# Explore the Principles of Prompt Tuning and the Progress of Research

*Tongxin Zheng*

College of Computer Science and Technology, Jilin University, Changchun , Jilin , 130012, China

**Abstract.** Prompt Tuning is a lightweight fine-tuning method that demonstrates efficient task adaptation and parameter efficiency for pre-trained language models (PLMs). Prompt Tuning highlights an important contribution to the advancement of NLP technology. The purpose of this paper is to explore the basic principles and methods of Prompt Tuning and to analyze the design features of discrete and continuous Prompts and their application performance in different scenarios. It is found that discrete Prompt performs well in interpretability and simple task adaptation, while continuous Prompt is more advantageous in complex tasks and cross-domain generalization; meanwhile, Prompt Tuning significantly improves the model performance in less sample scenarios, and the parameter efficiency is several times higher than that of traditional fine-tuning. However, the dependence of Prompt Tuning on Prompt design and the limitation of generalization to specific tasks still need to be further optimized. This paper hopes to provide an important reference for future theoretical research and practical applications of Prompt Tuning.

## 1 Introduction

In recent years, pre-trained language models (PLMs) such as BERT, GPT-3 and T5 have made breakthroughs in the field of natural language processing (NLP). These models have demonstrated strong language comprehension and generation capabilities through large-scale corpus pre-training. However, the traditional Fine-tuning (FT) approach requires tuning all model parameters for each downstream task, resulting in high computational costs, high storage requirements, and poor performance in low-sample scenarios. As the model size increases (e.g., GPT-3 has 175 billion parameters), the resource requirements of full-parameter fine-tuning are further exacerbated, making it difficult to apply in edge devices or resource-constrained environments.

To address the above challenges, Prompt Tuning emerged as a lightweight fine-tuning paradigm. The parametric efficiency by adding continuous Prompt vectors to the input. Prompt Tuning can achieve similar or even better performance in small-sample learning scenarios with tens of times the parameter efficiency compared to traditional full-parameter fine-tuning. This approach is particularly suitable for resource-constrained environments or tasks that require rapid adaptation.

zhengtx2121@mails.jlu.edu.cn

Similarly, Schick and Schütze demonstrated strong generalizability by analyzing the application of Prompt Tuning in a zero-sample scenario, verifying that it can still effectively guide the model to accomplish diverse tasks without the need for additional training data[1]. Together, these studies reveal the potential of Prompt Tuning in terms of efficiency and flexibility, and at the same time point out the direction for optimizing Prompt design and improving generalization ability.

The purpose of this paper is to systematically review the basic concepts of Prompt Tuning, its classification methods and its research progress in natural language processing, and to provide theoretical and practical references for the development of this field. First, this paper introduces the principle and core advantages of Prompt Tuning. Subsequently, based on the classification of discrete Prompt and continuous Prompt, it analyzes its design features and applicable scenarios. Best of all, we further sort out the related research results and explore its application performance and future direction in task adaptability, less sample learning and model generalization.

# 2 Prompt tuning basic concepts

## 2.1 Definitions

Prompt Tuning is a novel task adaptation method for PLMs, the core idea of which is to guide the model to accomplish downstream tasks without adjusting the pre-trained parameters by introducing specific prompts (Prompts) at the input side. Unlike the traditional Full-parameter Fine-tuning (Fine-tuning), Prompt Tuning freezes all the weights of the PLM and optimizes only the task-relevant part of the prompts, thus significantly reducing the computational and storage costs. Prompts can be discrete text in natural language form or continuous trainable embedding vectors. This approach makes full use of the general language knowledge acquired by the PLM in the pre-training phase, so that it shows strong generalization ability in Few-shot or even Zero-shot scenarios.

The concept of Prompt Tuning first originated from GPT-3 practiced in In-context Learning, and was subsequently systematized as a stand-alone fine-tuning paradigm. Its theoretical foundation is built on PLM's sensitivity to input distributions, and through well-designed cues, the model is able to transform downstream tasks into a form that is consistent with pre-training objectives (e.g., masked language modeling or autoregressive generation), thus enabling efficient task migration.

## 2.2 Classification

According to the form of prompts and optimization methods, Prompt Tuning can be divided into two main categories: discrete Prompt and continuous Prompt. As shown in Figure 1, Discrete Prompts mainly include methods such as PET and iPET, which guide the model to complete the task by designing fixed natural language prompts (e.g., specific words or phrases), emphasize the interpretability and intuitiveness of the prompts, and are suitable for scenarios that are more sensitive to task descriptions. Continuous Prompts, on the other hand, include methods such as LM-BFF, Prefix-Tuning, Prompt Tuning, Progressive Prompts, P-Tuning, P-Tuning v2, and PPT, which are optimized by adding learnable continuous vectors (usually in the form of embedding vectors) to the model inputs. prompts, which are more flexible and suitable for complex tasks and cross-domain generalization. Figure 1 illustrates prompt tuning based on discrete and continuous classification.

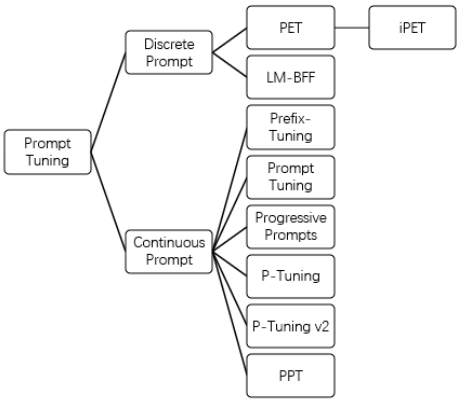


**Fig. 1** Prompt tuning based on discrete and continuous categorization

A discrete Prompt consists of words or phrases in natural language, selected directly from the PLM's vocabulary. For example, in the sentiment classification task, the prompt template "The sentiment of this text is [MASK]" can be designed, where [MASK] is to be predicted by the model. The advantage of discrete Prompt is that it is intuitive and interpretable, but its design relies on manual experience, is less flexible, and is difficult to adapt to the needs of diverse tasks. Continuous Prompt is represented by a set of trainable embedding vectors that are not restricted to discrete tokens in the vocabulary. e.g., "[V1][V2]... [Vj]" represents a set of continuous vectors whose parameters are optimized by gradient descent. The advantages of Continuous Prompt lie in its flexibility and adaptivity to capture task-specific semantic features through end-to-end training, but it is less interpretive and more dependent on the initialization strategy and training process.

## 2.3 Realization steps

The implementation steps of cue learning consist of three core components: template design, label mapping definition, and model training and optimization. Together, these steps translate downstream task requirements into a PLM-processable format and improve task performance while maintaining the stability of pre-trained knowledge.

Templates are designed to construct cue structures based on specific task requirements in order to effectively guide model output. Cue templates usually contain placeholders (e.g., [MASK]) and are spliced with the original input to form a complete input sequence. For example, in a text categorization task, for the input "Movie is great", a template can be designed to expand it to "Movie is great. The sentiment is [MASK]". The templates should be designed in such a way that they are consistent with the pre-training objectives of the PLM, while at the same time adequately characterizing the task. Discrete cues are selected directly from the vocabulary through words or phrases in natural language, which are intuitive and highly interpretable, but their design relies on human experience and is less flexible. In contrast, continuous cues consist of a set of trainable embedding vectors, e.g., "[V1][V2] ... [Vj]", which generates initial vectors by random initialization or predefined strategies, can flexibly adapt to diverse tasks, but has weak interpretability.

The label mapping definition is responsible for establishing the correspondence between the placeholders in the prompt and the task labels, thus transforming the task objectives into a PLM-understandable output form. Taking emotion classification as an example, if the [MASK] prediction is "positive", it is mapped to positive emotion; if it is "negative", it corresponds to negative emotion. The mapping can be as simple as a single token (e.g.,

"good") to a label, or it can involve complex multi-token predictions (e.g., "very good"). This step ensures that the output of the model accurately reflects the task requirements while remaining consistent with the design of the cue template.

Model training and optimization optimizes the cue component by annotating the data, while typically freezing the parameters of the PLM to preserve its pre-training knowledge. For discrete cues, the training process calculates losses (e.g., cross-entropy losses) by predicting the probability distribution of the placeholders, thus adjusting the model output. Continuous cues, on the other hand, directly optimize the parameters of the embedding vectors via gradient descent to capture task-specific semantic features. The optimization goal is to maximize the task performance of the model for a given cue while avoiding over-reliance on the initialization strategy or training process. Optimizers such as Adam are often used in training and combined with the early stopping method to select the best-performing model to ensure efficient convergence and model stability.

# 3 Research results

## 3.1 discrete prompt based approach

### *3.1.1 Pattern-Exploiting Training (PET)*

PET is a semi-supervised training method based on discrete prompts, which aims to improve the performance of PLM in few-shot learning. The method transforms the input samples into cloze-style phrases, combines Pattern and Verbalizer, and reconstructs the downstream task into a masked language modeling problem. The core process of PET includes: fine-tuning multiple PLMs with a small amount of labeled data, generating a large unlabeled dataset with soft labels based on different patterns, and finally training a standard classifier on this dataset. In addition, its iterative variant iPET further optimizes the performance by gradually expanding the size of the training set.

The introduction of PET marks the expansion of the Prompt Tuning field from zero-shot task description to few-shot semi-supervised learning. Its innovation lies in the clever combination of PLM pre-trained knowledge and task-specific natural language prompts, which significantly reduces the demand for labeled data. Compared with traditional full-parameter fine-tuning, PET freezes PLM parameters and only adjusts the task-related soft label generation process, reflecting the characteristics of parameter efficiency.

In the experimental verification by Schick et al., PET performed well on multiple benchmark datasets[1]. For example, in the Yelp sentiment classification task, when the number of training samples was only 10, PET's accuracy reached 52.9%, far exceeding the 21.1% of supervised learning, and iPET further improved to 57.6%. In the MNLI natural language inference task, when the scale was 1000, PET's accuracy was as high as 85.3%, surpassing the 73.1% of supervised learning. In addition, PET also showed superiority in multilingual tasks (such as x-stance). For example, on the German dataset, when the number of training samples was 1000, PET's F1 score was 66.4, a significant improvement over the 43.3 of supervised learning. Compared with contemporary semi-supervised methods (such as UDA and MixText), PET does not require complex data augmentation techniques and can achieve better results with intuitive pattern design.

The research significance of PET is that it provides a seminal framework for discrete Prompt methods, inspiring subsequent work such as LM-BFF on automated Prompt design [2] . However, PET is highly dependent on the quality of Patterns, and the limitations of manual design restrict its generalizability and may face the problem of insufficient semantic

capture in complex tasks. It can be further improved in the future by automating Pattern generation or combining multimodal data.

#### *3.1.2 LM-BFF*

LM-BFF is a few-shot fine-tuning method for medium-sized PLMs (such as RoBERTa), which aims to improve the performance of classification and regression tasks through prompt optimization. Inspired by GPT-3[3], this method combines the prompt idea of PET and proposes two innovations: one is automatic prompt generation, including template generation based on the T5 model and label word selection by pruning search, which reduces the burden of manual design; the other is a dynamic demonstration sampling strategy, which selects semantically similar examples to integrate into the input context and enhances the model's understanding of the task. LM-BFF assumes that only a small amount of labeled data (such as 16 samples per class) is used and the PLM parameters are updated during fine-tuning, which is suitable for practical scenarios with limited resources.

Compared with PET, LM-BFF is more radical in the few-shot setting. It abandons the semi-supervised nature of PET, relies only on small-scale annotated data, and is extended to regression tasks. In terms of method, LM-BFF converts the input into a prompt containing [MASK] (such as "<S1> It was [MASK]"), predicts the label word (such as "great" or "terrible") through pre-trained weights, introduces T5 to generate diversified templates (such as "<S1> A [MASK] piece"), and uses SBERT to select similar demonstration samples to optimize the context. In the experimental verification by Gao et al., the effectiveness of LM-BFF was verified on 15 NLP tasks (including SST-2, MNLI, etc.)[2]. Taking RoBERTa-large as an example, when the number of samples per class K=16, LM-BFF achieves an accuracy of 93.0% on SST-2, an increase of 11.6% over standard fine-tuning (81.4%); and 77.5% on SNLI, an increase of about 30%. Automatic prompts match or surpass manual designs, and demonstration sampling further improves performance, with an average improvement of 11%..

The research significance of LM-BFF lies in promoting the automation and universality of the discrete prompt method. Its T5 template generation and dynamic demonstration strategy provide ideas for subsequent work. However, its performance depends on whether the task is easy to transform into a fill-in-the-blank problem, and the automatically generated prompt may introduce bias, and the high variance problem also needs to be optimized. In the future, we can explore a wider range of task adaptability and stability improvements.

### 3.2 Continuous Prompt Based Approach

#### *3.2.1 Prefix-Tuning*

Prefix-Tuning is a lightweight fine-tuning method for Natural Language Generation (NLG) tasks that steers PLMs by optimizing successive, prefixed task-specific vectors (Prefix).

This approach is inspired by Prompting and GPT-3 context learning. The approach is inspired by Prompting and GPT-3's context learning, but different from discrete Prompt, Prefix-Tuning introduces continuous vectors as "virtual tokens", which influence the subsequent token generation through the Transformer's attention mechanism. The core innovation is that only about 0.1% of the task-specific parameters are updated (e.g., the 250K parameters of GPT-2 are compared with its 774M parameters), which realizes efficient storage and modular design for multi-task scenarios.

In contrast to traditional full-parameter fine-tuning, Prefix-Tuning extends the input sequence as "{PREFIX; x; y}" (autoregressive model) or "{PREFIX; x; PREFIX'; y}"

(encoding-decoding model), the prefix parameter$P_\theta$ is defined by the trainable matrix and injected directly into the activation layer$H_i$ , with subsequent activations computed by the frozen PLM. To enhance the optimization stability, the small matrix$P'_\theta$ is used after MLP reparameterization$P'_\theta$ . In Li and Liang's experimental validation, the Prefix-Tuning effect is verified on form-to-text generation (GPT-2) and summary generation (BART) [3,4] . Taking the E2E dataset as an example, Prefix-Tuning achieves a BLEU of 66.1 in the full-data scenario, which is close to the 68.2 of fine-tuning; in the low-data scenario (50-500 samples), it improves the average BLEU by 2.9 BLEUs; and the BLEU in the WebNLG unseen category test (unseen) is 43.8, which is superior to the 41.2 of fine-tuning.Compared with Adapter- Tuning compared to [5] , the Prefix-Tuning parameter is more efficient (0.1% vs. 3.6%), with comparable or better performance, especially in low-resource and extrapolation scenarios.

The research significance of Prefix-Tuning is that it provides an efficient paradigm for continuous Prompt methods, and its modular nature supports personalization (e.g., user isolation) and batch optimization. However, its performance is dependent on prefix length (e.g., 200 for summary tasks and 10 for table tasks) and initialization strategy (e.g., real word activation is preferred over random), and too long prefixes may lead to overfitting. Adaptive prefix design and broader applicability to generative tasks can be explored in the future.

### *3.2.2 Prompt Tuning*

Prompt Tuning is an efficient continuous Prompt method that adapts downstream tasks by adding a "Soft Prompt" of a small number of adjustable parameters before the input of frozen PLMs. Unlike discrete Prompt, Prompt Tuning encodes task-specific information into a continuous vector$P_e \epsilon R^{p \times e}$ and optimizes it by back-propagation, updating only the prompt parameter$P_e$ and keeping the PLM parameter$\theta$ frozen. Its innovation lies in the extremely low parameter overhead (just under 0.01% of the total parameters) and the end-to-end training capability, which supports signal distillation from any number of labeled data. Methodologically, the input sequence is "{[P; X]}", where e$P_e$ are spliced with the embedded input$X_e$ and processed through a T5 encoder-decoder, with the optimization objective of maximizing the conditional generation probability.

Compared with Prefix-Tuning, Prompt Tuning simplifies the design by adding hints only at the front input layer instead of prefix activation at each layer, with higher parameter efficiency (only 20,480 parameters vs. 1.1 billion full model parameters for 5 Token under T5-XXL). To overcome the Span Corruption pre-training limitation of T5, an LM adaptation strategy is proposed to train an additional 100K steps to bring the model closer to the natural text generation capability of GPT-3.Lester et al. experimentally validate the Prompt Tuning effect on SuperGLUE [6] , and with the increase of model size (e.g., T5-XXL, 11B parameters ), Prompt Tuning performance approaches full-parameter fine-tuning (89.3 vs. 89.3), far exceeding 71.8 for the GPT-3 fewer samples. ablation studies show that cue length (e.g., 20+ Token), initialization (e.g., class-label embedding), and LM adaptation significantly improve the performance, which is more robust especially in large models. In addition, performance is excellent in domain migration (e.g., SQuAD to TextbookQA, F1 improvement of 12.5) and Prompt integration.

The research significance of Prompt Tuning is to demonstrate that frozen PLM combined with a small number of tunable parameters can be comparable to fine-tuning, and its parametric efficiency supports multi-tasking services and storage optimization. However, its dependence on model size (small models are unstable) and pre-training target tuning limits generalizability. Interpretability of hints and cross-task migration capabilities can be explored in the future.

### *3.2.3 Progressive Prompts*

Progressive Prompts is a parameter-efficient Prompt method for Continuous Learning (CL) of language models. The method achieves knowledge retention and forward migration while freezing PLMs' parameters$\boldsymbol{\theta}$ by learning an independent soft prompt$P_k$ for each new task$T_k$ and splicing it sequentially with frozen prompts of previous tasks (e.g. $[P_k;P_{k-1};\ldots P_1;\mathrm{X}]$. The innovations are: 1) eliminating catastrophic forgetting by freezing previous hints; 2) supporting cross-task knowledge reuse by sharing inputs at$\mathbf{X}$ and adding hints step-by-step; and 3) introducing residual MLP reparameterization at$(P_k' = MLP(P_k) + P_k)$ to improve optimization stability. Compared to Prompt Tuning [6] which is designed for single-task only, Progressive Prompts focuses on multi-task sequential scenarios by updating only the current task cue parameter$\boldsymbol{\theta_{P_k}}$ with the optimization goal$max_{\theta_{P_k}} \Sigma_{x,y} \in T_k \log_p(y\,|[P_k;\ldots;P_1;x])$ and the parameter percentage is less than 0.1%.

Razdaibiedina et al. experimentally validate the Progressive Prompts effect on standard CL benchmarks (e.g. AG News, Yelp, etc.) and self-built 15-task long sequence benchmarks [7] . The average test accuracy reaches 75.1 on the T5-Large model in a small sample setup, an improvement of over 20% over the previous best method, LFPT5 (52.7 vs. 75.1) [8] ; and 77.9 on the BERT-Base model, outperforming IDBR (76.3) [9] . Long sequence experiments show that Progressive Prompts improves T5 and BERT by 21.9% and 33.3% in 20 samples/class less samples scenarios, demonstrating strong adaptability. Ablation studies show that residual reparameterization significantly improves BERT performance (e.g., 5-length cues on IMDB from 85.8 to 91.4), and attentional analysis reveals selective attention between cues (e.g., Amazon cues attention to Yelp), corroborating forward migration. Compared to Prompt Tuning, it is more memory efficient by eliminating the need for data playback or a large number of task-specific parameters.

The research significance of Progressive Prompts is to provide a lightweight solution for CL that balances forgetting mitigation and knowledge migration for Transformer architectures such as BERT and T5. Its excellent performance in sample less and long sequence tasks broadens the application scenarios such as multitasking services. However, the linear growth of cue length with the number of tasks may limit scalability, and cue compression or dynamic tuning strategies can be explored in the future.

### *3.2.4 P-tuning*

P-Tuning, a hybrid approach combining discrete and continuous cues, aims to improve the performance and stability of PLMs in natural language understanding (NLU) tasks. The method splices trainable continuous cue embeddings$[P_i]$ with manually designed discrete cues$[D_i]$ in the input to form a template$T = \{[\mathrm{P}_{0:i}], \mathbf{x}, [\mathrm{P}_{(i+1):j}], \mathbf{y}, [\mathrm{P}_{(j+1):k}]\}$ , maps$[P_i]$ into input embeddings$h_i = f([P_i])$ via a cue encoder (e.g., LSTM or MLP), and updates only the parameters of$[P_i]$ to optimize the task loss. Its innovations include 1) mitigating the instability of discrete cues (e.g., drastic performance degradation due to single word changes) through continuous cues, 2) introducing a cue encoder to model inter-cue dependencies to improve optimization efficiency, and 3) compatibility with freezing and fine-tuning PLMs. In contrast to Prompt Tuning [6] which uses only continuous cues, P-Tuning retains the semantic guidance of discrete cues to optimize for the goal of for $\max_{\{[\mathrm{P}_i]\}} \hat{p}(y|\{[\mathrm{P}_{0:i}], \mathbf{x}, [\mathrm{P}_{(i+1):k}]\})$.

Liu et al. experimentally verified the effect of P-Tuning on the LAMA knowledge detection and SuperGLUE benchmarks[10]. In LAMA (frozen model), P-Tuning improved P@1 on BERT-Base from 31.1 to 52.3 (+20.6), and on MegatronLM (11B) to 64.2 (+41.1),

surpassing AutoPrompt (43.3)[11]. In the SuperGLUE fully supervised setting, P-Tuning achieved an average score of 76.0 on BERT-Large, better than PET (73.0)[12]; in the few-shot setting, with ALBERT-xxLarge as the base model, the average score was 85.25, an improvement of more than 13 points over Prompt Tuning (61.50). Stability analysis shows that P-Tuning reduces the standard deviation of discrete prompts on LAMA-P17 from 10.1 to 0.46, and on FewGLUE from 5.68 to 3.52. Ablation studies show that LSTM encoders outperform MLP and direct embedding (EMB) on most tasks, suggesting that the position and number require task-specific adjustments.

The significance of P-Tuning is that it broadens the utility of NLU tasks by simultaneously improving performance and stability through a hybrid prompt design for freezing and fine-tuning scenarios. Unlike Prefix-Tuning (focusing on generative tasks) [4] , which focuses on NLU and combines discrete prompts, and Progressive Prompts (for continuous learning) [7] , P-Tuning is more generalized but lacks a mechanism for cross-task knowledge transfer. The limitation is that the cue design still requires manual initial templates, which can be further optimized with automatic cue search in the future.

### *3.2.5 P-tuning v2*

P-Tuning v2 is an improvement of P-Tuning, aiming at the generalization of Prompt Tuning (P-Tuning) to different model sizes and NLU tasks. It is based on Deep Prompt Tuning (DPT) [13,14] , which adds a continuous prompt embedding as a prefix to each layer of the pre-trained model (instead of only the input layer), with an input sequence of the form (l is the layer index), and updates only the prompt parameters to optimize the task loss. Its innovations include 1) the addition of adjustable parameters (0.1%-3% of the model) to the multilayer cues to improve the capacity and direct impact on the output; 2) the optimization details (e.g., cue length, multitask learning) to adapt to different task requirements; and 3) the direct use of [CLS] labels plus linear headers without a verbalizer. Compared to P-Tuning's hybrid cueing, P-Tuning v2 relies entirely on continuous cueing, with the optimization goals of .

Liu et al. experimentally verified the universality of P-Tuning v2 on SuperGLUE and sequence tagging tasks[13]. In SuperGLUE, P-Tuning v2's average performance on BERT-Large (335M) is close to fine-tuning (FT, e.g., BoolQ 75.8 vs. 77.7), comparable to FT on GLM-10B (e.g., RTE 93.1 vs. 93.1), and superior to P-Tuning and Lester et al. (e.g., BoolQ 67.2 on BERT-Large)[6]. In sequence tagging tasks (e.g., NER, QA, SRL), P-Tuning v2 achieves 92.8 on CoNLL03 (RoBERTa-Large), comparable to FT and far superior to P-Tuning's 86.1. Ablation studies show that deep cues (close to the output layer) are more effective than shallow ones), cue length varies by task (<20 for simple tasks, ~100 for complex tasks), and reparameterization (e.g., MLP) has varying effects.

The significance of P-Tuning v2 is that it achieves performance comparable to fine-tuning at a very low parameter cost for small to medium sized models (<10B) and complex tasks, compensating for the limitations of Prompt Tuning [6] . Unlike Prefix-Tuning [4] , it focuses on NLU rather than generative tasks; compared to P-Tuning [10] , it eschews discrete hints and is more pervasive; unlike Progressive Prompts [7] , it does not involve cross-task migration. The limitation is that the optimal prompt length and position need to be adjusted task-specifically, and adaptive prompt generation can be explored in the future.

### *3.2.6 Pre-trained Prompt Tuning (PPT)*

PPT aims to improve the performance of Prompt Tuning (PT) in sample less learning scenarios. Aiming at the poor performance of Vanilla PT [6] in sample less conditions, PPT obtains better initialization by introducing self-supervised task pre-training soft cues*P* in the pre-training phase. Its core innovations include 1) unifying the downstream classification

tasks into three formats (sentence pair, single sentence, multiple choice) and designing the corresponding self-supervised task (e.g., next sentence prediction, selection) pre-training cues; 2) further proposing to unify the multiple-choice format by converting all the tasks into $\arg\max_{\boldsymbol{P}} \sum_x \log p(\langle \mathbf{X} \rangle = v(y) | [\boldsymbol{P}; f(x)]; \boldsymbol{P})$ ; and 3) tuning the cues only in the downstream tasks (accounting for 0.004% of the model parameters) to keep the PLM frozen. Compared to P-Tuning's hybrid cues, PPT focuses on pure continuous cues and emphasizes pretraining initialization.

Gu et al. experimented with T5-XXL and CPM-2 based on 11B parameters to validate the PPT effect on English and Chinese few-sample tasks (32 samples) [15] . In the English task, PPT reached 93.5% on SST-2, outperforming 91.4% on FT and 70.5% on Vanilla PT, and Unified PPT reached 65.8% on RTE (64.1% on FT); in the Chinese task, PPT reached 90.1% on ChnSent, outperforming 89.1% on FT. Compared with Hybrid PT (SST-2 87.6%), Hybrid PPT further improved to 93.8%, showing that pretraining and hard cues complemented each other. The ablation study shows that PPT gradually approaches FT when the sample size increases, remains competitive in the full-data scenario (SST-2 96.9% vs. FT 96.1%), and outperforms Vanilla PT in terms of convergence speed.

The significance of PPT is to bridge the gap between pre-training and downstream tasks by prompting pre-training to achieve performance comparable to or even surpassing FT in less-sample scenarios while maintaining parameter efficiency. Unlike Prefix-Tuning [4] , PPT focuses on less-sample classification rather than generation; compared to Prompt Tuning [6] , its pre-training initialization significantly improves stability; and compared to P-Tuning v2 [13] , PPT emphasizes on a unified task format rather than multi-layer prompts. The limitations are that the pre-training task design relies on manual work and cross-domain single-sentence classification needs to be further optimized, and dynamic cue adjustment can be explored in the future.

## 4 Discussion

Prompt Tuning methods have their own characteristics in application. PET is known for its discrete prompts, which are highly interpretable and suitable for classification tasks, but rely on manually designed patterns. As a continuous method, Prompt Tuning has a parameter efficiency of 0.01%. It performs well in large model classification and generation tasks, but is unstable on small models. P-Tuning uses a mixed prompt with a parameter efficiency of 0.1%-1%. It is highly stable in NLU tasks, but requires a manual initial template. PPT performs well in few-sample classification with a parameter overhead of 0.004%. The pre-training prompts are efficient, but cross-domain applications still need to be optimized. In summary, different methods have their own trade-offs in efficiency and adaptability. In the future, more automated prompt design and cross-domain generalization capabilities can be explored.

Looking ahead, the research on Prompt Tuning can be further deepened in the following directions: First, automated prompt generation technology, such as combining generative models or reinforcement learning to reduce manual intervention and improve the scalability of the method; second, multimodal expansion, applying Prompt Tuning to visual-language models (such as CLIP) or speech-text tasks to broaden its application areas; third, cross-task knowledge transfer, by designing shared or reusable prompt structures to solve the problem of catastrophic forgetting in continuous learning, as shown in the initial attempts of Progressive Prompts. In addition, improving the interpretability of prompts and optimizing their stability in large models will also be the key to promoting the practical application of Prompt Tuning. Table 1 shows a comparison of Prompt Tuning methods.

**Table 1:** Comparison of Prompt Tuning Methods

| Method name | typology | Parameter efficiency (%) | Sample less performance | Total Supervisory Performance | Type of task to which it applies | Key Benefits | Main limitations |
|---|---|---|---|---|---|---|---|
| PET | Discrete Prompt | 0% | excellent | very much | Classification, Natural Language Reasoning | Highly interpretable with outstanding semi-supervisory capabilities | Dependent on Pattern quality, manual design is burdensome |
| LM-BFF | Discrete Prompt | 0% | excellent | very much | Classification, regression | Automated Prompt Generation, Dynamic Presentations for Improved Performance | Performance dependent task format, high variance problem |
| Prefix-Tuning | Continuous Prompt | 0.10% | very much | excellent | Generation (e.g., summary, table generation) | Parameter-efficient, modular design | Prefix length task dependency, initialization sensitive |
| Prompt Tuning | Continuous Prompt | 0.01% | very much | excellent | Classification, generation | Extremely low parameter overhead, strong performance for large models | Miniatures are unstable and need LM adaptation |
| Progressive Prompts | Continuous Prompt | <0.1% | excellent | very much | Classification (continuous | Knowledge retention to support | Cue length grows with the number |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | learning ) | forward migrati on | of tasks, limiting scalabili ty |
| P-Tuning | Hybrid Prompt | 0.1%-1% | excellen t | excellen t | Underst anding (e.g., NLU tasks) | Hybrid design enhance s stability and compati bility | Manual initializ ation of templat es required , comple x encoder selectio n |
| P-Tuning v2 | Continu ous Prompt | 0.1%-3% | very much | excellen t | Underst anding, sequenc e labeling | Multi-layer cueing generali zes to near fine-tuned perform ance | Cue length and placeme nt needs to be adjusted , task depende nt |
| PPT | Continu ous Prompt | 0.00% | excellen t | excellen t | Classifi cation (less samples ) | Pre-training cues improve sample less perform ance, efficient ly | Pre-training tasks are comple x in design and need to be optimiz ed across domain s |

## 5 Conclusion

This paper systematically sorts out the basic concepts and methods of Prompt Tuning. Through literature review and comparative analysis, it explores the classification characteristics of discrete Prompt and continuous Prompt and their application performance in natural language processing. The study found that discrete Prompt (such as PET and LM-BFF) is known for its intuitiveness and interpretability, and performs well in few-sample scenarios, especially in classification and semi-supervised tasks. Continuous Prompt (such as Prefix-Tuning, Prompt Tuning and P-Tuning v2) is more advantageous in large models due to its flexibility and adaptability. It is suitable for generation and understanding tasks, and its overall performance is comparable to full parameter fine-tuning. Prompt Tuning significantly reduces the computational and storage costs and shows strong generalization

ability in resource-constrained scenarios. However, the dependence of discrete Prompt on manual design limits its universality, the performance of continuous Prompt in small models is unstable, and the alignment problem of prompt design and task objectives remains to be solved. For example, PET is easily affected by pattern quality, and Prefix-Tuning is sensitive to prefix length. Current research focuses on single-modal NLP tasks, and the application potential of multimodal and cross-language scenarios has not been fully explored. In summary, Prompt Tuning provides a new perspective for PLM task adaptation with its efficiency and flexibility. In the future, we can further explore automated prompt design, multimodal expansion and cross-language application to promote its wider application and theoretical deepening in the field of artificial intelligence.

## References

1. Schick T, Schütze H. Exploiting cloze questions for few shot text classification and natural language inference[J]. arXiv preprint arXiv:2001.07676, 2020.
2. Gao T, Fisch A, Chen D. Making pre-trained language models better few-shot learners[J]. arXiv preprint arXiv:2012.15723, 2020.
3. Brown T, Mann B, Ryder N, et al. Language models are few-shot learners[J]. Advances in neural information processing systems, 2020, 33: 1877-1901.
4. Li X L, Liang P. Prefix-tuning: optimizing continuous prompts for generation[J]. arXiv preprint arXiv:2101.00190, 2021.
5. Houlsby N, Giurgiu A, Jastrzebski S, et al. Parameter-efficient transfer learning for NLP[C]//International conference on machine learning. pmlr,. 2019: 2790-2799.
6. Lester B, Al-Rfou R, Constant N. The power of scale for parameter-efficient prompt tuning[J]. arXiv preprint arXiv:2104.08691, 2021.
7. Razdaibiedina A, Mao Y, Hou R, et al. Progressive prompts: continuous learning for language models[J]. arXiv preprint arXiv:2301.12314, 2023.
8. Qin C, Joty S. Lfpt5: A unified framework for lifelong few-shot language learning based on prompt tuning of t5[J]. arXiv preprint arXiv:2110.07298, 2021.
9. Huang Y, Zhang Y, Chen J, et al. Continual learning for text classification with information disentanglement based regularization[J]. arXiv preprint arXiv:2104.05489, 2021.
10. Liu X, Zheng Y, Du Z, et al. GPT understands, too[J]. AI Open, 2024, 5: 208-215.
11. Shin T, Razeghi Y, Logan IV R L, et al. Autoprompt: eliciting knowledge from language models with automatically generated prompts[J]. arXiv preprint arXiv:2010.15980, 2020.
12. Schick T, Schütze H. It's not just size that matters: small language models are also few-shot learners[J]. arXiv preprint arXiv:2009.07118, 2020.
13. Liu X, Ji K, Fu Y, et al. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks[J]. arXiv preprint arXiv:2110.07602, 2021.
14. Qin G, Eisner J. Learning how to ask: Querying LMs with mixtures of soft prompts[J]. arXiv preprint arXiv:2104.06599, 2021.
15. Gu Y, Han X, Liu Z, et al. Ppt: pre-trained prompt tuning for few-shot learning[J]. arXiv preprint arXiv:2109.04332, 2021.