Adapt Once, Thrive with Updates: Transferable Parameter-Efficient Fine-Tuning on Evolving Base Models

Naibin Gu[1,2], Peng Fu[1,2*], Xiyu Liu[1,2], Ke Ma[3], Zheng Lin[1,2], Weiping Wang[1]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]School of Electronic, Electrical and Communication Engineering, UCAS, Beijing, China
{gunaibin,fupeng}@iie.ac.cn

Abstract

Parameter-efficient fine-tuning (PEFT) has become a common method for fine-tuning large language models, where a base model can serve multiple users through PEFT module switching. To enhance user experience, base models require periodic updates. However, once updated, PEFT modules fine-tuned on previous versions often suffer substantial performance degradation on newer versions. Re-tuning these numerous modules to restore performance would incur significant computational costs. Through a comprehensive analysis of the changes that occur during base model updates, we uncover an interesting phenomenon: continual training primarily affects task-specific knowledge stored in Feed-Forward Networks (FFN), while having less impact on the task-specific pattern in the Attention mechanism. Based on these findings, we introduce Trans-PEFT, a novel approach that enhances the PEFT module by focusing on the task-specific pattern while reducing its dependence on certain knowledge in the base model. Further theoretical analysis supports our approach. Extensive experiments across 7 base models and 12 datasets demonstrate that Trans-PEFT trained modules can maintain performance on updated base models without re-tuning, significantly reducing maintenance overhead in real-world applications[1].

1 Introduction

Large language models have shown impressive performance across various domains (Touvron et al., 2023a; OpenAI, 2023; Yang et al., 2024; Chen et al., 2024; Cai et al., 2024). While fine-tuning is a common approach to adapt these models for specific applications, it becomes increasingly challenging in terms of computational resources and storage requirements as model sizes grow. Parameter-efficient fine-tuning (PEFT) techniques (Houlsby et al., 2019; Hu et al., 2022; Gu et al., 2024; Liu et al., 2024) address this challenge by updating only a small subset of parameters while maintaining comparable performance. This enables efficient model customization where a single base model can serve numerous users through dynamic switching between PEFT modules.

In order to enhance user experience, base models require periodic updates to update their knowledge and improve their capabilities. These updates, which do not alter the architecture, are typically achieved through continual pre-training, as exemplified by the updating from Qwen2 (Yang et al., 2024) to Qwen2.5 (Qwen et al., 2025). However, a critical challenge emerges: when the base model is updated, the PEFT modules fine-tuned on the previous version cannot be applied to the new version, as such direct transfer would result in a significant drop in performance (Qin et al., 2023). For large-scale deployments with numerous PEFT modules, re-tuning not only incurs substantial computational overhead but also raises significant privacy concerns due to the necessity of maintaining long-term storage of users' data.

To tackle the challenge of performance degradation after transferring, we conduct an analysis of the changes that occur during base model updates and uncover several interesting observations: (1) Attention activation distributions remain similar in corresponding layers, suggesting that fine-tuned base models maintain similar task-specific patterns across versions. (2) FFN activation distributions differ between versions in corresponding layers, suggesting changes in task-specific knowledge storage within layers. (3) The influence of different FFN sub-layers on activation magnitudes changes, indicating shifts in both intra-layer and cross-layer knowledge storage. These changes in knowledge storage lead to a decline in task performance after transferring, highlighting the need to reduce dependence on certain knowledge.

14765

Building upon these findings, we introduce Trans-PEFT, a novel approach to enhance PEFT module transferability. During fine-tuning, Trans-PEFT employs two key strategies: Intra-layer Knowledge Masking and Cross-layer Knowledge Dropping. These strategies systematically reduce the dependence of PEFT modules on certain knowledge, thereby encouraging PEFT to capture the invariant task-specific patterns within the attention mechanisms. Furthermore, we provide a theoretical guarantee for the task performance after transferring by bounding the loss discrepancy between base model versions. In practical applications, Trans-PEFT can seamlessly integrate with existing PEFT methods without requiring architectural modifications or introducing additional computational costs. This practical design allows PEFT modules trained on old base models to be efficiently transferred to their new versions.

We validate our approach using 7 different base models from 3 different sources, across 12 datasets covering math reasoning, code generation, and commonsense reasoning. Experimental results demonstrate that Trans-PEFT maintains the performance across base model versions without re-tuning, achieving up to 30% performance gains compared to Direct Transfer. Furthermore, Trans-PEFT exhibits the capability not only to maintain performance but also to potentially leverage the improvements introduced by base model updates.

In summary, our contributions are as follows:

- We reveal the impact of base model updates: continual training significantly modifies task-specific knowledge stored in the FFN sub-layer, while it has a minor impact on the task-specific patterns of the attention mechanism.

- We introduce Trans-PEFT, a novel method enabling the transfer of PEFT modules from old base model versions to their new versions without re-tuning. We provide a theoretical analysis to establish the foundations of its effectiveness.

- Extensive experiments on 7 base models across 12 datasets show that Trans-PEFT trained modules can be used directly without re-tuning while maintaining performance.

2 Preliminary

In our Trans-PEFT approach, we select two widely used PEFT techniques, LoRA (Hu et al., 2022)

(a) InternLM2-7B   (b) InternLM2.5-7B
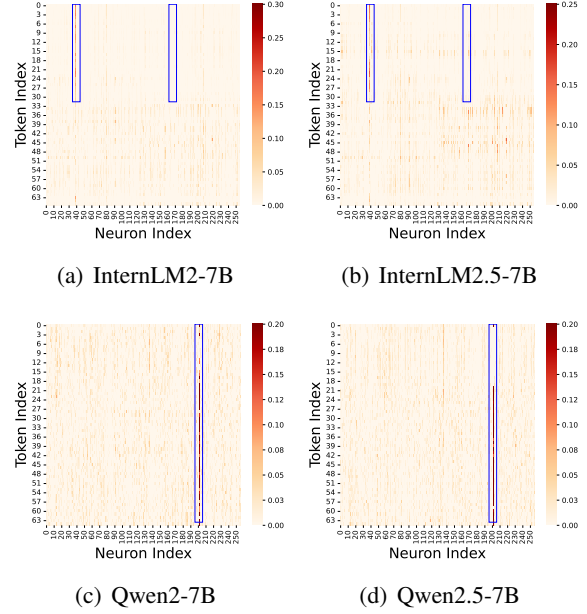
(c) Qwen2-7B   (d) Qwen2.5-7B

Figure 1: Comparison of activation distributions within attention sub-layers across different fine-tuned base model versions. The visualization shows the same dimensions from the 18th layer of four base models, with similar regularities observed across other layers.

and Adapter (Houlsby et al., 2019), to validate the effectiveness of transferability.

LoRA. For a linear weight $\mathbf{W} \in \mathbb{R}^{d \times k}$ in the base model, LoRA introduces the trainable module in parallel with the linear weight. Each module consists of a down-projection layer $\mathbf{W}_{\text{down}} \in \mathbb{R}^{d \times r}$ and an up-projection layer $\mathbf{W}_{\text{up}} \in \mathbb{R}^{r \times k}$, where d is the hidden size of the base model and $r \ll d$. After incorporating LoRA, the forward pass is modified as follows:

$$\mathbf{h} \leftarrow \mathbf{X}\mathbf{W} + \mathbf{X}\mathbf{W}_{\text{down}}\mathbf{W}_{\text{up}}, \qquad (1)$$

where $\mathbf{X} \in \mathbb{R}^d$ is the input of the $\mathbf{W}$.

Adapter. For each layer in the base model, the Adapter method adds two trainable modules sequentially after the attention sub-layer and the FFN sub-layer. After adding adapters, the forward pass is modified as follows:

$$\mathbf{h} \leftarrow \mathbf{h} + f(\mathbf{h}\mathbf{W}_{\text{down}})\mathbf{W}_{\text{up}}, \qquad (2)$$

where $h \in \mathbb{R}^d$ is the output of the sub-layer, f is a non-linear activation function.

3 Analysis of Updating Base Models

Modern base models typically employ the Transformer (Vaswani et al., 2017) architecture, which

(a) InternLM2-7B

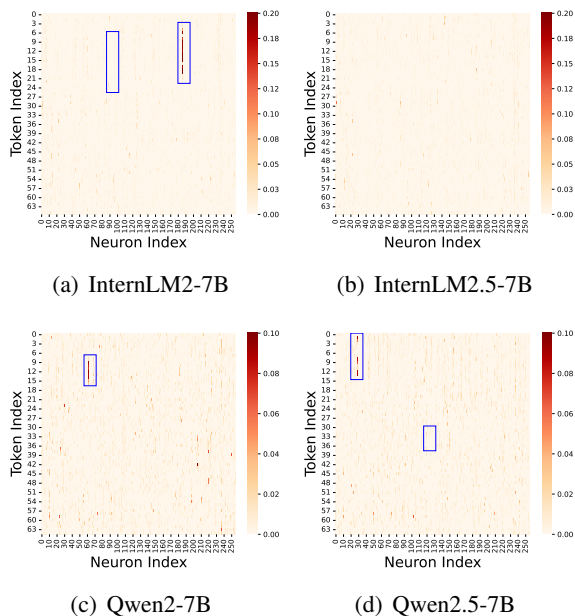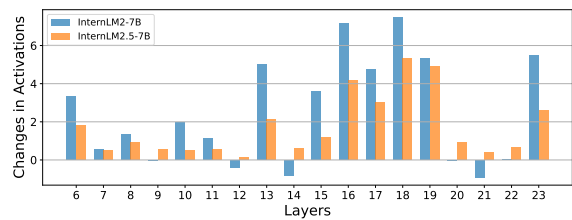(b) InternLM2.5-7B

(c) Qwen2-7B

(d) Qwen2.5-7B

Figure 2: Comparison of activation distributions within FFN sub-layers across different fine-tuned base model versions. The visualization shows the same dimensions from the 18th layer of four base models, with similar regularities observed across other layers.

(a) InternLM2-7B vs InternLM2.5-7B

(b) Qwen2-7B vs Qwen2.5-7B

Figure 3: Comparison of each FFN sub-layer's influence on activations across different fine-tuned base model versions. The influence is measured by the difference in the magnitude of activations between adjacent layers.
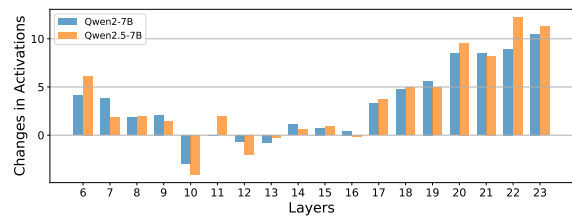
consists of multiple transformer layers, with each containing an attention sub-layer and an FFN sub-layer. To investigate the impact of updating, we fine-tune LoRA on Qwen2-7B and its newer version Qwen2.5-7B, InternLM2-7B (Cai et al., 2024) and its newer version InternLM2.5-7B with the MetaMathQA dataset (Yu et al., 2024) and analyze the activations in sub-layers of these models. See Appendix C for more implementation details.

3.1 Changes Occurring within the Layer

We compare the activation distributions of two fine-tuned model versions for the attention and FFN sub-layers in each layer, as shown in Figures 1 and 2. Blue rectangles highlight regions with relatively high activations. We find that after fine-tuning, the activation distributions of the attention sub-layer in both versions are quite similar. However, we observe notable differences in the intermediate activation distributions of the FFN sub-layer. Previous interpretability studies suggest that the primary role of the attention sub-layer is to extract dependencies between tokens (Hao et al., 2021), while the FFN sub-layer primarily functions as key-value memories to store knowledge (Geva et al., 2021). These results indicate that **after fine-tuning, the attention mechanism in both versions under-**

goes minimal changes regarding task-specific patterns for capturing token relationships. Typically, updates to the base model without changing its architecture aim to change model knowledge and enhance capabilities through continual training. **The differences in FFN activations confirm that the two versions differ in the task-specific knowledge regions.**

3.2 Changes Occurring across Different Layers

Furthermore, we investigate the knowledge storage changes across different FFN sub-layers. As illustrated in Figure 3, we analyze the influence of each FFN sub-layer on the activation magnitudes for both the old and new versions, which reflects modifications in task-specific knowledge storage. It is observed that many of the layers even exhibit opposing effects between the two versions, **suggesting knowledge storage also changes between different FFN sub-layers**. In summary, the unsuccessful direct transfer of PEFT modules to the updated base model can be attributed to two reasons: changes in both intra-layer and cross-layer knowledge storage of the base model.

4 Methodology

4.1 Trans-PEFT

Although we have discovered that transfer performance is influenced by redistributed knowledge
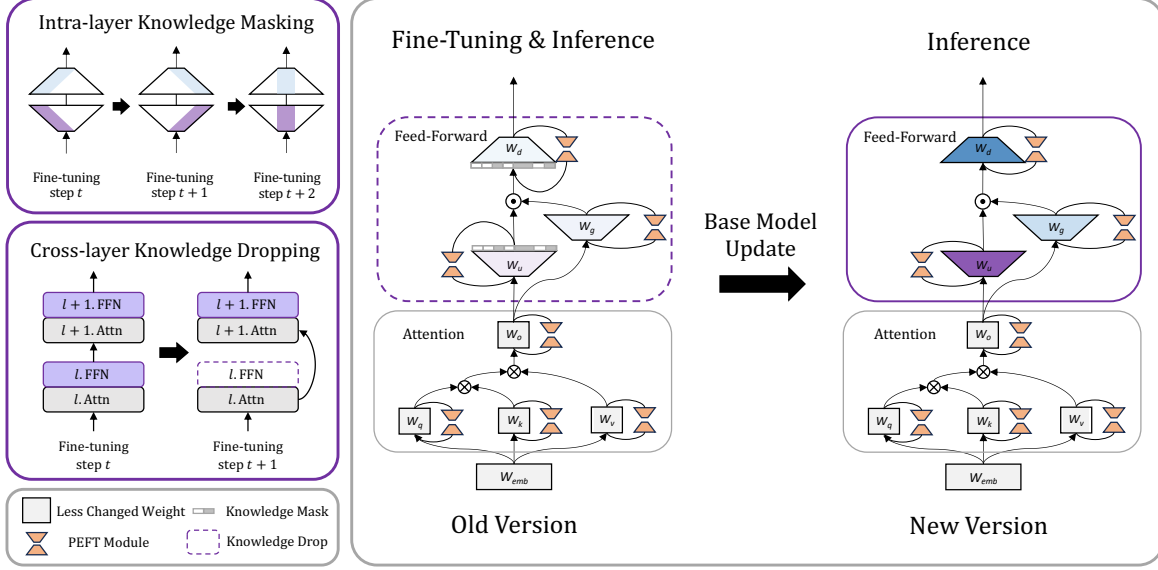
Figure 4: Illustration of Trans-PEFT. The Trans-PEFT approach mainly includes two strategies: intra-layer knowledge masking and cross-layer knowledge dropping. In each forward pass, it randomly chooses to drop the output of an entire FFN layer or to mask certain intermediate dimensions.

storage in FFN sub-layers, we are unable to precisely track how the base model modifies its storage for different tasks. This makes it impossible to determine the exact modifications needed for PEFT modules on their fine-tuning task.

To address this challenge, we propose the Trans-PEFT approach, which improves the transferability of PEFT modules by modifying the fine-tuning process on the old version of the model. As illustrated in Figure 4, our approach primarily incorporates two key strategies: Intra-layer knowledge masking and Cross-layer knowledge dropping.

Intra-layer Knowledge Masking. Due to the uncertainty in knowledge storage changes caused by updates, we are motivated to introduce randomness to reduce dependency on certain knowledge. Specifically, given a FFN sub-layer, it primarily consists of an up-projection matrix $\mathbf{W}_{\text{fc1}} \in \mathbb{R}^{d \times d_{ff}}$, a down-projection matrix $\mathbf{W}_{\text{fc2}} \in \mathbb{R}^{d_{ff} \times d}$, and an activation function σ, where d_{ff} is the intermediate size[2]. When PEFT modules are added, taking LoRA as an example, the output for a given input $\mathbf{X}$ is:

$$\text{FFN}(\mathbf{X}) = \sigma(\mathbf{X}(\mathbf{W}_{\text{fc1}} + \Delta\mathbf{W}_{\text{fc1}})) \cdot (\mathbf{W}_{\text{fc2}} + \Delta\mathbf{W}_{\text{fc2}}), \quad (3)$$

where $\Delta\mathbf{W}$ represents the added LoRA module on the corresponding weights.

---

[2]The detailed description of the gate_proj in LLaMA-style models are omitted for simplicity.

To achieve this goal, we introduce a non-trainable binary mask vector m for each FFN sub-layer during fine-tuning on the old base model version. This mask vector is applied element-wise to the output of the activation function, reducing its dependency on specific dimensions. The values of this mask vector are determined based on a rate p_i as follows:

$$m \sim \text{Bernoulli}(1 - p_i), \quad (4)$$

$$\text{FFN}(\mathbf{X}) = \sigma(\mathbf{X}(\mathbf{W}_{\text{fc1}} + \Delta\mathbf{W}_{\text{fc1}})) \odot m \cdot (\mathbf{W}_{\text{fc2}} + \Delta\mathbf{W}_{\text{fc2}}), \quad (5)$$

where m varies with each forward pass, randomly masking the outputs of the intermediate dimensions.

Cross-layer Knowledge Dropping. In addition to the intra-layer perspective strategy, based on our findings in Section 3.2, we introduce randomness from a cross-layer perspective to alleviate the PEFT module's dependency on specific layers' knowledge. Specifically, we introduce a binary element z before each FFN layer to decide whether to drop the output of that layer. For each layer, the value of z is determined based on the rate p_c as follows:

$$z \sim \text{Bernoulli}(1 - p_c), \quad (6)$$

$$\widetilde{\text{FFN}}(\mathbf{X}) = z \cdot \text{FFN}(\mathbf{X}), \quad (7)$$

where $\widetilde{\text{FFN}}(\mathbf{X})$ represents the final output of the FFN layer. This strategy introduces variability in

which layers contribute to the final output, reducing PEFT modules' dependency on any specific layer.

By integrating these two strategies, the intra-layer masking introduces fine-grained randomness, ensuring that the PEFT module does not overly rely on certain knowledge within the FFN layer. Meanwhile, the cross-layer knowledge dropping prevents the PEFT module from becoming overly dependent on the knowledge of certain layers, thereby encouraging PEFT modules to capture persistent task-specific patterns within the attention mechanisms. These strategies together enable PEFT modules to maintain effectiveness across different versions of the base model.

4.2 Theoretical Analysis

To provide theoretical insights into the effectiveness of our approach, we analyze how Trans-PEFT influences the transferability of PEFT modules across different model versions.

Let $\mathcal{M}_0$ and $\mathcal{M}_1$ denote two versions of a base transformer model, where $\mathcal{M}_1$ is updated from $\mathcal{M}_0$. We define $\mathbf{W}_{\text{ffn}} = \{\mathbf{W}_{\text{fc1}}, \mathbf{W}_{\text{fc2}}\}$ as the weights in FFN sub-layers, $\theta_{\text{ffn}} = \{\Delta\mathbf{W}_{\text{fc1}}, \Delta\mathbf{W}_{\text{fc2}}\}$ as the PEFT parameters for FFN sub-layers, $\mathbf{W}_{\text{att}}$ as the attention sub-layer weights, and $\Delta\mathbf{W}_{\text{att}}$ as the corresponding PEFT parameters. For an input $\mathbf{X} \in \mathbb{R}^d$, the output of a transformer layer can be expressed as:

$$y = \text{Attn}(\mathbf{X}) + \text{FFN}(\text{Attn}(\mathbf{X})), \quad (8)$$

where $\text{Attn}(\cdot)$ and $\text{FFN}(\cdot)$ denote the attention and FFN operations, respectively. When integrating PEFT modules (e.g., LoRA), the output becomes:

$$y = \text{Attn}(\mathbf{X}; \theta_{\text{att}}) + \text{FFN}(\text{Attn}(\mathbf{X}; \theta_{\text{att}}); \theta_{\text{ffn}}), \quad (9)$$

with task-specific loss $\mathcal{L}(\theta_{\text{att}}, \theta_{\text{ffn}}; \mathcal{M})$.

As observed in Figure 1 and 2, when the base model is updated from $\mathcal{M}_0$ to $\mathcal{M}_1$, the attention sub-layers in both versions related to task-specific fine-tuning remain largely unchanged: $\mathbf{W}_{\text{att}}^{(1)} \approx \mathbf{W}_{\text{att}}^{(0)}$. In contrast, the FFN sub-layers undergo significant parameter shifts. Therefore, we assume:

1. *Attention Stability*: $\|\mathbf{W}_{\text{att}}^{(1)} - \mathbf{W}_{\text{att}}^{(0)}\|_2 \leq \epsilon_{\text{att}} \ll 1$

2. *FFN Perturbation Boundedness*: $\|\mathbf{W}_{\text{ffn}}^{(1)} - \mathbf{W}_{\text{ffn}}^{(0)}\|_2 \leq \rho$

3. *Loss Smoothness*: $\mathcal{L}$ is L-Lipschitz in $\mathbf{W}_{\text{att}}, \mathbf{W}_{\text{ffn}}$ and β-Lipschitz in θ_{ffn}

4. *Loss Decomposability*: The total loss can be decomposed into additive components from the attention and FFN sub-layers $\mathcal{L}(\theta; \mathcal{M}) = \mathcal{L}_{\text{att}}(\theta_{\text{att}}; \mathcal{M}) + \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}; \mathcal{M})$

Based on these assumptions, Trans-PEFT is designed to maintain the effectiveness of PEFT modules when transferred from $\mathcal{M}_0$ to $\mathcal{M}_1$ by preserving dependencies on stable attention sub-layers while adapting to FFN changes through controlled randomness. Below, we provide an upper bound on the loss discrepancy when applying the PEFT parameters fine-tuned on the $\mathcal{M}_0$ to the updated model $\mathcal{M}_1$ by Trans-PEFT.

Theorem 1 *Let $\theta_{\text{att}}^{*(0)}$, $\theta_{\text{ffn}}^{*(0)}$ be the PEFT parameters fine-tuned on $\mathcal{M}_0$ with Trans-PEFT. For $\mathcal{M}_1$, the loss discrepancy satisfies:*

$$|\mathcal{L}(\theta^{*(0)}; \mathcal{M}_1) - \mathcal{L}(\theta^{*(0)}; \mathcal{M}_0)| \leq$$
$$\underbrace{L\rho}_{\text{FFN Shift}} + \underbrace{2\beta\|\theta_{\text{ffn}}^{*(0)} - \theta_{\text{ffn}}^{*(1)}\|}_{\text{Parameter Deviation}} + \underbrace{C \cdot (p_i + p_c)}_{\text{Regularization}},$$
(10)

where $\theta_{\text{ffn}}^{*(1)}$ is the hypothetical optimal PEFT parameter for $\mathcal{M}_1$. And the loss function satisfies smoothness and decomposability.

This bound reveals three key components affecting the transfer performance: First, the FFN shift term ($L\rho$) reflects the impact of base model updates; Second, the parameter deviation term measures the proximity between our transferred PEFT parameters and the optimal PEFT parameters for $\mathcal{M}_1$, which can be reduced as Trans-PEFT encourages PEFT modules to capture persistent patterns in attention; Third, the regularization term, which is controlled by Trans-PEFT, quantifies the performance penalty caused by its stochastic perturbations (e.g., masking and dropping). Specifically, Intra-layer Knowledge Masking reduces the model's sensitivity to intra-layer dimension perturbations by p_i, and Cross-layer Knowledge Dropping enhances parameter robustness against cross-layer variations by p_c. Through adjustment of p_i and p_c, we balance learning on $\mathcal{M}_0$ and adaptability to $\mathcal{M}_1$. The complete proof is in Appendix A.

5 Experiments

5.1 Experimental Setup

Base Models. To thoroughly evaluate the transferability of our approach, we perform experiments involving seven base models from three different

Type	Method	ARC-c	SIQA	WinoG.	BoolQ	ARC-e	PIQA	OBQA	HellaS.	Avg.
	Fine-tune$_n$	82.3	78.4	84.5	72.6	92.9	88.3	87.6	94.0	85.1
LoRA	Fine-tune$_o$	78.8	79.9	84.1	73.1	89.6	87.8	86.4	93.1	84.1
	Direct Trans.$_{o \to n}$	83.8	72.3	78.9	18.1	92.3	86.7	85.0	35.4	69.1
	Trans-PEFT$_{o \to n}$	84.2	80.1	85.2	63.3	92.9	87.3	88.2	93.1	84.3
	Fine-tune$_n$	87.1	80.2	84.6	72.3	95.3	90.3	91.8	95.0	87.1
Adapter	Fine-tune$_o$	86.5	80.6	85.3	75.3	93.7	90.1	88.4	95.0	86.9
	Direct Trans.$_{o \to n}$	81.7	76.7	70.6	39.6	90.1	85.4	85.8	84.2	76.8
	Trans-PEFT$_{o \to n}$	88.2	78.9	82.1	69.6	95.4	88.4	90.6	93.2	85.8
	Fine-tune$_n$	87.9	79.4	86.0	74.5	96.0	90.2	93.2	95.0	87.8
DoRA	Fine-tune$_o$	84.9	81.6	86.3	74.9	93.8	90.0	90.2	95.3	87.1
	Direct Trans.$_{o \to n}$	53.8	64.0	54.5	53.9	60.5	41.7	62.2	50.9	55.2
	Trans-PEFT$_{o \to n}$	88.1	81.1	84.0	65.6	95.0	88.5	90.8	94.1	85.9

Table 1: Results on commonsense reasoning tasks. We evaluate the transfer performance of various PEFT methods on Qwen models. $o \to n$ denotes directly applying PEFT modules fine-tuned on the old version to the new version without re-tuning.

Model	Method	Mathematical Reasoning		Code Generation		
		GSM8K	MATH	HumanEval	MBPP	Average
Qwen (2→2.5)	Fine-tune$_n$	84.91	49.32	75.00	71.96	70.30
	Fine-tune$_o$	80.97	44.10	62.80	68.52	64.10
	Direct Trans.$_{o \to n}$	40.03	32.08	75.61	69.84	54.39
	Trans-PEFT$_{o \to n}$	84.61	47.66	79.27	74.07	71.40
InternLM (2→2.5)	Fine-tune$_n$	79.61	39.08	63.41	64.81	61.73
	Fine-tune$_o$	74.37	28.88	47.56	62.43	53.31
	Direct Trans.$_{o \to n}$	39.50	25.34	59.76	62.70	46.83
	Trans-PEFT$_{o \to n}$	77.56	38.86	64.02	63.49	60.98

Table 2: Results on mathematical reasoning and code generation tasks. We evaluate the transfer performance on Qwen and InternLM base models with LoRA.

sources, all of which are updated through continual pre-training. These include representing general updates from Qwen2-7B to Qwen2.5-7B in the Qwen family, InternLM2-7B to InternLM2.5-7B in the InternLM family, and domain-specific updates from DeepSeek-7B (DeepSeek-AI et al., 2024) to DeepSeek-Coder-7B-v1.5 (Guo et al., 2024) and DeepSeek-Math-7B (Shao et al., 2024).

Datasets. The datasets used in our experiments primarily cover three domains: commonsense reasoning, mathematical reasoning, and code generation. In the commonsense reasoning tasks, we follow Hu et al. (2023) by fine-tuning and evaluating the model on the CommonsenseQA dataset including BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2019), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2019), ARC-

e, ARC-c (Clark et al., 2018), and OBQA (Mihaylov et al., 2018) datasets. In the mathematical reasoning tasks, we fine-tune the model on a 100K subset of the MetaMathQA dataset (Yu et al., 2024) and evaluate transfer performance on GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) datasets. In the code generation tasks, we fine-tune the model on the CodeFeedback105K dataset (Zheng et al., 2025; Meng et al., 2024) and evaluate performance on HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) datasets.

Baselines and Settings. We evaluate our approach using three representative PEFT methods: LoRA (Hu et al., 2022), Adapter (Houlsby et al., 2019), and DoRA (Liu et al., 2024), which is a popular variant of LoRA. We compare our method with the following baselines: **Fine-tune$_o$**, which de-

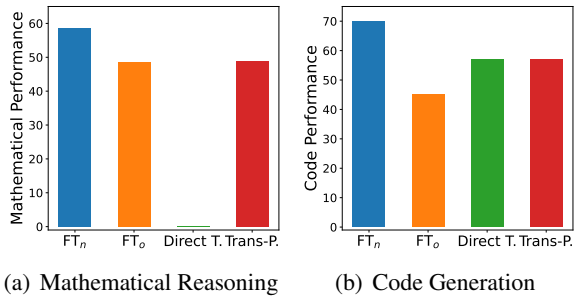(a) Mathematical Reasoning   (b) Code Generation

Figure 5: Results on base models with domain updates. We evaluate the transfer performance on DeepSeek base models with Adapter. For code tasks, we transfer from the base model to the code-updated version, while for math tasks, we transfer to the math-updated version.

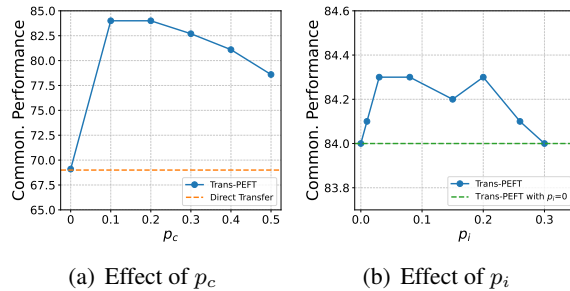

(a) Effect of p_c (b) Effect of p_i

Figure 6: Effects of p_c and p_i in our proposed strategies. The results are obtained on commonsense reasoning tasks using LoRA. In (a), we fix $p_i = 0$ and vary p_c. In (b), we use the optimal setting $p_c = 0.2$ and vary p_i.

notes fine-tuning PEFT on the old version of the base model and evaluating it on the old version; **Fine-tune$_n$**, which denotes fine-tuning PEFT on the new version and evaluating it on the new version, represents the ideal performance; and **Direct Transfer$_{o \to n}$**, which fine-tuning PEFT on the old version and directly transferring to the new version without any re-tuning. Our proposed method, **Trans-PEFT$_{o \to n}$**, applies two novel strategies to fine-tune PEFT on the old version, and then directly applies it to the new version without any re-tuning. More details are provided in Appendix C.

5.2 Experiments on Different PEFT Types

Table 1 presents the experimental results of three different types of PEFT on the Qwen base models for commonsense reasoning tasks. As shown, the Trans-PEFT method consistently outperforms the Direct Transfer approach across all three PEFT types, achieving substantial improvements of 15.2%, 9.0%, and 30.7% on LoRA, Adapter, and DoRA, respectively. Notably, the Direct Transfer method yields only 55.2% average performance with DoRA and underperforms on multiple tasks, indicating its impracticality for real-world applications. In contrast, Trans-PEFT achieves performance comparable to that of fine-tuning on the new version across all PEFT types, without the need for re-tuning. This demonstrates its ability to reduce dependency on certain knowledge and encourages the PEFT module to focus on the invariants in the attention mechanism effectively.

5.3 Experiments on Different Base Models

Table 2 presents results on different base models for mathematical reasoning and code generation tasks,

which are generally considered more challenging. In commonsense reasoning experiments, the two base model versions show comparable fine-tuning performance, suggesting that base model updates do not significantly enhance commonsense reasoning capability. However, for mathematical and coding tasks, both Qwen and InternLM model families show significantly higher scores when fine-tuning on the new versions (i.e., Fine-tune$_n$) compared to Fine-tune$_o$, indicating that the base model updates specifically enhance capabilities in these domains. As shown, Trans-PEFT not only outperforms Fine-tune$_o$ in overall performance but also approaches the effectiveness of the re-tuning method Fine-tune$_n$, demonstrating its ability to leverage the performance gains from continual pre-training of the base model. This success empirically validates our theoretical analysis, showing that Trans-PEFT can minimize the second term representing parameter deviation in fine-tuning processes. In contrast, the Direct Transfer approach not only fails to utilize the performance improvements from model updates but also underperforms compared to Fine-tune$_o$. This suggests that the fine-tuned modules become ineffective after transfer.

Furthermore, we conduct experiments on the base model updated in the specific domain, as shown in Figure 5. In mathematical and coding tasks, Trans-PEFT not only maintains the performance of the original model but also achieves improvements in coding tasks and slight enhancements in mathematical tasks. In comparison, Direct Transfer, while matching our performance in coding tasks, becomes nearly unusable after transferring to the more complex mathematical tasks. This demonstrates that our approach effectively captures task-specific patterns in attention mechanisms, en-

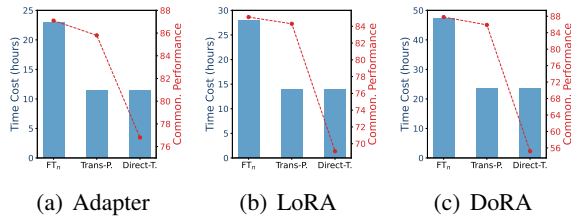|	(a) Adapter	(b) LoRA	(c) DoRA

Figure 7: Fine-tuning time cost and performance of different methods across different types of PEFT on Qwen models. The fine-tuning time is tested on an A800 GPU, which includes the time overhead from fine-tuning on both the old and new base model versions.

abling PEFT transfer across base model versions.

5.4 Analysis

Effect of p_c and p_i. We investigate the effect of two key parameters in Trans-PEFT: the layer drop probability p_c and dimension masking probability p_i. These parameters control the regularization effects that reduce dependence on certain knowledge, thereby capturing invariant task-specific patterns. The parameter p_c determines the probability of dropping each FFN layer during forward propagation. As illustrated in Figure 6(a), when p_c is zero, neither of our proposed strategies is employed, resulting in suboptimal transfer performance. As p_c increases, transfer performance improves, peaking at $p_c = 0.2$. Further increases in p_c begin to affect PEFT learning, leading to a decline in transfer performance.

The parameter p_i controls the proportion of FFN dimensions masked during forward propagation. As shown in Figure 6(b), when $p_i = 0$, only the layer knowledge drop is applied. Moderate increases in p_i further improve transfer performance compared to using layer knowledge drop alone, validating our dual-strategy approach. However, excessive p_i values negatively affect PEFT learning, similar to the effects of high p_c values.

Fine-tuning Efficiency. We conduct an analysis to assess the impact of Trans-PEFT on enhancing fine-tuning efficiency. As shown in Figure 7, fine-tuning different PEFT methods on the commonsense reasoning dataset typically takes 14 to 24 GPU hours on an A800 GPU. If re-tuning is conducted, including the initial fine-tuning time on the old version, the total time doubles to 28 to 48 hours (an additional 20 GPU hours on average). In practical applications, this efficiency gap becomes even more pronounced. For instance, when

dealing with thousands of customized PEFT modules, the cumulative re-tuning time could escalate to approximately 20,000 GPU hours[3]. Our Trans-PEFT approach offers a solution to this challenge by effectively eliminating the need for re-tuning while maintaining comparable performance levels. It requires only a single fine-tuning process on the old base model version, making it highly practical and efficient. This significant reduction in computational resources and time makes Trans-PEFT particularly valuable for large-scale applications and frequent model updates.

6 Related Work

6.1 Parameter Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) methods have several mainstream approaches to adding trainable modules: Prompt-based methods (Lester et al., 2021; Li and Liang, 2021; Liu et al., 2022) fine-tune soft prompts added before the hidden states in model layers, but are less commonly used due to their relatively unstable performance and reduced input length (Han et al., 2024). Adapter Tuning and its variants (Houlsby et al., 2019; He et al., 2022) insert trainable modules between model layers, offering efficient parameter updates. Reparameterization methods, such as LoRA and its variants (Hu et al., 2022; Liu et al., 2024; Jiang et al., 2024; Gu et al., 2025), take a different approach by inserting low-rank modules parallel to the model's linear layers to approximate updates of the original weights. Both of these methods typically achieve performance comparable to full parameter fine-tuning. Building on these established approaches, we propose the Trans-PEFT method, which enables efficient transfer of fine-tuned PEFT modules across different versions of the base model while maintaining performance.

6.2 Transfer Approaches on Updating Base Models

The challenge of transferring PEFT modules across base models is first introduced by Su et al. (2022). Subsequent studies (Qin et al., 2023; Lester et al., 2022) discover that while the homologous model could reuse PEFT modules to some extent, there remain significant performance gaps compared to

---

[3]It's important to note that, unlike pre-training, batching and updating thousands of user-customized PEFT modules in parallel presents significant challenges. This is primarily due to the divergent objectives of different modules, necessitating multiple forward passes to update each module individually.

re-tuning, thereby limiting their practical utility. To bridge this gap, Trans-PEFT focuses on transferring PEFT modules between models updated via continual pre-training and introduces a novel transfer mechanism that maintains performance while reducing computational costs. From a data privacy perspective, Wang et al. (2024) proposes to eliminate the need for storing user data for re-tuning by using synthetic data generated from the old model version. While this approach preserves privacy, it still incurs substantial computational costs due to the need for re-tuning the PEFT modules. Trans-PEFT provides a more efficient solution by enhancing the fine-tuning process of the old version, which both reduces computational demands and eliminates the need for storing sensitive user data. Further discussion of related work is provided in Appendix B.

7 Conclusion

This paper introduces Trans-PEFT, a novel method that enables PEFT modules to work on updated base models. Our analysis reveals that base model updates primarily affect task-specific knowledge stored in FFN sub-layers while having minimal impact on task-specific patterns in attention mechanisms. Leveraging these insights, we develop both within-layer and cross-layer strategies to decrease PEFT modules' dependence on FFN-stored knowledge, instead encouraging them to capture persistent patterns within the attention mechanisms. Extensive experiments show that Trans-PEFT successfully maintains performance across base model updates without requiring re-tuning, thereby significantly reducing the maintenance cost of PEFT.

Limitations

Trans-PEFT demonstrates efficacy in transferring PEFT modules between continuously trained base models. However, our method faces constraints when applied to re-pretraining scenarios. Such scenarios often include architectural changes or large-scale dataset expansions, like updating from LLaMA2 (Touvron et al., 2023b) to LLaMA3 (Grattafiori et al., 2024). This limitation stems from the fundamental differences in pre-trained parameter spaces between base models with different random initializations (Imfeld et al., 2024). Since PEFT operates within a specific parameter space (Aghajanyan et al., 2021), transferring parameters across completely distinct spaces remains infeasible. Exploring ways to adapt our method to these conditions represents a direction for future research.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328, Online. Association for Computational Linguistics.

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *CoRR*, abs/2108.07732.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. PIQA: reasoning about physical commonsense in natural language. *CoRR*, abs/1911.11641.

Lucas Caccia, Alan Ansell, Edoardo Ponti, Ivan Vulić, and Alessandro Sordoni. 2025. Training plug-n-play knowledge modules with deep context distillation. *Preprint*, arXiv:2503.08727.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang,

Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Yilong Chen, Junyuan Shang, Zhengyu Zhang, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. 2024. Mixture of hidden-dimensions transformer. *Preprint*, arXiv:2412.05644.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

DeepSeek-AI, :, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, Wenjie Hu, Panpan Huang, Erhang Li, Guowei Li, Jiashi Li, Yao Li, Y. K. Li, Wenfeng Liang, Fangyun Lin, A. X. Liu,

Bo Liu, Wen Liu, Xiaodong Liu, Xin Liu, Yiyuan Liu, Haoyu Lu, Shanghao Lu, Fuli Luo, Shirong Ma, Xiaotao Nie, Tian Pei, Yishi Piao, Junjie Qiu, Hui Qu, Tongzheng Ren, Zehui Ren, Chong Ruan, Zhangli Sha, Zhihong Shao, Junxiao Song, Xuecheng Su, Jingxiang Sun, Yaofeng Sun, Minghui Tang, Bingxuan Wang, Peiyi Wang, Shiyu Wang, Yaohui Wang, Yongji Wang, Tong Wu, Y. Wu, Xin Xie, Zhenda Xie, Ziwei Xie, Yiliang Xiong, Hanwei Xu, R. X. Xu, Yanhong Xu, Dejian Yang, Yuxiang You, Shuiping Yu, Xingkai Yu, B. Zhang, Haowei Zhang, Lecong Zhang, Liyue Zhang, Mingchuan Zhang, Minghua Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, Qihao Zhu, and Yuheng Zou. 2024. Deepseek llm: Scaling open-source language models with longtermism. *Preprint*, arXiv:2401.02954.

Weilong Dong, Xinwei Wu, Renren Jin, Shaoyang Xu, and Deyi Xiong. 2025. CONTRANS: Weak-to-strong alignment engineering via concept transplantation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4130–4148, Abu Dhabi, UAE. Association for Computational Linguistics.

Jessica Echterhoff, Fartash Faghri, Raviteja Vemulapalli, Ting-Yao Hu, Chun-Liang Li, Oncel Tuzel, and Hadi Pouransari. 2024. Muscle: A model update strategy for compatible llm evolution. *Preprint*, arXiv:2407.09435.

Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing transformer depth on demand with structured dropout. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

William Fleshman and Benjamin Van Durme. 2024. Re-adapt: Reverse engineered adaptation of large language models. *Preprint*, arXiv:2405.15007.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary,

Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu

Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Naibin Gu, Peng Fu, Xiyu Liu, Bowen Shen, Zheng Lin, and Weiping Wang. 2024. Light-PEFT: Lightening parameter-efficient fine-tuning via early pruning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7528–7541, Bangkok, Thailand. Association for Computational Linguistics.

Naibin Gu, Zhenyu Zhang, Xiyu Liu, Peng Fu, Zheng Lin, Shuohuan Wang, Yu Sun, Hua Wu, Weiping Wang, and Haifeng Wang. 2025. Beamlora: Beam-constraint low-rank adaptation. *Preprint*, arXiv:2502.13604.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. Deepseek-coder: When the large language model meets programming – the rise of code intelligence. *Preprint*, arXiv:2401.14196.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Transactions on Machine Learning Research*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. *Preprint*, arXiv:2004.11207.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning.
In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. *CoRR*, abs/2103.03874.

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

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.

Shih-Cheng Huang, Pin-Zu Li, Yu-chi Hsu, Kuang-Ming Chen, Yu Tung Lin, Shih-Kai Hsiao, Richard Tsai, and Hung-yi Lee. 2024. Chat vector: A simple approach to equip LLMs with instruction following and model alignment in new languages. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10943–10959, Bangkok, Thailand. Association for Computational Linguistics.

Moritz Imfeld, Jacopo Graldi, Marco Giordano, Thomas Hofmann, Sotiris Anagnostidis, and Sidak Pal Singh. 2024. Transformer fusion with optimal transport. In *The Twelfth International Conference on Learning Representations*.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. 2024. Mora: High-rank updating for parameter-efficient fine-tuning. *Preprint*, arXiv:2405.12130.

Feihu Jin, Jiajun Zhang, and Chengqing Zong. 2023. Parameter-efficient tuning for large language model without calculating its gradients. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 321–330, Singapore. Association for Computational Linguistics.

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

Brian Lester, Joshua Yurtsever, Siamak Shakeri, and Noah Constant. 2022. Reducing retraining by recycling parameter-efficient prompts. *Preprint*, arXiv:2208.05577.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32100–32121. PMLR.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. PiSSA: Principal singular values and singular vectors adaptation of large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Yujia Qin, Cheng Qian, Xu Han, Yankai Lin, Huadong Wang, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023. Recyclable tuning for continual pre-training. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11403–11426, Toronto, Canada. Association for Computational Linguistics.

Qwen, :, 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, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WINOGRANDE: an adversarial winograd schema challenge at scale. *CoRR*, abs/1907.10641.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Rana Muhammad Shahroz, Pingzhi Li, Sukwon Yun, Zhenyu Wang, Shahriar Nirjon, Chau-Wai Wong, and Tianlong Chen. 2025. Portllm: Personalizing evolving large language models with training-free and portable model patches. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. 2022. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3949–3969, Seattle, United States. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu,

Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023a. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Runqian Wang, Soumya Ghosh, David Cox, Diego Antognini, Aude Oliva, Rogerio Feris, and Leonid Karlinsky. 2024. *Trans-LoRA*: towards data-free transferable parameter efficient finetuning. *Preprint*, arXiv:2405.17258.

Zijun Wu, Yongkang Wu, and Lili Mou. 2024. Zero-shot continuous prompt transfer: Generalizing task semantics across language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Guangxuan Xiao, Ji Lin, and Song Han. 2023. Offsite-tuning: Transfer learning without full model. *Preprint*, arXiv:2302.04870.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Yang You, Guiming Xie, Xuejian Gong, and Kunlong Zhou. 2025. Train small, infer large: Memory-efficient lora training for large language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Kaiyan Zhang, Ning Ding, Biqing Qi, Xuekai Zhu, Xinwei Long, and Bowen Zhou. 2023. CRaSh: Clustering, removing, and sharing enhance fine-tuning without full large language model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9612–9637, Singapore. Association for Computational Linguistics.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. 2025. Opencodeinterpreter: Integrating code generation with execution and refinement. *Preprint*, arXiv:2402.14658.

A Proof of Theorem 1

A.1 Theorem Statement

Let $\mathcal{M}_0$ and $\mathcal{M}_1$ denote two versions of a base transformer model, where $\mathcal{M}_1$ is updated from $\mathcal{M}_0$. We define $\mathbf{W}^{(0)} = \{\mathbf{W}_{\text{ffn}}^{(0)}, \mathbf{W}_{\text{att}}^{(0)}\}$ as the weights of $\mathcal{M}_0$, $\theta^{(0)} = \{\theta_{\text{ffn}}^{(0)}, \theta_{\text{att}}^{(0)}\}$ as the fine-tuned PEFT weights on $\mathcal{M}_0$; $\mathbf{W}^{(1)} = \{\mathbf{W}_{\text{ffn}}^{(1)}, \mathbf{W}_{\text{att}}^{(1)}\}$ as the weights of $\mathcal{M}_1$, $\theta^{(1)} = \{\theta_{\text{ffn}}^{(1)}, \theta_{\text{att}}^{(1)}\}$ as the fine-tuned PEFT weights on $\mathcal{M}_1$.

Assumptions.

1. *Attention Stability*: $\|\mathbf{W}_{\text{att}}^{(1)} - \mathbf{W}_{\text{att}}^{(0)}\|_2 \leq \epsilon_{\text{att}} \ll 1$

2. *FFN Perturbation Boundedness*: $\|\mathbf{W}_{\text{ffn}}^{(1)} - \mathbf{W}_{\text{ffn}}^{(0)}\|_2 \leq \rho$

3. *Loss Smoothness*: $\mathcal{L}$ is L-Lipschitz in $\mathbf{W}_{\text{att}}, \mathbf{W}_{\text{ffn}}$ and β-Lipschitz in θ_{ffn}

4. *Loss Decomposability*: The total loss can be decomposed into additive components from the attention and FFN sub-layers $\mathcal{L}(\theta; \mathcal{M}) = \mathcal{L}_{\text{att}}(\theta_{\text{att}}; \mathcal{M}) + \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}; \mathcal{M})$

Theorem 1. *Let $\theta^{*(0)} = \{\theta_{\text{att}}^{*(0)}, \theta_{\text{ffn}}^{*(0)}\}$ be the optimal PEFT parameters fine-tuned on $\mathcal{M}_0$ using Trans-PEFT. Under the assumptions of **attention stability** and **FFN perturbation boundedness**, the loss discrepancy on $\mathcal{M}_1$ satisfies:*

$$
\begin{aligned}
|\mathcal{L}(\theta^{*(0)};&\mathcal{M}_1) - \mathcal{L}(\theta^{*(0)}; \mathcal{M}_0)| \\
&\leq L\rho + 2\beta\|\theta_{\text{ffn}}^{*(0)} - \theta_{\text{ffn}}^{*(1)}\| + C \cdot (p_i + p_c),
\end{aligned}
\tag{11}
$$

where $\theta_{\text{ffn}}^{*(1)}$ is the hypothetical optimal PEFT parameter for $\mathcal{M}_1$. And the loss function satisfies smoothness and decomposability.

A.2 Proof

Since Trans-PEFT introduces a mask in the FFN module $\tilde{\mathbf{W}}_{\text{ffn}}^{(0)} = \mathbf{W}_{\text{ffn}}^{(0)} + \delta(m, z)$, we can therefore derive:

$$
\begin{aligned}
|\mathcal{L}(\theta^{*(0)}; \mathcal{M}_1) &- \mathcal{L}(\theta^{*(0)}; \mathcal{M}_0)| \\
&= |\mathcal{L}(\theta^{*(0)}; [\mathbf{W}_{\text{att}}^{(1)}, \mathbf{W}_{\text{ffn}}^{(1)}]) - \mathcal{L}(\theta^{*(0)}; [\mathbf{W}_{\text{att}}^{(0)}, \tilde{\mathbf{W}}_{\text{ffn}}^{(0)}])|.
\end{aligned}
\tag{12}
$$

The total loss can be separated into attention and FFN components (Assumption 4):

$$
\begin{aligned}
|\mathcal{L}(\theta^{*(0)}; [\mathbf{W}_{\text{att}}^{(1)}&, \mathbf{W}_{\text{ffn}}^{(1)}]) - \mathcal{L}(\theta^{*(0)}; [\mathbf{W}_{\text{att}}^{(0)}, \tilde{\mathbf{W}}_{\text{ffn}}^{(0)}])| \\
=&|\mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(1)}) + \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(0)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})| \\
=&|\mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(1)}) - \mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(0)}) + \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})| \\
\leq&|\mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(1)}) - \mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(0)})| + |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})|.
\end{aligned}
\tag{13}
$$

Since $\mathbf{W}_{\text{att}}^{(1)} \approx \mathbf{W}_{\text{att}}^{(0)}$ (Assumption 1) and the L_{att}-Lipschitz continuity of $\mathcal{L}_{\text{att}}$ (Assumption 3), the attention loss discrepancy is negligible:

$$
|\mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(1)}) - \mathcal{L}_{\text{att}}(\theta_{\text{att}}^{*(0)}; \mathbf{W}_{\text{att}}^{(0)})| \leq L_{\text{att}}\epsilon_{\text{att}} \approx 0.
\tag{14}
$$

Thus, the total loss discrepancy is dominated by the FFN term:

$$
\begin{aligned}
|\mathcal{L}(\theta^{*(0)};&\mathcal{M}_1) - \mathcal{L}(\theta^{*(0)}; \mathcal{M}_0)| \\
&\leq |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})|.
\end{aligned}
\tag{15}
$$

14779

Thus, we can derive the sub-terms of the loss:

$$
\begin{aligned}
&|\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})| \\
&= |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)}) + \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})| \\
&\leq |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)})| + \underbrace{|\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \tilde{\mathbf{W}}_{\text{ffn}}^{(0)})|}_{\text{sub-term } \mathcal{R}} \\
&\leq \underbrace{|\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(1)})|}_{\text{sub-term } \mathcal{A}_1} + |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)})| + \mathcal{R} \\
&\leq \mathcal{A}_1 + \underbrace{|\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(0)})|}_{\text{sub-term } \mathcal{B}} + \underbrace{|\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(0)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)})|}_{\text{sub-term } \mathcal{A}_2} + \mathcal{R} \\
&= \mathcal{A}_1 + \mathcal{B} + \mathcal{A}_2 + \mathcal{R}.
\end{aligned}
\tag{16}
$$

Using the L-Lipschitz continuity of $\mathcal{L}_{\text{ffn}}$ (Assumption 3) and the bound of FFN weights (Assumption 2), we bound the sub-term $\mathcal{B}$:

$$
\begin{aligned}
\mathcal{B} &= |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(0)})| \\
&\leq L \cdot \|\mathbf{W}_{\text{ffn}}^{(1)} - \mathbf{W}_{\text{ffn}}^{(0)}\|_2 \leq L\rho.
\end{aligned}
\tag{17}
$$

Using the β-Lipschitz in θ_{ffn} (Assumption 3) we bound the sub-term $\mathcal{A}_1 + \mathcal{A}_2$:

$$
\begin{aligned}
\mathcal{A}_1 + \mathcal{A}_2 &= |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(1)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(1)})| + |\mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(1)}; \mathbf{W}_{\text{ffn}}^{(0)}) - \mathcal{L}_{\text{ffn}}(\theta_{\text{ffn}}^{*(0)}; \mathbf{W}_{\text{ffn}}^{(0)})| \\
&\leq 2\beta \|\theta_{\text{ffn}}^{*(0)} - \theta_{\text{ffn}}^{*(1)}\|.
\end{aligned}
\tag{18}
$$

On model $\mathcal{M}_0$, the fine-tuning objective of Trans-PEFT is to minimize the expected loss:

$$
\theta_{\text{ffn}}^{*(0)} = \arg\min_{\theta} \mathbb{E}_{m,z}\left[\mathcal{L}_{\text{ffn}}(\theta; \mathbf{W}_{\text{ffn}}^{(0)} + \delta(m, z))\right],
\tag{19}
$$

where $\delta(m, z)$ represents the implicit perturbation induced by masking m using probability p_i and dropping z using probability p_c. Perform a second-order Taylor expansion of the loss function at the weight $\mathbf{W}_{\text{ffn}}^{(0)}$:

$$
\mathcal{R} = |\mathcal{L}_{\text{ffn}}(\theta; \mathbf{W}_{\text{ffn}}^{(0)} + \delta) - \mathcal{L}_{\text{ffn}}(\theta; \mathbf{W}_{\text{ffn}}^{(0)})| \approx \nabla_{\mathbf{W}}\mathcal{L}_{\text{ffn}} \cdot \delta + \frac{1}{2}\delta^\top \nabla_{\mathbf{W}}^2 \mathcal{L}_{\text{ffn}} \cdot \delta.
\tag{20}
$$

After taking the expectation, because $\delta(m, z)$ is a zero-mean random variable, i.e., $\mathbb{E}_{m,z}[\delta(m, z)] = 0$, the first-order term vanishes. And the second-order term dominates the effect of the perturbation:

$$
\mathbb{E}[\mathcal{R}] \approx \frac{1}{2}\mathbb{E}\left[\delta^\top \nabla_{\mathbf{W}}^2 \mathcal{L}_{\text{ffn}} \cdot \delta\right].
\tag{21}
$$

Assume the maximum eigenvalue of the Hessian matrix $\nabla_{\mathbf{W}}^2 \mathcal{L}_{\text{ffn}}$ is $\lambda_{\max}$, then:

$$
\mathbb{E}\left[\delta^\top \nabla_{\mathbf{W}}^2 \mathcal{L}_{\text{ffn}} \cdot \delta\right] \leq \lambda_{\max} \cdot \mathbb{E}\left[\|\delta\|_2^2\right],
\tag{22}
$$

where $\mathbb{E}\left[\|\delta\|_2^2\right]$ is introduced by Trans-PEFT via orthogonally designed strategies: intra-layer masking with probability p_i and cross-layer dropping with probability p_c:

$$
\mathbb{E}[\|\delta\|_2^2] = \mathbb{E}[\|\delta_i\|_2^2] + \mathbb{E}[\|\delta_c\|_2^2] \approx C_1 p_i + C_2 p_c,
\tag{23}
$$

where C_1 and C_2 represent the sum of the squared norms of the masked weights by Trans-PEFT. Thus, we have the bound of $\mathcal{R}$:

$$
\mathbb{E}\left[\delta^\top \nabla_{\mathbf{W}}^2 \mathcal{L}_{\text{ffn}} \cdot \delta\right] \leq \lambda_{\max} \cdot (C_1 p_i + C_2 p_c),
\tag{24}
$$

$$
\begin{aligned}
\mathbb{E}[\mathcal{R}] &\leq \frac{1}{2}\lambda_{\max} \cdot (C_1 p_i + C_2 p_c) \\
&\leq C \cdot (p_i + p_c),
\end{aligned}
\tag{25}
$$

where $C = \frac{1}{2}\lambda_{\max} \cdot \max(C_1, C_2)$. Based on the above derivation, we obtain the three terms in the theorem.

14780

B Extended Related Work

The challenge of cross-model transferability for parameter-efficient modules is first introduced by Su et al. (2022), which observes that transferring prompt module parameters from the original model to a new model often leads to performance degradation or even total failure. The most straightforward solution to this problem is to restore performance through re-finetuning on the new model. MUS-CLE (Echterhoff et al., 2024) attempts to mitigate this by leveraging knowledge distillation to ensure that performance is maintained in the new model, while works such as Trans-LoRA (Wang et al., 2024) and Plug-n-Play (Caccia et al., 2025) further reduce or eliminate the need for additional downstream data for finetuning via distillation strategies. However, these methods ultimately still depend on re-finetuning on the new model and thus do not genuinely enable parameter transferability.

Existing studies aiming for module parameter transferability across models can be categorized into three directions: (1) cross-size transfer within the same model family, (2) transfer between entirely different model architectures, and (3) transfer between two models where one is updated from the other via continual training. The first direction, typified by works Offsite-Tuning (Xiao et al., 2023), CRaSh (Zhang et al., 2023), Plug-in (Jin et al., 2023), and LoRAM (Zhang et al., 2025), tackles finetuning efficiency. Here, the modules are fine-tuned on a smaller, compressed version of the base model for efficiency and then transferred back to the original large model. These works mainly address issues of dimensionality mismatch rather than substantial parameter changes in the base model, making their approach orthogonal to ours.

The second direction attempts to transfer prompt-based modules between two completely distinct models (Lester et al., 2022; Wu et al., 2024; Dong et al., 2025), leveraging the small parameter size of prompt modules and alignment of overlapping vocabularies between the models. However, since the base models are independently pre-trained, there exist fundamental differences in their weight parameter spaces due to distinct initialization and training dynamics (Imfeld et al., 2024). As PEFT methods operate within these model-specific parameter spaces (Aghajanyan et al., 2021), such approaches suffer from severely limited transferability and cannot be extended to mainstream PEFT methods like LoRA or to more complex tasks.

The third direction addresses a common real-world scenario where models are incrementally updated via continual pre-training rather than re-pretraining from scratch. In this context, direct transfer represents the most straightforward solution. For instance, Chat Vector (Huang et al., 2024) and RE-Adapt (Fleshman and Durme, 2024) both focus on direct transfer across fine-tuned base model versions, which involves limited continual training. For more substantial continual pre-training updates, Recycle Tuning (Qin et al., 2023) and PortLLM (Shahroz et al., 2025) show that direct transfer between versions can outperform zero-shot learning on models updated through continual pre-training. However, Recycle Tuning also finds that the performance of the direct transfer on newer versions is often much lower than that achieved by re-tuning, which is consistent with our findings. Our proposed method, Trans-PEFT, goes further by considering both the changes of knowledge during model updates in FFN sub-layers and the invariance of task-specific patterns in attention sub-layers, enabling the transferred parameter modules to maintain fine-tuned performance after the model is updated, and even achieve performance comparable to re-tuning.

From a methodological perspective, the Cross-layer Knowledge Dropping strategy in Trans-PEFT shares some similarities with LayerDrop (Fan et al., 2020), which focuses on inference efficiency; in contrast, our goal is to reduce the dependency of PEFT modules on specific sub-layers and to enhance cross-model transferability, marking a fundamental difference in motivation and outcome.

C Implementation Details

Observational Experiments. In the observational experiments, we fine-tune both the old and new versions of the model using LoRA with a rank of 64. To obtain the activations, we use a fixed subset for both fine-tuned models and obtain the activations via inference.

Commonsense Reasoning Experiments. In commonsense reasoning experiments, we set the rank of all PEFT types to 32 and perform fine-tuning on the Commonsense170K dataset. The fine-tuning follows the settings of Liu et al. (2024), with a batch size of 16 over three epochs. For all experiments, we set the random seed to 42. Finetune_n and Finetune_o denote typical fine-tuning and use on the corresponding versions. For Direct Transfer

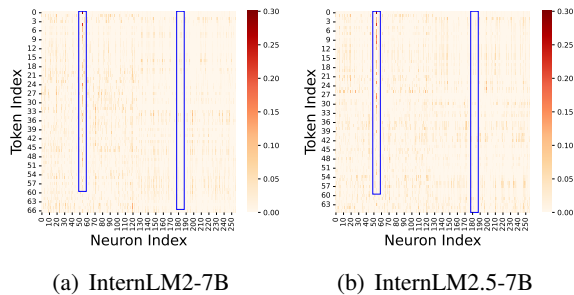(a) InternLM2-7B (b) InternLM2.5-7B

Figure 8: Comparison of activation distributions within attention sub-layers across different fine-tuned model versions on the code task. The visualization shows the same dimensions from the 18th layer, with similar regularities observed across other layers.
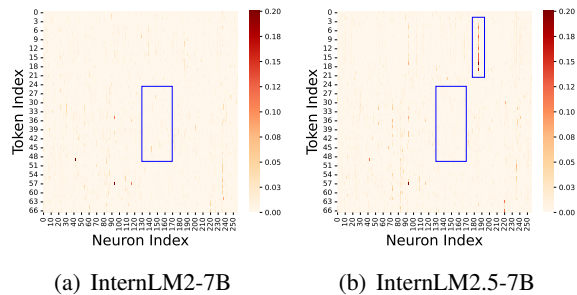

(a) InternLM2-7B (b) InternLM2.5-7B

Figure 9: Comparison of activation distributions within FFN sub-layers across different fine-tuned model versions on the code task. The visualization shows the same dimensions from the 18th layer, with similar regularities observed across other layers.

and Trans-PEFT, the PEFT modules are fine-tuned solely on the old version base model and then applied directly to the new version. For the Trans-PEFT parameters p_i and p_c, given the large dataset, we select from {0.1, 0.2, 0.3}.

Math Reasoning and Code Generation Experiments. In mathematical reasoning and code generation experiments, we set the rank of all PEFT types to 64. Following the experimental setup of Meng et al. (2024)'s setup, we fine-tune models on a 100K subset of MetaMathQA (Yu et al., 2024) and evaluate their 0-shot performance on GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) using the MetaMathQA codebase. For code generation tasks, we fine-tune models on the Code-Feedback105K dataset (Zheng et al., 2025; Meng et al., 2024) and assess their performance on HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) using the evalplus (Liu et al., 2023). For all experiments, we set the random seed to 42. As in commonsense experiments, we use a batch size of 16 and fine-tune over three epochs. For Trans-PEFT's parameter p_c, we consider values from {0.1, 0.2, 0.3}, and p_i from {0.01, 0.05, 0.1}. All experiments are conducted on two A800 GPUs.

D Extended Observations

D.1 Observations on Other Tasks

In Figures 8 and 9, we present the changes in activation distributions in the attention sub-layers and FFN sub-layers on the code generation task. As shown, the findings are consistent with those discussed in Section 3: task-specific pattern changes are less pronounced in the attention sub-layers, whereas knowledge storage changes are more significant in the FFN sub-layers.
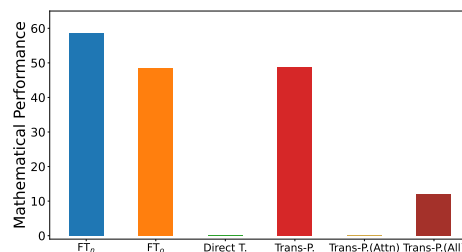

Figure 10: Results on mathematical reasoning tasks with applying dropping on the attention sub-layer. We evaluate the transfer performance on DeepSeek base models with Adapter.

D.2 Further Discussion on the Impact of Model Updates

To further validate our findings in Section 3 that continual pre-training primarily affects task-specific knowledge stored in FFN while having minimal impact on task-specific patterns in attention mechanisms, we conduct additional experiments with Trans-PEFT applied to different sub-layers. Specifically, we apply Trans-PEFT to the attention sub-layer to examine the effects of reducing PEFT's learning on these relatively stable task-specific patterns while maintaining its dependency on certain knowledge in FFN sub-layers. As shown in Figure 10, applying Trans-PEFT solely to attention sub-layers yields similar poor transfer results as Direct Transfer, failing to adapt to the updated base model. Moreover, when Trans-PEFT is applied to both attention and FFN sub-layers, the transfer performance declines compared to applying it to the FFN sub-layer alone. This is because introducing randomness in attention sub-layers affects the capture of task-specific patterns. These quantitative results provide additional empirical support for our findings.

14782

	GSM8K	MATH	HumanEval	MBPP	Avg. Perf. Improv.
$\text{Qwen}_{2\rightarrow2.5}$	p<0.01	p<0.01	p=0.03	p<0.01	$17.01_{\pm0.84}$
$\text{InternLM}_{2\rightarrow2.5}$	p<0.01	p<0.01	p=0.02	p=0.04	$14.15_{\pm0.40}$

Table 3: Results of t-tests comparing Trans-PEFT with Direct Transfer on mathematical reasoning and code generation tasks.

	Method	seed 1	seed 99	seed 1234	Average
GSM8K	Fine-tune_n	84.99	84.84	85.37	85.07
	Fine-tune_o	80.67	81.43	81.05	81.05
	$\text{Direct Trans.}_{o\rightarrow n}$	37.60	34.19	41.39	37.73
	$\text{Trans-PEFT}_{o\rightarrow n}$	84.76	85.14	84.91	84.94
MATH	Fine-tune_n	47.50	47.10	48.80	47.80
	Fine-tune_o	44.04	43.86	42.16	43.35
	$\text{Direct Trans.}_{o\rightarrow n}$	35.06	35.44	37.28	35.93
	$\text{Trans-PEFT}_{o\rightarrow n}$	47.76	45.74	48.78	47.43

Table 4: Results on mathematical reasoning tasks with multiple seeds. We evaluate the transfer performance on Qwen base models with LoRA.

E   Extended Experiments

E.1   Statistical Significance

Table 3 presents the statistical significance results for the experiments in Table 2, where we fix the random seed to 42 and perform three runs with different data orders. As shown, the p-values for all tasks comparing Trans-PEFT and Direct Transfer are less than 0.05, indicating that Trans-PEFT significantly outperforms Direct Transfer. Moreover, the difference in average performance is also statistically significant. These results demonstrate that Trans-PEFT can effectively enhance transfer performance on different tasks.

E.2   Stability Analysis of the Method

In all of our experiments, we use a fixed seed (seed=42) for fine-tuning. In Table 4, we further assess the performance stability of Trans-PEFT across different seeds. The overall performance trends are consistent with those in Table 2, and Trans-PEFT achieves significantly better results than Direct Transfer. This demonstrates that the ability of Trans-PEFT to reduce dependence on task-specific knowledge in the FFN sub-layers is minimally affected by seed.