-stage tuning to adapt MP^2 to various downstream tasks. In stage I, we only tune the router(s)[4] while keeping all other parameters frozen. In stage II, we fix the learned router(s) and only fine-tune the modular prompts selected by the router(s). The implementation details of the two-stage tuning can be different for gradient descent and black-box tuning. We provide a graphical illustration of the two-stage tuning using gradient descent and black-box tuning in Figure 6. **For gradient descent**, we fine-tune MP^2 for 1K epochs on each task, where the first 500 epochs as stage I and the last 500 epochs as stage II. For the shallow/deep MP^2, we use an Adam optimizer with learning rate of 1e-2/3e-3 for tuning the router(s) (stage I) and learning rate of 3e-4/2e-5 for tuning the prompts (stage II). **For black-box tuning**, we fine-tune shallow/deep MP^2 for 8K iterations (model forward computes) on each task, where the first 200/100 iterations as stage I and the rest as stage II. In stage I, we use Bayesian optimization (BO) with the acquisition function of upper confidence bound (UCB) with $\kappa = 2$ to tune the parameters of the router(s). In stage II, we use CMA-ES to optimize the prompts. For shallow MP^2, we use $\mu = 0$ and $\sigma = 0.1$ for initialization of the CMA-ES. For deep MP^2, we follow BBTv2 and use the divide-and-conquer algorithm to alternately optimize the prompt at each layer. For optimization of the prompt at the embedding layer, we initialize CMA-ES with $\mu = 0$ and $\sigma =$ 5e-2. For optimization of the prompt at intermediate layers, we adopt $\mu = 0$ and $\sigma =$ 1e-2. All the hyper-parameters are tuned manually in a lightweight manner on development sets. We perform fine-tuning a single NVIDIA 3090 GPU.

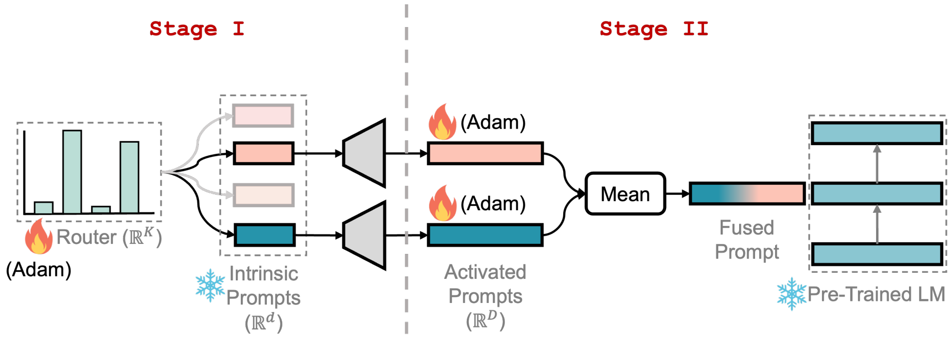
Methods	ChnSent	TNews	LCQMC	DRCD
		Shallow MP^2		
w/o FSL	90.46 $_{0.16}$	51.36 $_{1.12}$	72.50 $_{1.92}$	67.20 $_{2.96}$
w/ FSL	89.36 $_{0.63}$	51.36 $_{1.30}$	70.42 $_{1.27}$	58.96 $_{0.73}$
		Deep MP^2		
w/o FSL	91.61 $_{0.18}$	55.23 $_{0.29}$	82.30 $_{1.28}$	78.69 $_{0.72}$
w/ FSL	92.02 $_{0.11}$	54.71 $_{0.31}$	83.45 $_{1.00}$	80.64 $_{0.87}$

Table 5: Ablation of fast and slow learning (FSL).

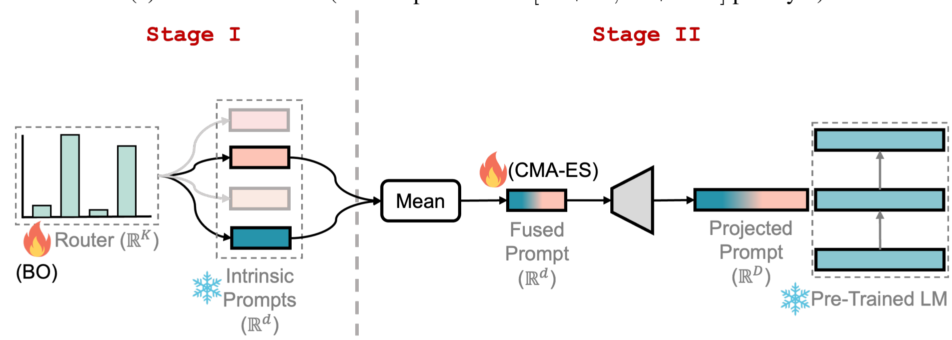
B Additional Results

Ablation of Fast and Slow Learning. We conduce ablation study on fast and slow learning (FSL), which is to assign different learning rates to routers

[4]A single router for shallow MP^2 and 24 routers for deep MP^2.



(a) Gradient descent (tunable parameters: $[K + D, K + KD]$ per layer)



(b) Black-box tuning (tunable parameters: $K + d$ per layer)

Figure 6: Illustration of the two-stage tuning for gradent descent and black-box tuning. For black-box tuning, which is a gradient-free optimization approach that cannot well handle high-dimensional optimization, we perform pre-fusion to obtain a low-dimensional fused prompt for optimization.

Dataset	Source
ChnSent	`https://github.com/SophonPlus/ChineseNlpCorpus`
TNews	`https://github.com/fatecbf/toutiao-text-classfication-dataset/`
OCNLI	Hu et al. (2020)
LCQMC	Liu et al. (2018)
DRCD	Shao et al. (2018)
C^3	Sun et al. (2020a)
COTE-BD	Li et al. (2018)

Table 6: Sources of the 7 downstream tasks in the UNSEEN TASK track.

and prompts. As demonstrated in Table 5, FSL exhibits positive effect on downstream tasks to deep MP^2 and negative effect to shallow MP^2. Therefore, we retain the shallow MP^2 pre-trained without FSL and the deep MP^2 pre-trained with FSL in our experiments.

C MRC Format

We unify upstream and downstream tasks into the machine reading comprehension (MRC) format, which takes as input a *context* and a *query*, and outputs the *answer* of the query. For topic classification and sentence-pair classification tasks, we use the original input text as the context and construct a query containing all valid labels. The context and the constructed query are concatenated and fed into the model. The model is trained to extract the answer in the query by predicting its start and end positions. For more complicated tasks such as relation extraction and poem understanding, we manually design task-specific templates to convert inputs to the desired contexts and queries. Some examples are shown in Table 7.

D Additional Details of Tasks

D.1 Upstream Tasks

Table 8 contains details of the 38 upstream tasks. We only use the training sets during pre-training. For tasks that also serve as a downstream task in the UNSEEN DATA track, we remove a small portion of training samples from pre-training to avoid data leakage.

D.2 Downstream Tasks

The downstream tasks are divided into two tracks, UNSEEN DATA and UNSEEN TASK. The tasks in the UNSEEN DATA track are a subset of upstream task, for which the details have been provided in Table 8. For the 7 tasks in the UNSEEN TASK track, we provide the sources in Table 6.

Dataset	Task	Template
Amazon	TC	打分：$\langle S\rangle$的评价是？选项：非常差，较差，一般，较好，非常好。 (Rating: $\langle S\rangle$ What do you think about it? Options: very bad, bad, okay, good, very good.)
ChnSent	TC	情感分析：$\langle S\rangle$的情感是？选项：负面，正面。 (Sentiment analysis: What is the sentiment of $\langle S\rangle$? Options: negative, positive.)
TNews	TC	主题识别：$\langle S\rangle$的主题是？选项：房产，汽车，金融，体育，文化... (Topic classification: What is the topic of $\langle S\rangle$? Options: housing, car, finance, sports, culture, ...)
FinRe	TC	关系判别：$\langle S1\rangle$和$\langle S2\rangle$在句子中的关系是？选项：未知，注资，拥有，纠纷，自己... (Relation classification: What is the relationship between $\langle S1\rangle$ and $\langle S2\rangle$? Options: unknown, capital injection, possess, dispute, oneself...)
CMNLI	NLI	意思判别：$\langle S1\rangle$与$\langle S2\rangle$的关系是？选项：矛盾，蕴含，中立。 (Textual entailment: What is the relationship between $\langle S1\rangle$ and $\langle S2\rangle$? Options: contradiction, entailment, neutral.)
CCPM	MCQA	诗句理解：与句子$\langle S\rangle$最相近的诗句是？选项：$\langle A1\rangle$，$\langle A2\rangle$，$\langle A3\rangle$，$\langle A4\rangle$。 (Poem understanding: Which verse comes closest to $\langle S\rangle$? Options: $\langle A1\rangle$, $\langle A2\rangle$, $\langle A3\rangle$, $\langle A4\rangle$.)
C^3	MCQA	阅读选择：文档：$\langle S1\rangle$，问题：$\langle S2\rangle$，选项：$\langle A1\rangle$，$\langle A2\rangle$，$\langle A3\rangle$。 (Reading comprehension: Document: $\langle S1\rangle$, Question: $\langle S2\rangle$, Options: $\langle A1\rangle$, $\langle A2\rangle$, $\langle A3\rangle$.)

Table 7: Example templates to formulate non-MRC tasks into the MRC format.

ID	Dataset	Task	Domain	|Train|	|Dev|	|Test|	|Labels|	Reference
1	AFQMC	NLI	Financial	31k	3k	4k	2	Xu et al. (2020)
2	Paws	NLI	General	44k	5k	2k	2	Yang et al. (2019a)
3	CMNLI	NLI	General	380k	12k	12k	3	Xu et al. (2020)
4	CSL	NLI	Academic	18k	2k	3k	2	Xu et al. (2020)
5	BQ	NLI	Financial	90k	10k	10k	2	Chen et al. (2018)
6	CHIP-STS	NLI	Biomedical	14k	2k	4k	2	Zhang et al. (2022b)
7	KUAKE-QQR	NLI	Clinical	14k	2k	2k	3	Zhang et al. (2022b)
8	XNLI	NLI	General	380k	12k	2k	3	Conneau et al. (2018)
9	NLPCC-DBQA	NLI	General	170k	12k	41k	2	http://tcci.ccf.org.cn/conference/2016
10	Finance-zhidao	NLI	Financial	64k	12k	38k	2	https://github.com/SophonPlus/ChineseNlpCorpus
11	Law-zhidao	NLI	Law	23k	3k	7k	2	https://github.com/SophonPlus/ChineseNlpCorpus
12	Liantong-zhidao	NLI	Telecom	150k	12k	20k	2	https://github.com/SophonPlus/ChineseNlpCorpus
13	Nonghang-zhidao	NLI	Financial	29k	3k	4k	2	https://github.com/SophonPlus/ChineseNlpCorpus
14	Touzi-zhidao	NLI	Investment	487k	12k	29k	2	https://github.com/SophonPlus/ChineseNlpCorpus
15	Baoxian-zhidao	NLI	Insurance	5k	0.6k	2k	2	https://github.com/SophonPlus/ChineseNlpCorpus
16	Dianxin-zhidao	NLI	Telecom	99k	11k	31k	2	https://github.com/SophonPlus/ChineseNlpCorpus
17	THUCNews	TC	General	45k	5k	5k	10	https://github.com/thunlp/THUCTC
18	CHIP-CTC	TC	Biomedical	23k	8k	10k	44	Zong et al. (2021)
19	FinRe	TC	Financial	12k	1k	1k	44	Li et al. (2019)
20	Fudan-TC	TC	General	9k	1k	10k	20	Not found†
21	KUAKE-QIC	TC	Clinical	6k	0.7k	2k	11	Zhang et al. (2022b)
22	NLPCC-TC	TC	General	6k	0.7k	2k	2	http://tcci.ccf.org.cn/conference/2016
23	Amazon	TC	Shopping review	3.6M	12k	41k	5	https://github.com/SophonPlus/ChineseNlpCorpus
24	DianPing	TC	Shopping review	2.6M	12k	30k	5	https://github.com/SophonPlus/ChineseNlpCorpus
25	DMSC	TC	Movie review	1.6M	12k	92k	5	https://github.com/SophonPlus/ChineseNlpCorpus
26	Online-Shopping	TC	Shopping review	45k	5k	6k	2	https://github.com/SophonPlus/ChineseNlpCorpus
27	Waimai	TC	Shopping review	8k	0.8k	2k	2	https://github.com/SophonPlus/ChineseNlpCorpus
28	Weibo-sentiment	TC	General	76k	8k	24k	5	https://github.com/SophonPlus/ChineseNlpCorpus
29	Toutiao-TC	TC	General	321k	12k	11k	14	https://github.com/aceimnorstuvwxz/toutiao-text-classfication-dataset
30	SanWen	TC	Literature	13k	1k	2k	10	Xu et al. (2017)
31	CLUE-WSC	CR	General	1k	0.1k	0.3k	2	Xu et al. (2020)
32	COTE-DP	KE	Shopping review	16k	2k	5k	N/A	Li et al. (2018)
33	COTE-MFW	KE	Shopping review	26k	3k	8k	N/A	Li et al. (2018)
34	DuReader-Checklist	MRC	General	3k	0.3k	1k	N/A	https://github.com/baidu/DuReader
35	DuReader-Robust	MRC	General	13k	1k	1k	N/A	Tang et al. (2021)
36	CMRC-2018	MRC	General	8k	0.9k	3k	N/A	Xu et al. (2020)
37	CCPM	MCQA	Poem	19k	2k	3k	4	https://github.com/SophonPlus/ChineseNlpCorpus
38	DogWhistle	MCQA	General	218k	12k	29k	4	Xu et al. (2021)
	Total	-	-	10.7M	213k	499k	-	-

Table 8: Pre-training datasets for MP^2. NLI: natural language inference. TC: text classification. CR: coreference resolution. KE: keyword extraction. MRC: machine reading comprehension. MCQA: multiple choice question answering. † We did not find the official source of the Fudan-TC dataset.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
The limitations are discussed in the first section after the conclusion.

☑ A2. Did you discuss any potential risks of your work?
The potential risks are discussed in the first section after the conclusion.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and 1. Introduction.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

4.1 Datasets and Tasks and Appendix D Additional Details of Tasks.

☑ B1. Did you cite the creators of artifacts you used?
4.1 Datasets and Tasks and Appendix D Additional Details of Tasks.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
All the datasets used in the submission are publicly accessible for research use, as listed in Table 6 and Table 8.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
all the datasets used in the submission are publicly accessible for research use

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Appendix D Additional Details of Tasks.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
4.1 Datasets and Tasks and Appendix D Additional Details of Tasks.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
4.1 Datasets and Tasks.

C ☑ Did you run computational experiments?

4 Experiments, Appendix A.1 Upstream Pre-training, Appendix A.2 Downstream Fine-tuning

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4.2 Backbones and Baselines, Appendix A.1 Upstream Pre-training, Appendix A.2 Downstream Fine-tuning

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A.1 Upstream Pre-training, Appendix A.2 Downstream Fine-tuning

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4.3 Results

☒ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
We did not use existing packages.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.