Artificial Intelligence Review (2025) 58:227
https://doi.org/10.1007/s10462-025-11236-4

Parameter-efficient fine-tuning in large language models: a survey of methodologies

Luping Wang[1] · Sheng Chen[1] · Linnan Jiang[1] · Shu Pan[1] · Runze Cai[1] · Sen Yang[1] · Fei Yang[1]

Accepted: 10 April 2025

Abstract
The large language models, as predicted by scaling law forecasts, have made groundbreaking progress in many fields, particularly in natural language generation tasks, where they have approached or even surpassed human levels. However, the unprecedented scale of their parameters brings significant computational and storage costs. These large language models require substantial computational resources and GPU memory to operate. When adapting large language models to specific downstream tasks, their massive parameter scale poses a significant challenge in fine-tuning on hardware platforms with limited computational power and GPU memory. To address this issue, parameter-efficient fine-tuning (PEFT) offers a practical solution by efficiently adjusting the parameters of large pre-trained models to suit various downstream tasks. Specifically, PEFT adjusts the parameters of pre-trained large language models to adapt to specific tasks or domains, minimizing the introduction of additional parameters and the computational resources required. This review mainly introduces the preliminary knowledge of PEFT, the core ideas and principles of various PEFT algorithms, the applications of PEFT, and potential future research directions. By reading this review, we believe that interested parties can quickly grasp the PEFT methodology, thereby accelerating its development and innovation.

Keywords Fine-tuning · Parameter-efficient · Large language model · Deep learning · Artificial intelligence

1 Introduction

In recent years, large pre-trained models, commonly referred to as "large language models", have emerged as a significant advancement in the field of artificial intelligence. Due to their outstanding performance and versatility in various application contexts, these models have attracted plenty of attention and provoked much discussion. These models have impressive computing capabilities and extensive data resources, allowing them to excel in tackling intricate jobs. Within the field of natural language processing (NLP), notable interest is given to large language models (LLMs). These models demonstrate remarkable ingenuity

Extended author information available on the last page of the article

Published online: 03 May 2025

in text generation (Wu et al. 2023; Li et al. 2024b), machine translation (Zhu et al. 2023; Wang et al. 2023a), personalized chatbots (Zheng et al. 2023; Kim et al. 2023; Dan et al. 2023), text summarization (Zhang et al. 2019), sentiment analysis (Zhang et al. 2023a), and question-answering systems (Pan et al. 2024).

Nevertheless, the development of LLMs faces significant challenges and controversies. These models require substantial computational resources and data support, which can potentially jeopardize the environment and compromise privacy protection (Yao et al. 2024). Despite their impressive performance in specific tasks, these models still have limitations and error rates that need continuous optimization and improvement (Huang et al. 2024a; Huang and Chang 2022; Saparov and He 2022). When directly using LLMs for specific tasks, their performance often falls below desired levels. Consequently, fine-tuning LLMs has become a crucial method for enhancing model performance.

Parameter-efficient fine-tuning (PEFT) is a transfer learning method specifically developed to adapt the parameters of the large pre-trained models to suit new tasks and scenarios. This approach involves dynamically adjusting the model to enhance its effectiveness in performing certain tasks, taking into account the distinct features and requirements of the target task. The fine-tuning process typically entails improving the model architecture (Houlsby et al. 2019), optimizing parameters (Li and Liang 2021; Lester et al. 2021), and adapting learning strategies (Chen et al. 2020), among other considerations, to achieve better performance in new tasks. As the field of deep learning continues to evolve, techniques for optimizing and fine-tuning LLMs have also made significant advancements. Notable PEFT approaches include LoRA (Hu et al. 2021), adapter tuning (Zhang et al. 2023f), prefix tuning (Li and Liang 2021), prompt tuning (Lester et al. 2021), P tuning (Liu et al. 2024), BitFit (Zaken et al. 2021), and others. However, despite the significant achievements of large model fine-tuning techniques across several fields, there are always challenges and difficulties that need to be resolved. Overfitting mitigation, optimizing fine-tuning efficiency, and striking a learning balance between pre-training and fine-tuning tasks are a few examples of issues that need more investigation.

In recent years, hundreds of articles on PEFT have been published, with some studies offering informative overviews of the most prevalent approaches. A comparative analysis of these surveys in terms of taxonomy and application is shown in Table 1.

Ding et al. (2022) introduce a theoretical abstraction for Delta Tuning, which is analyzed from the viewpoints of optimization and optimum control. This abstraction offers a unified approach to describe the current PEFT methods which provides a distinct perspective for future investigations. Nonetheless, while the study predominantly concentrates on NLP applications, the generalizability and efficacy of these methods in diverse domains merit

Table 1 A comparative analysis of survey methodologies: taxonomy and application domains

Survey	Taxonomy					Application			
	Add	Sel	Rep	Hybrid	Unified	NLP	Vision	Multi	Diffusion
Ding et al. (2022)	✓	✓	✓			✓			
Lialin et al. (2023)	✓	✓	✓	✓					
Xu et al. (2023a)	✓	✓	✓	✓	✓	✓		✓	
Xin et al. (2024)		✓	✓		✓		✓		
Han et al. (2024)	✓	✓	✓	✓	✓	✓			✓
Ours	✓	✓	✓	✓		✓	✓	✓	✓

Add. Additive, *Sel.*, selective, Rep. *reparameterized*, *Multi.* multi-task, *Diffusion* diffusion model

additional investigation. Lialin et al. (2023) provide a comprehensive analysis and classification that covers a broad range of methods and compares approximately 30 approaches across five dimensions: storage efficiency, memory efficiency, computational efficiency, accuracy, and inference overhead. However, while the article primarily focuses on detailed methods with practical efficiency for fine-tuning multibillion-scale language models, the exploration of real-world application scenarios is relatively limited. Xu et al. (2023a) provide a thorough evaluation and analysis of current PEFT approaches, assessing their performance, parameter efficiency, and memory utilization within a range of NLP tasks. Nonetheless, the paper does not fully expound on the practical applications of these methodologies in actual operational environments, nor does it deeply investigate their adaptability and the domain-specific challenges they might encounter. Xin et al. (2024) offer a comprehensive overview and future directions for visual PEFT, with a systematic review of the latest advancements. While the article spans multiple visual tasks, the experiments are primarily focused on several common tasks and do not fully encompass the broader range of potential application scenarios. Han et al. (2024) provide a detailed classification of PEFT approaches and explores the application of PEFT techniques across various model architectures and downstream tasks, as well as the systematic design challenges of PEFT methods. It offers researchers and engineers a comprehensive overview of PEFT approaches, but there is still room for improvement in terms of practical application coverage.

Our contributions are as follows:

- This survey comprehensively reviews the latest literature PEFT, covering cutting-edge methods and related research. It establishes a theoretical framework and offers a solid knowledge base for future research.
- We make extensive use of intuitive schematic diagrams and structured tables to elaborate on PEFT methodologies. By means of visualization, we demonstrate the complex principles of these methods, carry out comparative analyses of different approaches, and organically combine intuitiveness with systematicness, which significantly enhances the readability and academic value of the research content.
- Breaking traditional boundaries, this survey explores PEFT in natural language processing, computer vision, multimodal fusion, and diffusion models. It uncovers application potential, offers practical guidelines, and broadens the application scope of fine-tuning technology.

This survey aims to comprehensively review the recent advancements in large model fine-tuning techniques. By conducting a thorough examination of existing research, our objective is to identify and fill the gaps in our current knowledge system. This will result in the development of a comprehensive and systematic framework of knowledge, which will provide researchers with a concise perspective on the topic and guide their future research. In conclusion, our work offers valuable resources and perspectives that can be utilized for both academic and practical purposes in related domains. The remainer of this survey is structured in the following manner:

In Sect. 2, we offer a succinct summary of the fundamental components of LLMs, including their past development, emerging capabilities, and the scaling laws that govern their size. Subsequently, we offer a brief overview of the dominant classifications of comprehensive language models and introduce the fundamental principles and framework of multi-modal

comprehensive models. Furthermore, we investigate the primary methodologies employed in the fine-tuning domain of extensive language models, including instruction fine-tuning, alignment, and Reinforcement Learning from Human Feedback (RLHF). Ultimately, we present a brief summary of the most used benchmarks and assessment datasets in the field of big model fine-tuning.

In Sect. 3, we offer a comprehensive analysis and summary of PEFT approaches, presenting a cohesive framework for classifying current PEFT methodologies, encompassing over 100 research articles published from June 2019 to July 2024. Expanding on the conventional tripartite classification of additive, reparameterized, and subtractive PEFT, we incorporate summaries of hybrid, quantization, and multi-task categorization PEFT approaches.

In Sect. 4, we present a comprehensive analysis and description of the prevailing PEFT approaches in the fields of multimodal, visual, and diffusion models. Our objective is to provide a deep understanding and recommendations for choosing and improving PEFT in different application scenarios.

In Sect. 5, we encapsulate our extensive survey and put forward multiple promising avenues for future advancements, encompassing both algorithmic refinements and task scenarios, hoping to provide valuable insights for further research and development in this burgeoning field.

2 Preliminary

2.1 Large language models: foundations and variants

2.1.1 Large language models

2.1.1.1 Background LLMs refer to neural language models with a large number of parameters, typically over billions of parameters. These models are built on the transformer architecture (Vaswani et al. 2017) and are pre-trained on vast text corpora (Devlin et al. 2018). Prior to the emergence of LLMs, the advent of transformers revolutionized the development approach for neural language models, shifting from end-to-end training to a pre-train then fine-tune paradigm. Under the pre-train fine-tune paradigm, pre-trained models can be repeatedly utilized, significantly enhancing the scalability of neural language models. Consequently, the scale of parameters is continuously growing larger. For instance, OpenAI's GPT-1 possessed 120 million parameters, while GPT-2 boasted 1.5 billion parameters. This number surged to 175 billion for GPT-3 and soared to 1.76 trillion for the latest GPT-4 (Achiam et al. 2023).

2.1.1.2 Emergent abilities Research suggests that the rapid expansion of the parameter scale may lead to emergent abilities (Wei et al. 2022), which are formally defined as abilities that are not present in small models but arise in LLMs, constituting one of the most

prominent characteristics distinguishing LLM from previous PLM. In conclusion, emerging abilities can be categorized into threefolds.

In-context learning In-context learning (Wei et al. 2022; Sanh et al. 2021), known as ICL defined in GPT-3 (Brown et al. 2020), illustrates the ability of LLMs to acquire new task capabilities based on a small set of examples in context. Importantly, this process does not require additional training or gradient updates, indicating that the LLM is capable of completing new tasks with only prompts. In addition, Wei et al. (2022) reveals that ICL is associated with both the LLM and the downstream task.

Instruction following. Natural language descriptions, known as instructions, are essential for fine-tuning LLMs. Instruction tuning organizes fine-tuning datasets in the format of natural language descriptions (instructions). Research (Ouyang et al. 2022) shows that with instruction tuning, LLMs are enabled to follow task instructions for new tasks without using explicit examples, demonstrating better generalization capability across inputs of various tasks. Chung et al. (2024) discovered that to achieve evident efficacy, instruction tuning should be conducted on a relatively large-scale LLM, e.g., over 60B parameters.

Step-by-step reasoning. Constrained by parameter size, PLMs often struggle to solve tasks requiring intricate reasoning. In contrast, scaling up in parameter size equips language models with the Chain-of-Thought (CoT) (Wei et al. 2022). CoT enhances language models' performance on tasks involving logic, calculation, and decision making by structuring the input prompt to human reasoning. Thanks to CoT, LLMs are enabled to tackle tasks that demand intermediate reasoning steps to derive the final answer, akin to constructing a step-by-step prompt that invokes a thinking and inference process within the model.

Emergent abilities in LLMs have significantly boosted various real-world applications, across fields such as natural language (Kalla et al. 2023; Team et al. 2023; Liu et al. 2024b), healthcare (Wang 2023; Biswas 2023), legal (Li et al. 2024a), financial (Xing 2024) and multiple scientific disciplines (Ahn et al. 2024; Wang et al. 2023d). Despite the promising emergent capabilities, there are three main limitations that restrict the further and deeper applications of LLMs. Firstly, the inconsistency across models and tasks. LLMs trained on different architectures or datasets may demonstrate emergent behavior to varying degrees. Some models might excel in certain tasks while failing to exhibit the same level of ability in others, resulting in unpredictable performance when applied to diverse real-world scenarios (Bommasani et al. 2021). Secondly, the hallucinations and factual errors. LLMs often generate text that is fluent and coherent. However, they can also produce hallucinations, outputs that seem plausible but contain factual inaccuracies or misleading information (Bender et al. 2021; Lin et al. 2021). This tendency is particularly problematic in contexts where precise and reliable information is crucial, such as legal, medical, or scientific applications. Finally, the deficiency in deep understanding. The performance of LLMs largely stems from recognizing statistical patterns in vast datasets rather than a genuine semantic understanding of the content (Bender et al. 2021). This superficial grasp of language limits their effectiveness in tasks requiring in-depth logical reasoning and nuanced comprehension across models and tasks.

In conclusion, emergent abilities grant LLMs remarkable problem-solving capabilities, though they remain imperfect. To bridge the gap between LLMs and real-world applications, integrating traditional algorithms, expert systems, or hybrid models may be necessary to enhance reliability, accuracy, and domain-specific expertise.

2.1.1.3 Scaling laws of LLMs Thanks to the exceptional scalability of the transformer architecture (Vaswani et al. 2017), language models also exhibit high scalability. The scaling laws for LLM describe how the model grows and performs as the volume of training data increases.

In general, a scaling law includes four parameters, which also characterize a language model: (1) Parameters count *N*. The number of parameters of an LLM is often associated with the number of transformer layers and the hidden size, except for some MoE LLMs. (2) Data size *D*. In LLM, this refers to the number of tokens for training. (3) Computation cost *C*. This is typically measured in terms of time and computational resources. (4) Loss *L*. The performance of training is usually evaluated by the training loss. There are two representative scaling laws for transformer LLMs.

The Kaplan scaling law Proposed by Kaplan et al. (2020), the law examines the statistical relations between the parameters *C*, *N*, *D* and *L* over a wide range of values, models and data tokens. The relationships can be expressed through the following equations:

$$L(N) = \left(\frac{N_c}{N}\right)^{\alpha_N}, \alpha_N \sim 0.076, N_c \sim 8.8 \times 10^{13} \tag{1}$$

$$L(D) = \left(\frac{D_c}{D}\right)^{\alpha_D}, \alpha_D \sim 0.095, D_c \sim 5.4 \times 10^{13} \tag{2}$$

$$L(C) = \left(\frac{C_c}{C}\right)^{\alpha_C}, \alpha_C \sim 0.050, N_c \sim 3.1 \times 10^{8}, \tag{3}$$

where the loss *L* is influenced by parameters *N*, *D*, and *C*, shedding light on decision-making processes when computational resources are limited.

The Chinchilla scaling law Proposed by DeepMind (Hoffmann et al. 2022), the law provides guidelines for compute-optimal training of LLMs, specifically when computational resources are limited. Through rigorous experiments spanning a wide range of model sizes from 70 M to 16B and dataset sizes from 5B to 500B tokens, they derived a scaling law with different coefficients compared to Kaplan's, as shown below:

$$L(N, D) = E + \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}}, \tag{4}$$

where E denotes the loss of an ideal generative process on the test data. Furthermore, claimed by the research, the constants in this formula are $\alpha = 0.34, \beta = 0.28, A = 406.4, B = 410.7, L_0 = 1.69$. Moreover, there is a general constraint that model the relationship between *C* and (*N*, *D*): $C = 6ND$, which means that it costs six FLOPs per parameter to train one token. Thus, the optimal selection of model size and data size can be determined and expressed as:

$$N_{opt} = 0.6\, C^{0.45} \tag{5}$$

$$D_{opt} = 0.3\, C^{0.55} \tag{6}$$

$$L_{opt} = 1070\, C^{-0.154} + 1.7. \tag{7}$$

From the equations, scaling laws can guide decisions regarding model size. Given a fixed compute budget (e.g., 100K GPU hours), they enable predictions on whether a smaller model trained for a longer duration or a larger model trained for a shorter time would yield better performance. Additionally, scaling laws provide insight into the benefits of continued training. The diminishing returns they imply suggest that beyond a certain point, increasing compute resources may not lead to a substantial enough performance gain to justify the additional cost.

In addition, based on the statistical modeling illustrated by Eq. 4, one approximate estimation for Chinchilla efficient model size and training dataset size can be denoted as:

$$N_{opt} = 0.1\, C^{0.5} \tag{8}$$

$$D_{opt} = 1.7\, C^{0.5}. \tag{9}$$

This suggests that the model size and training data volume should be scaled in accordance with the available computational budget. The expected ratio of training tokens to model parameters is approximately 17:1. However, in real-world applications, this ratio is often slightly higher, as additional training data beyond the 17× scaling rule can still contribute to performance improvements when sufficient computational resources are available. For instance, GPT-2 was trained on 40B tokens with 1.5B parameters, LLaMA was trained on 1.4T tokens with 65B parameters, and DeepSeek-V3 was trained on 14.8T tokens with 0.671T parameters. While all these ratios exceed 17, they remain close to this scaling guideline.

PEFT and Sustainability of AI Research Training large models from scratch is highly energy-intensive. For example, training LLaMA-3.1 405B can demand 40 million GPU hours on H100, resulting in a substantial carbon footprint. While fully Supervised Fine-Tuning (SFT) can enhance an existing LLM using a relatively smaller set of training samples, it still requires updating the entire parameter network. In contrast, Parameter-Efficient Fine-Tuning (PEFT) methods-such as adapters or low-rank adaptations-enable fine-tuning a large pre-trained model for specific tasks by updating only a small subset of parameters (typically just 1–2% of the total). As a result, PEFT significantly reduces computational costs; for instance, a full SFT process that requires 4 million GPU hours can be reduced to 400K GPU hours or less with PEFT.

By lowering GPU usage, PEFT not only decreases energy consumption but also mitigates the environmental impact. Moreover, this reduction in compute requirements is crucial for sustainable AI research, as PEFT provides a cost-effective and efficient approach for the AI community and researchers to conduct experiments and develop new models.

2.1.2 Prevalent LLMs

2.1.2.1 The GPT family Generative Pre-trained Transformers (GPT) constitute a series of decoder-only Transformer-based language models, pioneered by OpenAI. This family

encompasses GPT-1 (Radford et al. 2018), GPT-2 (Radford et al. 2019), GPT-3, InstructGPT (Ouyang et al. 2022), ChatGPT, GPT-4, GPT-4o, CODEX (Chen et al. 2021), and WebGPT (Nakano et al. 2021). GPT-1 and GPT-2 belong to PLMs, while following GPT-3, all subsequent models in this family are classified as LLMs.

GPT-3 (Brown et al. 2020) is widely recognized as the first LLM due to its significantly larger size compared to previous PLMs, showcasing emergent abilities not observed in smaller PLMs before. A key emergent ability demonstrated by GPT-3 is in-context learning (Kojima et al. 2022), enabling the model to solve various downstream tasks without the need for fine-tuning. Distinct with other GPT-family LLMs, GPT-4 and GPT-4o are both multi-modal LLMs. GPT-4 (Achiam et al. 2023) is one of the most powerful LLM reported to train on a transformer network of 1.8 trillion parameters which exhibits great capabilities in image understanding and reasoning. GPT-4o, while inheriting the powerful intelligence of GPT-4, has further enhanced its capabilities in text, image, and speech processing. Compared to existing models, it particularly excels in visual and audio comprehension.

2.1.2.2 The LLaMA family LLaMA stands as a series of open-source LLMs developed by Meta. To date, the official release includes: LLaMA, LLaMA-2, and LLaMA-3.x, spanning parameter scales from 1 billion to 405 billion. Beyond the weights provided by Meta, the qualities of these LLMs are further extended through supervised fine-tuning and parameter-efficient fine-tuning.

LLaMA-1 (Touvron et al. 2023) was released in February 2023. Although LLaMA is open-sourced and possesses fewer parameters, LLaMA-13B demonstrates significant improvements over GPT-3 (175 billion parameters) across various benchmarks. As a consequence, LLaMA has emerged as a widely adopted and exemplary base model for LLM research. LLaMA-2 (Touvron et al. 2023) was developed in partnership with Microsoft and released half a year later. The model maintains the same architecture as the LLaMA-1 but is trained with 40% more data. LLaMA-3 was released by Meta in April 2024, offering two parameter sizes: 8B and 70B. These models underwent pre-training on approximately 15 trillion tokens of text sourced from publicly available data and are fine-tuned over 10 million human-annotated examples. Subsequently, Meta released LLaMA-3.1 (Vavekanand and Sam 2024), a 405B open-sourced LLM, which focuses on improving text generation capabilities and achieves performance comparable to leading models like GPT-4. Then, in September 2024, LLaMA-3.2 was released, introducing both vision models (11B and 90B) and lightweight text-only models (1B and 3B) for mobile device use. LLaMA-3.2 marked Meta's first open-source AI model capable of processing both images and text, broadening the scope of potential applications. The smaller models were designed for efficient performance on mobile devices, promoting wider adoption in edge computing scenarios.

2.1.2.3 The OpenAI o1 family In September 2024, a new series of LLM, OpenAI-o1[1] (Jaech et al. 2024), excels in complex reasoning tasks, using Chain-of-Thought (CoT) reasoning to outperform GPT-4o in areas like math, coding, and science. The release includes two versions: o1-preview and o1-mini. The o1-preview is an early iteration of the full model, while the o1-mini is a lightweight version optimized for size and speed. When solving prob-

[1] https://openai.com/index/introducing-openai-o1-preview/.

lems, o1 uses the CoT[2] strategy like human deep thinking. Reinforcement learning helps o1 refine its thinking and strategies, find and correct errors, break down complex steps, and change approaches when necessary, improving reasoning. The reward model combines text and number scores for evaluation.

Then previewed in December 2024, OpenAI o3-mini,[3] the newest, most cost-efficient model was offically released in January 2025, which provides a specialized alternative for technical domains requiring precision and speedwhich. It delivers exceptional STEM capabilities-with particular strength in science, math, and coding-all while maintaining the low cost and reduced latency of OpenAI o1-mini.

2.1.2.4 The DeepSeek family DeepSeek-LLM is a newly established LLM series that has garnered significant attention from both academia and industry. Developed by the company DeepSeek, the first version, DeepSeek-V1 (Bi et al. 2024), was trained on 2 trillion tokens and released in January 2024, featuring two core models: 7B and 67B, along with their respective chat variants. In the same month, DeepSeek introduced DeepSeek-MoE (Mixture of Experts) (Dai et al. 2024a) 16B, which delivers performance comparable to LLaMA 2 7B while requiring only 40% of the computational cost. This model introduces an innovative Mixture of Experts (MoE) architecture, integrating shared expert isolation with fine-grained expert segmentation. Additionally, it incorporates a novel load-balancing strategy that optimizes both expert and device balance, enhancing computational efficiency. They made significant progress with DeepSeek-V2 (Liu et al. 2024a), a large MoE-LLM trained on 8.1 trillion tokens, featuring 2 shared experts, 160 routed experts, and 236 billion parameters. This version introduced Multi-head Latent Attention (MLA), which significantly reduces GPU memory consumption while maintaining the same level of precision. It outperforms the widely used Grouped-Query Attention (GQA) strategy adopted by LLaMA 3. Subsequently, they released DeepSeek-V3 (Liu et al. 2024b) in December 2024. Building upon V2, the V3 model introduces its Multi-Token Prediction (MTP) approach and an Auxiliary-Free Load Balancing strategy to further enhance efficiency. Additionally, it integrates DualPipe (Li and Hoefler 2021), cross-node all-to-all communication techniques, and a minimal-overhead memory-saving strategy, achieving a groundbreaking industrial milestone-training a 671B-parameter MoE-LLM with FP8 precision. The performance of the DeepSeek-V3 model is remarkable, achieving state-of-the-art (SOTA) results among all open-source LLMs and demonstrating performance comparable to GPT-4o and Claude 3.5 Sonnet. Moreover, it offers significant advantages in training and inference costs, requiring less than 10% of the training cost of LLaMA 3-405B and only 9% of the inference cost of Claude 3.5 Sonnet, revolutionizing the development of industrial LLMs. Then, DeepSeek released R1 (Guo et al. 2025), a reinforcement learning-focused model leveraging the Group Relative Policy Optimization (GRPO) (Shao et al. 2024) algorithm. R1 delivers performance comparable to OpenAI-o1 in mathematical and logical reasoning tasks, while

[2] https://openai.com/index/learning-to-reason-with-llms/.

[3] https://openai.com/index/openai-o3-mini/.

requiring only 2% of the computational cost, marking a major breakthrough in efficiency and scalability.

2.1.2.5 The Claude family Anthropic (2025) represents a series of conversational AI models developed by Anthropic, designed with a focus on safety, helpfulness, and natural language understanding. This family includes Claude 1, Claude 2, Claude 2.1, Claude 3 Opus, Claude 3 Sonnet, and Claude 3 Haiku.

Claude 1 marked the initial release of Anthropic's conversational AI, introducing the concept of Constitutional AI to the field. Claude 2 and its subsequent update, Claude 2.1, brought significant improvements in language understanding, context retention, and response coherence. These versions demonstrated enhanced capabilities in handling complex queries and maintaining longer, more contextually rich conversations.

Claude 3 models (Opus, Sonnet, and Haiku) represent the latest advancements in the Claude family, each tailored for distinct applications. Opus, the most advanced model, integrates cutting-edge multimodal capabilities, enabling it to process both textual and visual inputs with deep reasoning and high-level comprehension, excelling in complex problem-solving tasks. Sonnet, optimized for efficiency and speed, is ideal for scenarios requiring rapid, precise, and contextually appropriate replies. Haiku prioritizes simplicity and elegance, delivering concise, poetic, and highly relevant responses, making it particularly well-suited for creative and literary applications. Together, these models set new benchmarks for AI-driven interaction and analytical reasoning.

Each model in the Claude family is continuously refined to improve performance, safety, and alignment with user needs, ensuring that they remain at the forefront of conversational AI technology.

2.1.2.6 The Gemini family Gemini (Anil et al. 2023) constitutes a series of multimodal Transformer-based language models, developed by Google DeepMind. This family includes Gemini 1, Gemini 1.5, and Gemini 2, each introducing significant advancements in multimodal understanding, long-context reasoning, and integration with Google's ecosystem. Unlike GPT family models, which initially focused on text generation, Gemini models were designed from the ground up to be native multimodal models, enabling seamless processing of text, images, audio, and video. Gemini 1 marked Google's transition from its Bard chatbot to a more advanced multimodal LLM, introducing cross-modal reasoning and excelling in mathematical problem-solving, coding, and knowledge retrieval, though it faced limitations in real-world usability. Gemini 1.5 introduced a 1 million-token context window, significantly improving long-document processing, dialogue coherence, and complex multi-step reasoning. Additionally, it implemented memory capabilities, allowing it to retain user-specific context across interactions. The latest version, Gemini 2 further enhanced reasoning, tool integration, and inference speed, introducing a "Flash Thinking" mode that enables

intermediate reasoning steps for improved transparency. It also deepened integration with Google Search, Docs, and other productivity tools, optimizing it for real-world applications.

2.1.2.7 Other representative LLMs Mistral Series (Jiang et al. 2023) is an open-sourced LLM developed by Mistral AI. The basic Mistral-7B demonstrates superior performance across all evaluated benchmarks, surpassing all open-sourced 13B LLMs and even outperforming LLaMA-34B in reasoning, mathematics, and code generation tasks. Mistral 7B employs Grouped Query Attention (GQA) to enable faster inference and Sliding Window Attention (SWA) to handle longer text sequences efficiently. Subsequently, Mistral AI introduced two additional models: Mixtral 8×7B and Mixtral 8×22B. These models utilize the Sparse Mixture of Experts (SMoE) technique (Riquelme et al. 2021), which selectively activates a subset of experts for each input, thereby significantly reducing computational load.

The PaLM (Chowdhery et al. 2023) (Pathwaysutilized Language Models) is developed by Google as a collection of decoder-only LLMs. The first PaLM model was trained on a high-quality text corpus of 780 billion tokens, boasting a remarkable 540 billion parameters. Unlike prevalent LLMs which primarily utilize GPUs for training, PaLM is pre-trained with the Pathways system on 6144 TPU v4 chips to facilitate rapid and efficient training. In the following days, U-PaLM (Tay et al. 2022), FlAN-PaLM (Chung et al. 2024) and PaLM-2 were released.

2.1.3 Multimodal large language models

2.1.3.1 MLLM: background Multimodal large language model (MLLM), is an extension of LLM which adopts multimodal information as input such as text, sound, video, etc. to enable multiple dimensional reasoning and text generation.

Before the emergence of MLLM, significant research efforts were dedicated to multimodality. These efforts can generally be categorized into representative and generative paradigms. An exemplary work in the representative paradigm is CLIP (Radford et al. 2021), which serves as a foundational contribution.

This process yields a visual encoder (Cherti et al. 2023; Sun et al. 2023) and a text encoder, effectively establishing a bridge for downstream multimodal tasks. In contrast, generative frameworks (Wang et al. 2022a; Cho et al. 2021) approach multimodal tasks by transforming them into sequence-to-sequence tasks. MLLM distinguishes itself from previous multimodal research in two key aspects. (1) Composition: MLLM is comprised of at least one LLM with billion-scale parameters. (2) Training techniques: MLLM introduces and incorporates novel training techniques derived from LLM to enhance multimodal performance.

2.1.3.2 MLLM: architecture Figure 1 illustrates the mainstream architecture of MLLMs, typically composed of three modules: a multimodal encoder, an LLM, and a modal connector.

Multimodal Encoder. This module incorporates non-text inputs, such as images or audio, and encoding the raw information into a more compact representation. It is notewor-

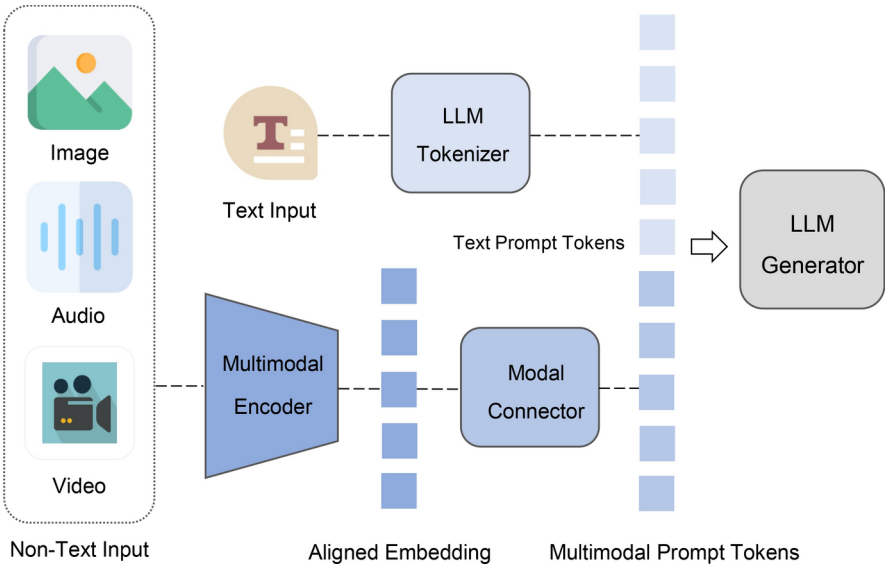


Fig. 1 Architecture of MLLM: this figure shows a common architecture and workflow of an MLLM

thy that the encoder is aligned with one or several encoders in advance to ensure associated meanings are preserved. It is more advisable to directly adopt and fine-tune a pre-trained multimodal encoder, such as CLIP (Radford et al. 2021), EVA-CLIP (Sun et al. 2023), or ViT-G (Zhai et al. 2022), rather than starting from scratch to train a new encoder for generalized data.

LLM. It is also more efficient to adopt a pre-trained LLM instead of training from the start. Through tremendous pre-training on web corpus, LLMs have been embedded with rich world knowledge, and demonstrate strong generalization and reasoning capabilities.

Modal Connector. This module serves as a crucial bridge between different modalities, allowing efficient communication with the LLM. It accomplishes this by projecting information into a space that the LLM can readily comprehend. Through training the connector, the encoded multimodal tokens can be transformed to LLM prompt tokens that illustrate the content presented by the image, video, etc. Consequently, the LLM will generate the expected content based on the request and prompt.

2.2 Optimization, datasets, and evaluation of large language models

2.2.1 Instruction tuning

Instruction tuning in LLMs has undergone significant development, evolving from initial efforts in multi-task fine-tuning without explicit instruction prompts to sophisticated techniques leveraging diverse tasks and templates. Early work focused on improving downstream task performance through large-scale multi-task fine-tuning (Raffel et al. 2020; Liu et al. 2019; Aghajanyan et al. 2021; Aribandi et al. 2021), while other efforts (Khashabi et al. 2020; McCann et al. 2018; Keskar et al. 2019) converted a range of NLP tasks into a single generative question answering format using prompt instructions. The instruction tun-

ing began in 2020 with the release of several task collections, including Natural Instructions (Mishra et al. 2021), Flan 2021 (Wei et al. 2021), and PromptSource (Bach et al. 2022). These collections aggregated large NLP datasets and provided templatized instructions for zero-shot prompting, enabling models to generalize to unseen instructions. MetaICL (Min et al. 2021) emphasized few-shot prompting without explicit instructions, using input–output examples to teach tasks in-context. Research confirmed the benefits of task and template diversity, with some studies highlighting the advantages of inverting inputs and outputs to create new tasks (Min et al. 2021). The subsequent phase saw the expansion and combination of resources, with collections like SuperNatural Instructions (Wang et al. 2022c) and OPT-IML (Iyer et al. 2022) integrating more datasets and tasks. This phase also introduced multilingual instruction tuning, as seen in xP3 (Muennighoff et al. 2022), and incorporated Chain-of-Thought training prompts in Flan 2022 (Chung et al. 2022). These expanded collections included most tasks from previous resources, establishing a strong foundation for future open-source work. Current and future research is exploring new directions, such as synthetic data generation for creative and open-ended dialogue tasks (Wang et al. 2022b; Honovich et al. 2022; Ye et al. 2022; Gupta et al. 2022) and integrating human feedback on model responses (Ouyang et al. 2022; Glaese et al. 2022; Nakano et al. 2021; Bai et al. 2022b). These approaches are viewed as complementary to foundational instruction tuning methods, driving further advancements in the field.

A recent advance in instruction tuning is the potential to complement or replace few-shot in-context learning with PEFT. Compared to instruction tuning, PEFT can achieve performance comparable to full parameter tuning while being computationally more cost-effective. Previous studies (Liu et al. 2022c; Wei et al. 2021; Vu et al. 2021; Singhal et al. 2023) have demonstrated that PEFT can be effectively integrated with instruction tuning, either before or after the instruction tuning process. Additionally, this body of research highlights that PEFT can enhance the performance and applicability of instruction tuning across different domains.

2.2.2 Alignment tuning and RLHF

Despite the emergent abilities brought by increasing parameters of language models, hallucination exhibit to become a challenge for LLMs to produce satisfying response. To address this issue, alignment tuning is applied to align the models with specific human preferences. There are three primary targets for alignment tuning, respectively presented as helpfulness, honesty and harmlessness. From the targets' names, it can be concluded that the alignment criteria are closely associated with human's recognition, making it difficult to formulate them as optimization objectives for LLMs. Therefore, human feedback is widely adopted as an assistance to reinforce LLMs' performance.

RLHF (Knox and Stone 2008; Christiano et al. 2017) emerged as a method to fine-tune language models using human feedback, aiming to align the LLMs with human preferences, and consequently enhancing alignment performance.

Generally, an RLHF system (Ouyang et al. 2022) comprises three key components: a pre-trained language model, a reward model learned from human feedback, and a reinforcement learning algorithm to train the language model. Figure 2 shows the three key steps.

- **Supervised Fine-Tuning (SFT):** Initially, a supervised dataset consisting of input

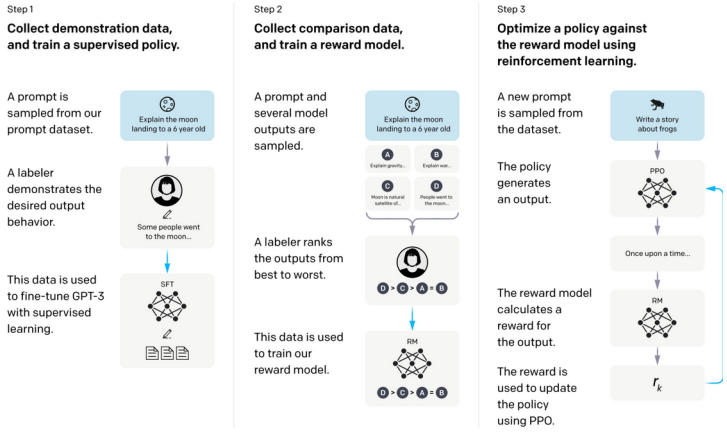


Fig. 2 RLHF workflow: this figure is from InstructGPT, which interprets the RL process

prompts and desired outputs is applied to fine-tune the language model. These prompts and outputs can be written by human labelers for some specific tasks while ensuring the diversity of tasks. This step helps the model learn expected behaviors.

- **Reward Model Training:** A reward model is trained using human feedback data. The LLM is employed to generate a certain number of output texts using sampled prompts as input. Then human labelers rank these output pairs based on their preferences. Given human predictions, the reward model is trained to predict these rankings, effectively learning human preferences. Notably, Lee et al. (2023) proposes an approach, namely Reinforcement Learning from AI Feedback (RLAIF), the annotation of preference on response pairs can be generated by an AI agent, increasing the automatic ability of the reinforcement process.
- **Reinforcement Learning Fine-Tuning:** The final step involves formalizing the alignment process as a reinforcement learning problem. Here, the pre-trained language model acts as a policy generating text, with the reward model providing feedback scores. To prevent the model from deviating too far from its initial state, a penalty term is often included in the reward function. The language model is then optimized using algorithms like SARSA (Sutton 1995), DQN (Fan et al. 2020), PPO (Schulman et al. 2017), DPO (Rafailov et al. 2024), and GRPO (Shao et al. 2024), iteratively improving its performance based on human-aligned rewards.

2.2.3 Datasets for LLM

A critical component of the development and deployment of LLM is the datasets used at various stages of their lifecycle, which significantly influence their capabilities and performance. In this section, we delve into the datasets that are instrumental in the Pre-training, SFT, and RLHF. The Pre-training phase is where an LLM absorbs the foundational knowledge from a diverse array of textual data. This stage is pivotal, as it sets the stage for the model's general understanding of language. The datasets used in Pre-training are vast and varied, encompassing everything from the sprawling expanse of the internet to curated collections of literature and encyclopedias. SFT is the process where the LLM is fine-tuned

on specific tasks or domains. This phase refines the model's abilities, enabling it to perform with greater precision and relevance in targeted applications. SFT datasets are often more specialized and may include annotated examples that guide the model towards desired behaviors and outputs. RLHF is the stage where the LLM is further optimized based on human feedback. This phase enhances the model's alignment with human preferences and values, ensuring that its outputs are more aligned with user expectations. RLHF datasets typically consist of human-labeled examples and feedback, which help the model learn to prioritize high-quality and contextually appropriate responses.

2.2.3.1 Commonly used datasets for pre-training In the realm of LLM, the pre-training phase is instrumental in establishing a robust foundation upon which the model's linguistic prowess is built. LLM, with their exponentially larger parameter counts, necessitate an extensive and diverse corpus of training data that spans a multitude of topics and linguistic expressions. This data not only serves as the bedrock for the model's comprehension of language but also influences its ability to generalize and adapt to new contexts and tasks. To meet these requirements, a variety of comprehensive and accessible datasets have been curated and made available for the research community.

In this section, we embark on an overview of the datasets that are pivotal in the pre-training of LLM. We categorize these datasets based on the type of content they provide, which can be broadly divided into seven distinct groups: Webpages, Books, Code, Social Media, Wikipedia, and a diverse array of other sources. Each of these categories contributes unique elements to the model's knowledge base, ensuring a well-rounded understanding of human language and its myriad uses. Here are 2 typical Pre-training Datasets and their importance in evaluating PEFT Methods:

- **Common Crawl:** The Common Crawl corpus is an extensive, unstructured, multilingual dataset of webpages, encompassing over eight years of web crawler data. This dataset is available in various formats, including web archive, web archive transformation, and web-extracted text. Many pre-training corpora are obtained through data preprocessing based on this corpus, which provides a vast and diverse source of text for language models. Its unstructured nature and multilingual content make it an ideal resource for training models that need to handle a wide variety of text types and languages. Importantly, the Common Crawl corpus plays a crucial role in evaluating PEFT methods. Its vast and varied content provides a comprehensive base for pre-training models that can then be fine-tuned using PEFT techniques. This allows researchers to assess how effectively PEFT methods can enhance model performance across diverse linguistic contexts.
- **The Pile:** The Pile is a large-scale, diverse language modeling dataset consisting of 22 data subsets, designed to capture text in as many forms as possible and cover a wide range of textual content. The corpus includes academic papers, code, legal materials, patents, subtitles, chat content, parallel corpora, and more. This diversity ensures that models trained on The Pile are exposed to a broad spectrum of language use cases, making them more adaptable to various downstream tasks. In the context of evaluating PEFT methods, The Pile offers a robust testbed. Its rich diversity of text types allows researchers to evaluate how well these fine-tuning methods can adapt models to different

domains and tasks, thereby enhancing their understanding of the effectiveness of PEFT methods in various applications (Table 2).

2.2.3.2 Commonly used datasets for SFT and RLHF Two critical stages in LLM are SFT and RLHF. These stages are designed to enhance the model's performance on specific tasks and align its outputs with human preferences. This section provides an overview of these two stages, highlighting their significance and the datasets used to support them.

SFT is a process where LLM are trained on specialized datasets to improve their performance on specific tasks. This stage is crucial for adapting the model to particular domains or applications. SFT involves using annotated datasets that provide examples of desired outputs for given inputs. By training on these datasets, the model learns to generate more accurate and contextually relevant responses. RLHF is particularly effective in enhancing the model's ability to follow human instructions. These datasets provide a comprehensive set of examples that help the model learn to discern correct answers from plausible alternatives (Table 3).

2.2.4 LLM evaluation

The burgeoning field of LLM research has necessitated the development of robust evaluation frameworks to accurately gauge the capabilities and limitations of these sophisticated AI systems. Evaluation serves multiple critical functions: it benchmarks model performance across a spectrum of tasks, identifies areas for improvement, and ensures that advancements in LLM technology align with ethical and practical standards. In the academic and professional realms of LLM evaluation, it is widely recognized that a multifaceted approach is essential to gauge the capabilities and limitations of these advanced AI systems comprehensively. The Qwen blog's evaluation of the Qwen2.5 base language model,[4] underscore the importance of using multiple benchmarks to assess the model's performance across various domains thoroughly.

Platforms such as Hugging Face offer a suite of datasets for this purpose,[5] including IFEval, BBH, MATH (Hendrycks et al. 2021b), GPQA (Rein et al. 2024), and MUSR (Sprague et al. 2023). These datasets encompass a broad spectrum of tasks, ranging from language modeling to problem-solving in mathematics, ensuring a comprehensive evaluation of model competencies. Models like Qwen2.5 is evaluated using a diverse array of datasets that cover general tasks such as MMLU, and HellaSwag, as well as specialized tasks in math and science with datasets like GPQA and MATH, and coding tasks including HumanEval and MBPP. Additionally, multilingual capabilities are assessed through datasets like Multi-Exam and Multi-Translation.

To achieve a comprehensive evaluation of a language model's performance, it is often necessary to employ a combination of benchmarks. These benchmarks should be representative of real-world scenarios and cover diverse domains and linguistic complexities. The evaluations include a variety of tests that measure the model's ability to handle extended dialogues and manage a variety of tasks. By leveraging these diverse datasets and assess-

[4] https://qwenlm.github.io/blog/qwen2.5-llm/.

[5] https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.

Table 2 A curated list of datasets for pre-training

Collections	Categories	Publication time	Size	URL
Common Crawl[a]	Webpages	2023	400 TB	https://commoncrawl.org/
WuDaoCorpora-Text (Yuan et al. 2021)	Webpages	2023	5 TB	https://data.baai.ac.cn/details/WuDaoCorporaText
BookCorpusOpen (Zhu et al. 2015)	Books	2015	9.05 GB	https://huggingface.co/datasets/defunct-datasets/bookcorpusopen
PG-19 (Rae et al. 2020)	Books	2020	11.74 GB	https://huggingface.co/datasets/deepmind/pg19
The Stack (Kocetkov et al. 2023)	Code	2022	3 TB	https://huggingface.co/datasets/bigcode/the-stack
OpenWebText (Gokaslan and Cohen 2019)	Social Media	2019	38 GB	https://huggingface.co/datasets/Skylion007/openwebtext
Pushshift Reddit (Baumgartner et al. 2020)	Social Media	2020	89.1 GB	https://zenodo.org/records/3608135
Wikipedia[b]	Wikipedia	2023	71.8 GB	https://huggingface.co/datasets/wikimedia/wikipedia
The Pile (Gao et al. 2020)	Others	2020	800 GB	https://pile.eleuther.ai/
S2ORC (Lo et al. 2020)	Others	2020	80.5 GB	https://huggingface.co/datasets/sentence-transformers/s2orc
MultiUN (Eisele and Chen 2010)	Others	2010	31.8 GB	https://huggingface.co/datasets/Helsinki-NLP/multiun

This table provides a comprehensive overview of various datasets used for pre-training purposes in natural language processing tasks. It includes details such as the collection name, the corpus it belongs to, publication year, size in terms of tokens, and the URL for accessing the dataset. The datasets listed cover a range of sources from web pages to books, offering a diverse set of data for training models in different domains

[a]https://commoncrawl.org/

[b]https://www.wikipedia.org/

ments, researchers can effectively benchmark LLM and guide their development towards practical applications, ensuring alignment with ethical and practical standards. This section explores benchmarking in two parts: general tasks and specialized tasks.

- **General Tasks:** General tasks are designed to assess the broad capabilities of LLM across a wide range of subjects and skills. These benchmarks are essential for evaluating the foundational knowledge and general reasoning abilities of LLM. These benchmarks help determine how well models can understand and generate text in various contexts, ensuring that they possess a solid understanding of language fundamentals. Datasets such as MMLU, ARC, and HellaSwag are commonly used for general evaluations.**Specialized Tasks:** Specialized tasks focus on evaluating LLM in specific domains, such as mathematics, coding, and natural language understanding. These benchmarks are designed to assess the model's proficiency in particular areas, providing a deeper understanding of their specialized skills. Specialized tasks are crucial for identifying domain-specific strengths and weaknesses, ensuring that models can effectively apply their knowledge in practical scenarios (Table 4).

3 PEFT taxonomy

PEFT techniques are typically divided into three primary categories: **Additive PEFT** (Sect. 3.1), which introduces additional trainable components or parameters into the pre-existing model; **Reparameterized PEFT** (Sect. 3.2), a method that restructures the model's parameters during the training phase and then reverts to the original form for inference; and **Selective PEFT** (Sect. 3.3), which focuses on optimizing a specific subset of the model's parameters. Besides these, there is the **Hybrid PEFT** (Sect. 3.4), which combines the strengths of various PEFT approaches. Additionally, there are specialized adaptations such as **Quantization PEFT** (Sect. 3.5) designed for the quantization process, and **Multi-task PEFT** (Sect. 3.6) aimed at enhancing multi-task learning capabilities. A conceptual illustration of the core principles underlying these PEFT methodologies is presented in Fig. 3. A comprehensive classification of PEFT methods is depicted in Fig. 4. The main ideas, number of trainable parameters, applications, and limitations of different types of PEFT methods are summarized in Table 5. To facilitate a more intuitive understanding of the performance differences among various PEFT methods, Table 6 presents the performance results of representative PEFT methods of different types across various base models and tasks.

3.1 Additive PEFT

Full-parameter fine-tuning is computationally expensive and could adversely affect the model's capacity to generalize. To address this, additive PEFT methods add a small set of trainable parameters to a pre-trained model, carefully integrated into its architecture. When fine-tuning for particular downstream tasks, it is only these extra components or parameters are adjusted, keeping the original pre-trained model parameters unchanged. This approach significantly reduces the need for storage, memory, and computation. Based on where and how these additional trainable parameters are incorporated into the model's architecture,

Table 3 A curated List of datasets for SFT and RLHF

Collections	Categories	Publication time	Examples	URL
E2E NLG (Dušek et al. 2020)	NLP Task	2020	50,000	https://sites.google.com/site/hwinteractionlab/E2E/
WikiSQL (Zhong et al. 2017)	NLP Task	2017	80,654	https://huggingface.co/datasets/Salesforce/wikisql
WebNLG (Gardent et al. 2017)	NLP Task	2017	27,731	https://huggingface.co/datasets/web_nlg
SAMSum (Gliwa et al. 2019)	Daily Chat	2019	16,369	https://huggingface.co/datasets/Samsung/samsum
OASST1 (Wang et al. 2024a)	Daily Chat	2023	161,443	https://huggingface.co/datasets/OpenAssistant/oasst1
WMT[a]	Others	2019	124,448,248	https://huggingface.co/datasets/wmt/wmt19
XSUM (Narayan et al. 2018)	Others	2018	200,000	https://huggingface.co/datasets/EdinburghNLP/xsum
DART (Nan et al. 2021)	Text generation	2021	82,000	https://github.com/Yale-LILY/dart
HH-rlhf (Bai et al. 2022a)	Dialogue and preference	2022	169,000	https://huggingface.co/datasets/Anthropic/hh-rlhf
PKU-SafeRLHF (Ji et al. 2023)	Dialogue and preference	2023	362,000	https://huggingface.co/datasets/PKU-Alignment/PKU-SafeRLHF
HotpotQA (Yang et al. 2018)	Question–Answering	2018	113,000	https://huggingface.co/datasets/hotpotqa/hotpot_qa
SHP (Ethayarajh et al. 2022)	Community preference	2022	385,000	https://huggingface.co/datasets/stanfordnlp/SHP

This table provides an overview of the datasets used in SFT and RLHF phases, categorized by their primary purposes and characteristics. This categorization helps in understanding the diversity and scope of data used to train and fine-tune models in different phases of development. The URLs provided allow researchers and practitioners to access these datasets for further analysis and experimentation

[a]https://www.statmt.org/wmt19/

there are primarily three types of additive PEFT techniques: *Adapter*, *Soft Prompt*, and *Scale and Shift*. We will delve into some of the principal studies on these techniques.

3.1.1 Adapter

Adapter methods enable PEFT by inserting small adapter layers into pre-trained models, which learn task-specific transformations while keeping the base model frozen. These adapters, typically consisting of a down-projection, a non-linear activation function and an up-projection layer (the standard adapter shown in Fig. 5a), adapt the representations to downstream tasks with minimal overhead. For example, in **Sequential Adapter** (Houlsby et al. 2019), two serial adapters are inserted after the attention layer and the feed-forward layer in transformer blocks. **Residual Adapter** (Lin et al. 2020) dynamically adapts a pre-trained language model, such as GPT-2, to various downstream tasks using low-rank residual adapters and task embeddings, with the adapter module formulated as:

$$\text{Adapter}(H_i) = (\text{ReLU}(\text{LN}(H_i)W_i^E))W_i^D + H_i, \tag{10}$$

where H_i is the hidden representation of the ith layer, W_i^E and W_i^D are the adapter parameters, and LN denotes layer normalization. **AdapterDrop** (Rücklé et al. 2020) dynamically removes adapters from the lower layers of a transformer during training and inference, which significantly enhances inference speed in multi-task settings with minimal impact on task performance. **Tiny-Attn Adapter** (Zhao et al. 2022) applies a multi-head attention mechanism with tiny per-head dimension the intermediate embeddings of each token to obtain the modified embeddings, and employs parameter-averaging technique to reduce inference cost during deployment. **Parallel Adapter** (He et al. 2021) integrates the adapter network to both the attention and feed-forward layers of the transformer in a parallel manner, facilitating a more efficient incorporation of the module. **CIAT** (Counter-Interference Adapter for Multilingual Machine Translation) (Zhu et al. 2021) employs an embedding adapter to refine multilingual word embeddings and parallel layer adapters to de-noise the multilingual interference in intermediate layers, improving the translation performance with a small parameter overhead. **CoDA** (Condition Adapter) (Lei et al. 2024) enhances inference efficiency by selectively activating computations on a subset of input tokens, determined by a soft top-k operation, thus balancing model expressivity and computational efficiency. **Hadamard Adapter** (Chen et al. 2023e) (shown in Fig. 5b) employs a weight vector and a bias vector, applying the Hadamard product (element-wise multiplication) and element-wise addition to the self-attention outputs, resulting in new self-attention outputs. **Compacter** (Karimi Mahabadi et al. 2021) incorporates concepts from adapters, low-rank methods, and hypercomplex multiplication layers. It introduces task-specific weight matrices by combining shared "slow" weights with "fast" rank-one matrices computed through Kronecker products, tailored to each COMPACTER layer's requirements. **Sparse-Adapter** (He et al. 2022) prunes a significant portion of parameters at initialization, using a sparsity-inducing method to maintain performance while reducing computational overhead, and further improving capacity through a "Large-Sparse" configuration that scales up the bottleneck dimension with an increased sparsity ratio.

Table 4 A typical list of available datasets for LLM evaluation

Collections	Task	Publication time	Examples	URL
MMLU (Hendrycks et al. 2021a)	General	2021	15,908	https://huggingface.co/datasets/cais/mmlu
ARC (Clark et al. 2018)	General	2018	7787	https://huggingface.co/datasets/allenai/ai2_arc
HellaSwag (Zellers et al. 2019)	General	2019	59,950	https://huggingface.co/datasets/Rowan/hellaswag
GLUE (Wang et al. 2019b)	Natural language understanding	2018	1,485,043	https://huggingface.co/datasets/nyu-mll/glue
SuperGLUE (Wang et al. 2019a)	Natural language understanding	2019	196,309	https://huggingface.co/datasets/aps/super_glue
GSM8K (Cobbe et al. 2021)	Science and mathematics	2021	17,584	https://huggingface.co/datasets/openai/gsm8k
Theoremqa (Chen et al. 2023d)	Science and mathematics	2023	800	https://huggingface.co/datasets/TIGER-Lab/TheoremQA
Humaneval (Chen et al. 2021)	Code	2021	164	https://huggingface.co/datasets/openai/openai_humaneval
MBPP (Austin et al. 2021)	Code	2021	1401	https://huggingface.co/datasets/google-research-datasets/mbpp
AGIEval (Zhong et al. 2024)	Exam	2023	8062	https://github.com/ruixiangcui/AGIEval
GAOKAO-Bench (Zhang et al. 2023h)	Exam	2023	2811	https://github.com/OpenLMLab/GAOKAO-Bench
TruthfulQA (Lin et al. 2021)	Other	2021	1634	https://huggingface.co/datasets/truthfulqa/truthful_qa
BBH (Suzgun et al. 2023)	Other	2022	6511	https://huggingface.co/datasets/lukaemon/bbh

This table provides an exhaustive compilation of datasets pertinent to the evaluation of LLM. These datasets span a diverse array of tasks, from general to domain-specific, aiming to holistically assess the performance of LLM across various scenarios. The table delineates the publication timeline, the number of examples, and the access points (URLs) for each dataset, facilitating researchers in procuring and utilizing these resources

3.1.2 Soft prompt

Soft prompt methods involve appending a sequence of trainable continuous vectors, known as soft prompts, to the input of pre-trained language models. These soft prompts act as additional context that guides the model towards the desired output for a specific task. During training, the soft prompts are optimized to facilitate the model's adaptation to the new task, while the rest of the model remains largely unchanged, making the approach parameter-efficient. Based on the intuition that a properly optimized context, in the form of continuous word embeddings, can guide the language model towards performing an NLG task without altering its parameters, **Prefix-tuning** (Li and Liang 2021) and **prompt-tuning** (Lester et al. 2021) involve prepending a prefix P_θ of trainable vectors θ to the input. The activations for these prefix indices are treated as free parameters. To stabilize the optimization process, P_θ is parametrized by reparameterizing it through a smaller matrix P'_θ, which is then composed with a feedforward neural network (MLP), i.e., $P_\theta = \mathrm{MLP}(P'_\theta)$. **p-tuning** (Liu et al. 2021b) leverages trainable continuous prompt embeddings, which are concatenated with discrete prompts to form an input sequence for a pretrained language model. This sequence is then mapped to a hidden representation through an embedding function parameterized by a prompt encoder, such as an LSTM or MLP, and is optimized via backpropagation to minimize a task-specific loss function. **p-tuning v2** (Liu et al. 2021a) is an optimized prompt tuning method that universally matches the performance of fine-tuning across various model scales and NLU tasks by applying trainable continuous embeddings to every layer of the pre-trained model as prefix tokens, thus increasing the capacity of continuous prompts and reducing the gap to fine-tuning, especially for smaller models and more challenging tasks. **SMoP** (Sparse Mixture-of-Prompts) (Choi et al. 2023) utilizes a gating mechanism to route each input instance to one of multiple short soft prompts, which are specialized in handling different subsets of the data, thereby achieving efficient training and inference while maintaining performance gains typically induced by longer soft prompts. The routing probability for the *j*-th prompt is calculated as $p_j(X) = \mathrm{softmax}(L_\mu(\bar{X}))]_j$, where L_μ is a small linear router model, $\bar{X}$ is the average of input embeddings, and μ are the parameters of the router model. **APT** (Adaptive Prefix Tuning) (Zhang et al. 2023i) dynamically customizes the prefix at each layer of a Transformer model through a gate mechanism. It utilizes both fine-grained gated weight assignment and coarse-grained scaled weight specification. The pseudo prefix tokens $\hat{P}_i$ in the *i*th layer are updated as follows:

$$\hat{P}_i = \lambda_i \odot \alpha_i \cdot [P_{ik}, P_{iv}], \tag{11}$$

where $[P_{ik}, P_{iv}]$ represents the keys-values pair of the original pseudo prefix tokens, λ_i is a learnable scaled weight, $\odot$ denotes element-wise multiplication, and α_i represents the gated weights, which are calculated as:

$$\alpha_i = \mathrm{sigmoid}(h_{i-1} W_i), \tag{12}$$

where h_{i-1} represents the hidden states from the previous layer, and W_i are the parameters to be learned. **IDPG** (Instance Dependent Prompt Generation) (Wu et al. 2022) works on the principle of generating prompts for each input instance using a lightweight model *G* that takes the instance representation *x* and task *T* as inputs to produce a task-specific prompt

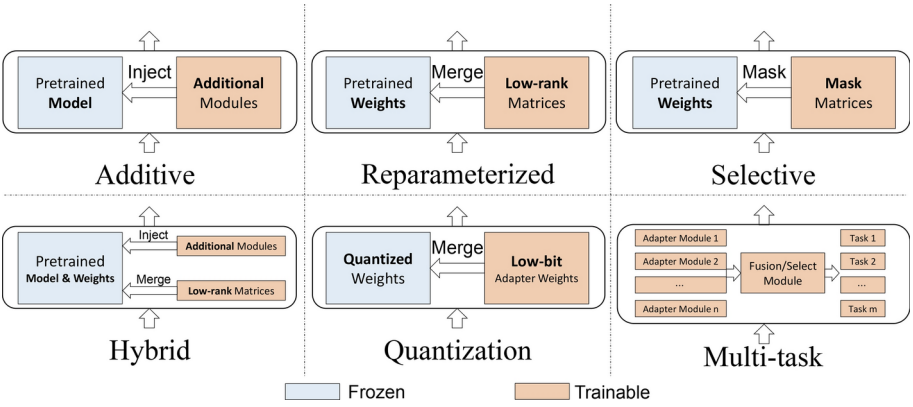


Fig. 3 Illustration of the main idea of different types of PEFT methods

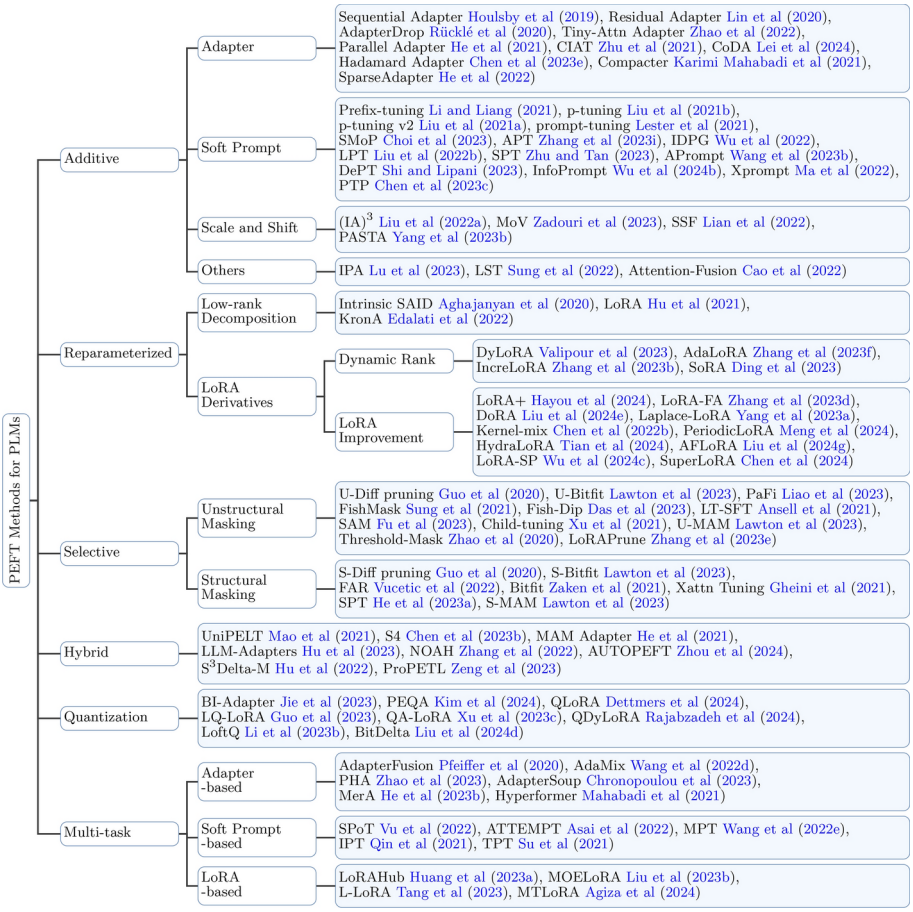


Fig. 4 Taxonomy of PEFT methods

Table 5 An overview of different types of PEFT methods: main idea, number of trainable parameters, applications, and limitations

Category	Main idea	Representative methods	# Trainable parameters	Applications	Advantages	Limitations
Additive	Add trainable components, freeze original.	Sequential Adapter (Houlsby et al. 2019), Prefix-tuning (Li and Liang 2021), $(IA)^3$ (Liu et al. 2022c), IPA (Lu et al. 2023)	#Params of additional modules	Single-task, rapid adaptation.	Minimal updates, flexible insertion.	Computational overhead, design-sensitive.
Reparameterized	Low-rank decomposition, tune low-rank matrices.	LoRA (Hu et al. 2021), AdaLoRA (Zhang et al. 2023f), DoRA (Liu et al. 2024e)	#Params of low-rank matrices	Large-scale, efficient updates.	Fewer parameters, no inference latency.	Low-rank constraints, hyperparameter tuning.
Selective	Update subsets (e.g., biases, masked params).	U-Bitfit (Lawton et al. 2023), FAR (Vucetic et al. 2022)	#Params of selected subsets (e.g., biases, masked params)	Resource-constrained environments.	Critical updates, low memory.	Task-sensitive, parameter selection.
Hybrid	Combine multiple PEFT methods dynamically.	UniPELT (Mao et al. 2021), MAM-Adapter (He et al. 2021)	#Params of used PEFT modules	Complex tasks, multimodal.	Task flexibility, improved performance.	High complexity, search overhead.
Quantization	Quantize model, enable efficient tuning.	QLoRA (Dettmers et al. 2024), BitDelta (Liu et al. 2024d)	#Params of used PEFT modules	Edge devices, low resources.	Low storage, low-precision inference.	Precision loss, quantization balance.
Multi-task	Share parameters and dynamic adapters for multi-task.	AdapterFusion (Pfeiffer et al. 2020), SPoT (Vu et al. 2022), MOELoRA (Liu et al. 2023b)	#Params of shared and task-specific modules	Multi-task, cross-task knowledge.	Redundant reduction, task transfer.	Task conflicts, routing complexity.

$W_p(T, x)$, which is then inserted into the input sequence x for fine-tuning the pre-trained language model M with a unified template, as denoted by the equations:

$$\begin{aligned} W_p(T, x) &= G(M(x), T), \quad x \in D_{train}, \\ h[CLS] &= M(\text{concat}[x, W_p(T, x)]). \end{aligned} \tag{13}$$

LPT (Late Prompt Tuning) (Liu et al. 2022b) is a method that inserts a "late prompt" into a pre-trained model (PTM) at an intermediate layer. This late prompt is created by a neural prompt generator (NPG) which uses the hidden states from the model layer just before the prompt insertion. This process generates a prompt that is tailored to each specific instance, enhancing the model's performance and efficiency. The generation of this instance-aware prompt involves a series of steps that include transformations and combinations of various elements derived from the model's hidden states. Once created, the prompt is reshaped to be integrated into the model's processing workflow. **SPT** (Selective Prompt Tuning) (Zhu and Tan 2023) initializes a prompt hyper-network where each intermediate layer of the pre-trained model (PTM) has a prompt generation layer controlled by a learnable probabilistic gate α_i, which is optimized to determine the importance of each layer for the task at hand, using the formulation $a_i = \sigma(\alpha_i)$, where σ is the sigmoid function, and p_i, the prompt at layer I, is calculated as $p_i = (1 - \tau \cdot a_i) \cdot p_{\text{prev},i} + \tau \cdot a_i \cdot p_{\text{new},i}$, with τ being a hyper-parameter that decides whether to discard the previous layer's prompt when a new one is generated. **APrompt** (Wang et al. 2023b) introduces trainable query, key, and value prompts, denoted as P_q, P_k, and P_v, into the self-attention mechanism of a Transformer encoder layer, which are integrated into the respective matrices to guide the attention computation during fine-tuning, while keeping the majority of the model parameters frozen. The new attention computations are formulated as:

$$\begin{aligned} L(\cdot) &= \text{MLP}(\text{LN}(\text{MSA}(\cdot))), \\ \text{MSA}(\cdot) &= \text{softmax}\left(\frac{Q_{\text{new}}^T K_{\text{new}}}{\sqrt{d}}\right) V_{\text{new}}, \end{aligned} \tag{14}$$

where MLP and LN represent the frozen multi-layer perceptron and layer norm, MSA is the multi-head self-attention module, Q_{new} is the new query matrix, K_{new} and V_{new} are the new key and value matrices augmented with attention prompts, and d is the dimension of the embeddings. **DePT** (Decomposed Prompt Tuning) (Shi and Lipani 2023) decomposes a trainable soft prompt matrix $P \in \mathbb{R}^{l \times d}$ into a shorter trainable prompt matrix $P_s \in \mathbb{R}^{m \times d}$ and a pair of low-rank matrices $A \in \mathbb{R}^{s \times r}$ and $B \in \mathbb{R}^{r \times d}$, where the rank $r \ll \min(s, d)$. These components are optimized with different learning rates α_1 and α_2 respectively. The updated word embedding matrix for the ith sample is given by $W_i' = W_i + BA$, where W_i is the original word embedding matrix. The loss function to be optimized is $L_{\text{DePT}} = -\sum_{i=1}^{N} \log P(y_i|[P_s, W_i']; \Theta)$, where Θ represents the frozen pretrained model weights. **Xprompt** (Ma et al. 2022) operates on the principle of hierarchical structured pruning to identify and retain only the most effective soft prompt tokens, denoted as p_i, and their components, denoted as $q_{i,e}$, by calculating their importance scores I_{p_i} and $I_{q_{i,e}}$ using the following expressions:

Table 6 Performance evaluation across various PEFT methods for fine-tuning common base models (RoBERTa-base, RoBERTa-large, and DeBERTaV3-base) on the GLUE benchmark

Model	PEFT type	PEFT method	#TPs	CoLA	SST2	MRPC	STS-B	QQP	MNLI	QNLI	RTE
RoBERTa-base		FT	124.6M	59.07	92.89	88.24/91.58	90.87/90.61	90.81/87.72	86.27	91.07	72.2
	Additive	AdapterS	7.41M	63.32	94.31	90.44/93.18	91.25/90.94	90.81/86.55	87.33	92.06	73.56
		Prefix-tuning	0.96M	59.31	93.81	87.25/91.03	88.48/88.32	87.75/84.09	85.21	90.77	54.51
		(IA)3	0.66M	59.58	93.92	87.00/90.52	90.30/90.32	87.99/84.10	83.95	90.88	71.12
	Reparameterized	LoRA	0.89M	62.09	94.04	87.50/90.68	90.66/90.83	88.83/85.21	86.54	92.02	72.92
		AdaLoRA	1.03M	59.82	93.92	87.99/91.33	90.83/90.73	88.58/84.98	86.26	91.43	70.04
	Selective	BitFit	0.69M	61.32	94.72	89.22/92.41	90.34/90.27	88.12/84.11	84.64	91.09	77.98
		Child-Tuning	/	60.33	93.58	89.22/92.20	91.14/90.93	90.98/88.04	87.4	92.2	77.62
	Hybrid	MAM Adapter	46.78M	61.42	94.87	89.31/92.21	90.74/90.42	88.31/83.20	86.63	90.19	72.62
RoBERTa-large		FT	355.3M	65.78	95.54	89.22/92.28	91.74/91.76	89.30/86.68	89.42	93.61	81.23
	Additive	AdapterS	19.77M	67.03	96.37	89.94/92.54	92.58/92.42	92.19/88.50	91	94.31	85.25
		Prefix-tuning	2.03M	59.01	95.76	88.24/91.37	90.92/91.07	88.88/85.45	89.3	93.32	74.01
		(IA)3	1.22M	61.15	94.61	86.52/90.33	92.22/92.03	89.45/86.25	88.63	94.25	81.23
	Reparameterized	LoRA	1.84M	64.47	96.67	87.50/91.19	91.66/91.44	90.15/86.91	90.76	95	79.78
		AdaLoRA	2.23M	65.85	94.95	89.46/92.34	92.05/91.80	89.60/86.30	90.36	94.62	77.98
	Selective	BitFit	1.32M	68.01	96.1	90.93/93.38	91.93/91.77	89.48/86.43	89.98	94.47	87.73
		Child-Tuning	/	63.08	95.07	90.69/93.43	92.36/92.18	91.52/88.75	35.45	93.15	86.25
	Hybrid	MAM Adapter	122.2M	67.39	95.81	90.12/92.77	92.44/92.18	90.87/86.65	90.62	94.31	86.62
DeBERTaV3-base		FT	/	69.2	95.3	89.5/93.3	91.6/91.1	92.4/89.8	90.5	94	82
	Quantization	QLoRA	/	N.A	86.5	73.8/82.8	83.0/82.8	86.8/82.3	75.4	82.4	55.9
		LoftQ	/	37.4	90.2	83.8/88.6	87.1/86.9	90.3/86.9	84.7	86.6	61.4

All performance metrics are cited from prior published works (Xu et al. 2023a; Li et al. 2023b). Metrics may vary by task: Matthews correlation for COLA, accuracy/F1 score for MRPC and QQP, Pearson/Spearman correlation for STS-B, average matched accuracy for MNLI, and accuracy for the remaining tasks. Higher metric values indicate superior performance. $\#TP$ denotes the number of trainable parameters for each method

$$I_{p_i} = \mathbb{E}_{x \sim D_x} \left| \frac{\partial L(x)}{\partial \gamma_i} \right|, \quad I_{q_{i,e}} = \mathbb{E}_{x \sim D_x} \left| \frac{\partial L(x)}{\partial \zeta_i} \right|, \tag{15}$$

where L is the loss function, D_x is the training data distribution, γ_i and ζ_i are mask variables for token-level and piece-level pruning respectively, and the importance scores determine the contribution of each prompt token and piece to the model's performance. **InfoPrompt** (Wu et al. 2024b) maximizes the mutual information between the prompt P and the parameters of the classification head θ, denoted as $I(P; \theta|X)$, and between the prompt P and the encoded representation from the pretrained language model $Z = \Phi(P, X)$, denoted as $I(P; Z|X)$, by optimizing two novel loss functions, referred to as the head loss and the representation loss, respectively. **PTP** (Prompt Tuning with Perturbation-based Regularizer) (Chen et al. 2023c) introduces perturbation-based regularizers to stabilize prompt tuning by smoothing the loss landscape. This can be formulated as:

$$\min_{\theta} \mathbb{E}_{(s,y)\sim D} \left[L \left(M \left(\theta, s + \delta, y \right) \right) \right], \tag{16}$$

where δ is the perturbation sampled from either a Gaussian distribution ($\delta \sim \mathcal{N}$ for PTP-RN) or generated by an adversarial attack algorithm ($\delta = \text{argmax}_{\|\delta\| \leq \epsilon} L\left(\theta, s + \delta, y\right)$ for PTP-ADV). s is the input sequence, y is its label, M is the LLM, θ represents the trainable prompt parameters, and L is the loss function.

3.1.3 Scale and shift

$(\text{IA})^3$ (Infused Adapter by Inhibiting and Amplifying Inner Activations) (Liu et al. 2022c) shown in Fig. 6a is a PEFT method for scaling inner activations of a model by learned vectors. For a decoder with L layers, $(\text{IA})^3$ adds scaling vectors l_k, l_v, and l_{ff} (initialized as ones) to scale key, value, and feed-forward activations, respectively. This allows for task-specific adaptations while updating a tiny fraction ($\leq 0.01\%$) of the model's parameters, facilitating mixed-task batches. The method can be applied permanently to weight matrices if the model is dedicated to a single task, avoiding extra computations. **MoV** (Mixture of Vectors) (Zadouri et al. 2023) introduces a parameter-efficient Mixture of Experts (MoE) architecture that updates only lightweight experts, less than 1% of an 11B parameter model. It generalizes well to unseen tasks. Computation is routed with soft merging: $E_{\text{mix}} = \sum_{i=1}^{n} s_i \cdot E_i$; $y = E_{\text{mix}}(x)$, where E_i represents each expert, s_i is the gating weight for each expert, and x is the input. This approach ensures robust performance under strict parameter constraints. **SSF** (Lian et al. 2022) shown in Fig. 6b modifies deep features extracted by a pre-trained model through linear transformations to match the distribution of the target dataset. Given an input $x \in \mathbb{R}^{(N^2+1)\times d}$, the output y is computed as:

$$y = [\gamma \odot x + \beta]^T, \tag{17}$$

where γ and β are learnable scale and shift parameters, respectively, and $\odot$ denotes element-wise multiplication. This approach requires tuning far fewer parameters than full fine-tuning. **PASTA** (PArameter-efficient tuning with Special Token Adaptation) (Yang et al. 2023b), as illustrated in Fig. 6c, modifies special token representations in pretrained mod-

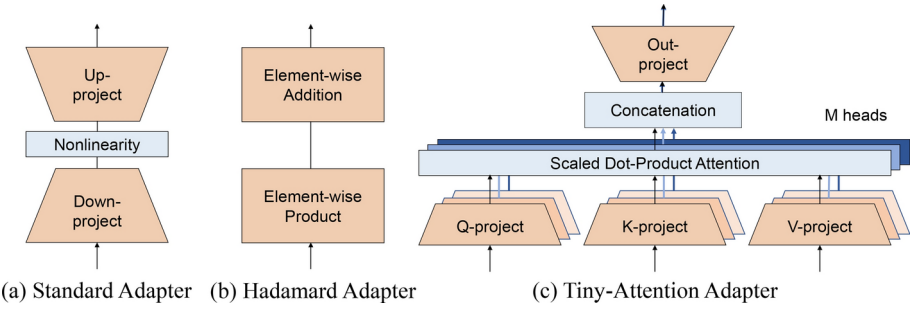


Fig. 5 Illustration of three representative types of adapter

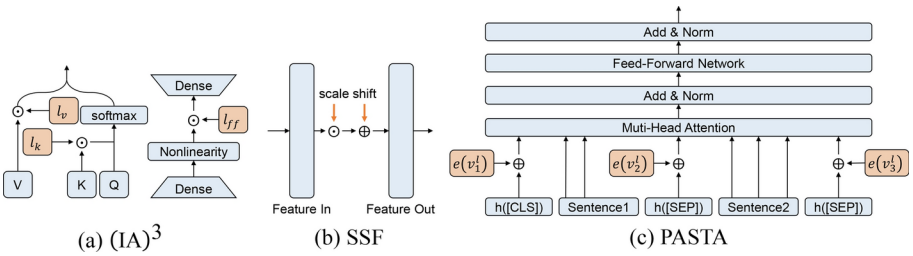


Fig. 6 Illustration of three representative scale and shift algorithms

els. For the l^{th} Transformer layer, given input $H^{(l)} = \{h_i^{(l)}\}_{i=1}^N$, where $h_i^{(l)} \in \mathbb{R}^d$, PASTA updates the input as $H_{\text{mod}}^{(l)} = \{h_i^{(l)} + m_i^{(l)}\}_{i=1}^N$, where $m_i^{(l)}$ is defined as:

$$m_i^{(l)} = \begin{cases} 0 & \text{if } i \text{ is not a special token} \\ e(v_p^{(l)}) & \text{if } i \text{ is the } p\text{th special token} \end{cases}, \tag{18}$$

with $e(v_p^{(l)}) \in \mathbb{R}^d$ being the trainable vector for the pth special token at layer I.

3.1.4 Others

IPA (Inference-time Policy Adapters) (Lu et al. 2023) tailors LLMs to specific objectives without fine-tuning. IPA combines the output distribution of a base LLM with a smaller, trainable adapter policy. The adapter is optimized via reinforcement learning (RL) to align the LLM's output with user-defined goals. At inference, the base model's distribution and the trained adapter's distribution are merged for decoding as follows:

$$p_{\text{combined}}(\text{output} \mid \text{input}) = \alpha p_{\text{base}}(\text{output} \mid \text{input}) + (1-\alpha) p_{\text{adapter}}(\text{output} \mid \text{input}), \tag{19}$$

where p_{base} is the base model's probability distribution, p_{adapter} is the adapter's distribution, and α controls their mixture. **LST** (Ladder Side-Tuning) (Sung et al. 2022) introduces a side network that predicts outputs using shortcuts (ladders) from a pre-trained backbone, avoiding backpropagation through the entire backbone. Formally, given a backbone $f_N(f_{N-1}(\ldots f_2(f_1(x))\ldots))$, the side network g takes intermediate activations z_i as inputs, where $z_i = f_i(x)$. The final output $\hat{y}$ is computed by $g(z_i; \theta_g)$, significantly reducing mem-

ory cost. Here, x is the input, f_i represents the i-th layer function, and θ_g are the parameters of the side network. **Attention-Fusion** (Cao et al. 2022) aggregates intermediate layer representations from a pre-trained model to compute task-specific token representations. This module trains only 0.0009% of total parameters and achieves competitive performance to full fine-tuning. Formally, given a pre-trained model with L layers, the output $\mathrm{h}_i^{(l)}$ of each layer l for token i is used to compute a weighted sum $\mathrm{r}_i = \sum_{l=1}^{L} \alpha_i^{(l)} \mathrm{h}_i^{(l)}$, where $\alpha_i^{(l)}$ represents the attention weight for layer l on token i.

3.2 Reparameterized PEFT

Reparameterization is a technique for improving the training efficiency and performance of a model by transforming its parameters. In the context of PEFT, the transformation involves low-rank parameterization, which entails constructing a low-rank learnable parameter matrix to adapt to specific downstream tasks. During training, only the low-rank parameter matrix is fine-tuned, and at inference time, the learned matrix is combined with the pre-trained parameters to ensure that inference speed is not affected.

3.2.1 Low-rank decomposition

LoRA (Low-rank Adaptation) (Hu et al. 2021) introduces low-rank trainable matrices $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$ to update the pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$ via $\Delta W = BA$, where $W = W_0 + \Delta W$ is used for inference without additional latency. **KronA** (Edalati et al. 2022) is a Kronecker product-based adapter module for efficient fine-tuning of Transformer-based pre-trained language models (PLMs). The tuned weight matrix W_{tuned} is computed as the original PLM weight matrix W plus a scaled Kronecker product of two learnable matrices A_k and B_k:

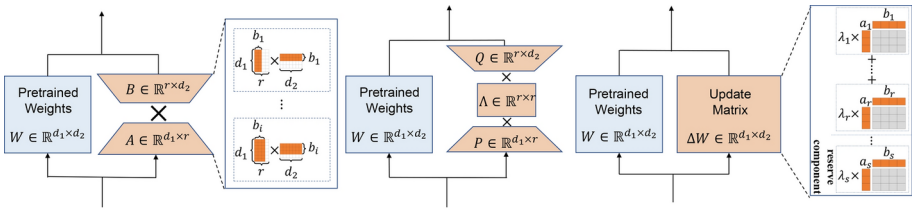
$$W_{\text{tuned}} = W + s[A_k \otimes B_k], \tag{20}$$

where s is a scaling factor, and $\otimes$ denotes the Kronecker product operator.

3.2.2 LoRA derivatives

3.2.2.1 Dynamic rank

DyLoRA (Valipour et al. 2023) shown in Fig. 7a introduces a dynamic low-rank adaptation technique by training Low-Rank Adapter (LoRA) blocks for a range of ranks during training, where the representation learned by the adapter module is sorted at different ranks, enabling the model to be flexible and perform well across a wider range of ranks without additional training time or the need for rank selection. **AdaLoRA** (Zhang et al. 2023f) illustrated in Fig. 7b dynamically allocates the budget among weight matrices based on their importance scores, where incremental updates are parameterized in the form of a singular value decomposition as $W = W_0 + P\Lambda Q$, with $P \in \mathbb{R}^{d_1\times r}$, $Q \in \mathbb{R}^{r\times d_2}$, and $\Lambda \in \mathbb{R}^{r\times r}$ being the left singular vectors, right singular vectors and singular values, respectively. **IncreLoRA** (Zhang et al. 2023b) presented in Fig. 7c incrementally allocates trainable parameters during the training process based on the importance scores of each module, which is formulated as follows:



(a) DyLoRA (b) AdaLoRA (c) IncreLoRA

Fig. 7 Illustration of three representative dynamic rank methods in LoRA

$$W = W_0 + \sum_{i=1}^{r} \lambda_i w_i = W_0 + \sum_{i=1}^{r} \lambda_i b_i a_i, \tag{21}$$

where W_0 is the pretrained weight matrix, $r \ll \min(in, out)$, w_i is a rank-1 matrix, $a_i \in \mathbb{R}^{in}$, $b_i \in \mathbb{R}^{out}$, and λ_i is a scaling factor updated through backpropagation, with λ_i initialized to zero to ensure the initial update matrix is zero. **SoRA** (Sparse low-rank Adaption) (Ding et al. 2023) introduces a gate unit, optimized with a proximal gradient method to control the sparsity of the LoRA's low-rank matrices. The gate unit enables dynamic adjustment of the rank of LoRA during training, enhancing representation power while maintaining parameter efficiency. During inference, blocks corresponding to zero entries in the gate unit are eliminated, reducing the SoRA module to a concise, rank-optimal LoRA.

3.2.2.2 LoRA improvement **LoRA+** (Hayou et al. 2024) introduces a novel technique by applying different learning rates to the down- and up-projection matrices A and B: $\eta_B = \lambda \eta_A$, where λ is a fixed value greater than 1, focusing on tuning η_A for enhanced model adaptability. Designed to mitigate the significant memory requirements for activations that are intrinsic to LoRA, **LoRA-FA** (Low-Rank Adaptation with Frozen-A) (Zhang et al. 2023d) freezes the pre-trained weight W and the projection-down weight A, and only update the projection-up weight B during the fine-tuning process, which results in a model weight change ΔW that resides in a low-rank space defined by the column space of A. The method is designed to reduce the activation memory footprint without incurring additional computational overhead. **DoRA** (Weight-Decomposed Low-Rank Adaption) (Liu et al. 2024e) aims to bridge the gap in performance between LoRA and full fine-tuning (FT) by leveraging a novel weight decomposition approach. It decomposes the pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$ into *magnitude* and *direction*. During fine-tuning, only the *direction* component is updated using a low-rank approximation $\Delta W = BA$, where $B \in \mathbb{R}^{d \times r}$ and $A \in \mathbb{R}^{r \times k}$, and $r \ll \min(d, k)$. Here, r denotes the rank of the low-rank approximation, d and k represent the dimensions of the weight matrix. This allows for efficient parameter updates while preserving the original weight's magnitude, enhancing learning capacity and stability. **Laplace-LoRA** (Yang et al. 2023a) introduces a Bayesian approach to LoRA for fine-tuning LLMs. It addresses the issue of overconfidence in fine-tuned LLMs by estimating predictive uncertainty. Laplace-LoRA approximates the posterior distribution over LoRA parameters using a Laplace approximation, leading to better-calibrated models. Mathematically, given a maximum a posteriori (MAP) estimate θ_{MAP}, the predictive distribution for a new input x^* is approximated as:

$$f_\theta(x^*) \sim \mathcal{N}\left(f_{\theta_{\text{MAP}}}(x^*), \Lambda\right), \tag{22}$$

where $\Lambda = (\nabla_\theta f_\theta(x^*)|_{\theta=\theta_{\text{MAP}}})\Sigma(\nabla_\theta f_\theta(x^*)|_{\theta=\theta_{\text{MAP}}})^\top$. Here, $\nabla_\theta f_\theta(x^*)$ represents the gradient of the prediction with respect to the parameters, and Σ is the covariance matrix of the Laplace approximation. The prior precision λ is optimized using the Laplace marginal likelihood on the training dataset:

$$P(y|X) \approx \exp(L(y, X; \theta_{\text{MAP}}))(2\pi)^{D/2}|\Sigma|^{1/2}, \tag{23}$$

Samples from the predictive distribution are obtained by:

$$\tilde{f}_\theta(x^*) = f_{\theta_{\text{MAP}}}(x^*) + L\xi, \tag{24}$$

where L is the Cholesky factor of Λ and ξ is a vector of independent standard normal random variables. This method improves calibration without requiring a separate validation set, making it suitable for small datasets. **PeriodicLoRA** (PLoRA) (Meng et al. 2024) enhances LoRA's learning capacity by periodically accumulating low-rank updates to form a higher-rank matrix. During each stage, only LoRA weights W_{LoRA} are updated. At the end of each stage, W_{LoRA} is unloaded into the backbone parameters W_{backbone}, i.e., $W_{\text{backbone}} \leftarrow W_{\text{backbone}} + \Delta W_{\text{LoRA}}$, and then W_{LoRA} is reinitialized. This increases the effective update rank without additional memory cost. **HydraLoRA** (Tian et al. 2024) enhances LoRA by adopting an asymmetric structure for efficient fine-tuning. It segments the LoRA into multiple "intrinsic components," each with a distinct matrix B_k, sharing a common matrix A. The update formula is given by:

$$\alpha W = W_0 + r \sum_{k=1}^{N} AB_k, \tag{25}$$

where W_0 is the original weight matrix, r is a scaling factor, A and B_k are low-rank matrices, and N is the number of components. A trainable MoE router dynamically allocates samples to these components for fine-tuning. **AFLoRA** (Liu et al. 2024g) incrementally freezing trainable low-rank matrices based on a novel freezing score, computed using smoothed gradient $\bar{I}(t)_{A_l}$, uncertainty tensor $\bar{U}(t)_{A_l}$, and their Hadamard product to determine the stability of weights throughout training, as described by the equations:

$$\begin{aligned}
I(t)_{A_l} &= |\nabla L(\theta)|, \\
\bar{I}(t)_{A_l} &= \beta_1 \bar{I}(t-1)_{A_l} + (1-\beta_1) I(t)_{A_l}, \\
U(t)_{A_l} &= |I(t)_{A_l} - \bar{I}(t)_{A_l}|, \\
\bar{U}(t)_{A_l} &= \beta_2 \bar{U}(t-1)_{A_l} + (1-\beta_2) U(t)_{A_l}, \\
s(t)_{A_l} &= \text{mean}(\bar{I}(t)_{A_l} \odot \bar{U}(t)_{A_l}),
\end{aligned} \tag{26}$$

where A_l represents the low-rank tensor, $L(\theta)$ is the loss function, β_1 and β_2 are smoothing factors, and t denotes the current training step. **LoRA-SP** (Wu et al. 2024c) selectively freezes half of the parameters in the matrices A and B during fine-tuning, with the adapted

weight matrix ΔW calculated as $\Delta W = (A \odot S)(B \odot S)^\top$, where S is a binary selection matrix that determines which parameters to update or freeze, and $\odot$ denotes element-wise multiplication. **SuperLoRA** (Chen et al. 2024) generalizes LoRA approach by jointly adapting all weight updates ΔW across layers through a high-order tensor decomposition, where $\Delta W_{\text{group}_g}$ is computed as

$$F(\Delta W_{\text{lora}_g}) = F\left(\bigotimes_{k=1}^{K}\left(C_{gk}\prod_{m=1}^{M}{}_{\times m}A_{gkm}\right)\right), \tag{27}$$

with F being a projection function, M the order of tensor modes, K the number of Kronecker splits, C_{gk} the core tensor, A_{gkm} the plane factors, $\prod_{m=1}^{M}{}_{\times m}$ the tensor products from model-1 to model-M, and $\bigotimes$ the Kronecker product.

3.3 Selective PEFT

Contrary to Additive PEFT, Selective PEFT selects a very small subset of the pre-trained model's parameters for fine-tuning to adapt to specific downstream tasks through a parameter masking matrix. Depending on the way the parameters are masked, Selective PEFT can be divided into unstructured masking and structured masking.

3.3.1 Unstructural masking

U-Diff pruning (Guo et al. 2020) introduces a task-specific "diff" vector δ_τ that is added to pretrained model parameters θ. The task-specific parameters are defined as $\theta_\tau = \theta + \delta_\tau$. During training, δ_τ is adaptively pruned using a differentiable L_0-norm approximation to encourage sparsity. θ remains fixed. This method enables efficient transfer learning, modifying only a small fraction of the parameters per task. **U-Bitfit** (Lawton et al. 2023) determines which components of the bias update vector Δb should be zero or non-zero, based on a first-order approximation of the change in training loss from pruning a bias parameter θ, calculated as $-\theta \cdot \frac{\partial L}{\partial \theta}$. **PaFi** (Liao et al. 2023) generates a universal sparse mask for parameter selection without training. PaFi identifies the least significant pre-trained parameters by their magnitude and fine-tuning only those, represented as selecting parameters θ_i where $|\theta_i| \leq \text{sort}(|\theta|)_k$ for the mask m. **FishMask** (Sung et al. 2021) precomputes a fixed sparse mask for neural network parameters, selecting the top k parameters based on their Fisher information to be updated during training. This "FISH (Fisher-Induced Sparse uncHanging) mask" enables efficient training by updating only a subset of parameters, which reduces memory and communication costs compared to full model updates. k represents the number of parameters to be selected for updates, and Fisher information measures parameter importance for the given task. **Fish-Dip** (Das et al. 2023) dynamically updates the importance of model parameters for fine-tuning based on feedback from the most regressing samples, using the empirical Fisher information to create a sparsity mask that focuses training on a subset of parameters, as denoted by the equation:

$$\hat{F}_\theta \approx \frac{1}{n}\sum_{\{(x_i,y_i)|L_{tr}(x_i,y_i)\in \text{top}_n\}}\left(\frac{\partial \log p_\theta(y_i|x_i)}{\partial \theta}\right)^2, \tag{28}$$

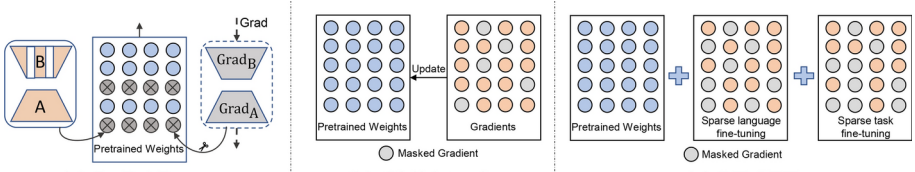
where $\hat{F}_\theta$ represents the empirical Fisher information, n is the number of most regressing training examples, $p_\theta(y_i|x_i)$ is the output probability for the given input x_i and parameters θ, and the sum is taken over the top n regressing examples as determined by their loss L_{tr} during training. **LT-SFT** (see Fig. 8c) (Ansell et al. 2021) introduces a composable sparse fine-tuning method for cross-lingual transfer learning. It learns sparse, real-valued masks based on a variant of the Lottery Ticket Hypothesis (LTH). Task-specific masks are derived from supervised data in the source language, while language-specific masks are obtained through masked language modeling in the target language. These masks are composed with the pre-trained model to enable zero-shot cross-lingual transfer. The sparsity of the masks reduces parameter overlap and interference, improving modularity and preventing overfitting. **SAM** (Second-order Approximation Method) (Fu et al. 2023) approximates the original optimization problem using a second-order Taylor expansion to make it analytically solvable, and directly determines the parameters to optimize by solving the approximation function, which is formulated as:

$$\min_{\Delta\theta} \left[L(\theta_0) + \nabla L(\theta_0)^T M\Delta\theta + \frac{1}{2}(M\Delta\theta)^T H M\Delta\theta \right], \tag{29}$$

subject to $\|M\|_0 = \lfloor mp \rfloor; M_{ij} = 0, \forall i \neq j; M_{ii} \in \{0, 1\}$, where θ_0 are the pre-trained parameters, $\Delta\theta$ is the difference vector, M is the parameter mask matrix, L is the loss function, $\nabla L(\theta_0)$ is the gradient of the loss function at θ_0, and H is an approximated diagonal Hessian matrix. **Child-tuning** (see Fig. 8b) (Xu et al. 2021) updates only a subset of parameters, referred to as the child network, during fine-tuning while masking out the gradients of the remaining parameters in the backward pass, which can be formulated as:

$$w_{t+1} = w_t - \eta \odot \frac{\partial L(w_t)}{\partial w_t} \odot M_t, \tag{30}$$

where w_t represents the model parameters at the t^{th} iteration, η is the learning rate, $L(w_t)$ is the loss function, and M_t is a 0–1 mask indicating the child network. **U-MAM** (Lawton et al. 2023) is an unstructured neural architecture search approach for parameter-efficient tuning of large pre-trained language models. It involves pruning a dense low-rank update from an initial parameter-efficient tuning architecture to find an efficient subset of parameters to fine-tune. **Threshold-Mask** (Zhao et al. 2020) learns selective binary masks for pre-trained language model weights without fine-tuning, where each linear layer W_l is associated with a real-valued matrix M_l initialized randomly, and a binary mask M_l^{bin} is obtained by applying a thresholding function, used to select important weights: $(m_l^{bin})_{i,j} = 1(m_{l,i,j} \geq \tau)$ with $m_{l,i,j} \in M_l$ and the global thresholding hyperparamter τ, and the masked weights are



(a) LoRAPrune (b) Child-tuning (c) LT-SFT

Fig. 8 Illustration of three representative unstructural masking methods

computed as $\hat{W}_l = W_l \odot M_l^{bin}$, with M_l updated during training via the straight-through estimator: $M_l \leftarrow M_l - \eta \frac{\partial L(\hat{W}_l)}{\partial M_l^{bin}}$. **LoRAPrune** (see Fig. 8a) (Zhang et al. 2023e) approximates the importance of each parameter in the pre-trained model weights W_0 by utilizing the gradients of the low-rank matrices A and B, which are then used to perform structured pruning in an iterative and progressive manner, efficiently reducing the model's size while maintaining performance.

3.3.2 Structural masking

S-Diff pruning (Guo et al. 2020) introduces a structured pruning strategy by dividing the weight parameters into local groups and strategically removing them collectively. **S-Bitfit** (Lawton et al. 2023) selects whether to update each bias parameter b with a learned update Δb, where the decision is based on a pruning criterion that sums the first-order approximation of the loss change over the entire bias update Δb, expressed as $-\sum_{\theta \in \Delta b} \theta \cdot \frac{\partial L}{\partial \theta}$. **FAR** (Freeze And Reconfigure) (Vucetic et al. 2022) leverages overparameterization in BERT-like models to efficiently fine-tune them on resource-constrained devices. FAR selectively updates parameters based on their importance, determined through priming, while freezing others. This reduces memory usage and fine-tuning time, with minimal impact on performance. Notation-wise, if P represents the total parameters, $P_{\text{frozen}} \subset P$ denotes frozen parameters, and $P_{\text{active}} = P \backslash P_{\text{frozen}}$ are active parameters updated during fine-tuning. P_{frozen} is selected using priming to ensure optimal performance. **BitFit** (Zaken et al. 2021) modifies only the bias terms of a pre-trained BERT model, demonstrating competitive performance with full fine-tuning on small to medium datasets and practical utility for deploying multi-task models in memory-constrained environments. **Xattn Tuning** (Gheini et al. 2021) updates only cross-attention parameters in Transformer models for machine translation, showing it can achieve near-equivalent performance to fine-tuning the entire model, while also leading to crosslingually aligned embeddings that can mitigate catastrophic forgetting and enable zero-shot translation capabilities. **SPT** (He et al. 2023a) identifies task-specific sensitive parameters by measuring their impact on loss reduction, denoted as s_n, and then adaptively allocates trainable parameters to these positions under a given budget τ, utilizing both unstructured tuning for individual parameters and structured tuning for weight matrices with a high number of sensitive parameters, as indicated by σ_{opt}. **S-MAM** (Lawton et al. 2023) is a structured neural architecture search approach for parameter-efficient tuning of large pre-trained language models. It selects and fine-tunes a fixed rank of parameters within the model's attention mechanisms and feed-forward networks.

3.4 Hybrid PEFT

Due to the significant performance differences of different types of PEFT methods on various tasks, many studies aim to enhance model performance by combining the advantages of different types of PEFT methods. These research efforts are summarized as Hybrid PEFT methods. A representative hybrid PEFT method, known as MAM-Adapter, is illustrated in Fig. 9.

UniPELT (Mao et al. 2021) operates on the principle of dynamically activating the most suitable parameter-efficient language model tuning (PELT) submodules for a given task

through a gating mechanism, which is mathematically represented as $h'_A = G_A h_A + h_F$, where h'_A is the final output, h_A is the output of the adapter submodule, h_F is the direct input to the adapter, and G_A is the gating function that modulates the contribution of the adapter submodule based on the specific data and task setup. **S4** (Chen et al. 2023b) discovers design patterns by grouping layers in a spindle pattern, uniformly allocating trainable parameters, tuning all groups, and assigning tailored strategies to different groups, consistently outperforming existing fine-tuning strategies across various NLP tasks and models. **MAM Adapter** (He et al. 2021) is a unified framework for parameter-efficient transfer learning methods by reframing them as modifications to specific hidden states in pretrained models, which can be mathematically represented as $h \leftarrow (1 - \lambda(x))h + \lambda(x)\Delta h$, where h is the original hidden representation, $\lambda(x)$ is a gating scalar, and Δh is the modification vector computed by a function f applied to the input x. **LLM-Adapters** (Hu et al. 2023) discusses the use of different adapters such as Series Adapters, Parallel Adapters, and LoRA (Low-Rank Adaptation), which are incorporated into the model's architecture at optimal locations. **NOAH** (Zhang et al. 2022) employs neural architecture search to automatically design optimal "prompt modules" for large vision models, tailored to each downstream dataset, enhancing transfer learning, few-shot learning, and domain generalization. **AUTOPEFT** (Zhou et al. 2024) automates the configuration selection for PEFT of large pre-trained language models. It employs a multi-objective Bayesian optimization approach to discover a set of Pareto-optimal configurations that balance task performance with parameter efficiency, significantly outperforming existing PEFT methods with minimal training costs. S^3Delta-M (Hu et al. 2022) automatically searches for an optimal trainable structure within pre-trained models by using a unified framework of various Delta Tuning methods. It employs bi-level optimization and a shifted global sigmoid function to control sparsity, achieving high performance with minimal trainable parameters. **ProPETL** (Zeng et al. 2023) enables the sharing of a single prototype network across different layers and

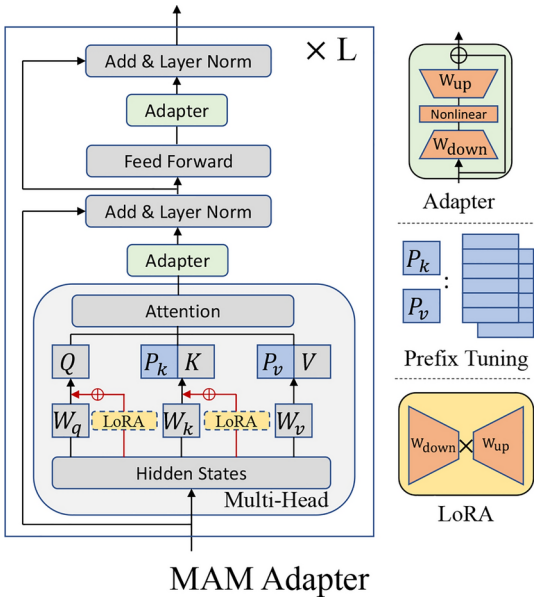


Fig. 9 Illustration the principles of MAM-Adapter, a representative hybrid PEFT method

tasks, with binary masks learned to prune sub-networks, significantly reducing parameter storage while improving efficiency and performance over other methods.

3.5 Quantization PEFT

Quantization is another widely used and studied technique aimed at improving computational efficiency and reducing memory usage. We summarize the PEFT methods that use and research quantization technology, as Quantization PEFT.

BI-Adapter (Jie et al. 2023) introduces a novel method for low-precision adapter training in vision models. It utilizes the observation that adapter parameters converge to flat minima, suggesting robustness to precision reduction. The method employs a quantization-aware training strategy, minimizing the quantization error by clustering weight parameters into Gaussian distributions. Specifically, weights w are standardized $w' = \frac{w-\mu}{\sigma}$, quantized, and then de-standardized to backpropagate gradients effectively. This approach significantly reduces model size with minimal impact on performance, addressing storage and transmission inefficiencies in multi-task learning. **PEQA** (Kim et al. 2024) involves a two-step process: first, decomposing the parameter matrix of each fully-connected layer into a low-bit integer matrix and quantization scales, and second, fine-tuning only the quantization scale while keeping the integer matrix frozen, which can be mathematically represented as:

$$\tilde{W} = (s_0 + \Delta s) \cdot \left(\text{clamp} \left(\lfloor \frac{W_0}{s_0} \rceil + z_0, 0, 2^b - 1 \right) - z_0 \right), \tag{31}$$

where the notation $A \cdot B$ denotes the element-wise product of matrices A and B. The symbol $\lfloor \cdot \rceil$ represents the rounding function, which rounds its argument to the nearest integer. The function $\text{clamp}(\cdot, a, b)$ signifies the clamping operation that constrains its input within the range $[a, b]$. Here, W_0 denotes the original weight matrix, s_0 represents the initial scale factor, and z_0 is the zero-point value. The variable $\Delta s \in \mathbb{R}^{n \times 1}$ signifies the gradient update of s_0, obtained through adaptation to a downstream task, and b indicates the bit-width. **QLORA** (Dettmers et al. 2024), a quantized version of LoRA, utilizes 4-bit NormalFloat (NF4) precision for quantizing pretrained models, enhanced by double quantization and a paged optimizer to prevent the gradient checkpointing memory spikes. The NF4 is an information theoretically optimal quantization data type for normally distributed data, delivering enhanced empirical performance over 4-bit Integer and Float representations. While QLoRA converts the FP16 pretrained weights W to the NF4 precision to enable LLM fine-tuning on a reduced number of GPUs, the auxiliary weights of the LoRA matrix re-quantize the final weights back to FP16 post-finetuning. Therefore, **QA-LoRA** (Quantization-Aware Low-Rank Adaptation) (Xu et al. 2023c) addresses the imbalance between quantization and adaptation by employing group-wise operations, which increase the flexibility of low-bit quantization while reducing that of the adaptation process. The algorithm is straightforward to implement and provides two key benefits: during fine-tuning, LLM weights are quantized (e.g., to *INT*4) to conserve time and memory; post fine-tuning, the LLM and auxiliary weights are seamlessly integrated into a quantized model without accuracy loss. The comparative analysis and conceptual distinctions among LoRA, QLoRA, and QA-LoRA methodologies are visually illustrated in Fig. 10. **LoftQ** (Li et al. 2023b) introduces a simultaneous process of quantizing an LLM and initializing LoRA with low-rank matrices to mitigate performance gaps. The algorithm approximates the original weights $W \in \mathbb{R}^{d_1 \times d_2}$ with a quantized version $Q \in \mathbb{R}_N^{d_1 \times d_2}$ and low-rank matrices $A \in \mathbb{R}^{d_1 \times r}$ and $B \in \mathbb{R}^{d_2 \times r}$

, minimizing the Frobenius norm $\|W - Q - AB^\top\|_F$. LoftQ alternates between quantization and SVD, efficiently approximating the original weights for improved downstream task performance, especially in 2-bit and 2/4-bit mixed precision scenarios. **LQ-LoRA** (Guo et al. 2023) iteratively decomposes a pretrained matrix W into a quantized component Q and a low-rank component L_1L_2 by solving the optimization problem:

$$\arg\min_{Q,L_1,L_2} \|W - (Q + L_1L_2)\|_F, \tag{32}$$

where Q is fixed during finetuning and only L_1 and L_2 are updated. **QDyLoRA** (Rajabzadeh et al. 2024) is a quantized dynamic low-rank adaptation technique for efficient tuning of LLMs. It builds upon the DyLoRA (Valipour et al. 2023) method, which enables training across a spectrum of ranks dynamically, and combines it with quantization techniques from QLoRA (Dettmers et al. 2024). The core principle is to allow the model to finetune on a set of predefined ranks and then select the optimal rank for inference, achieving efficiency without compromising performance. Mathematically, the forward pass is given by $h = W_{\text{NF4}}^{\text{DDequant}} x + \alpha \sum_{b=1}^{r} (W_{\text{up}})_{:,b} (W_{\text{dw}})_{b,:} x$, where $W_{\text{NF4}}^{\text{DDequant}}$ is the dequantized pretrained weight, x is the input, α is the LoRA scalar, r is the sampled rank, and W_{up} and W_{dw} are the up- and down-projection matrices, respectively. This approach reduces memory usage during training and inference, making it suitable for large-scale LLMs. **BitDelta** (Liu et al. 2024d) is an efficient post-training quantization method for compressing large language models after fine-tuning. The core idea is to represent the fine-tuning induced weight delta, $\Delta = W_{\text{fine}} - W_{\text{base}}$, where W_{fine} is the weight matrix of the fine-tuned model and W_{base} is the base pre-trained model's weight, using only 1 bit. This is achieved by quantizing Δ to its sign bits and a trainable scaling factor α, resulting in $\hat{\Delta} = \alpha \odot \text{Sign}(\Delta)$. The scaling factor is initialized to minimize the L2 norm of the error and further refined through distillation to align the quantized model's output with the original fine-tuned model. This approach dramatically reduces memory requirements and can enhance inference speed, with minimal impact on performance.

3.6 Multi-task PEFT

The previously introduced PEFT methods were mainly designed for single downstream task. This section focuses on PEFT for multi-task learning. Figure 11 illustrates three multi-task PEFT approaches: AdaMix (Adapter-based), ATTEMPT (Soft Prompt-based), and MOELoRA (LoRA-based).

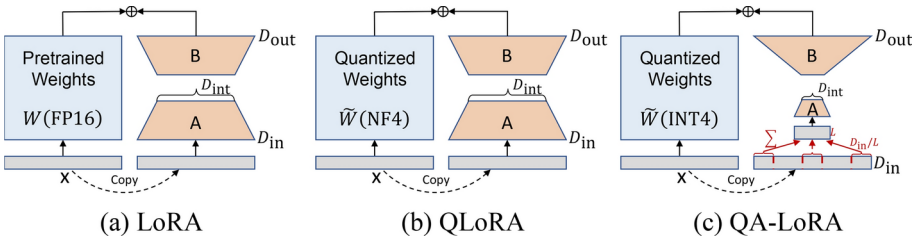


Fig. 10 Illustration of the difference among LoRA, QLoRA and QA-LoRA

3.6.1 Adapter-based

AdapterFusion (Pfeiffer et al. 2020) employs a two-stage approach to transfer learning, where it first extracts knowledge into task-specific adapters and then composes this knowledge in a separate step to exploit multi-task representations without destructive interference. **AdaMix** (Wang et al. 2022d) integrates multiple adaptation modules within each Transformer layer of a pre-trained language model, enabling efficient tuning with a mixture of these modules while maintaining most of the model's weights unaltered. **PHA** Zhao et al. (2023) leverages an instance-dense retriever and a prototypical hypernetwork to efficiently generate task-specific adapter layers by retrieving prototype embeddings and feeding them into the hypernetwork, enabling sample-efficient multi-task learning and new task generalization. **AdapterSoup** (Chronopoulou et al. 2023) improves the generalization of pretrained language models to new domains by averaging the weights of adapters trained on different domains, without the need for additional training or increasing inference cost. **MerA** (He et al. 2023b) efficiently incorporates pretrained adapters into a single model through model fusion, aligning the parameters via optimal transport based on weights and activations to enhance performance in few-shot learning scenarios. **Hyperformer** (Mahabadi et al. 2021) integrates hypernetwork-based adapter layers into a transformer model, enabling the model to share knowledge across tasks while adapting to each individual task through task-specific adapters generated by shared hypernetworks.

3.6.2 Soft prompt-based

SPoT (Soft Prompt Transfer) (Vu et al. 2022) leverages soft prompts to adapt pre-trained language models efficiently. It first trains a soft prompt p on one or more source tasks, where $p \in \mathbb{R}^d$ represents a sequence of continuous vectors with dimensionality d. This learned prompt is then used to initialize the prompt for a target task, facilitating transfer learning. SPoT significantly improves upon the performance of prompt tuning and matches or outperforms full model fine-tuning while using significantly fewer task-specific parameters. **ATTEMPT** (ATTEntional Mixtures of Prompt Tuning) (Asai et al. 2022) leverages pre-trained soft prompts $P_1, \ldots, P_t$ for different high-resource tasks and a new target prompt P_{target}. An attention module G computes attention scores between input X and each prompt token to produce an instance-wise prompt $P_{\text{instance}} = \sum_{j=1}^{t+1} a_j P_j$, where a_j represents the attention weight for prompt P_j. Only P_{target} and G are updated during training, keeping the original language model frozen. This approach is parameter-efficient and flexible for multi-task learning. **MPT** (Multitask Prompt Tuning) (Wang et al. 2022e) is a method for efficient transfer learning of LLMs across multiple downstream tasks. The core idea is to distill

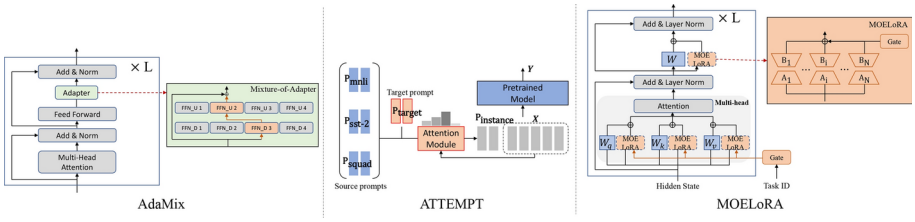


Fig. 11 Illustration of three representative multi-task PEFT methods: AdaMix (Adapter-based), ATTEMPT (Soft Prompt-based), and MOELoRA (LoRA-based)

knowledge from multiple task-specific source prompts into a single transferable prompt, P^*, which is then adapted to each target task with minimal additional parameters. The prompt for each source task is decomposed into a shared matrix P^* and a low-rank task-specific matrix $W_k = u_k \otimes v_k^T$, where u_k and v_k are task-specific vectors. This decomposition is learned through a knowledge distillation process that minimizes the KL-divergence between teacher and student prompts, L_{Logits}, and an additional mean squared loss on the hidden states, L_{Hidden}. The total training loss is $L_{\text{Total}} = L_{\text{PLM}} + \lambda(L_{\text{Logits}} + L_{\text{Hidden}})$, where L_{PLM} is the task loss and λ balances the distillation impact. The innovation lies in leveraging cross-task knowledge within a parameter-efficient framework, which outperforms full finetuning with far fewer task-specific parameters. **IPT** (Intrinsic Prompt Tuning) (Qin et al. 2021) is a method to reparameterize the adaptation of pre-trained language models to various tasks within a low-dimensional intrinsic task subspace. The key idea is to decompose the soft prompts P for multiple NLP tasks into a shared, lower-dimensional space using an auto-encoder with projection $\text{Proj}(\cdot)$ and back-projection $\text{Projb}(\cdot)$ functions. The auto-encoder is trained to minimize the reconstruction loss $L_{AE} = ||P^* - P||_2^2$, where $P^* = \text{Projb}(\text{Proj}(P))$. The intrinsic dimension d_I determines the size of this subspace. After finding the subspace, IPT tunes only d_I parameters to adapt PLMs to new tasks or data, suggesting that the adaptations can be generalized across tasks by optimizing a small set of free parameters in a unified subspace. **TPT** (transferable prompt tuning) (Su et al. 2021) investigates transferring soft prompts across tasks and models to improve prompt tuning (PT) efficiency. Soft prompts $P = \{p_1, p_2, \ldots, p_l\}$, where $p_i \in \mathbb{R}^d$ and d is the input dimension, are prepended to input sequences $X = \{x_1, x_2, \ldots, x_n\}$. The objective is to maximize the likelihood $L = p(y|P, x_1, \ldots, x_n)$ of generating desired outputs y, with P being the only trainable component. Transferability is explored through initializing with similar tasks' prompts and using a cross-model projector. The overlapping rate of activated neurons is found to be a strong indicator of transferability.

3.6.3 LoRA-based

LoRAHub (Huang et al. 2023a) is a dynamic composition of multiple LoRA modules, represented as $\hat{m} = (w_1 A_1 + w_2 A_2 + \ldots + w_N A_N)(w_1 B_1 + w_2 B_2 + \ldots + w_N B_N)$, followed by a gradient-free optimization to determine the coefficients w_i that best adapt the combined module for performance on new, unseen tasks. **MOELoRA** (Liu et al. 2023b) integrates a Mixture-of-Experts (MOE) model with trainable experts $\{E_i\}_{i=1}^N$, each consisting of a pair of low-rank matrices $B_i \in \mathbb{R}^{d_{in} \times r}$ and $A_i \in \mathbb{R}^{r \times d_{out}}$, along with a task-motivated gate function that outputs expert weights ω_{ji} for task T_j, to efficiently fine-tune LLMs for multi-task medical applications while maintaining a compact set of trainable parameters. **L-LoRA** (Linearized LoRA) (Tang et al. 2023) is a novel partial linearization method for PEFT models, which enhances weight disentanglement and improves multi-task fusion capability with a low computational cost overhead by linearizing only the adapter modules and applying model fusion algorithms over the linearized adapters. **MTLoRA** (Agiza et al. 2024) revolves around the use of Task-Agnostic and Task-Specific Low-Rank Adaptation modules to efficiently adapt a shared transformer backbone for multiple downstream tasks in a Multi-Task Learning architecture, balancing between learning shared features and those specific to individual tasks.

4 Applications of PEFT

This section presents a comprehensive overview of PEFT methodologies specifically developed for several prominent applications, categorized as follows: **PEFT in Vision Models** (Sect. 4.1), which primarily focuses on adapting pretrained vision models to specialized computer vision tasks (e.g., image classification, image segmentation, object detection, and depth estimation); **PEFT in Diffusion Models** (Sect. 4.2), which addresses the adaptation of diffusion models for vision generation tasks; and **PEFT in MLLM** (Sect. 4.3), which emphasizes training model connectors on domain-specific datasets to bridge multimodal data discrepancies while maintaining input consistency for LLMs. For a structured overview of these applications and their corresponding recommended PEFT techniques, refer to Fig. 12.

4.1 PEFT in vision models

Over the past decade, deep learning has achieved significant advancements in the field of computer vision, particularly with the introduction of the ImageNet dataset and the widespread adoption of the pre-training-fine-tuning paradigm based on pretrained vision models (PVMs). Numerous studies have shown that better ImageNet pre-training performance typically leads to improved performance on downstream tasks. As visual pre-trained models continue to evolve, especially with the introduction of Vision Transformer (ViT) architectures, the scale of model parameters has increased significantly, highlighting the inefficiencies of traditional full fine-tuning methods in terms of parameter efficiency. To address these issues and improve parameter efficiency during the fine-tuning process of PVMs, various PEFT methods have emerged. These methods have demonstrated their advantages across multiple domains, including image classification, dense prediction, video analysis, and 3D point cloud analysis. This section will focus on the application of PEFT methods in image classification and dense prediction tasks.

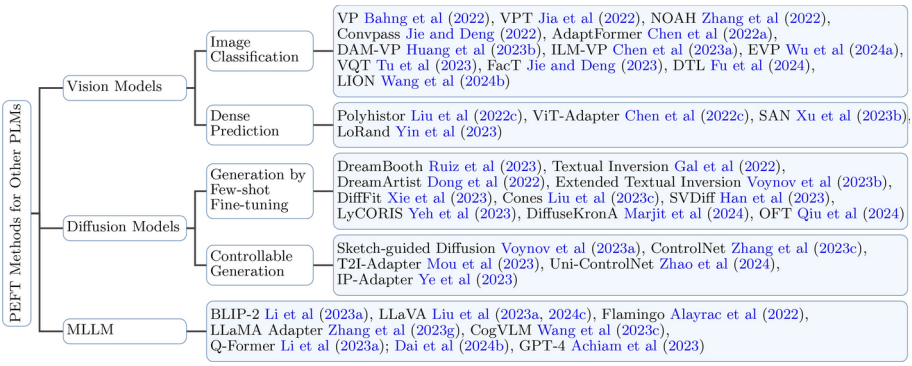


Fig. 12 Taxonomy of PEFT methods for vision models, diffusion models and MLLM

4.1.1 Image classification

In this subsection, we introduce PEFT methods for image classification tasks in vision models. Figure 13 illustrates the principles of three representative PEFT methods discussed in this subsection.

VP (Bahng et al. 2022) investigates visual prompting as a means to adapt large-scale pre-trained models for new tasks without updating model parameters. A single image perturbation (δ) is learned such that when added to input images (x), the prompted image ($x' = x + \delta$) steers the model's prediction towards a target task. This method is akin to adversarial reprogramming, but it aims for constructive task adaptation. Its effectiveness is demonstrated through experiments, which show competitive performance compared to linear probes. Notably, the approach is input-agnostic and dataset-wide. **VPT** (Visual Prompt Tuning) (Jia et al. 2022) adapts pre-trained vision Transformers for downstream tasks by introducing task-specific, learnable parameters ($P = \{p_k \in \mathbb{R}^d | k \in \mathbb{N}, 1 \leq k \leq m\}$) into the input sequence, while keeping the backbone of the model frozen. Here, d represents the dimensionality of the input features, while m signifies the total number of prompts. These prompts P are prepended to the input sequence of each Transformer layer and learned alongside a linear classification head during fine-tuning. **NOAH** (Neural prOmpt seArcH) (Zhang et al. 2022) automatically searches for the optimal design of prompt modules for large vision models through Neural Architecture Search (NAS). NOAH encompasses three prompt modules: Adapter, LoRA, and VPT, each inserted into Transformer blocks. The search space includes parameters like embedding dimensions $D = \{5, 10, 50, 100\}$ and depths

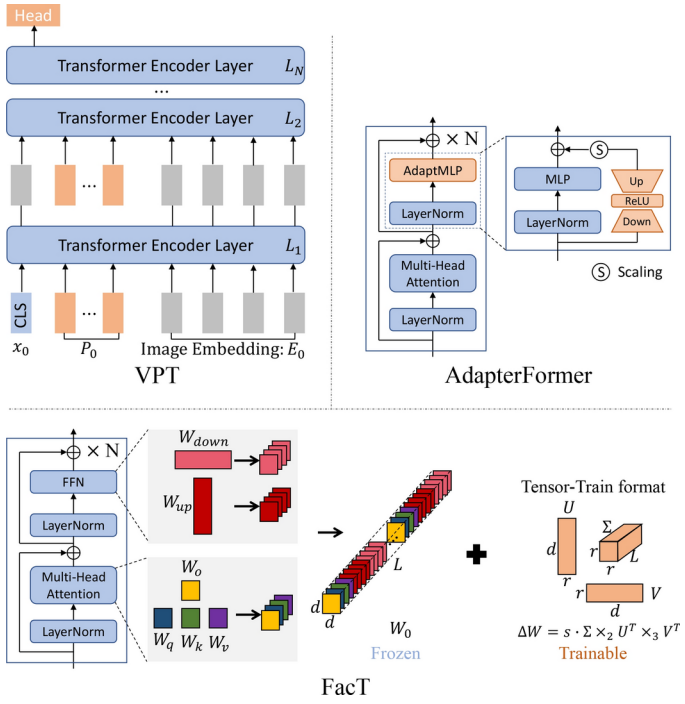


Fig. 13 Illustration of the principles of three PEFT methods for image classification: VPT (Soft Prompt-based), AdapterFormer (Adapter-based), and FacT (LoRA-based). $\times_i$ in FacT is mode-i product

$L = \{3, 6, 9, 12\}$, determining the range of applications. An AutoFormer-based one-shot NAS algorithm is employed to select the best configuration for each downstream dataset. **Convpass** (Jie and Deng 2022), convolutional bypasses for ViTs, to serve as adaptation modules during finetuning. Convpass, introduced as a parallel convolutional bottleneck block to the Multi-Head Self-Attention (MHSA) or MLP blocks, "bypasses" the original ViT block. For a ViT layer, the input sequence $X \in \mathbb{R}^{N \times d}$ is processed through Convpass, reconstructing the spatial structure of the token sequence. During finetuning, only Convpass modules and the classification head are updated. Convpass leverages the inductive bias of convolutional layers, enhancing its suitability for visual tasks, particularly in low-data scenarios. **AdaptFormer** (Chen et al. 2022a) is a lightweight module designed for efficient fine-tuning of pre-trained ViTs on diverse visual recognition tasks. It introduces additional trainable parameters, consisting of two fully connected layers $\mathrm{FC}_1, \mathrm{FC}_2$, a non-linear activation function (σ), and a scaling factor (α). These components are placed in parallel with the feed-forward network (FFN) of the original ViT. The learnable parameters of AdaptFormer are updated during the fine-tuning phase, while the pre-trained ViT parameters remain frozen. This design enables AdaptFormer to enhance the transferability of ViTs with minimal parameter updates, thereby improving scalability and performance on various visual tasks. **DAM-VP** (Diversity-Aware Meta Visual Prompting) (Huang et al. 2023b) partitions a dataset into homogeneous subsets based on diversity, optimizing a unique prompt for each subset. Prompts are initialized with a meta-prompt learned across multiple datasets, improving convergence speed and performance. During inference, the appropriate prompt is selected based on the feature distance between input and subset prototypes. Formally, for a dataset D divided into K subsets $D_1, D_2, ..., D_K$, the optimal prompts $p_1^*, ..., p_K^*$ are found by minimizing the cross-entropy loss:

$$p_1^*, ..., p_K^* = \arg \min_{p_1, ..., p_K} \sum_{k=1}^{K} \sum_{x \in D_k} L_{\mathrm{CE}}(M(x + p_k), y), \tag{33}$$

where p_k is the prompt for subset D_k, M is the pre-trained model, x is an input image, y is the ground truth label, and L_{CE} is the cross-entropy loss function. **ILM-VP** (Chen et al. 2023a) is an iterative label mapping-based visual prompting method. It optimizes the mapping between source and target labels to improve the accuracy of reprogramming pre-trained models for new tasks. The key equation is:

$$\min_{\delta} \sum_{yt \in Ttr} \min_{ys \in Ss} L(f_\theta(x + \delta), ys; yt), \tag{34}$$

where δ is the visual prompt, L is the cross-entropy loss, f_θ is the pre-trained model, x is the input image, *Ttr* is the target training set, *Ss* is the set of source labels, and *ys* and *yt* are the source and target labels, respectively. ILM-VP enhances interpretability by providing meaningful mappings. **EVP** (Enhanced Visual Prompting) (Wu et al. 2024a) is a method for adapting pre-trained models to downstream tasks without substantial parameter updates. Instead of directly combining the prompt P and the image I, they shrink I and pad P around it, ensuring independence. They also reintroduce input diversity and gradient normalization techniques, originally used in adversarial example generation, to improve the optimization

and generalizability of the prompt. This approach outperforms linear probing and matches fully fine-tuning in some cases, with significantly fewer parameters. **VQT** (Visual Query Tuning) (Tu et al. 2023) leverages learnable "query" tokens in each Transformer layer to summarize intermediate features effectively. VQT introduces a set $Q = \{q_1, q_2, \ldots, q_n\}$ where $q_i \in \mathbb{R}^d$ represents the i-th query token with d being the feature dimension. These queries interact with the intermediate features $X \in \mathbb{R}^{N \times d}$ through the attention mechanism, where N is the number of tokens. The output $Z = \{z_1, z_2, \ldots, z_n\}$ summarizes the layer's information, with z_i denoting the summary for q_i. This enables efficient transfer learning with memory and parameter savings. **FacT** (Jie and Deng 2023) is a method for efficient fine-tuning of pre-trained ViTs by updating only a fraction of parameters. The key idea is to tensorize the weights of ViT into a 3D tensor and decompose the weight increments into lightweight factors. During fine-tuning, only these factors are updated and stored. Mathematically, if ΔW represents the increment of a weight matrix W, then ΔW is approximated as $\Delta W \approx A \times B$, where A and B are the decomposed factors. A and B are learned during fine-tuning, reducing storage requirements. **DTL** (Disentangled Transfer Learning) (Fu et al. 2024) addresses the inefficiency of Parameter-Efficient Transfer Learning (PETL) methods in GPU memory usage. DTL employs a Compact Side Network (CSN) to disentangle trainable parameters from the backbone. CSN uses low-rank linear mappings to extract and reintegrate task-specific information. Formally, given a backbone with N blocks, the output z_{i+1} of the i-th block is updated as $z'_{i+1} = z_{i+1} + \theta(h_{i+1})$ for $i \geq M$, where θ is a non-linear activation function, and h_{i+1} captures the task-specific information extracted by CSN. This disentanglement significantly reduces GPU memory footprint and trainable parameters while maintaining or improving accuracy. **LION** (impLicit vIsion prOmpt tuNing) (Wang et al. 2024b) inserts two equilibrium implicit layers (P_1, P_2) at the start and end of a frozen pre-trained backbone (θ). P_1 and P_2 are defined as:

$$P_1 = f_{eq}^{(1)}(x; \phi_1), \quad P_2 = f_{eq}^{(2)}(z; \phi_2), \tag{35}$$

where x is the input, z is the output of the backbone, and ϕ_1, ϕ_2 are parameters of the implicit layers. f_{eq} denotes the equilibrium function. To reduce computational burden, parameters are pruned based on the lottery ticket hypothesis. LION adapts the backbone to downstream tasks efficiently with minimal parameter updates.

4.1.2 Dense prediction

Dense prediction, encompassing tasks such as image segmentation, object detection, depth estimation, etc., is another crucial task in the field of 2D vision. Unlike image classification tasks, which typically generate a single prediction label for an entire image, dense prediction tasks require making predictions for every pixel in the image, usually resulting in an output image with the same resolution as the input image. Fine-tuning pre-trained models from image classification is a common approach for dense prediction tasks. With the application of PEFT methods in vision tasks, various PEFT methods tailored for dense prediction tasks have been proposed. Figure 14 illustrates a representative PEFT method for dense prediction.

Polyhistor (Liu et al. 2022a) employs a strategy of hypernetworks that are broken down into components, along with scaling kernels applied at each layer, to facilitate the sharing

of information across various tasks efficiently and with a minimal number of parameters. In this approach, the weight matrix of each adapter, denoted as W, is decomposed into two distinct elements: a template kernel T and a scaling kernel S. The weight matrix is then reconstructed through the Kronecker product of these two kernels, represented as $W = T \otimes S$. This method effectively reduces the number of parameters required while still preserving the level of accuracy in the system. **ViT-Adapter** (Chen et al. 2022c) leverages the inherent representation power of a plain ViT backbone and augments it with an adapter that incorporates image-specific inductive biases during fine-tuning. This enables the model to capture high-frequency details crucial for tasks like object detection and segmentation. **SAN** (Side Adapter Network) (Xu et al. 2023b) decouples mask proposal generation and class recognition for open-vocabulary semantic segmentation. A lightweight side network is attached to a frozen CLIP model, predicting mask proposals and attention bias to guide CLIP's recognition of the mask's class. This design leverages CLIP's robustness while minimizing additional parameters and computational cost. The attention bias is applied in CLIP's attention mechanism $\text{Attention}(Q, K, V, \text{bias})$, where Q, K, and V represent query, key, and value vectors, enhancing CLIP's awareness of the proposed regions. **LoRand** (Yin et al. 2023) adds lightweight, low-rank adapter modules to a pre-trained vision model, such as the Swin Transformer, without updating the original model's parameters. These adapters consist of multi-branch low-rank projections and non-linearities, enabling them to capture complex representations with minimal parameters. Specifically, for a backbone with parameters θ, LoRand trains a small subset ϕ $(1\% - 3\%)$ of θ, where $\phi \subset \theta$, achieving competitive performance with full fine-tuning while significantly reducing the number of trainable parameters.

4.2 PEFT in diffusion models

As diffusion models evolve, these models have now surpassed GANs as the mainstream method in the image generation domain. Given their success in image generation, their potential applications in video generation, 3D content generation, and speech synthesis are also becoming increasingly apparent. Additionally, many application domains involve fine-

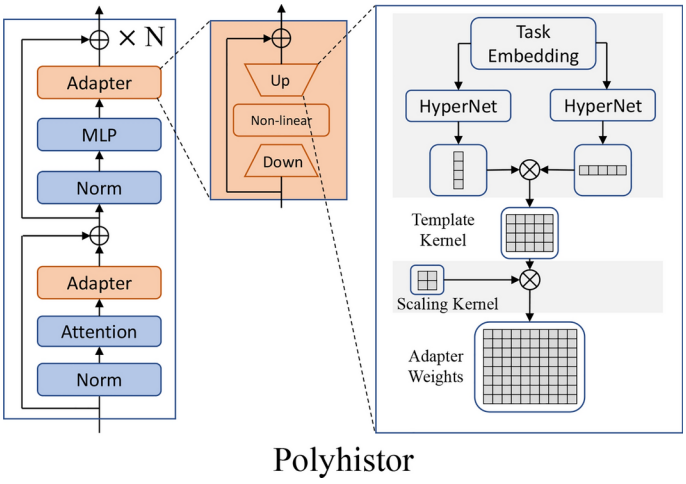


Fig. 14 Illustration of a representative PEFT method for dense prediction: Polyhistor

tuning diffusion models, including embedding personalized concepts in image generation, customizing generated images based on reference images, and training multi-view image generation capabilities based on pre-trained text-to-image diffusion models in the 3D content generation domain. Compared to the NLP field, research on PEFT for diffusion models is relatively scarce. Current research mainly focuses on two areas: generation by few-shot finetuning and controllable generation in image generation:

4.2.1 Generation by few-shot finetuning

Generation by few-shot finetuning involves providing a few images (or even just one) of an object or style, and fine-tuning the model on these images. This process allows the model to generate new images that reflect the unique characteristics of the provided examples.

DreamBooth (Ruiz et al. 2023) is a method for personalizing text-to-image diffusion models using just a few images of a subject. The technique fine-tunes a pre-trained model with a novel autogenous class-specific prior preservation loss, to bind a unique identifier to the subject and preserve class diversity. This enables generating photorealistic images of the subject in various scenes while maintaining key features. The fine-tuning process involves adjusting the model parameters based on input images and text prompts, leveraging the model's semantic prior and the new loss function to enhance subject fidelity and versatility in image synthesis. **Textual Inversion** (Gal et al. 2022) is a method that personalizes text-to-image generation by embedding unique concepts as new "pseudo-words" in the latent space of a pre-trained model. This allows intuitive composition into sentences guiding image creation, capturing both semantics and details without retraining the model. The innovation lies in optimizing a single word embedding to represent a concept through reconstruction, balancing distortion and editability. The method's strength is its simplicity and compatibility with existing models, while its limitation is the potential for less precise shape retention. **DreamArtist** (Dong et al. 2022) leverages positive–negative prompt-tuning to enable one-shot text-to-image generation. Given a reference image I, it learns a positive embedding S_p^* that captures the image's characteristics and a negative embedding S_n^* that rectifies deficiencies. S_p^* drives diverse generation, while S_n^* ensures corrections, improving controllability. The embeddings are combined through a fusion function $f_m(z_p, z_n)$ where z_p and z_n represent the latent representations of positive and negative prompts, respectively. This approach facilitates the synthesis of high-quality, diverse, and controllable images from a single reference. In paper (Voynov et al. 2023b), an **Extended Textual Conditioning (P+)** space is introduced for text-to-image generation, allowing for more granular control over image synthesis through per-layer textual prompts. The innovation, **Extended Textual Inversion**, inverts images into P+ space using a set of token embeddings, enhancing expressiveness and precision without compromising editability. This method is advantageous due to its faster convergence and the ability to achieve finer control over image attributes by leveraging the distinct sensitivities of U-net layers to shape or appearance. The downside includes imperfect concept reconstruction and the relatively slow inversion process. **DiffFit** (Xie et al. 2023) fine-tunes only the bias terms and introduces scaling factors γ in specific layers, initialized to 1.0, to adapt to new domains quickly. The method achieves significant training efficiency and reduced storage costs, with γ enhancing feature scaling for better adaptation. The efficacy is theoretically justified by analyzing the shift in distributions caused by the scaling factors. **SVDiff** (Han et al. 2023) is a method for fine-tuning

text-to-image diffusion models by adjusting the singular values (σ_i) of weight matrices (W), represented as $W = \sum_i \sigma_i u_i v_i^\top$, where u_i and v_i are the left and right singular vectors, respectively. This approach leads to a compact parameter space, reducing overfitting and model size ($\approx 2,200\times$ fewer parameters than DreamBooth). They also introduce CutMix-Unmix for improved multi-subject generation and a single-image editing framework. **LyCORIS** (Yeh et al. 2023) is an open-source library for fine-tuning Stable Diffusion models. It implements methods like LoRA, LoHa, LoKr, GLoRA, and $(IA)^3$. The library aims to simplify the integration and evaluation of these methods. A comprehensive evaluation framework is proposed, using metrics for concept fidelity, text-image alignment, diversity, and style preservation. Experiments highlight the nuanced impacts of hyperparameters and the suitability of different methods for specific tasks. **DiffuseKronA** (Marjit et al. 2024) utilizes a Kronecker product-based adaptation mechanism to efficiently fine-tune large diffusion models for personalized text-to-image generation. The method reduces the parameter count by applying truncated singular value decomposition on critical model layers, enabling subject-specific image synthesis with enhanced stability, interpretability, and text alignment. The approach offers a $\geq 50\%$ parameter reduction compared to state-of-the-art methods, with comparable or superior image quality. **OFT** (Orthogonal Finetuning) (Qin et al. 2024) is a method to adapt text-to-image diffusion models for downstream tasks without losing generative performance. OFT preserves the hyperspherical energy which characterizes neuron relationships by applying a layer-shared orthogonal transformation R to the pretrained weights W_0. This maintains the pairwise angles among neurons, crucial for semantic information. The transformation is constrained as $R^T R = RR^T = I$, ensuring minimal deviation from the original model. A variant, Constrained Orthogonal Finetuning (COFT), further limits angular deviation with $\|R - I\| \leq \epsilon$. The method aims to balance flexibility and stability in finetuning.

4.2.2 Controllable generation

Controllable generation primarily involves adding control sources beyond the prompt to guide the image generation. These control sources can include sketches, keypoints, or other forms of guidance to shape the generated output more precisely. A representative implementation of controllable generation method is shown in Fig. 15

Sketch-guided diffusion (Voynov et al. 2023a) is a method to guide pre-trained text-to-image diffusion models using spatial maps like sketches. It involves training a lightweight per-pixel multi-layer perceptron (MLP), named the latent guidance predictor (LGP), to map noisy image features to spatial maps. The LGP is trained on a small dataset, predicting spatial layouts from latent features $F(\mathrm{z}_t|\mathrm{c}, t)$ extracted from a denoising diffusion probabilistic model (DDPM) network, where z_t is a noisy image at timestep t, and c presents the conditioning text prompt. **ControlNet** (Zhang et al. 2023c) enhances pretrained text-to-image diffusion models by adding spatially localized conditions. For a neural block $F(x; \Theta)$ transforming input x to output y, ControlNet freezes Θ and introduces a trainable copy. Conditions c are injected through zero-initialized convolution layers (zero convolutions) ensuring no initial noise. $y_c = F(x, c; \Theta')$ represents the output with conditions, where Θ' denotes the updated parameters. This approach facilitates robust finetuning and sudden convergence. **T2I-Adapter** (Mou et al. 2023) enhances controllability of pre-trained text-to-image (T2I) models by learning lightweight adapter models that align the model's internal

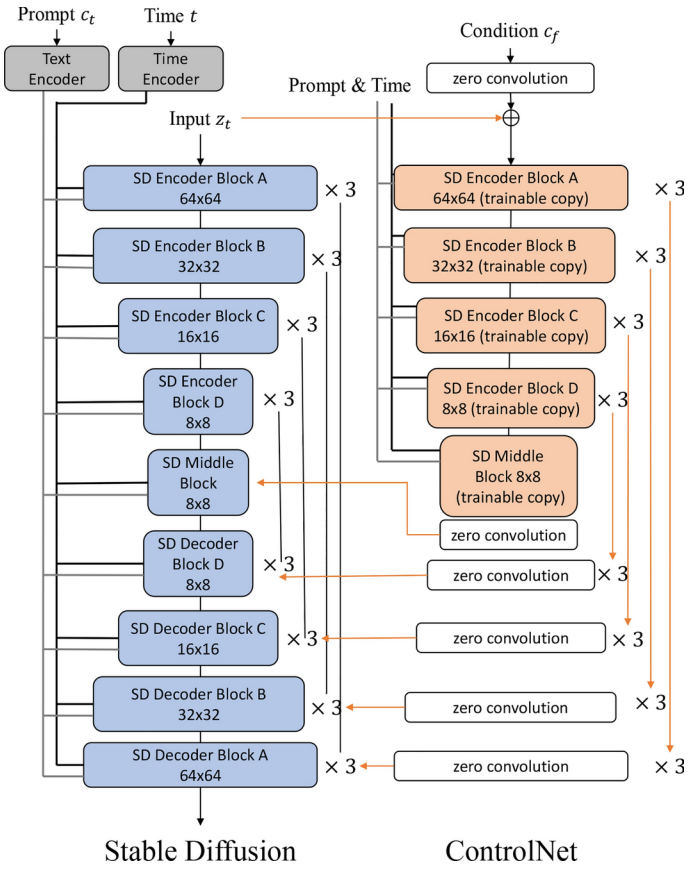


Fig. 15 Illustration of the principle of ControlNet, a representative implementation of a controllable generation method

knowledge with external control signals. This is achieved without modifying the original T2I model, allowing for granular control over generated images' structure and color. Mathematically, let $\mathcal{M}$ denote the pre-trained T2I model, $\mathcal{A}$ the adapter, and x_c the control signal (e.g., sketches, masks). The adapted model generates images x from text prompts t and control signals x_c as follows:

$$\mathrm{x} = \mathcal{M}_{\text{adapted}}(t, \mathrm{x}_c) = \mathcal{M}(t) + \omega \cdot \mathcal{A}(\mathrm{x}_c), \tag{36}$$

where ω is a weighting factor balancing the influence of the control signal. The adapter $\mathcal{A}$ is trained to translate x_c into a form that can steer $\mathcal{M}$ towards desired outputs, enabling precise control. **Uni-ControlNet** (Zhao et al. 2024) integrates diverse control signals into pre-trained text-to-image (T2I) diffusion models through two lightweight adapters, facilitating efficient and composable control. It employs a multi-scale condition injection strategy, using Feature Denormalization (FDN) to modulate noise features with local conditions:

$$\mathrm{F}_{\mathrm{DN}r}(\mathrm{Z}_r, c_l) = \mathrm{norm}(\mathrm{Z}_r) \cdot (1 + \mathrm{conv}_\gamma(\mathrm{zero}(h_r(c_l)))) + \mathrm{conv}_\beta(\mathrm{zero}(h_r(c_l))), \tag{37}$$

where Z_r are noise features at resolution r, c_l are concatenated local conditions, h_r extracts features at resolution r, and conv_γ converts features into modulation coefficients. Global controls are aligned with text embeddings via a condition encoder. $h_g(c_g) \rightarrow K$ global tokens Here, c_g is the global condition, and K is the number of global tokens. **IP-Adapter** (Ye et al. 2023) enables pretrained text-to-image models to utilize image prompts effectively. It introduces a decoupled cross-attention mechanism, adding extra layers dedicated to image features while keeping the original text-focused layers intact. During training, these new layers learn to process image embeddings extracted by a CLIP encoder. At inference, the image and text features are processed separately then combined, improving controllability and fidelity of generated images. The core equation is:

$$\hat{\epsilon}_\theta(x_t, c, t) = w\epsilon_\theta(x_t, c, t) + (1 - w)\epsilon_\theta(x_t, t), \tag{38}$$

where $\hat{\epsilon}_\theta(x_t, c, t)$ is the predicted noise, w is the guidance scale adjusting the influence of condition c, $\epsilon_\theta(x_t, c, t)$ is the conditional noise prediction, and $\epsilon_\theta(x_t, t)$ is the unconditional prediction.

4.3 PEFT in MLLM

The PEFT of MLLM primarily focuses on the model connector. It is because maintain consistency for both multimodal and textual data is challenging. As a consequence, a modal connector is serially connected right before the LLM, converting multimodal embeddings into understandable text prompt tokens for the LLM. Training the model connector on PEFT dataset bridges the gap between different modal data while ensuring consistency in the input to the LLM. As a representative PEFT approach within the MLLM framework, the schematic diagram of LLaMA-Adapter (Zhang et al. 2023g) is illustrated in Fig. 16.

Generally, the parameter scale of the model connector will not be very large, much smaller than the prevalent LLMs. Therefore, full-parameter training instead of PEFT is more prevalent for model connector. Studies of the model connector primarily focus on the structural design, which will be dedicated to improving the training performance. A classic design of the modal connector involves employing a set of learnable query tokens to extract information in a query-based manner, a technique first introduced in **BLIP-2** (Li et al. 2023a) and subsequently adopted by various projects (Dai et al. 2024b). These query-based approaches, reminiscent of Q-Former-style methods, condense visual tokens into a smaller

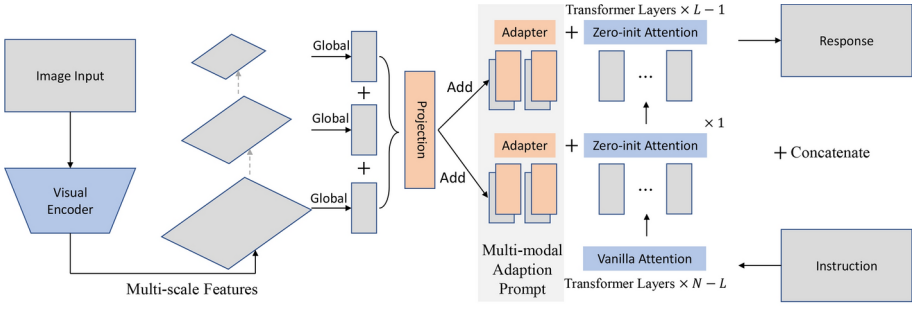


Fig. 16 Illustration of the principle of LLaMA-Adapter, which is a representative PEFT method in MLLM

set of representation vectors. In the meantime, some methods utilize an MLP-based interface to bridge the modality gap. For instance, the **LLaVA** series (Liu et al. 2023a, 2024c) employs one or two linear MLPs to project visual tokens and align feature dimensions with word embeddings. In feature-level fusion, additional modules facilitate deep interaction and fusion between text features and visual features. For example, **Flamingo** (Alayrac et al. 2022) introduces extra cross-attention layers between the frozen Transformer layers of LLMs, enhancing language features with external visual cues. In addition, adapters and prompt embedding are also applied to add learnable parameters to fill the gap, such as **LLaMA Adapter** (Zhang et al. 2023g) and **CogVLM** (Wang et al. 2023c).

Figure 17 illustrates the concrete structures of the two designs. The first one, pioneered by the LLaVA series, is characterized by its simplicity. As highlighted by Liu et al. (2024c), an MLP composed of basic linear layers is adept at transforming multimodal embeddings into LLM prompt tokens.

In contrast, the second paradigm, known as the **Q-Former** (Li et al. 2023a; Dai et al. 2024b), introduces a transformer neural network for modal information conversion. Unlike traditional approaches of directly applying self-attention on input embeddings, Q-Former employs a set of trainable query tokens. This approach bears resemblance to LLM PEFT methods such as prefix-tuning and p-tuning, which incorporate external trainable embedding tokens. However, the key distinction lies in how these methods handle the tokens: prefix-tuning and p-tuning append them to the input text tokens to form a comprehensive LLM input, while Q-Former accepts the query tokens as the primary input.

From both the structural design and training intricacies, it becomes evident that Q-Former is considerably more complicated compared to the MLP-based LLaVA. However, this complexity comes with its advantages. A comprehensive transformer network like Q-Former enables the execution of numerous pre-trained tasks, facilitating explicit alignment between non-textual and textual modalities. This, in turn, reduces the quality requirements on the multimodal data. Nevertheless, LLaVA, as detailed by Liu et al. (2024c), which incorporates **GPT-4** (Achiam et al. 2023) as the LLM, reports a slight performance improvement

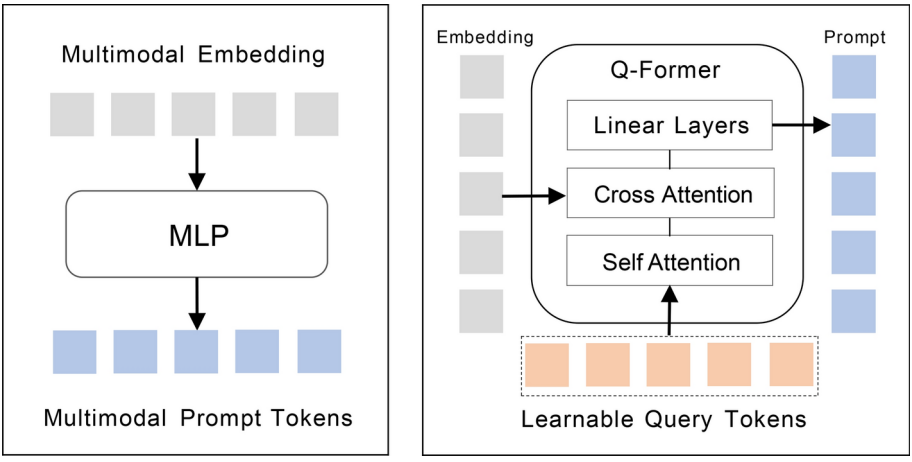


Fig. 17 Modal connector design: this figure shows two different mainstream design of the modal connector in MLLM. The first one, a simple MLP for converting the modal. The second one, layers with cross attention and query tokens for training

over BLIP-2. This is largely attributed to the inherent superiority of GPT-4 over BLIP-2's Flan-T5 across various aspects. Specifically, GPT-4 possesses innate multimodal reasoning capabilities, a feature lacking in Flan-T5. This observation underscores the fact that a comprehensive modal connector design may not be necessary when the LLM itself possesses significant power and capabilities.

To further quantify the performance of different PEFT methods in specific applications, we present Table 7, which compares various methods based on key metrics such as accuracy and the number of trainable parameters across multiple benchmark tasks. Since existing literature does not provide detailed computational cost analysis, we use the number of trainable parameters as an approximate measure of computational efficiency, serving as a practical proxy for resource consumption across different PEFT methods. As shown in Table 7, compared to full fine-tuning, PEFT methods in specific applications significantly reduce the number of trainable parameters while maintaining competitive performance. These results highlight the advantage of PEFT methods in various applications, where they enable efficient adaptation of large models with lower computational and storage costs while preserving task-specific performance.

5 Future directions

In this section, focusing on potential issues with existing PEFT techniques and aspects that have not received sufficient attention, we propose a series of possible research directions. These directions encompass **task**, **data**, **model**, **learning mechanisms**, and **fundamental flaws**.

1. *PEFT methods for multi-objective tasks:* Current PEFT methods mainly focus on optimizing for single objectives (e.g., task accuracy), but real-world applications often require balancing multiple objectives (e.g., privacy, fairness, latency). For example, in healthcare, models must preserve patient privacy while maintaining diagnostic accuracy. Existing methods like LoRA or Adapters lack explicit mechanisms to handle such trade-offs. In recent work (Yang et al. 2024), the authors addressed the program repair task by incorporating a dual-objective optimization framework, wherein the two objectives were combined through linear weighting with manually predefined coefficients to formulate the model's loss function. Although this study presents a straightforward and effective approach to PEFT for multi-objective tasks, determining the optimal weighting coefficients remains non-trivial. This limitation highlights the need for developing more flexible and task-adaptive methodologies to enhance the robustness and generalizability of such approaches.
2. *PEFT methods in multimodal learning:* Multimodal models (e.g., vision-language models) face unique challenges in aligning heterogeneous data streams (text, images, audio). Current PEFT methods (e.g., adapters) are primarily designed for unimodal LLMs, leading to suboptimal performance in tasks like visual question answering. Recent work on CLIP adaptations (Zavras et al. 2024) highlights the need for modality-specific parameter-efficient tuning to bridge domain gaps. Multimodal learning has emerged as one of the most prominent research topics in contemporary machine learning. However, significant challenges persist in effectively integrating cross-modal

information through PEFT approaches, particularly in achieving optimal inter-modal alignment and representation learning while maintaining computational efficiency.

3. *Automated design of adapter modules:* Adapter architectures (e.g., bottleneck layers) rely on manually tuned hyperparameters (e.g., dimension, placement), which limits scalability. Neural Architecture Search (NAS) techniques (Xu and Wen 2024) could automate adapter design, optimizing for both parameter efficiency and task performance. However, the extensive design space of adapter modules significantly compromises the efficiency of NAS approaches. This limitation necessitates further investigation into more efficient and flexible automated design methodologies that can navigate the complex parameter space effectively while maintaining architectural optimality.
4. *Heuristic search strategies for hybrid PEFT methods:* Hybrid methods (e.g., combining LoRA and adapters) often rely on trial-and-error combinations, lacking principled strategies. For example, in paper (Chen et al. 2023b), the authors, under a predefined design space, conduct numerous experiments to determine an ideal hybrid strategy. However, the optimal hybrid strategy may not be included within this artificially predefined design space. Therefore, introducing heuristic search strategies to find the best hybrid strategy is a promising direction for future research.
5. *Continual learning for PEFT methods:* Deployed models must adapt to evolving data distributions (e.g., user preferences in chatbots). Traditional PEFT lacks mechanisms to prevent catastrophic forgetting. Current work (Wei et al. 2024) proposed a method for task-free online continual learning that dynamically adapts pretrained Vision Transformer models by adding new low-rank adaptation parameters when the loss surface plateaus, indicating data distribution shifts, and uses online weight regularization to mitigate catastrophic forgetting. The experimental results presented in this paper demonstrate significant performance improvements through the application of LoRA, establishing a valuable reference framework for investigating continual learning paradigms in other types of PEFT methodologies.
6. *Improving the calibration of fine-tuned LLMs:*To date, numerous PEFT approaches developed for the purpose of adeptly tailoring LLMs to downstream tasks have achieved notable advancements in computational and storage efficiency. Nonetheless, when subjected to fine-tuning on modest datasets, LLMs are often prone to overconfidence in their predictions (Jiang et al. 2021; Tian et al. 2023; Achiam et al. 2023). This phenomenon is especially pernicious for decision-making processes within safety-critical applications or domains where data is scarce, such as medical diagnostics, financial services, and experimental design (Singhal et al. 2023; Lee et al. 2024; Huang et al. 2024b). Hence, there exists an exigent demand for the formulation of strategies aimed at refining the calibration of fine-tuned LLMs, ensuring that their predictive outputs are not only dependable but also robust.
7. *Differential privacy for PEFT methods:* Different downstream tasks often involve varying levels of sensitve and personal data, which further emphasizes the need for privacy in LLM fine-tuning, particularly with PEFT methods. The integration of LLM fine-tuning and differential privacy holds significant promise for future research. However, existing differential privacy techniques, such as DP-SGD (Abadi et al. 2016) and DP-AdamW (Li et al. 2021), often result in limited performance and substantial computaitional cost. Therefore, future reasearch should focus on developing methods that preserve privacy while simultaneously optimizing performance and minimizing

Table 7 Performance of PEFT methods in specific applications

Task	Model	PEFT method	#TPs (M)	Result										
				CIFAR 100	CIFAR 10	Flowers	Food	EuroSAT	SUN	DMLab	SVHN	Pets	DTD	RESISC	CLEVR
Image Classification	CLIP	FT	151.28	82.1	95.8	97.4	87.8	99	79	63.5	95.7	88.5	72.3	98.1	94.4
		VP	0.07	75.3	94.2	62	83.2	95.6	68.4	41.9	88.4	86.5	57.1	84.1	81.4
		VPT	0.064	76.6	95	76.2	84.7	94.6	69.3	48.4	86.1	92.1	61.6	84.3	58.6
		EVP	0.062	81.2	96.6	82.3	84.1	97.6	71	62.3	90.5	90	68.4	89.7	75.9

Task	Model	PEFT method	#TPs (M)	Seg.	H.Part	Sal.	Normals.
Dense Prediction	Swin Transformer -Tiny	Single-task FT	112.62	67.21	61.93	62.35	17.97
		Multi-task FT	30.06	68.71	62.13	64.18	17.35
		Bitfit	2.85	68.57	55.99	60.64	19.42
		Relative bias	2.64	63.51	52.35	57.74	21.07
		VPT-shallow	2.57	62.96	52.27	58.31	20.9
		VPT-deep	3.43	64.35	55.24	58.15	21.07
		PHM layer	3.14	68.55	56.28	60.35	19.23
		Compacter	2.78	68.38	56.69	59.47	19.54
		Compacter++	2.66	67.26	55.69	59.47	19.54
		LoRA	2.87	67.26	55.69	59.47	19.54
		Adapter	11.24	69.21	57.38	61.28	18.83
		Low-rank adapter	2.89	68.31	56.53	60.29	19.36
		Shared Adapter	4.74	70.21	59.15	62.29	19.26
		Hyperformer	75.32	71.43	60.73	65.54	17.77
		Polyhistor	8.96	70.87	59.54	65.47	17.47
		Polyhistor-Lite	2.96	70.24	59.12	64.75	17.4

Table 7 (continued)

Task	Model	PEFT method	#TPs (M)	Food	SUN	DF- 20 M	Caltech	CUB-Bird	ArtBench	Oxford Flowers	Standard Cars	Average FID
Generation-by Few-shot Finetuning	DiT-XL-2	FT	673.8	10.46	7.96	17.26	35.25	5.68	25.31	21.05	9.79	16.59
		Adapt-Parallel	4.28	13.67	11.47	22.38	35.76	7.73	38.43	21.24	10.73	20.17
		Adapt-Sequential	4.28	11.93	10.68	19.01	34.17	7	35.04	21.36	10.45	18.7
		BitFit	0.61	9.17	9.11	17.78	34.21	8.81	24.53	20.31	10.64	16.82
		VPT-Deep	2.81	18.47	14.54	32.89	42.78	17.29	40.74	25.59	22.12	26.8
		LoRA-R8	1.15	33.75	32.33	120.25	86.05	56.03	80.99	164.13	76.24	81.31
		LoRA-R16	2.18	34.34	32.15	121.51	86.51	58.25	80.72	161.68	75.35	81.31
		DiffFit	0.83	6.96	8.55	17.35	33.84	5.48	20.87	20.18	9.9	15.39

Task	Model	PEFT method	#TPs (M)	CLIP-T	CLIP-I
Controllable Generation	CLIP ViT-L/14	Uni-ControlNet (Global Control)	47	0.51	0.74
		T2I-Adapter (Style)	39	0.49	0.65
		ControlNet Shuffle	361	0.42	0.62
		IP-Adapter	22	0.59	0.83

All performance metrics are cited from prior published work (Wu et al. 2024a; Liu et al. 2022a; Xie et al. 2023; Mou et al. 2023). Metrics vary by task: 1. Image Classification: 12 datasets with CLIP. 2. Dense Prediction: 4 datasets with Swim Transformer-Tiny. 3. Generation by Few-shot Finetuning: 9 datasets with DiT-XL-2. 4. Controllable Generation: 2 datasets with CLIP ViT-L/14

computational costs. Additionally, exploring scalable, privacy preserving methods tailored to PEFT methods is essential. These advancements will enable secure and efficient fine-tuning of LLMs, ensuring robust privacy protections.

6 Conclusions

LLMs have garnered widespread attention due to their exceptional performance across a broad spectrum of natural language tasks, beginning with the release of ChatGPT in November 2022. These models have acquired the capability for general-purpose language understanding and generation by training billions of parameters on vast amounts of textual data, as predicted by scaling laws. Traditional full-parameter fine-tuning methods pose significant challenges when customizing these models for specific downstream tasks, particularly on hardware platforms with limited computational capabilities, due to their enormous parameter scale and computational demands. PEFT has emerged as an efficient method for adapting to various downstream tasks, minimizing the number of additional parameters introduced or the computational resources required, thereby enabling the fine-tuned model's performance to approach or even surpass that of full-parameter fine-tuning methods. This survey provides a systematic overview of the latest advancements in PEFT, encompassing introductions to classic pre-trained large models, classification and principle explanation of PEFT algorithms, applications of PEFT methods, and prospects for future research directions in PEFT. This survey not only offers readers a comprehensive and systematic organization of PEFT work but also inspires researchers in various fields to identify potential research directions in PEFT research, accelerating the research process of PEFT methods.

Author contributions I have read the Nature Portfolio journal policies on author responsibilities and submit this manuscript in accordance with those policies.

Funding This work was supported by the National Key Research and Development Program of China (Grant No. 2023YFE0108600), National Natural Science Foundation of China (Grant No. U22A6001), Shanghai Artificial Intelligence Laboratory (Grant No. P22KN00581) and "Pioneer" and "Leading Goose" R&D Program of Zhejiang (Grant No. 2024SSYS0002).

Data availability No, I do not have any research data outside the submitted manuscript file.

Declarations

Conflict of interest Yes, the authors have Conflict of interest as defined by Springer, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Open Access I confirm that I understand Artificial Intelligence Review is an open access journal that levies an article processing charge per articles accepted for publication. By submitting my article I agree to pay this charge in full if my article is accepted for publication.

Consent to publication The results/data/figures in this manuscript have not been published elsewhere, nor are they under consideration (from you or one of your Contributing Authors) by another publisher.

Third party material All of the material is owned by the authors and/or no permissions are required.

References

Abadi M, Chu A, Goodfellow I et al (2016) Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp 308–318

Achiam J, Adler S, Agarwal S et al (2023) GPT-4 technical report. arXiv preprint. arXiv:2303.08774

Aghajanyan A, Zettlemoyer L, Gupta S (2020) Intrinsic dimensionality explains the effectiveness of language model fine-tuning. arXiv preprint. arXiv:2012.13255

Aghajanyan A, Gupta A, Shrivastava A et al (2021) MUPPET: massive multi-task representations with pre-finetuning. arXiv preprint. arXiv:2101.11038

Agiza A, Neseem M, Reda S (2024) MTLORA: a low-rank adaptation approach for efficient multi-task learning. arXiv preprint. arXiv:2403.20320

Ahn J, Verma R, Lou R et al (2024) Large language models for mathematical reasoning: progresses and challenges. arXiv preprint. arXiv:2402.00157

Alayrac JB, Donahue J, Luc P et al (2022) FLAMINGO: a visual language model for few-shot learning. Adv Neural Inf Process Syst 35:23716–23736

Anil R, Borgeaud S, Wu Y et al (2023) Gemini: A family of highly capable multimodal models. arXiv preprint. arXiv:2312.11805 1

Ansell A, Ponti EM, Korhonen A et al (2021) Composable sparse fine-tuning for cross-lingual transfer. arXiv preprint. arXiv:2110.07560

Anthropic (Online) Claude. https://www.anthropic.com/claude. Accessed 11 Feb 2025

Aribandi V, Tay Y, Schuster T et al (2021) EXT5: towards extreme multi-task scaling for transfer learning. arXiv preprint. arXiv:2111.10952

Asai A, Salehi M, Peters ME et al (2022) Attempt: parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In: Proceedings of the 2022 conference on empirical methods in natural language processing, pp 6655–6672

Austin J, Odena A, Nye M et al (2021) Program synthesis with large language models. arXiv preprint. arXiv:2108.07732

Bach SH, Sanh V, Yong ZX et al (2022) Promptsource: an integrated development environment and repository for natural language prompts. arXiv preprint. arXiv:2202.01279

Bahng H, Jahanian A, Sankaranarayanan S et al (2022) Exploring visual prompts for adapting large-scale models. arXiv preprint. arXiv:2203.17274

Bai Y, Jones A, Ndousse K et al (2022a) Training a helpful and harmless assistant with reinforcement learning from human feedback. arXiv preprint. arXiv:2204.05862

Bai Y, Kadavath S, Kundu S et al (2022b) Constitutional ai: Harmlessness from ai feedback. arXiv preprint. arXiv:2212.08073

Baumgartner J, Zannettou S, Keegan B et al (2020) The pushshift reddit dataset. In: ICWSM. AAAI Press, pp 830–839

Bender EM, Gebru T, McMillan-Major A et al (2021) On the dangers of stochastic parrots: Can language models be too big? In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, pp 610–623

Bi X, Chen D, Chen G et al (2024) Deepseek llm: Scaling open-source language models with longtermism. arXiv preprint. arXiv:2401.02954

Biswas SS (2023) Role of Chat GPT in public health. Ann Biomed Eng 51(5):868–869

Bommasani R, Hudson DA, Adeli E et al (2021) On the opportunities and risks of foundation models. arXiv preprint. arXiv:2108.07258

Brown T, Mann B, Ryder N et al (2020) Language models are few-shot learners. Adv Neural Inf Process Syst 33:1877–1901

Cao J, Prakash CS, Hamza W (2022) Attention fusion: a light yet efficient late fusion mechanism for task adaptation in NLU. In: Findings of the Association for Computational Linguistics: NAACL 2022, pp 857–866
Chen S, Hou Y, Cui Y et al (2020) Recall and learn: fine-tuning deep pretrained language models with less forgetting. arXiv preprint. arXiv:2004.12651
Chen M, Tworek J, Jun H et al (2021) Evaluating large language models trained on code. arXiv preprint. arXiv:2107.03374
Chen S, Ge C, Tong Z et al (2022a) ADAPTFORMER: adapting vision transformers for scalable visual recognition. Adv Neural Inf Process Syst 35:16664–16678
Chen Y, Hazarika D, Namazifar M et al (2022b) Empowering parameter-efficient transfer learning by recognizing the kernel structure in self-attention. In: Findings of the Association for Computational Linguistics: NAACL 2022, pp 1375–1388
Chen Z, Duan Y, Wang W et al (2022c) Vision transformer adapter for dense predictions. arXiv preprint. arXiv:2205.08534
Chen A, Yao Y, Chen PY et al (2023a) Understanding and improving visual prompting: a label-mapping perspective. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 19133–19143
Chen J, Zhang A, Shi X et al (2023b) Parameter-efficient fine-tuning design spaces. arXiv preprint. arXiv:2301.01821
Chen L, Huang H, Cheng M (2023c) PTP: boosting stability and performance of prompt tuning with perturbation-based regularizer. arXiv preprint. arXiv:2305.02423
Chen W, Yin M, Ku M et al (2023d) THEOREMQA: a theorem-driven question answering dataset. In: EMNLP. Association for Computational Linguistics, pp 7889–7901
Chen Y, Fu Q, Fan G et al (2023e) Hadamard adapter: an extreme parameter-efficient adapter tuning method for pre-trained language models. In: Proceedings of the 32nd ACM international conference on information and knowledge management, pp 276–285
Chen X, Liu J, Wang Y et al (2024) SUPERLORA: parameter-efficient unified adaptation of multi-layer attention modules. arXiv preprint. arXiv:2403.11887
Cherti M, Beaumont R, Wightman R et al (2023) Reproducible scaling laws for contrastive language-image learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 2818–2829
Cho J, Lei J, Tan H et al (2021) Unifying vision-and-language tasks via text generation. In: International conference on machine learning, PMLR, pp 1931–1942
Choi JY, Kim J, Park JH et al (2023) SMOP: towards efficient and effective prompt tuning with sparse mixture-of-prompts. In: The 2023 conference on empirical methods in natural language processing
Chowdhery A, Narang S, Devlin J et al (2023) PALM: scaling language modeling with pathways. J Mach Learn Res 24(240):1–113
Christiano PF, Leike J, Brown T et al (2017) Deep reinforcement learning from human preferences. In: Advances in neural information processing systems, vol 30
Chronopoulou A, Peters ME, Fraser A et al (2023) Adaptersoup: Weight averaging to improve generalization of pretrained language models. arXiv preprint. arXiv:2302.07027
Chung HW, Hou L, Longpre S et al (2022) Scaling instruction-finetuned language models. arXiv preprint. arXiv:2210.11416
Chung HW, Hou L, Longpre S et al (2024) Scaling instruction-finetuned language models. J Mach Learn Res 25(70):1–53
Clark P, Cowhey I, Etzioni O et al (2018) Think you have solved question answering? Try arc, the AI2 reasoning challenge. arXiv preprint. arXiv:1803.05457v1
Cobbe K, Kosaraju V, Bavarian M et al (2021) Training verifiers to solve math word problems. arXiv preprint. arXiv:2110.14168
Dai D, Deng C, Zhao C et al (2024a) DeepSeekMoE: towards ultimate expert specialization in mixture-of-experts language models. arXiv preprint. arXiv:2401.06066
Dai W, Li J, Li D et al (2024b) InstructBLIP: towards general-purpose vision-language models with instruction tuning. In: Advances in Neural Information Processing Systems, vol 36
Dan Y, Lei Z, Gu Y et al (2023) Educhat: a large-scale language model-based chatbot system for intelligent education. arXiv preprint. arXiv:2308.02773
Das SSS, Zhang RH, Shi P et al (2023) Unified low-resource sequence labeling by sample-aware dynamic sparse finetuning. arXiv preprint. arXiv:2311.03748
Dettmers T, Pagnoni A, Holtzman A et al (2024) QLORA: efficient finetuning of quantized LLMS. In: Advances in neural information processing systems. vol 36
Devlin J, Chang MW, Lee K et al (2018) BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint. arXiv:1810.04805

Ding N, Qin Y, Yang G et al (2022) Delta tuning: a comprehensive study of parameter efficient methods for pre-trained language models. arXiv preprint. arXiv:2203.06904
Ding N, Lv X, Wang Q et al (2023) Sparse low-rank adaptation of pre-trained language models. In: Proceedings of the 2023 conference on empirical methods in natural language processing, pp 4133–4145
Dong Z, Wei P, Lin L (2022) DreamArtist: towards controllable one-shot text-to-image generation via positive-negative prompt-tuning. arXiv preprint. arXiv:2211.11337
Dušek O, Novikova J, Rieser V (2020) Evaluating the state-of-the-art of end-to-end natural language generation: the E2E NLG challenge. Comput Speech Lang 59:123–156
Edalati A, Tahaei M, Kobyzev I et al (2022) Krona: parameter efficient tuning with kronecker adapter. arXiv preprint. arXiv:2212.10650
Eisele A, Chen Y (2010) Multiun: a multilingual corpus from united nation documents. In: LREC
Ethayarajh K, Choi Y, Swayamdipta S (2022) Understanding dataset difficulty with $\mathcal{V}$-usable information. In: Chaudhuri K, Jegelka S, Song L et al (eds) Proceedings of the 39th international conference on machine learning, proceedings of machine learning research, vol 162. PMLR, pp 5988–6008
Fan J, Wang Z, Xie Y et al (2020) A theoretical analysis of deep Q-learning. In: Learning for dynamics and control, PMLR, pp 486–489
Fu Z, Yang H, So AMC et al (2023) On the effectiveness of parameter-efficient fine-tuning. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp 12799–12807
Fu M, Zhu K, Wu J (2024) DTL: disentangled transfer learning for visual recognition. In: Proceedings of the AAAI conference on artificial intelligence, pp 12082–12090
G Team, Anil R, Borgeaud S et al (2023) Gemini: a family of highly capable multimodal models. arXiv preprint. arXiv:2312.11805
Gal R, Alaluf Y, Atzmon Y et al (2022) An image is worth one word: personalizing text-to-image generation using textual inversion. arXiv preprint. arXiv:2208.01618
Gao L, Biderman S, Black S et al (2020) The Pile: ann 800gb dataset of diverse text for language modeling. arXiv preprint. arXiv:2101.00027
Gardent C, Shimorina A, Narayan S et al (2017) Creating training corpora for NLG micro-planning. In: 55th Annual meeting of the association for computational linguistics, ACL 2017. Association for Computational Linguistics (ACL), pp 179–188
Gheini M, Ren X, May J (2021) Cross-attention is all you need: adapting pretrained transformers for machine translation. arXiv preprint. arXiv:2104.08771
Glaese A, McAleese N, Trbacz M et al (2022) Improving alignment of dialogue agents via targeted human judgements. arXiv preprint. arXiv:2209.14375
Gliwa B, Mochol I, Biesek M et al (2019) Samsum corpus: a human-annotated dialogue dataset for abstractive summarization. arXiv preprint. arXiv:1911.12237
Gokaslan A, Cohen V (2019) Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus
Guo D, Rush AM, Kim Y (2020) Parameter-efficient transfer learning with diff pruning. arXiv preprint. arXiv:2012.07463
Guo D, Yang D, Zhang H et al (2025) Deepseek-r1: incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint. arXiv:2501.12948
Guo H, Greengard P, Xing EP et al (2023) LQ-LORA: low-rank plus quantized matrix decomposition for efficient language model finetuning. arXiv preprint. arXiv:2311.12023
Gupta P, Jiao C, Yeh YT et al (2022) Instructdial: Improving zero and few-shot generalization in dialogue through instruction tuning. arXiv preprint. arXiv:2205.12673
Han L, Li Y, Zhang H et al (2023) SVDIFF: compact parameter space for diffusion fine-tuning. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 7323–7334
Han Z, Gao C, Liu J et al (2024) Parameter-efficient fine-tuning for large models: A comprehensive survey. arXiv preprint. arXiv:2403.14608
Hayou S, Ghosh N, Yu B (2024) Lora+: efficient low rank adaptation of large models. arXiv preprint. arXiv:2402.12354
He J, Zhou C, Ma X et al (2021) Towards a unified view of parameter-efficient transfer learning. arXiv preprint. arXiv:2110.04366
He S, Ding L, Dong D et al (2022) Sparseadapter: an easy approach for improving the parameter-efficiency of adapters. arXiv preprint. arXiv:2210.04284
He H, Cai J, Zhang J et al (2023a) Sensitivity-aware visual parameter-efficient fine-tuning. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 11825–11835
He S, Fan RZ, Ding L et al (2023b) MERA: merging pretrained adapters for few-shot learning. arXiv preprint. arXiv:2308.15982
Hendrycks D, Burns C, Basart S et al (2021a) Measuring massive multitask language understanding. In: ICLR. OpenReview.net

Hendrycks D, Burns C, Kadavath S et al (2021b) Measuring mathematical problem solving with the math dataset. In: Thirty-fifth conference on neural information processing systems datasets and benchmarks track (Round 2), pp 1–11
Hoffmann J, Borgeaud S, Mensch A et al (2022) Training compute-optimal large language models. arXiv preprint. arXiv:2203.15556
Honovich O, Scialom T, Levy O et al (2022) Unnatural instructions: tuning language models with (almost) no human labor. arXiv preprint. arXiv:2212.09689
Houlsby N, Giurgiu A, Jastrzebski S et al (2019) Parameter-efficient transfer learning for nlp. In: International conference on machine learning, PMLR, pp 2790–2799
Hu EJ, Shen Y, Wallis P et al (2021) LORA: low-rank adaptation of large language models. arXiv preprint. arXiv:2106.09685
Hu S, Zhang Z, Ding N et al (2022) Sparse structure search for delta tuning. Adv Neural Inf Process Syst 35:9853–9865
Hu Z, Wang L, Lan Y et al (2023) LLM-ADAPTERS: an adapter family for parameter-efficient fine-tuning of large language models. arXiv preprint. arXiv:2304.01933
Huang J, Chang KCC (2022) Towards reasoning in large language models: A survey. arXiv preprint. arXiv:2212.10403
Huang C, Liu Q, Lin BY et al (2023a) LORAHUB: efficient cross-task generalization via dynamic lora composition. arXiv preprint. arXiv:2307.13269
Huang Q, Dong X, Chen D et al (2023b) Diversity-aware meta visual prompting. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10878–10887
Huang K, Mo F, Li H et al (2024a) A survey on large language models with multilingualism: Recent advances and new frontiers. arXiv preprint. arXiv:2405.10936
Huang K, Qu Y, Cousins H et al (2024b) CRISPR-GPT: an llm agent for automated design of gene-editing experiments. arXiv preprint. arXiv:2404.18021
Iyer S, Lin XV, Pasunuru R et al (2022) OPT-IML: scaling language model instruction meta learning through the lens of generalization. arXiv preprint. arXiv:2212.12017
Jaech A, Kalai A, Lerer A et al (2024) OPENAI O1 system card. arXiv preprint. arXiv:2412.16720
Ji J, Liu M, Dai J et al (2023) BeaverTails: towards improved safety alignment of LLM via a human-preference dataset. Adv Neural Inf Process Syst 36:24678–24704
Jia M, Tang L, Chen BC et al (2022) Visual prompt tuning. In: European conference on computer vision. Springer, pp 709–727
Jiang Z, Araki J, Ding H et al (2021) How can we know when language models know? On the calibration of language models for question answering. Trans Assoc Comput Ling 9:962–977
Jiang AQ, Sablayrolles A, Mensch A et al (2023) MISTRAL 7B. arXiv preprint. arXiv:2310.06825
Jie S, Deng ZH (2022) Convolutional bypasses are better vision transformer adapters. arXiv preprint. arXiv:2207.07039
Jie S, Deng ZH (2023) Fact: Factor-tuning for lightweight adaptation on vision transformer. In: Proceedings of the AAAI conference on artificial intelligence, pp 1060–1068
Jie S, Wang H, Deng ZH (2023) Revisiting the parameter efficiency of adapters from the perspective of precision redundancy. In: Proceedings of the IEEE/CVf international conference on computer vision, pp 17217–17226
Kalla D, Smith N, Samaah F et al (2023) Study and analysis of ChatGPT and its impact on different fields of study. Int J Innov Sci Res Technol 8(3):827
Kaplan J, McCandlish S, Henighan T et al (2020) Scaling laws for neural language models. arXiv preprint. arXiv:2001.08361
Karimi Mahabadi R, Henderson J, Ruder S (2021) Compacter: efficient low-rank hypercomplex adapter layers. Adv Neural Inf Process Syst 34:1022–1035
Keskar NS, McCann B, Xiong C et al (2019) Unifying question answering, text classification, and regression via span extraction. arXiv preprint. arXiv:1904.09286
Khashabi D, Min S, Khot T et al (2020) UNIFIEDQA: crossing format boundaries with a single qa system. arXiv preprint. arXiv:2005.00700
Kim JK, Chua M, Rickard M et al (2023) Chatgpt and large language model (LLM) chatbots: The current state of acceptability and a proposal for guidelines on utilization in academic medicine. J Pediatr Urol 19(5):598–604
Kim J, Lee JH, Kim S et al (2024) Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. In: Advances in neural information processing systems, vol 36
Knox WB, Stone P (2008) TAMER: training an agent manually via evaluative reinforcement. In: 2008 7th IEEE international conference on development and learning. IEEE, pp 292–297
Kocetkov D, Li R, Allal L et al (2022) The stack: 3 tb of permissively licensed source code. arXiv preprint. https://arxiv.org/abs/2211.15533

Kojima T, Gu SS, Reid M et al (2022) Large language models are zero-shot reasoners. Adv Neural Inf Process Syst 35:22199–22213
Lawton N, Kumar A, Thattai G et al (2023) Neural architecture search for parameter-efficient fine-tuning of large pre-trained language models. arXiv preprint. arXiv:2305.16597
Lee H, Phatale S, Mansoor H et al (2023) RLAIF: scaling reinforcement learning from human feedback with ai feedback. arXiv preprint. arXiv:2309.00267
Lee J, Stevens N, Han SC et al (2024) A survey of large language models in finance (FINLLMS). arXiv preprint. arXiv:2402.02315
Lei T, Bai J, Brahma S et al (2024) Conditional adapters: parameter-efficient transfer learning with fast inference. In: Advances in neural information processing systems, vol 36
Lester B, Al-Rfou R, Constant N (2021) The power of scale for parameter-efficient prompt tuning. arXiv preprint. arXiv:2104.08691
Li S, Hoefler T (2021) Chimera: efficiently training large-scale neural networks with bidirectional pipelines. In: Proceedings of the international conference for high performance computing, networking, storage and analysis, pp 1–14
Li XL, Liang P (2021) Prefix-tuning: Optimizing continuous prompts for generation. arXiv preprint. arXiv:2101.00190
Li X, Tramer F, Liang P et al (2021) Large language models can be strong differentially private learners. arXiv preprint. arXiv:2110.05679
Li J, Li D, Savarese S et al (2023a) BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint. arXiv:2301.12597
Li Y, Yu Y, Liang C et al (2023b) LOFTQ: lora-fine-tuning-aware quantization for large language models. In: The Twelfth international conference on learning representations
Li H, Chen J, Yang J et al (2024a) LegalAgentBench: evaluating LLM agents in legal domain. arXiv preprint. arXiv:2412.17259
Li J, Tang T, Zhao WX et al (2024b) Pre-trained language models for text generation: a survey. ACM Comput Surv 56(9):1–39
Lialin V, Deshpande V, Rumshisky A (2023) Scaling down to scale up: a guide to parameter-efficient fine-tuning. arXiv preprint. arXiv:2303.15647
Lian D, Zhou D, Feng J et al (2022) Scaling & shifting your features: a new baseline for efficient model tuning. Adv Neural Inf Process Syst 35:109–123
Liao B, Meng Y, Monz C (2023) Parameter-efficient fine-tuning without introducing new latency. arXiv preprint. arXiv:2305.16742
Lin Z, Madotto A, Fung P (2020) Exploring versatile generative language model via parameter-efficient transfer learning. arXiv preprint. arXiv:2004.03829
Lin S, Hilton J, Evans O (2021) TRUTHFULQA: measuring how models mimic human falsehoods. arXiv preprint. arXiv:2109.07958
Liu X, He P, Chen W et al (2019) Multi-task deep neural networks for natural language understanding. arXiv preprint. arXiv:1901.11504
Liu X, Ji K, Fu Y et al (2021a) P-tuning v2: prompt tuning can be comparable to fine-tuning universally across scales and tasks. arXiv preprint. arXiv:2110.07602
Liu X, Zheng Y, Du Z et al (2021b) GPT understands, too. arXiv preprint. arXiv:2103.10385
Liu YC, Ma CY, Tian J et al (2022a) POLYHISTOR: parameter-efficient multi-task adaptation for dense vision tasks. Adv Neural Inf Process Syst 35:36889–36901
Liu X, Sun T, Huang X et al (2022b) Late prompt tuning: a late prompt could be better than many prompts. arXiv preprint. arXiv:2210.11292
Liu H, Tam D, Muqeeth M et al (2022c) Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. Adv Neural Inf Process Syst 35:1950–1965
Liu H, Li C, Li Y et al (2023a) Improved baselines with visual instruction tuning. arXiv preprint. arXiv:2310.03744
Liu Q, Wu X, Zhao X et al (2023b) MOELORA: an MOE-based parameter efficient fine-tuning method for multi-task medical applications. arXiv preprint. arXiv:2310.18339
Liu Z, Feng R, Zhu K et al (2023c) Cones: Concept neurons in diffusion models for customized generation. arXiv preprint. arXiv:2303.05125
Liu A, Feng B, Wang B et al (2024a) DEEPSEEK-V2: a strong, economical, and efficient mixture-of-experts language model. arXiv preprint. arXiv:2405.04434
Liu A, Feng B, Xue B et al (2024b) DEEPSEEK-V3 technical report. arXiv preprint. arXiv:2412.19437
Liu H, Li C, Wu Q et al (2024c) Visual instruction tuning. In: Advances in neural information processing systems, vol 36
Liu J, Xiao G, Li K et al (2024d) BITDELTA: your fine-tune may only be worth one bit. arXiv preprint. arXiv:2402.10193

Liu SY, Wang CY, Yin H et al (2024e) DORA: weight-decomposed low-rank adaptation. arXiv preprint. arXiv:2402.09353
Liu X, Zheng Y, Du Z et al (2024f) GPT understands, too. AI Open 5:208–215
Liu Z, Kundu S, Li A et al (2024g) AFLORA: adaptive freezing of low rank adaptation in parameter efficient fine-tuning of large models. arXiv preprint. arXiv:2403.13269
Lo K, Wang LL, Neumann M et al (2020) S2ORC: the semantic scholar open research corpus. In: ACL. Association for Computational Linguistics, pp 4969–4983
Lu X, Brahman F, West P et al (2023) Inference-time policy adapters (IPA): tailoring extreme-scale lms without fine-tuning. In: Proceedings of the 2023 conference on empirical methods in natural language processing, pp 6863–6883
Ma F, Zhang C, Ren L et al (2022) XPROMPT: exploring the extreme of prompt tuning. arXiv preprint. arXiv:2210.04457
Mahabadi RK, Ruder S, Dehghani M et al (2021) Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. arXiv preprint. arXiv:2106.04489
Mao Y, Mathias L, Hou R et al (2021) UNIPELT: a unified framework for parameter-efficient language model tuning. arXiv preprint. arXiv:2110.07577
Marjit S, Singh H, Mathur N et al (2024) DIFFUSEKRONA: a parameter efficient fine-tuning method for personalized diffusion model. arXiv preprint. arXiv:2402.17412
McCann B, Keskar NS, Xiong C et al (2018) The natural language decathlon: multitask learning as question answering. arxiv preprint. arXiv:1806.08730
Meng X, Dai D, Luo W et al (2024) PERIODICLORA: breaking the low-rank bottleneck in lora optimization. arXiv preprint. arXiv:2402.16141
Min S, Lewis M, Zettlemoyer L et al (2021) METAICL: learning to learn in context. arXiv preprint. arXiv:2110.15943
Mishra S, Khashabi D, Baral C et al (2021) Cross-task generalization via natural language crowdsourcing instructions. arXiv preprint. arXiv:2104.08773
Mou C, Wang X, Xie L et al (2023) T2I-ADAPTER: learning adapters to dig out more controllable ability for text-to-image diffusion models. arXiv preprint. arXiv:2302.08453
Muennighoff N, Wang T, Sutawika L et al (2022) Crosslingual generalization through multitask finetuning. arXiv preprint. arXiv:2211.01786
Nakano R, Hilton J, Balaji S et al (2021) WEBGPT: browser-assisted question-answering with human feedback. arXiv preprint. arXiv:2112.09332
Nan L, Radev DR, Zhang R et al (2021) DART: open-domain structured data record to text generation. In: NAACL-HLT. Association for Computational Linguistics, pp 432–447
Narayan S, Cohen SB, Lapata M (2018) Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In: EMNLP. Association for Computational Linguistics, pp 1797–1807
Ouyang L, Wu J, Jiang X et al (2022) Training language models to follow instructions with human feedback. Adv Neural Inf Process Syst 35:27730–27744
Pan Z, Luo H, Li M et al (2024) CONV-COA: improving open-domain question answering in large language models via conversational chain-of-action. arXiv preprint. arXiv:2405.17822
Pfeiffer J, Kamath A, Rücklé A et al (2020) ADAPTERFUSION: non-destructive task composition for transfer learning. arXiv preprint. arXiv:2005.00247
Qin Y, Wang X, Su Y et al (2021) Exploring universal intrinsic task subspace via prompt tuning. arXiv preprint. arXiv:2110.07867
Qiu Z, Liu W, Feng H et al (2024) Controlling text-to-image diffusion by orthogonal finetuning. In: Advances in neural information processing systems, vol 36
Radford A, Narasimhan K, Salimans T et al (2018) Improving language understanding by generative pre-training. Technical Report, OpenAI
Radford A, Wu J, Child R et al (2019) Language models are unsupervised multitask learners. OpenAI Blog 1(8):9
Radford A, Kim JW, Hallacy C et al (2021) Learning transferable visual models from natural language supervision. In: International conference on machine learning, PMLR, pp 8748–8763
Rae JW, Potapenko A, Jayakumar SM et al (2020) Compressive transformers for long-range sequence modelling. In: ICLR. OpenReview.net
Rafailov R, Sharma A, Mitchell E et al (2024) Direct preference optimization: your language model is secretly a reward model. In: Advances in neural information processing systems, vol 36
Raffel C, Shazeer N, Roberts A et al (2020) Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res 21(140):1–67
Rajabzadeh H, Valipour M, Zhu T et al (2024) QDYLORA: quantized dynamic low-rank adaptation for efficient large language model tuning. arXiv preprint. arXiv:2402.10462

Rein D, Hou BL, Stickland AC et al (2024) GPQA: a graduate-level google-proof q &a benchmark. In: First conference on language modeling
Riquelme C, Puigcerver J, Mustafa B et al (2021) Scaling vision with sparse mixture of experts. Adv Neural Inf Process Syst 34:8583–8595
Rücklé A, Geigle G, Glockner M et al (2020) ADAPTERDROP: on the efficiency of adapters in transformers. arXiv preprint. arXiv:2010.11918
Ruiz N, Li Y, Jampani V et al (2023) DREAMBOOTH: fine tuning text-to-image diffusion models for subject-driven generation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 22500–22510
Sanh V, Webson A, Raffel C et al (2021) Multitask prompted training enables zero-shot task generalization. arXiv preprint. arXiv:2110.08207
Saparov A, He H (2022) Language models are greedy reasoners: a systematic formal analysis of chain-of-thought. arXiv preprint. arXiv:2210.01240
Schulman J, Wolski F, Dhariwal P et al (2017) Proximal policy optimization algorithms. arXiv preprint. arXiv:1707.06347
Shao Z, Wang P, Zhu Q et al (2024) DEEPSEEKMATH: pushing the limits of mathematical reasoning in open language models. arXiv preprint. arXiv:2402.03300
Shi Z, Lipani A (2023) DEPT: decomposed prompt tuning for parameter-efficient fine-tuning. arXiv preprint. arXiv:2309.05173
Singhal K, Azizi S, Tu T et al (2023) Large language models encode clinical knowledge. Nature 620(7972):172–180
Sprague Z, Ye X, Bostrom K et al (2023) MUSR: testing the limits of chain-of-thought with multistep soft reasoning. arXiv preprint. arXiv:2310.16049
Su Y, Wang X, Qin Y et al (2021) On transferability of prompt tuning for natural language processing. arXiv preprint. arXiv:2111.06719
Sun Q, Fang Y, Wu L et al (2023) EVA-CLIP: improved training techniques for clip at scale. arXiv preprint. arXiv:2303.15389
Sung YL, Nair V, Raffel CA (2021) Training neural networks with fixed sparse masks. Adv Neural Inf Process Syst 34:24193–24205
Sung YL, Cho J, Bansal M (2022) LST: ladder side-tuning for parameter and memory efficient transfer learning. Adv Neural Inf Process Syst 35:12991–13005
Sutton RS (1995) Generalization in reinforcement learning: Successful examples using sparse coarse coding. In: Advances in neural information processing systems, vol 8
Suzgun M, Scales N, Schärli N et al (2023) Challenging big-bench tasks and whether chain-of-thought can solve them. In: ACL (findings). Association for Computational Linguistics, pp 13003–13051
Tang A, Shen L, Luo Y et al (2023) Parameter efficient multi-task model fusion with partial linearization. arXiv preprint. arXiv:2310.04742
Tay Y, Wei J, Chung HW et al (2022) Transcending scaling laws with 0.1% extra compute. arXiv preprint. arXiv:2210.11399
Tian K, Mitchell E, Zhou A et al (2023) Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. arXiv preprint. arXiv:2305.14975
Tian C, Shi Z, Guo Z et al (2024) HYDRALORA: an asymmetric lora architecture for efficient fine-tuning. arXiv preprint. arXiv:2404.19245
Touvron H, Lavril T, Izacard G et al (2023) Llama: Open and efficient foundation language models. arXiv preprint. arXiv:2302.13971
Tu CH, Mai Z, Chao WL (2023) Visual query tuning: towards effective usage of intermediate representations for parameter and memory efficient transfer learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 7725–7735
Valipour M, Rezagholizadeh M, Kobyzev I et al (2023) DYLORA: parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In: Proceedings of the 17th conference of the European chapter of the Association for Computational Linguistics, pp 3274–3287
Vaswani A, Shazeer N, Parmar N et al (2017) Attention is all you need. In: Advances in neural information processing systems, vol 30
Vavekanand R, Sam K (2024) LLAMA 3.1: an in-depth analysis of the next-generation large language model. https://doi.org/10.13140/RG.2.2.10628.74882
Voynov A, Aberman K, Cohen-Or D (2023a) Sketch-guided text-to-image diffusion models. In: ACM SIGGRAPH 2023 conference proceedings, pp 1–11
Voynov A, Chu Q, Cohen-Or D et al (2023b) p+[CDATA[p+]]: extended textual conditioning in text-to-image generation. arXiv preprint. arXiv:2303.09522
Vu T, Lester B, Constant N et al (2021) SPoT: better frozen model adaptation through soft prompt transfer. arXiv preprint. arXiv:2110.07904

Vu T, Lester B, Constant N et al (2022) SPoT: better frozen model adaptation through soft prompt transfer. In: Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers), pp 5039–5059
Vucetic D, Tayaranian M, Ziaeefard M et al (2022) Efficient fine-tuning of bert models on the edge. In: 2022 IEEE international symposium on circuits and systems (ISCAS). IEEE, pp 1838–1842
Wang J (2023) The power of ai-assisted diagnosis. EAI Endorsed Transactions on e-Learning 8(4)
Wang A, Pruksachatkun Y, Nangia N et al (2019a) SuperGLUE: a stickier benchmark for general-purpose language understanding systems. In: NeurIPS, pp 3261–3275
Wang A, Singh A, Michael J et al (2019b) GLUE: a multi-task benchmark and analysis platform for natural language understanding. In: ICLR (Poster). OpenReview.net
Wang P, Yang A, Men R et al (2022a) OFA: unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. In: International conference on machine learning, PMLR, pp 23318–23340
Wang Y, Kordi Y, Mishra S et al (2022b) Self-instruct: aligning language models with self-generated instructions. arXiv preprint. arXiv:2212.10560
Wang L, Lyu C, Ji T et al (2023a) Document-level machine translation with large language models. arXiv preprint. arXiv:2304.02210
Wang Q, Mao Y, Wang J et al (2023b) APROMPT: attention prompt tuning for efficient adaptation of pre-trained language models. In: Proceedings of the 2023 conference on empirical methods in natural language processing, pp 9147–9160
Wang W, Lv Q, Yu W et al (2023c) COGVLM: visual expert for pretrained language models. arXiv preprint. arXiv:2311.03079
Wang X, Hu Z, Lu P et al (2023d) SCIBENCH: evaluating college-level scientific problem-solving abilities of large language models. arXiv preprint. arXiv:2307.10635
Wang Y, Mishra S, Alipoormolabashi P et al (2022c) Benchmarking generalization via in-context instructions on 1,600+ language tasks. arXiv preprint. arXiv:2204.07705 2
Wang Y, Mukherjee S, Liu X et al (2022d) Adamix: mixture-of-adapter for parameter-efficient tuning of large language models. arXiv preprint. arXiv:2205.12410 1(2):4
Wang Z, Panda R, Karlinsky L et al (2022e) Multitask prompt tuning enables parameter-efficient transfer learning. In: The Eleventh international conference on learning representations
Wang G, Cheng S, Zhan X et al (2024a) OpenChat: advancing open-source language models with mixed-quality data. In: ICLR. OpenReview.net
Wang H, Chang J, Zhai Y et al (2024b) LION: implicit vision prompt tuning. In: Proceedings of the AAAI conference on artificial intelligence, pp 5372–5380
Wei J, Bosma M, Zhao VY et al (2021) Finetuned language models are zero-shot learners. arXiv preprint. arXiv:2109.01652
Wei J, Tay Y, Bommasani R et al (2022) Emergent abilities of large language models. arXiv preprint. arXiv:2206.07682
Wei X, Li G, Marculescu R (2024) ONLINE-LORA: task-free online continual learning via low rank adaptation. arXiv preprint. arXiv:2411.05663
Wu Z, Wang S, Gu J et al (2022) IDPG: an instance-dependent prompt generation method. arXiv preprint. arXiv:2204.04497
Wu S, Fei H, Qu L et al (2023) NEXT-GPT: any-to-any multimodal llm. arXiv preprint. arXiv:2309.05519
Wu J, Li X, Wei C et al (2024a) Unleashing the power of visual prompting at the pixel level. In: TMLR
Wu J, Yu T, Wang R et al (2024b) Infoprompt: Information-theoretic soft prompt tuning for natural language understanding. In: Advances in neural information processing systems, vol 36
Wu Y, Xiang Y, Huo S et al (2024c) LORA-SP: streamlined partial parameter adaptation for resource-efficient fine-tuning of large language models. arXiv preprint. arXiv:2403.08822
Xie E, Yao L, Shi H et al (2023) Difffit: Unlocking transferability of large diffusion models via simple parameter-efficient fine-tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 4230–4239
Xin Y, Luo S, Zhou H et al (2024) Parameter-efficient fine-tuning for pre-trained vision models: a survey. arXiv preprint. arXiv:2402.02242
Xing F (2024) Designing heterogeneous llm agents for financial sentiment analysis. ACM Trans Manag Inf Syst 16(1):1–24
Xu S, Wen X (2024) Automatic design of adapter architectures for enhanced parameter-efficient fine-tuning. In: ICASSP 2024–2024 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 12536–12540
Xu R, Luo F, Zhang Z et al (2021) Raise a child in large language model: towards effective and generalizable fine-tuning. arXiv preprint. arXiv:2109.05687

Xu L, Xie H, Qin SZJ et al (2023a) Parameter-efficient fine-tuning methods for pretrained language models: a critical review and assessment. arXiv preprint. arXiv:2312.12148
Xu M, Zhang Z, Wei F et al (2023b) Side adapter network for open-vocabulary semantic segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2945–2954
Xu Y, Xie L, Gu X et al (2023c) QA-LORA: quantization-aware low-rank adaptation of large language models. In: The Twelfth international conference on learning representations
Yang Z, Qi P, Zhang S et al (2018) HOTPOTQA: a dataset for diverse, explainable multi-hop question answering. In: EMNLP. Association for Computational Linguistics, pp 2369–2380
Yang AX, Robeyns M, Wang X et al (2023a) Bayesian low-rank adaptation for large language models. In: The Twelfth international conference on learning representations
Yang X, Huang JY, Zhou W et al (2023b) Parameter-efficient tuning with special token adaptation. In: Proceedings of the 17th conference of the European chapter of the association for computational linguistics, pp 865–872
Yang B, Tian H, Ren J et al (2024) Multi-objective fine-tuning for enhanced program repair with LLMS. arXiv preprint. arXiv:2404.12636
Yao Y, Duan J, Xu K et al (2024) A survey on large language model (LLM) security and privacy: the good, the bad, and the ugly. High-Confidence Computing p 100211
Ye S, Kim D, Jang J et al (2022) Guess the instruction! making language models stronger zero-shot learners. arXiv preprint. arXiv:2210.02969
Ye H, Zhang J, Liu S et al (2023) IP-ADAPTER: text compatible image prompt adapter for text-to-image diffusion models. arXiv preprint. arXiv:2308.06721
Yeh SY, Hsieh YG, Gao Z et al (2023) Navigating text-to-image customization: from lycoris fine-tuning to model evaluation. arXiv preprint. arXiv:2309.14859
Yin D, Yang Y, Wang Z et al (2023) 1% vs 100%: parameter-efficient low rank adapter for dense predictions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 20116–20126
Yuan S, Zhao H, Du Z et al (2021) Wudaocorpora: a super large-scale Chinese corpora for pre-training language models. AI Open 2:65–68
Zadouri T, Üstün A, Ahmadian A et al (2023) Pushing mixture of experts to the limit: extremely parameter efficient moe for instruction tuning. arXiv preprint. arXiv:2309.05444
Zaken EB, Ravfogel S, Goldberg Y (2021) Bitfit: simple parameter-efficient fine-tuning for transformer-based masked language-models. arXiv preprint. arXiv:2106.10199
Zavras A, Michail D, Demir B et al (2024) Mind the modality gap: towards a remote sensing vision-language model via cross-modal alignment. arXiv preprint. arXiv:2402.09816
Zellers R, Holtzman A, Bisk Y et al (2019) Hellaswag: can a machine really finish your sentence? In: Proceedings of the 57th annual meeting of the Association for Computational Linguistics
Zeng G, Zhang P, Lu W (2023) One network, many masks: towards more parameter-efficient transfer learning. arXiv preprint. arXiv:2305.17682
Zhai X, Kolesnikov A, Houlsby N et al (2022) Scaling vision transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 12104–12113
Zhang H, Xu J, Wang J (2019) Pretraining-based natural language generation for text summarization. arXiv preprint. arXiv:1902.09243
Zhang Y, Zhou K, Liu Z (2022) Neural prompt search. arXiv preprint. arXiv:2206.04673
Zhang B, Yang H, Zhou T et al (2023a) Enhancing financial sentiment analysis via retrieval augmented large language models. In: Proceedings of the fourth ACM international conference on AI in finance, pp 349–356
Zhang F, Li L, Chen J et al (2023b) Increlora: Incremental parameter allocation method for parameter-efficient fine-tuning. arXiv preprint. arXiv:2308.12043
Zhang L, Rao A, Agrawala M (2023c) Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 3836–3847
Zhang L, Zhang L, Shi S et al (2023d) LORA-FA: memory-efficient low-rank adaptation for large language models fine-tuning. arXiv preprint. arXiv:2308.03303
Zhang M, Shen C, Yang Z et al (2023e) Pruning meets low-rank parameter-efficient fine-tuning. arXiv preprint. arXiv:2305.18403
Zhang Q, Chen M, Bukharin A et al (2023f) ADALORA: adaptive budget allocation for parameter-efficient fine-tuning. arXiv preprint. arXiv:2303.10512
Zhang R, Han J, Liu C et al (2023g) LLAMA-ADAPTER: efficient fine-tuning of language models with zero-init attention. arXiv preprint. arXiv:2303.16199
Zhang X, Li C, Zong Y et al (2023h) Evaluating the performance of large language models on gaokao benchmark. arXiv preprint. arXiv:2305.12474

Zhang ZR, Tan C, Xu H et al (2023i) Towards adaptive prefix tuning for parameter-efficient language model fine-tuning. arXiv preprint. arXiv:2305.15212
Zhao M, Lin T, Mi F et al (2020) Masking as an efficient alternative to finetuning for pretrained language models. arXiv preprint. arXiv:2004.12406
Zhao H, Tan H, Mei H (2022) Tiny-attention adapter: contexts are more important than the number of parameters. arXiv preprint. arXiv:2211.01979
Zhao H, Fu J, He Z (2023) Prototype-based hyperadapter for sample-efficient multi-task tuning. arXiv preprint. arXiv:2310.11670
Zhao S, Chen D, Chen YC et al (2024) UNI-CONTROLNET: all-in-one control to text-to-image diffusion models. In: Advances in neural information processing systems, vol 36
Zheng L, Chiang WL, Sheng Y et al (2023) Judging llm-as-a-judge with mt-bench and chatbot arena. Adv Neural Inf Process Syst 36:46595–46623
Zhong V, Xiong C, Socher R (2017) SEQ2SQL: generating structured queries from natural language using reinforcement learning. arXiv preprint. arXiv:1709.00103
Zhong W, Cui R, Guo Y et al (2024) AGIEVAL: a human-centric benchmark for evaluating foundation models. In: NAACL-HLT (Findings). Association for Computational Linguistics, pp 2299–2314
Zhou H, Wan X, Vulić I et al (2024) AUTOPEFT: automatic configuration search for parameter-efficient fine-tuning. Trans Assoc Comput Ling 12:525–542
Zhu W, Tan M (2023) SPT: learning to selectively insert prompts for better prompt tuning. In: The 2023 conference on empirical methods in natural language processing
Zhu Y, Kiros R, Zemel RS et al (2015) Aligning books and movies: towards story-like visual explanations by watching movies and reading books. In: ICCV. IEEE Computer Society, pp 19–27
Zhu Y, Feng J, Zhao C et al (2021) Counter-interference adapter for multilingual machine translation. arXiv preprint. arXiv:2104.08154
Zhu W, Liu H, Dong Q et al (2023) Multilingual machine translation with large language models: empirical results and analysis. arXiv preprint. arXiv:2304.04675

Authors and Affiliations

Luping Wang[1] · Sheng Chen[1] · Linnan Jiang[1] · Shu Pan[1] · Runze Cai[1] · Sen Yang[1] · Fei Yang[1]

✉ Fei Yang
yangf@zhejianglab.org

Luping Wang
wangluping@zhejianglab.org

Sheng Chen
scucs@zhejianglab.org

Linnan Jiang
jianglinnan@zhejianglab.org

Shu Pan
shu.pan@zhejianglab.org

Runze Cai
cairz@zhejianglab.org

Sen Yang
yangsen@zhejianglab.org

[1] Zhejiang Laboratory, Kechuang Avenue, Zhongtai Sub-district, Yuhang District, Hangzhou 311121, Zhejiang Province, China