Prefix-Tuning+: Modernizing Prefix-Tuning by Decoupling the Prefix from Attention

Haonan Wang[1,*] **Brian K. Chen**[1,2,*] **Siquan Li**[1,*] **Xinhe Liang**[1]
Hwee Kuan Lee[1,2,‡] **Kenji Kawaguchi**[1] **Tianyang Hu**[1,†]
[1]National University of Singapore [2]Bioinformatics Institute, A*STAR

Abstract

Parameter-Efficient Fine-Tuning (PEFT) methods have become crucial for rapidly adapting large language models (LLMs) to downstream tasks. Prefix-Tuning, an early and effective PEFT technique, demonstrated the ability to achieve performance comparable to full fine-tuning with significantly reduced computational and memory overhead. However, despite its earlier success, its effectiveness in training modern state-of-the-art LLMs has been very limited. In this work, we demonstrate empirically that Prefix-Tuning underperforms on LLMs because of an inherent tradeoff between input and prefix significance within the attention head. This motivates us to introduce Prefix-Tuning+, a novel architecture that generalizes the principles of Prefix-Tuning while addressing its shortcomings by shifting the prefix module out of the attention head itself. We further provide an overview of our construction process to guide future users when constructing their own context-based methods. Our experiments show that, across a diverse set of benchmarks, Prefix-Tuning+ consistently outperforms existing Prefix-Tuning methods. Notably, it achieves performance on par with the widely adopted LoRA method on several general benchmarks, highlighting the potential modern extension of Prefix-Tuning approaches. Our findings suggest that by overcoming its inherent limitations, Prefix-Tuning can remain a competitive and relevant research direction in the landscape of parameter-efficient LLM adaptation.

1 Introduction

Large Language Models (LLMs) have advanced at a remarkable pace within recent years, driven primarily by larger models and bigger training datasets [12, 28]. As a result, training and fine-tuning LLMs has become prohibitively expensive, with all but the biggest players unable to implement full parameter fine-tuning on state-of-the-art models. To remedy this, parameter-efficient fine-tuning (PEFT) methods have been introduced. One such approach is Prefix-Tuning (PT) [16], a technique which prepends trainable vectors to future inputs of each attention layer in the transformer. PT is extremely cheap to implement, while matching and even surpassing other bulkier methods in a variety of studies. However, as LLMs have grown to record sizes, PT has failed to perform well on the largest models, gradually losing popularity to other methods such as LoRA [11] and GaLore [40].

Earlier studies have primarily attributed this behaviour to PT's failure to reshape attention patterns within attention heads [27]. We show empirically that, while this applies to more shallow transformers, it does not extend to modern LLMs which tend to have a deep transformer architecture. Prefix-Tuning large language models can in fact result in a significant shift in the attention pattern. This leads to our

*Equal contribution.
†Correspondence to Tianyang Hu. Email: hutianyang.up@outlook.com.
‡Also affiliated with Nanyang Technological University, Singapore Eye Research Institute, Singapore International Research Laboratory on AI, and Singapore Institute for Clinical Sciences.

Preprint. Under review.

conclusion that an inability to alter attention patterns is not the reason behind PT's bad performance on state-of-the-art LLMs.

In this work, we argue that the real reason PT performs sub-optimally is its inherent tradeoff between prefix and input significance. When the prefix is long relative to input length, the model risks losing input specificity and being dominated by the prefix. When the input is long relative to prefix length, the impact of PT itself is greatly diminished. This tradeoff is a result of prefixes being included in the attention head itself. Motivated by this, we build on previous work [4] to propose Prefix-Tuning+ (PT+), which relocates the prefix outside the attention head and approximates it with an external module consisting of a trainable matrix and representation function. Diagnostic experiments suggest that PT+ is substantially more expressive than standard PT, reinforcing our choice of using the external module. We also provide a unified overview to the choices we make when extending PT to PT+, discussing how readers can potentially pick and choose what to keep when designing future context-based methods.

To evaluate the performance of PT+, we run extensive experiments in the few-shot data setting comparing it to other popular training methods such as LoRA and PT. Our experiments show that across multiple popular benchmarks, PT+ can compare directly with LoRA which is considered state-of-the-art. In a few cases it can even exceed it. Regular PT flounders in comparison.

Our work presents the following key contributions:

- We demonstrate empirically that Prefix-Tuning performs badly on modern LLMs because of an inherent tradeoff between input and prefix significance within the attention head.
- We introduce Prefix-Tuning+, a novel architecture based on Prefix-Tuning that isolates the prefix module outside of the attention head. We further provide a unified overview of our decision making process in constructing PT+ to guide users when constructing future context-based methods.
- We perform extensive experiments to show the efficacy of PT+. Our experiments show that, in the few-shot setting, PT+ is competitive with state-of-the-art approaches such as LoRA—achieving an average absolute improvement of **8.1%** over LoRA and **29.4%** over Prefix-Tuning across all six evaluated settings.

This serves as a proof of concept that, when the prefix information is isolated from the attention head like in PT+, prefix-tuning methods can serve as a viable alternative to current SOTA methods and is an exciting future area of research.

2 Related Work

Parameter-efficient fine-tuning (PEFT) [21, 8] adapts large language models (LLMs) by optimizing only a limited set of parameters while freezing the majority of pre-trained weights, reducing computational and memory demands. This approach enables rapid model adaptation to downstream tasks, facilitating deployment in resource-constrained environments without sacrificing performance [9].

Weight-Based PEFT Methods. LoRA [11] represents the most widely adopted weight-based PEFT method, introducing small, trainable low-rank matrices into transformer layers while freezing the original weight matrices. Variants such as QLoRA [7] and LoRA+ [10] refine this concept further, projecting the model's weights onto low-dimensional subspaces to achieve efficiency comparable to full fine-tuning at significantly reduced computational cost. However, these methods primarily adjust linear layers within transformer blocks, indirectly affecting internal attention patterns, and potentially limiting their flexibility in adapting attention patterns and behaviors explicitly.

Context-Based PEFT Methods. In contrast to weight-based methods, context-based PEFT methods directly alter the input context provided to LLMs without modifying the model's weights. Prominent examples include P-Tuning [18, 19], Prompt Tuning [15], and Prefix-Tuning [16]. Among these, Prefix-Tuning has been recognized for its exceptional parameter efficiency, achieving performance close to full fine-tuning on generation tasks. Nevertheless, Prefix-Tuning faces significant scalability issues, as performance quickly saturates or even declines with increasing prefix length [26, 27], thereby limiting its effectiveness in learning novel tasks that substantially differ from the pretraining distributions. Addressing these limitations is crucial for enhancing the versatility and applicability of context-based PEFT approaches. In this work, we present a unified view to better understand context-based PEFT methods and propose advancements that extend beyond traditional prefix-tuning.

3 Preliminaries

Transformer models were introduced to address sequence-to-sequence tasks and primarily consist of attention layers, feed forward networks, and other task specific modules [34]. In this paper, we assume inputs take the form $X = [x_1, \ldots, x_n]$, where each X is a sequence of tokens X_i where $X_i \in \mathbb{R}^d$ for all $i \in [n]$ such that $X \in \mathbb{R}^{n \times d}$.

3.1 The Attention Mechanism

Attention modules are a key component of transformers which accepts the entire sequence as an input. Typically, attention layers consist of multiple heads, each with a separate set of parameters. For notational simplicity we focus on single headed attention. A single attention head takes the form:

Definition 1 (Single-headed Attention) *Given input $X \in \mathbb{R}^{N \times d}$ and trainable matrices $W_Q, W_K \in \mathbb{R}^{d \times d_K}, W_V \in \mathbb{R}^{d \times d_V}$. A single attention head takes the form:*

$$O = \mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_K}} + M\right)V, \tag{1}$$

where O is the output, $Q = XW_Q$, $K = XW_K$ and $V = XW_V$ and M is a causal mask. Based on [13], the attention head can be expressed as:

$$o_i^\top = \frac{\sum_{j \leq i} \mathrm{sim}(q_i, k_j) v_j^\top}{\sum_{j \leq i} \mathrm{sim}(q_i, k_j)}. \tag{2}$$

o_i is the i-th output token whilst $q_i = x_i W_Q$, $k_i = x_i W_K$ and $v_i = x_i W_V$ and $\mathrm{sim}(q, k) = \exp(\frac{qk^\top}{\sqrt{d_K}})$ is a similarity score.

3.2 Prefix-Tuning

Prefix-tuning was initially motivated by the phenomenon of prompting and in-context learning (ICL):

Definition 2 (In-Context Learning) *ICL allows large language models to adapt to new tasks by prepending demonstration prompts to the input based on a specified criteria. Given context prompt $[x'_1, ..., x'_p]$ and input X, the new prompt becomes: $X^{ICL} = [x'_1, ..., x'_p, x_1, ..., x_n]$.*

Given the broad success of ICL, prefix tuning was introduced as a natural generalization of this principle. Rather than selecting tokens which correspond to elements available in the model's vocabulary, soft-tokens (i.e trainable vectors) are prepended to future model inputs:

Definition 3 (Prefix-Tuning) *Prefix-Tuning (PT) is a form of parameter-efficient fine-tuning that prepends a sequence of vectors to the inputs. Given prefix $[s_1, ..., s_p]$, where $s_i \in \mathbb{R}^d$ for all i, and input X, the new prompt becomes $X^{pt} = [s_1, ..., s_p, x_1, ..., x_n]$. The vectors $\{s_i\}_{i=1}^p$ are then trained based on traditional gradient based methods while the rest of the model weights are frozen.*

Referring to Equation (2), the inclusion of prefix $[s_1, ..., s_p]$ yields the following output:

$$o_i^{pt\,\top} = \frac{\sum_{j \leq i} \mathrm{sim}(q_i, k_j) v_j^\top + \sum_{j \leq p} \mathrm{sim}(q_i, W_K s_j)(W_V s_j)^\top}{\sum_{j \leq i} \mathrm{sim}(q_i, k_j) + \sum_{j \leq p} \mathrm{sim}(q_i, W_K s_j)}. \tag{3}$$

Any form of ICL is a special instance of prefix-tuning but *not* vice-versa, making prefix-tuning a more flexible and expressive form of fine-tuning compared with prompting and ICL methods.

Compared with full parameter fine-tuning and even most other PeFTs, prefix-tuning offers an extremely light-weight training approach. Research shows that prefix-tuning excels in low-data or few-shot settings and when guiding the model to leverage a mix of its pretrained tasks, rather than learning entirely new tasks from scratch.

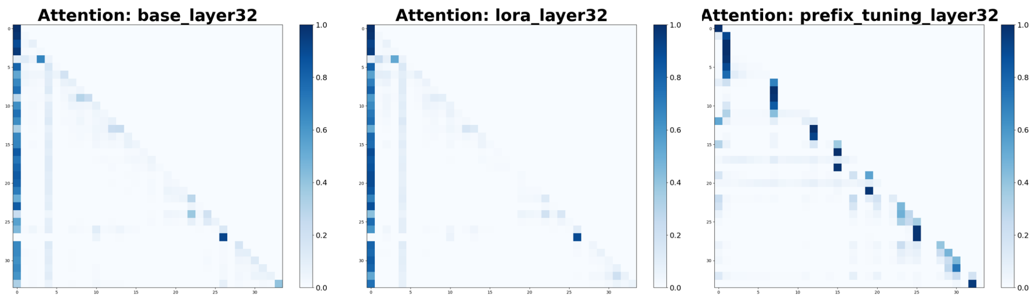

Figure 2: Attention Map of LLaMA2-7B-Chat, and its LoRA and Prefix-Tuning fine-tuned versions.

4 Limitations of prefix-tuning in LLMs

In the previous section, we noted that PT is particularly effective when leveraging pre-trained tasks. With the continual increase in the size and capability of large language models (LLMs), supported by an expanding pretraining corpus, one might anticipate a corresponding rise in the prominence and effectiveness of PT. However, contrary to expectations, the adoption of prefix-tuning has significantly declined in recent years, as evidenced by its sparse implementation on state-of-the-art models available in repositories such as Hugging Face. This diminished popularity is primarily due to PT's underwhelming performance with larger and more complex models, which manifests in reduced accuracy and instability. As depicted in Figure 1, Prefix-Tuning consistently performs under compared to LoRA on three commonly used generative classification benchmarks, despite introducing a similar number of new parameters (see Section 6 for details). With the advent of LoRA, a Parameter-Efficient Fine-Tuning (PEFT) method that consistently outperforms Prefix-Tuning on established benchmarks—the overall relevance and applicability of Prefix-Tuning methods have been increasingly questioned.

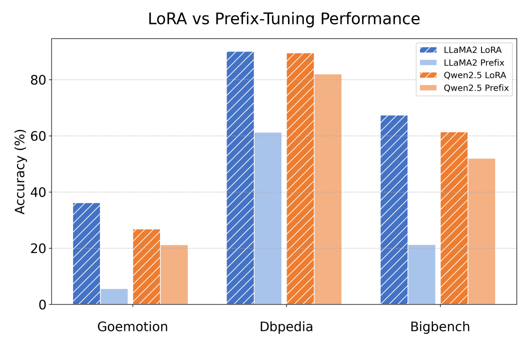

Figure 1: Performance comparison between Prefix-Tuning and LoRA.

4.1 Does Prefix-Tuning alter the attention pattern?

So why doesn't Prefix-Tuning behave well on state-of-the-art LLMs? The popular stance is that PT cannot alter the attention distribution in the attention heads [27]. Previous work [27] demonstrates that prefix-tuning is only capable of biasing the attention layer activations which forms a severe limitation. This is shown to be true for single-layer transformers and shallow transformers in general. In this study, we argue that, while this analysis is indicative for shallow transformers, it does not capture how PT behaves on LLMs, which are deep multi-layer transformers. Our experiments in Figure 2 show that PT can modify the attention pattern of LLMs significantly, despite having bad performance (experiment details are differed to Appendix B.2). This leads us to believe that an inability to affect the attention pattern is not why PT performs badly.

4.2 Tradeoff between prefix and input significance

In this section, we argue that the fundamental limitation of Prefix-Tuning is the inherent tradeoff between the significance of the prefix and the input. This can be observed by rewriting Equation (3) based on the work by [27] as follows:

$$o_i^{pt\,\top} = (1 - \alpha_i) o_i^\top + \sum_{j \le p} \alpha_{ij} {v'_j}^\top, \tag{4}$$

where $\alpha_{ij} = \frac{\text{sim}(q_i, W_K s_j)}{\sum_{j \le i} \text{sim}(q_i, k_j) + \sum_{j \le p} \text{sim}(q_i, W_K s_j)}$, $\alpha_i = \sum_{j \le p} \alpha_{ij}$ and ${v'_j}^\top = W_V s'_j$.

Equation (4) shows that the output with prefix-tuning can be represented as a linear combination between the attention from the input o_i and the attention from each prefix v'_j with weights α_{ij}. Prefix-

Tuning mainly does two things: re-weights the original attention output and adds query-dependent bias vectors.

When the prefix is long relative to input length: In this case, we can expect the value of α to be large, which results in a greater change in the attention pattern since the base model's attention pattern is mainly dependent on o_i; this explains our observations in Figure 2. To further verify, we have conducted experiments with different prefix lengths and measured the attention pattern changes using the REEF framework [39]. Our results in Table 4 confirms that as prefix length increases, the deviation from the base attention pattern grows. Details can be found in Appendix B.3. What happens when you have a large α is a smaller contribution from the input itself. The model then has reduced specificity regarding each input and risks being dominated by the prefixes. Too little significance may be placed upon the input itself.

This is further exacerbated by the fact that, as the length of the prefix increases, prefix-tuning is unable to make full use of the space spanned by the vectors $\{W_V s_i\}_{i=1}^{p}$. This phenomenon is also noticed by [27] and is attributed to the competing optimization goals for the prefix s_i. The prefix both needs to grab attention through $W_K s_i$ and determine direction through $W_V s_i$.

When the input is long relative to prefix length: we can expect the value of α to be small. The opposite issue arises because when each α_i is small, the contribution of the prefix term is diminished. As LLMs get more and more capable, relying more on long sequences arising from techniques such as chain-of-thought reasoning [35], it is understandable for the effectiveness of prefix-tuning to be severely limited. Too little significance has been placed upon the prefix-tuning.

5 Prefix-Tuning+: Method and Framework

5.1 Motivation and Construction

A key insight from Section 4.2 is that the trade-off between prefix and input importance stems from the prefix's confinement within the attention head. This motivates Prefix-Tuning+, a novel extension of PT which seeks to bring the prefix information out of the attention head itself.

We first draw the terms containing the prefix information out of the attention head by splitting Equation (3) into:

$$o_i^{pt\,\top} = \lambda \frac{\sum_{j\le i} \mathrm{sim}(q_i, k_j) v_j^\top}{\sum_{j\le i} \mathrm{sim}(q_i, k_j)} + (1-\lambda) \frac{\sum_{j\le p} \mathrm{sim}(q_i, W_K s_j)(W_V s_j)^\top}{\sum_{j\le p} \mathrm{sim}(q_i, W_K s_j)}, \tag{5}$$

where $\lambda \in [0, 1]$ is a constant. This replaces the softmax regularization tradeoff, which is dependent on the length of the input and context, with a fixed convex linear combination similar to previous works [24, 36]. Then, we approximate the similarity metric $\mathrm{sim}(\cdot,\cdot)$ with a kernel feature map ϕ such that $\mathrm{sim}(\cdot,\cdot) \approx \phi(\cdot)^\top \phi(\cdot)$. We have

$$o_i^{pt\,\top} = \lambda \frac{\sum_{j\le i} \mathrm{sim}(q_i, k_j) v_j^\top}{\sum_{j\le i} \mathrm{sim}(q_i, k_j)} + (1-\lambda) \frac{\phi(q_i)^\top \sum_{j\le p} \phi(W_K s_j)(W_V s_j)^\top}{\phi(q_i)^\top \sum_{j\le p} \phi(W_K s_j)}. \tag{6}$$

A similar approach is used in [4] to approximate in-context learning prompts, which has shown that the bias term $b_1 = \sum_{j\le p} \phi(W_K s_j)(W_V s_j)^\top$ is capable of capturing contextual prompt or prefix information. The natural generalization of this step is to replace the bias b_1 by a more expressive, trainable matrix M, and the analogous term $b_2 = \sum_{j\le p} \phi(W_K s_j)$ by a trainable matrix N, which yields:

$$o_i^{pt\,\top} = \lambda \frac{\sum_{j\le i} \mathrm{sim}(q_i, k_j) v_j^\top}{\sum_{j\le i} \mathrm{sim}(q_i, k_j)} + (1-\lambda) \frac{\phi(q_i)^\top M}{\phi(q_i)^\top N}. \tag{7}$$

In practice, we make two more modifications during application. First, due to the trainable nature of M and layer normalization, λ can be absorbed into the trainable weights and is not necessary. Second, $\phi(q_i)^\top N$ is no longer meaningful for regularization, so we remove it. Therefore, the final attention output of the Prefix-Tuning+ architecture has the following form:

$$o_i^{pt+\,\top} = \frac{\sum_{j\le i} \mathrm{sim}(q_i, k_j) v_j^\top}{\sum_{j\le i} \mathrm{sim}(q_i, k_j)} + \phi(q_i)^\top M. \tag{8}$$

Choice of feature map. Regarding the choice of ϕ there are several viable options which represent a tradeoff between expressivity and cost. A few from existing literature include $\phi(x) = \text{elu}(x)$ [13] and $\phi_W(x) = \text{ReLU}(Wx + b)$ [23]. In this study, as a proof of concept, we conduct all experiments with $\phi(x) = \text{elu}(x)$. This is because it is the easiest to implement and offers a good proof of concept regarding the viability of our approach. Other choices may offer more expressiveness and better performance but would require significantly more detailed tuning so we leave it to future work. Further details on construction of the Prefix-Tuning+ modules are offered in the appendix.

Remark 1 (Expressiveness) *By choosing $\phi_W(x) = \text{ReLU}(Wx + b)$, the term $\phi_W(q_i)M$ becomes effectively a single-layer MLP. Depending on future choices for $\phi(\cdot)$, Prefix-Tuning+ has the ability to be extremely expressive, matching methods such as full fine-tuning and LoRA.*

5.2 A Unified View for Context Based Methods

This section outlines the design choices and intermediate stages behind PT and PT+ in general, offering the rationale for each to guide future implementation decisions. We first refer to Equation (4). The most elementary version of prefix-tuning is ICL, where each vector v'_j corresponds to an input-vocabulary token prepended to the input of the transformer. Based on a decision to increase expressivity with the added need for training, we have prompt-tuning, where v'_j are replaced with trainable soft prompts. Last but not least, there is the decision to improve expressivity with added computational and memory cost. This leads to PT which prepends these soft prompts to the inputs of the individual attention heads of the transformer. To arrive at PT+, there are two following decisions to be made:

1. Shift the prefix module out of the attention head
2. Approximate $\sum_{j \le p} \text{sim}(., W_K s_j)$ by $\phi(\cdot)^\top M$

Choice 1: Shifting the prefix module out of the attention head is to avoid the limitations highlighted in Section 4.2. By doing so we avoid the α scaling on both the input and prefixes so there is no longer the same tradeoff between input contribution and prefix significance/contribution.

Choice 2: Replacing the original similarity metric by $\phi(\cdot)^\top M$ shifts the output from Equation (6) to Equation (7). By doing so, we lose some of the inherent structure of the attention mechanism. In return, we have an increase in model expressivity from the flexibility of a training matrix M. Since both PT and PT+ can be viewed as adding query-dependent d-dimensional bias terms to the transformer, we calculate the covariate output matrices of the bias from each and find the respective eigenvalue decay. From Figure 3, we see that with PT+, the top eigenvalues corresponding to the main principle components are large and decay slowly compared to PT. This indicates that the output bias spans many principal components rather than collapsing onto a handful of axes. In other words, Prefix-Tuning+ adds a bias from a more diverse, high-dimensional subspace. This is an intuitive proxy which indicates higher expressivity.

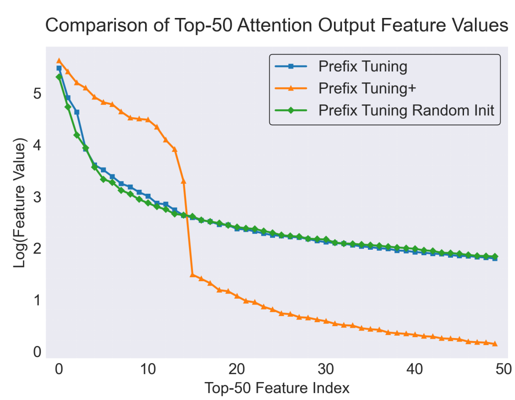

Figure 3: Spectrum of prefix representations.

In Prefix-Tuning+, both choices are used in conjunction. This does not have to be the case. Users can choose to keep the prefix term within the attention head and only apply choice 2. The resulting output is expressed as:

$$o_i^{pt\,\top} = \frac{\sum_{j \le i} \text{sim}(q_i, k_j) v_j^\top + \phi(q_i)^\top M}{\sum_{j \le i} \text{sim}(q_i, k_j) + \phi(q_i)^\top N}. \tag{9}$$

The opposite is also true, and prefix information can be brought out of the attention head through another module. In this work we choose to combine the two because we consider it expressive, easy

Table 1: Fine-Tuning Method Performance Comparison (Accuracy %). Results across datasets and models; best-performing results are in boldface, highlighting the effectiveness of Prefix-Tuning+.

Dataset	LLaMA2-7B-Chat				Qwen2.5-3B-Instruct			
	Prefix-Tuning+	**Full**	**LoRA**	**Prefix**	**Prefix-Tuning+**	**Full**	**LoRA**	**Prefix**
Goemotion	**45.2**	32.7	36.2	5.6	37.3	**37.8**	26.8	21.2
Dbpedia	**92.7**	92.6	90.1	61.3	**96.9**	94.4	89.5	82.0
Bigbench	**71.2**	38.8	67.4	21.3	**76.6**	67.4	61.4	52.0

to implement and a good proof of concept. However, in future research, what choices to implement for the optimal architecture is an interesting direction.

Remark 2 (The Memory Perspective) *We can view our method as explicitly treating the learnable matrix M as an internal memory store. Traditional context-based PEFTs, such as Prefix-Tuning, incorporate context memory by extending the KV inputs, tying the memory capacity to the prefix length. By linearizing attention and summing over the KV circuit, our approach decouples the memory capacity from sequence length and instead makes it proportional to the dimensionality of M, enabling more flexible storage of attention patterns. In practice, M allows the model to record and retrieve token interactions without altering the core attention weights by acting as an external memory module. This memory interface is both more direct and more parameter-efficient than auxiliary MLP-based memory modules, which typically require deep architectural changes, incurring higher costs.*

6 Experiments

In this section, we evaluate Prefix-Tuning+ across diverse tasks, models, and training settings, focusing on rapid adaptation, IID accuracy, and OOD generalization. We also investigate the impact of attention mechanisms and extend evaluations to practical alignment scenarios.

6.1 Experimental Setup

Datasets. We evaluate on four generative QA tasks:BigBench [31, 32], GoEmotions [6], DBpedia [14] and Banking77 [3]. We leave the detailed description of those dataset in Appendix A.1.

Training and Evaluation Protocol. We assess each method's ability to quickly adapt to downstream tasks in a few-shot setting by fine-tuning on up to five independent rounds of minimal data. In each round, we randomly sample one example per class (6 examples for BIG-bench, 28 for GoEmotions, and 14 for DBpedia) to form the entire training set. After fine-tuning, we report in-distribution (IID) accuracy on each dataset's standard test split, averaging results over the five rounds to mitigate sampling variability. Since the ability to quickly adapt to new tasks often comes at the cost of generalization, we also evaluate out-of-distribution (OOD) performance using the Banking77 intent-classification dataset without additional fine-tuning. During inference, models receive a multiple-choice prompt listing all 77 Banking77 intents and must select the most appropriate label for each query. OOD accuracy is computed as the proportion of test queries correctly classified, measuring how effectively learned features generalize to unseen domains. We perform this evaluation independently for each of the five models fine-tuned on different source datasets.

Models and Training Configuration. We experiment with two pre-trained language models to assess architectural effects: LLaMA2-7B-Chat and Qwen2.5-3B-Instruct. The LLaMA2 series models employ the multi-head attention (MHA) [34] and Qwen2.5 use grouped-query attention (GQA) [1]. GQA ties together query heads by sharing key/value projections, offering faster inference and lower memory usage, which allows us to examine if such architectural differences impact adaptation efficacy. Both models are used in their instruction version in order to test the OOD performance. We fine-tune these models using the AdamW [20] optimizer with a small learning rate and a fixed number of training steps (4000 steps). All methods use same small batch size (batch size 2).

Baselines. We compare Prefix-Tuning+ against several baseline approaches for adapting large language models, covering both parameter-efficient and traditional full fine-tuning, as well as a training-free prompt-based baseline:

- **Full Fine-Tuning**: All model parameters are fine-tuned on the minimal training set for each round. This represents the conventional approach where all weights of models are updated.

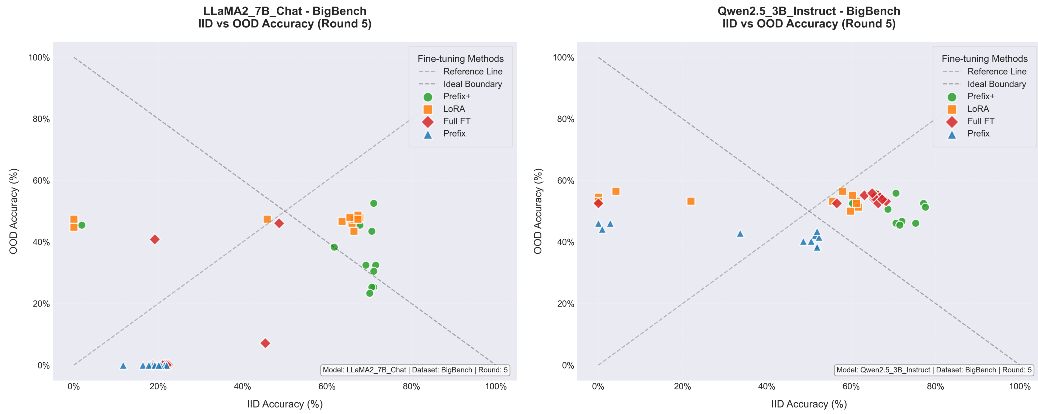

Figure 4: Pareto plots illustrating the trade-off between IID performance (on Bigbench) and OOD performance (on Banking77) for checkpoints of LLaMA2 and Qwen2.5 during training.

- **Low-rank adaptation (LoRA [11])**: LoRA freezes original model parameters and introduces trainable low-rank update matrices into each Transformer layer. Only these small rank-r matrices are learned, substantially reducing the number of trainable parameters. We set $r = 64$ to approximately match the parameter count introduced by Prefix-Tuning+.
- **Prefix-Tuning (PT [16])**: Standard prefix-tuning keeps all model weights fixed, learning only a continuous prefix vector that is prepended to the input at each Transformer layer. We follow the original implementation and set the prefix length $m = 32$.
- **In-Context Learning (ICL [2])**: Unlike the previous methods, ICL involves no parameter updates. Instead, the training examples are directly provided as demonstrations in the context at inference.

6.2 Supervised Fine-Tuning Performance Across Tasks

PEFT techniques aim to rapidly adapt large pre-trained language models (LLMs) to downstream tasks by updating a limited number of parameters. To study the effectiveness and adaptability of our proposed Prefix-Tuning+ across diverse classification scenarios, we conduct experiments on several tasks with the five round data setting. We summarize the accuracy results in Table 1, comparing Prefix-Tuning+ with various baseline approaches across the evaluated datasets. Our Prefix-Tuning+ consistently demonstrates superior or highly competitive performance compared to all baseline methods. Specifically, on BIG-bench, Prefix-Tuning+ achieves an accuracy of 71.2% with LLaMA2-7B-Chat and 76.6% with Qwen2.5-3B-Instruct, significantly outperforming LoRA, Prefix-Tuning, and full fine-tuning. On DBpedia, Prefix-Tuning+ also achieves top results (92.7% for LLaMA2, 96.9% for Qwen2.5), matching or exceeding the performance of the strongest baselines. For GoEmotions, Prefix-Tuning+ remains robust, reaching 45.2% accuracy with LLaMA2-7B-Chat and achieving a competitive 37.3% with Qwen2.5-3B-Instruct. These outcomes underscore Prefix-Tuning+ 's capability to effectively generalize and perform across varied classification tasks and model architectures.

6.3 Balancing In-Distribution Accuracy and Out-of-Distribution Generalization

An inherent IID-OOD performance trade-off typically emerges when models are trained to optimize for specific downstream tasks. In this section, we aims to study robustness of various fine-tuning approaches in effectively balancing IID performance with OOD resilience. Specifically, we examine the performance of the LLaMA2-7B-Chat and Qwen2.5-3B-Instruct models trained on the three datasets (BigBench, GoEmotions, and DBpedia). IID performance is measured directly on the hold-out part of those datasets, while OOD performance is evaluated using the Banking77 dataset. To provide a clear visualization, we present Pareto plots that depict the trade-off between IID (x-axis) and OOD (y-axis) performance. Each point on these plots represents the performance throughout training (from various checkpoints saved in different steps), with points of the same color corresponding to checkpoints from the same fine-tuning approach. The results of two models on BigBench are shown in Figure 4. These plots clearly demonstrate the performance trade-offs and highlight the differences in how each model generalizes from IID conditions to OOD scenarios. Notably, our proposed method consistently appears on the Pareto front, indicating that it achieves an optimal balance between IID and OOD performance. We leave results on more datasets in Appendix A.2.

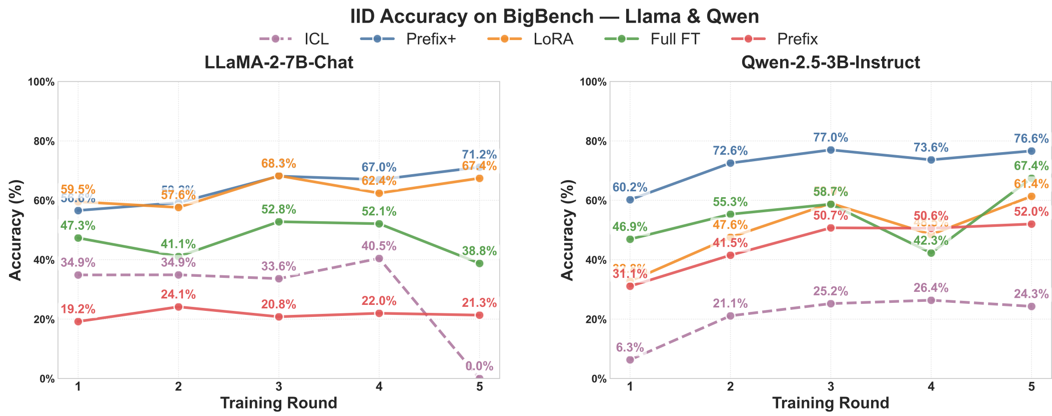

Figure 5: Performance over five incremental rounds of training data on BigBench. Prefix-Tuning+ consistently matches or exceeds baselines, with the largest gains observed on Qwen-2.5-3B-Instruct.

6.4 Performance Across Varying Data Sizes and Attention Mechanisms

To evaluate how effectively Prefix-Tuning+ scales with training set size and different attention mechanisms, we conducted experiments using the BigBench dataset, incrementally increasing dataset size over five rounds. We fine-tuned two distinct models, LLaMA-2-7B-Chat with standard attention and Qwen2.5-3B-Instruct with grouped-query attention (GQA)—using Prefix-Tuning+, Prefix-Tuning, LoRA, and full-parameter fine-tuning. Figure 5 illustrates the average performance across these rounds. Our analysis highlights two points: first, Prefix-Tuning+ maintains strong and consistent performance across different data scales and attention mechanisms, effectively matching or surpassing all baseline methods. Second, Prefix-Tuning+ shows particularly notable improvements when combined with GQA, outperforming both LoRA and full-parameter fine-tuning. These results indicates that Prefix-Tuning+ is effective when paired with the widely adopted grouped-query attention (GQA) mechanism, yielding superior performance compared to existing approaches. Additional results can be found in Appendix A.3.

6.5 Practical Alignment Tasks across Larger Datasets and Diverse Optimization Objectives

To study the effectiveness of our proposed Prefix-Tuning+ beyond generative text classification, we performed experiments aimed at aligning large language models (LLMs) more closely with human values and intentions. Specifically, we evaluated how well Prefix-Tuning+ performs when integrated with different preference optimization strategies. We employed the Qwen2.5-3B model optimized with Prefix-Tuning+ and compared its performance against LoRA, using three different training approaches: supervised fine-tuning (SFT)[25] on the Magpie-Ultra v0.1 dataset[37], and two preference-based methods—Direct Preference Optimization (DPO)[29] and Simple Preference Optimization (SimPO)[22]—using the binarized UltraFeedback dataset [5]. For each training method, we used a consistent dataset size of 10,000 samples to ensure fairness and comparability of results. Following training, we evaluated the models using AlpacaEval 2 [17], a standardized benchmark for alignment tasks. All experiments were implemented and executed using the LLaMAFactory framework [41]. Table 2 summarizes the improvement in win-rates achieved by each method. Prefix-Tuning+ consistently delivered higher win-rate increases compared to LoRA across all training objectives, highlighting its robustness and versatility. The advantage of Prefix-Tuning+ was particularly pronounced in preference-based settings (DPO and SimPO), where it notably outperformed LoRA. Interestingly, our experiments revealed a slight but consistent advantage of DPO over SimPO, contrary to prior findings [22]. We hypothesize that SimPO's comparatively weaker performance in our setup may stem from its sensitivity to hyperparameter configurations [30].

Table 2: Performance improvements of Prefix-Tuning+ over LoRA on alignment tasks using SFT, DPO, and SimPO objectives (evaluated with AlpacaEval 2).

Method	SFT	DPO	SimPO
LoRA	+0.49	+3.52	+1.24
Prefix-Tuning+	**+0.76**	**+4.66**	**+1.74**

7 Discussion

To conclude, in this work we argue that the reason why Prefix-Tuning has been ineffective when applied to modern large language models is that prefixes are "trapped" within the attention head. To remedy this, we introduce a novel architecture that generalizes upon existing Prefix-Tuning methods by approximating the prefix module and shifting it out of the attention head. Surprisingly, even with this slightly naive implementation, our model is able to match state-of-the-art methods such as LoRA on popular benchmarks in a few-shot setting, far outpacing previous prefix-tuning methods. We treat this as proof of concept that, if approached correctly, Prefix-Tuning methods can be competitive and are an exciting future avenue of research.

We also acknowledge the existing limitations of our work. Rather than presenting a clear alternative to existing PEFTs, Prefix-Tuning+ is primarily a proof of concept. The design of our method has yet to be thoroughly ablated. For instance, this line of work can potentially be improved utilizing a more powerful choice of feature map ϕ such as a learnable one. Further studies are needed to test the limits of our method in more tasks and with more training objectives.

References

[1] Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders, 2020. URL https://arxiv.org/abs/2003.04807.

[4] Brian K Chen, Tianyang Hu, Hui Jin, Hwee Kuan Lee, and Kenji Kawaguchi. Exact conversion of in-context learning to model weights in linearized-attention transformers. *International Conference on Machine Learning*, 2024.

[5] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback, 2023.

[6] Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. Goemotions: A dataset of fine-grained emotions. *arXiv preprint arXiv:2005.00547*, 2020.

[7] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[8] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

[9] Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

[10] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models, 2024. URL https://arxiv.org/abs/2402.12354.

[11] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[12] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

[13] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

[14] Fanshuang Kong, Richong Zhang, Zhijie Nie, and Ziqiao Wang. Rethink the evaluation protocol of model merging on classification task. *arXiv preprint arXiv:2412.13526*, 2024.

[15] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[16] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[17] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 5 2023.

[18] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[19] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI Open*, 5:208–215, 2024.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[21] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft, 2022.

[22] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

[23] Jean Mercat, Igor Vasiljevic, Sedrick Scott Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. Linearizing large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=soGxskHGox.

[24] Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention, 2024. URL https://arxiv.org/abs/2404.07143.

[25] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[26] Yawen Ouyang, Yongchang Cao, Yuan Gao, Zhen Wu, Jianbing Zhang, and Xinyu Dai. On prefix-tuning for lightweight out-of-distribution detection. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1533–1545, 2023.

[27] Aleksandar Petrov, Philip HS Torr, and Adel Bibi. When do prompting and prefix-tuning work? a theory of capabilities and limitations. *arXiv preprint arXiv:2310.19698*, 2023.

[28] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling language models: Methods, analysis and insights from training gopher, 2022. URL https://arxiv.org/abs/2112.11446.

[29] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[30] schrieffer-z. can't reproduce AE-LC numbers in hf ckpt (Llama-3-8b-SFT-DPO, Llama-3-8b-SFT-SimPO). GitHub issue #77, urlhttps://github.com/princeton-nlp/SimPO/issues/77, December 2024. princeton-nlp/SimPO repository.

[31] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[32] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[33] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[35] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

[36] Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers, 2022. URL https://arxiv.org/abs/2203.08913.

[37] Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing, 2024.

[38] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[39] Jie Zhang, Dongrui Liu, Chen Qian, Linfeng Zhang, Yong Liu, Yu Qiao, and Jing Shao. REEF: Representation encoding fingerprints for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[40] Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection, 2024. URL https://arxiv.org/abs/2403.03507.

[41] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

A Appendix: More Experiment Results

A.1 Datasets

We use four generative classification datasets:

- **(1) BigBench** [31, 32]: A comprehensive evaluation suite consisting of 23 challenging tasks. We focus on the *Date Understanding* task, formulated as a 6-class QA problem in which the model must choose one of six answer categories. For simplicity, we refer to this setting as BigBench.
- **(2) GoEmotions** [6]: A fine-grained emotion classification dataset containing 58K Reddit comments labeled with 27 emotion categories plus neutral (28 classes total). As the largest human-annotated English emotion dataset, GoEmotions covers a broad taxonomy of emotions. We cast this as a generative QA task: the model reads a comment and generates the corresponding emotion label.
- **(3) DBpedia** [14]: A widely used ontology classification dataset consisting of Wikipedia abstracts assigned to 14 top-level classes. We formulate this as a generative QA task where the model must output the correct class name given an abstract.
- **(4) Banking77** [3]: A challenging intent classification dataset designed for conversational systems, consisting of 13,083 customer service queries annotated across 77 categories. We formulate this as a generative QA task where the model must generate the correct label given a customer query.

A.2 In-Distribution Accuracys and Out-of-Distribution Generalization

In this appendix, we provide additional Pareto plots to complement the analysis presented in expriments section. Specifically, Figures6 and 7 illustrate the trade-offs between in-distribution (IID) and out-of-distribution (OOD) performances for fine-tuned LLaMA2-7B-Chat and Qwen2.5-3B-Instruct models across two additional datasets: GoEmotions and DBPedia.

Each plot shows the IID accuracy (x-axis) evaluated directly on the respective dataset's held-out test set, and the OOD accuracy (y-axis) evaluated on the Banking77 dataset without further fine-tuning. Points within each plot represent model checkpoints captured at different training intervals, with colors indicating the respective fine-tuning methods used.

Consistent with our observations in the main text, the proposed method frequently occupies positions near the Pareto front. This indicates its effectiveness in maintaining a balanced performance between achieving high accuracy on IID tasks and exhibiting strong generalization to OOD scenarios.

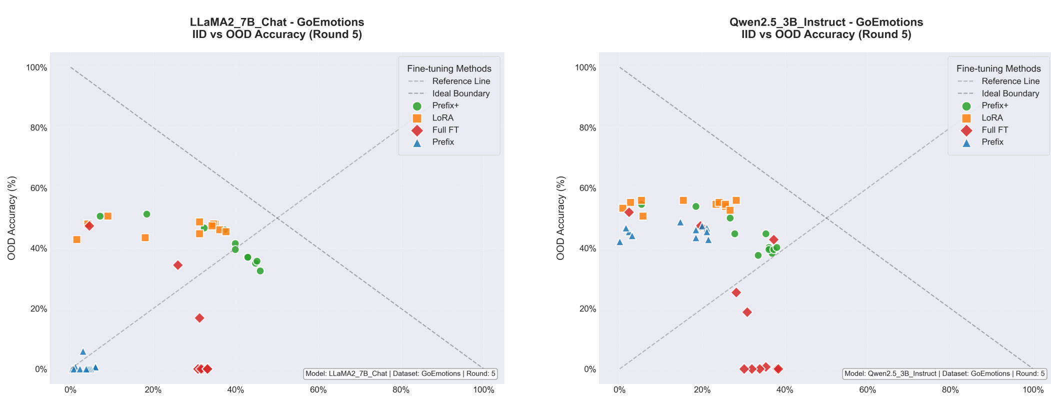

Figure 6: Pareto plots illustrating the trade-off between IID performance (on GoEmotions) and OOD performance (on Banking77) for checkpoints of LLaMA2-7B-Chat and Qwen2.5-3B-Instruct during training.

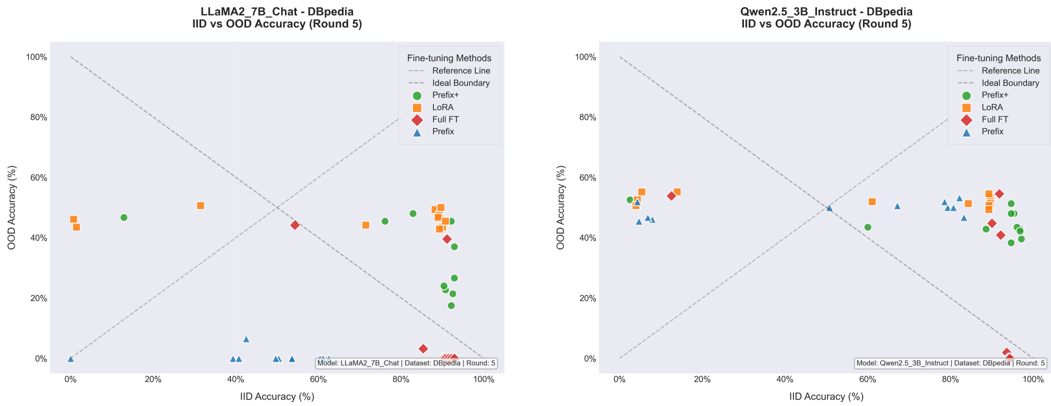

Figure 7: Pareto plots illustrating the trade-off between IID performance (on DBPedia) and OOD performance (on Banking77) for checkpoints of LLaMA2-7B-Chat and Qwen2.5-3B-Instruct during training.

A.3 Performance Across Varying Data Sizes and Attention Mechanisms

To further validate the robustness and adaptability of Prefix-Tuning+ across different tasks and attention mechanisms, we provide additional experiment results on two more datasets: GoEmotions and DBpedia. Similar to the main experiment, we incrementally increased the training set size across five rounds, fine-tuning two models—LLaMA-2-7B-Chat (multi-head attention, MHA) and Qwen-2.5-3B-Instruct (grouped-query attention, GQA)—using Prefix-Tuning+, Prefix-Tuning, LoRA, full-parameter fine-tuning, and the In-context Learning (ICL) baseline. Figure 8 and 9 illustrates the results across these additional datasets. Overall, these supplementary results reinforce our primary findings that Prefix-Tuning+ scales effectively with data size and adapts particularly well to the grouped-query attention mechanism, outperforming existing parameter-efficient methods.

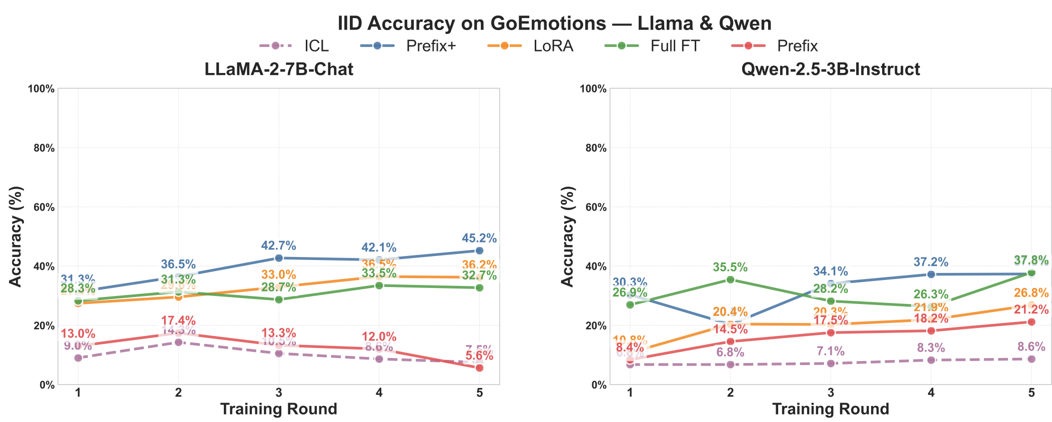

Figure 8: Performance comparison over five incremental rounds of training data on GoEmotions.

B Appendix: Verification Experiment Setup

To better understand how different methods affect model behavior, we design three *comprehension-oriented experiments* that focus on analyzing attention patterns and internal representations. These experiments aim to shed light on the mechanisms and effects of each approach. For consistency and comparability, we use the GoEmotions dataset as the in-distribution (IID) dataset and the Banking77 dataset as the out-of-distribution (OOD) dataset across all experiments. The following subsections detail the setup of each experiment.

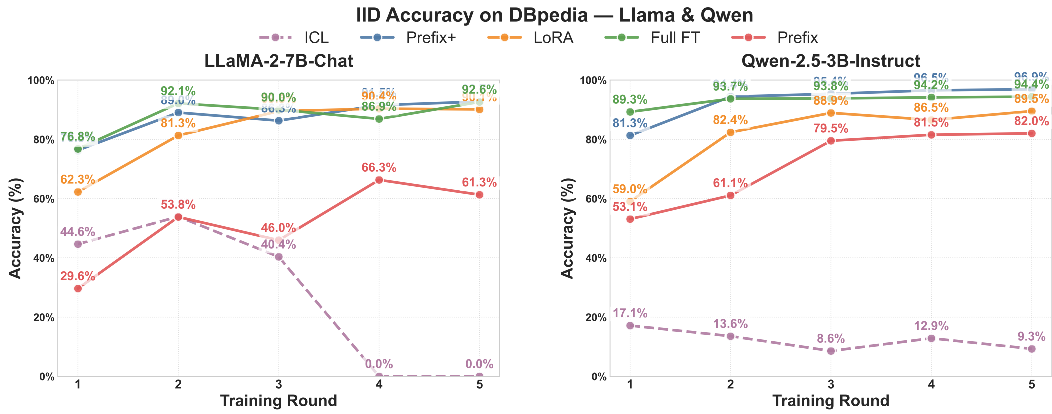

Figure 9: Performance comparison over five incremental rounds of training data on DBpedia dataset.

B.1 Spectrum Analysis of Prefix Representations

In this experiment, we use Qwen2.5-3B-Instruct as the base model. We fine-tune two variants—prefix-tuning (with a prefix length of 32) and prefix-tuning+—on the GoEmotions dataset using identical training configurations and a consistent sampling strategy (5 rounds).

Let $F_b \in \mathbb{R}^{n\times d}$ denote the base model's final layer attention outputs for n input tokens in total with representation dimension d, and $F_t \in \mathbb{R}^{n\times d}$ represent the corresponding fine-tuned model outputs. The representation effect (bias) matrix is computed as:

$$\Delta F = F_t - F_b$$

After normalization, we perform eigenvalue decomposition on the covariance matrix of representation effects:

$$\Sigma = \frac{1}{n-1}\Delta F^\top \Delta F = V\Lambda V^\top$$

where $\Lambda = \text{diag}(\lambda_1, ..., \lambda_d)$ contains eigenvalues ($\lambda_1 \geq ... \geq \lambda_d$), and V is the orthogonal eigenvector matrix.

We concatenate examples from the GoEmotions test split into the input sequences and extract the self_attn.attn_output from the final layer. We then compute the corresponding attention outputs bias from the two fine-tuned variants, analyze their eigenvalue spectra, and visualize the top 50 eigenvalues to quantify how prefix tuning and our method alters the representation space geometry.

B.2 Attention Pattern Visualization

This experiment examines how different fine-tuning methods impact attention behavior. We use LLaMA2-7B-Chat and Qwen2.5-3B-Instruct as base models, and fine-tune their respective prefix-tuning and prefix-tuning+ variants using the same data and settings as in the previous experiment. We select one example each from the IID (GoEmotions) and OOD (Banking77) datasets as test inputs. For each model, we extract the self.attn.attn_weight from the final layer and visualize it as a heatmap to reveal attention patterns. For the prefix-tuning variants, we isolate the attention weights corresponding only to real tokens (excluding prefix tokens), normalize them, and then produce the heatmap visualization.

B.3 Representation Similarity via CKA

Inspired by the REEF framework [39], which utilizes centered kernel alignment (CKA) to quantify representation-level differences, we evaluate the similarity between base and fine-tuned models. The CKA similarity between two sets of representations X (base model) and Y (fine-tuned model) is computed as:

$$\text{CKA}(X, Y) = \frac{\text{HSIC}(X, Y)}{\sqrt{\text{HSIC}(X, X) \cdot \text{HSIC}(Y, Y)}},$$

where the Hilbert-Schmidt Independence Criterion (HSIC) is defined as:

$$\text{HSIC}(X, Y) = \frac{1}{(m-1)^2} \text{tr}(K_X H K_Y H).$$

Here, $H = I - \frac{1}{m} 11^T$ is the centering matrix, and K_X, K_Y are Gram matrices with $(K_X)_{ij} = k(X_i, X_j)$ and $(K_Y)_{ij} = k(Y_i, Y_j)$, where k is a kernel function (we use linear kernel in our experiments). X_i denotes the i-th representation vector from layer outputs.

We use Qwen2.5-3B-Instruct as the base model, and obtain its prefix-tuning and prefix-tuning+ variants using the same training data and setup. The TruthfulQA dataset is used for evaluation. Following the sampling and CKA computation protocol from the REEF paper, we extract decoder representations from the 18$^{\text{th}}$ layer of each model and compute the CKA similarity with the base model. This allows us to quantitatively assess how each method alters the internal representations while controlling for computational variance.

Table 3: CKA Similarity Between Different Methods And Base Model

Method	**CKA Similarity**
Base Model	1.0000
LoRA	0.9978
Prefix Tuning+	0.9432
Prefix Tuning (32)	0.8242

As shown in Table 3, we present the CKA similarity between the base model and the models fine-tuned using three PEFT methods: LoRA, prefix-tuning+, and prefix-tuning. It is evident that prefix-tuning+ and LoRA exhibit notably different effects on the model's internal representations. Our proposed prefix-tuning+ method induces more substantial shifts in the model's representation space, indicating a stronger impact on the model's expressive capacity. On the other hand, although prefix-tuning causes significant changes in the attention patterns, this also leads to much larger representation shifts, which may partly explain its relatively weaker downstream performance.

Table 4: CKA Similarity Between Prefix Tuning And Base Model

Method	**CKA Similarity**
Base Model	1.0000
Prefix Tuning (16)	0.8802
Prefix Tuning (32)	0.8242
Prefix Tuning (64)	0.7957

As shown in Table 4, we further examine how the prefix length affects the representation similarity between the prefix fine-tuned model and the base model under the same dataset and training settings. It is clear that as the prefix length increases from 16 to 64, the model's internal representations deviate more significantly from those of the base model, indicating that longer prefixes introduce more substantial changes in representation space.

In our experiments, since both Prefix-Tuning and Prefix-Tuning+ only modify parameters within the self-attention mechanism—without affecting other components of the decoder layers—the resulting changes in representations can be regarded as a close approximation of changes in the attention pattern.

C Appendix: Implementation Details

We implemented our experiments using PyTorch and trained our models utilizing the DeepSpeed optimization library with ZeRO Stage 3 to efficiently manage memory usage during training. To further optimize memory and computational efficiency, we offloaded both optimizer states and model parameters to CPU with pinned memory enabled, facilitating faster data transfers. Gradient communication and computation were overlapped, and contiguous gradients were enforced to enhance training throughput.

The AdamW optimizer was employed with a weight decay of 0.1, momentum terms set as $\beta_1 = 0.9, \quad \beta_2 = 0.95$, and epsilon of 1×10^{-8}. Training was executed using automatic precision selection between FP16 and BF16 modes for optimal balance between performance and stability. The learning rate was held constant at 2×10^{-5} throughout the training process. Each GPU processed a micro-batch size of one sample per step, while gradient accumulation was automatically managed to simulate larger batch sizes effectively. Gradient clipping was automatically controlled by DeepSpeed to maintain stable training dynamics.

For supervised fine-tuning (SFT) experiments, training was conducted using 2 GPUs, whereas human preference alignment experiments utilized 8 GPUs.

D Appendix: Limitation

Despite the promising results demonstrated by Prefix-Tuning+, several areas remain open for exploration. Firstly, our implementation utilizes the kernel approximation for simulating attention, specifically the exponential linear unit (ELU). While this choice enabled efficient experimentation and a clear proof-of-concept demonstration, other feature mappings or kernel functions could potentially yield improved performance. Exploring more sophisticated kernel approximations or trainable kernel designs remains an exciting area for further enhancement of expressivity and effectiveness. Secondly, although Prefix-Tuning+ effectively addresses the trade-off between prefix length and input specificity within attention heads, our experiments did not extensively explore the effects of varying internal dimensionalities or architectures of the externalized prefix module. Further studies investigating these architectural choices and their optimization could unlock additional performance gains. Lastly, our evaluations were primarily conducted in supervised fine-tuning (SFT) and human alignment scenarios. Extending evaluations to contexts involving abundant data would provide deeper insights into Prefix-Tuning+'s maximum capacity to acquire new knowledge. However, due to computational resource constraints at our institution, such comprehensive studies were beyond our current capabilities. We acknowledge this limitation and leave extensive evaluations to future research.

E Appendix: Broader Impacts

The introduction of Prefix-Tuning+ offers significant positive impacts by making large language model (LLM) adaptation more efficient and accessible, thus enabling broader participation in AI research and application, particularly for resource-constrained communities and organizations. Additionally, by reducing computational requirements, Prefix-Tuning+ contributes positively to sustainability efforts in AI development. On the other hand, the enhanced ease of adapting powerful LLMs also carries risks, such as potential misuse in generating misinformation or biased content. It is essential for researchers and practitioners to incorporate ethical practices, robust monitoring, and mitigation strategies to address these risks, ensuring that the societal benefits of Prefix-Tuning+ significantly outweigh its potential negative impacts.

F Licenses

We use standard licenses from the community. We include the following licenses for the codes, datasets and models we used in this paper.

Datasets & Benchmarks:

- BigBench [31]: MIT
- GoEmotions [6]: Apache License 2.0
- DBPedia [14]: Creative Commons 3.0
- Banking77 [3]: MIT

Codes:

- LLaMA-Factory [41]: Apache License 2.0
- Alpaca-eval [41]: Apache License 2.0

Models:

- Qwen2.5-3B-Instruct [38]: Apache License 2.0
- LLaMA2-7B-Chat [33]: LLaMA2 Community License