LoRA Meets Foundation Models: Unlocking Efficient Specialization for Scalable AI

Jianhong Shun*
jianhong.shun@mail.xjtu.edu.cn
The Frontier Institute of Science and Technology (FIST), Xi'an Jiaotong University, China

Abstract

The proliferation of foundation models—massive pre-trained architectures with billions of parameters—has redefined the landscape of deep learning. While these models achieve remarkable performance across a wide range of tasks, their fine-tuning poses significant computational and storage challenges, particularly in low-resource or multi-task scenarios. Low-Rank Adaptation (LoRA) has emerged as a principled and practical solution to this bottleneck, enabling efficient specialization of frozen base models via lightweight, trainable rank-constrained updates. This survey provides a comprehensive overview of LoRA in the context of foundation models. We begin by formalizing the core intuition behind low-rank updates, analyzing their expressivity, regularization properties, and connections to classical matrix theory. We then explore the expanding ecosystem of LoRA variants and extensions, including quantized, sparse, and task-conditioned forms. Comparisons with alternative parameter-efficient fine-tuning (PEFT) methods such as adapters, prompt tuning, and BitFit highlight LoRA's distinctive strengths in mergeability, modularity, and scalability. Practical applications are discussed across NLP, vision, and multimodal domains, with attention to open-source ecosystems and real-world deployments. Finally, we outline key open challenges—from automatic rank selection and robustness to cross-modal generalization—and chart promising directions for future research. This work positions LoRA as a foundational primitive for scalable, modular, and democratized adaptation in the age of foundation models.

Index Terms

Low-Rank Adaptation (LoRA), Parameter-Efficient Fine-Tuning (PEFT), Foundation Models, Transfer Learning, Modular Adaptation, Deep Learning, Natural Language Processing (NLP), Multimodal Models, Scalable AI, Machine Learning Optimization.

I. Introduction

In recent years, the field of machine learning has experienced a paradigm shift catalyzed by the emergence and widespread adoption of foundation models—extremely large-scale, pre-trained models capable of exhibiting remarkable generalization abilities across diverse tasks and domains [1]. Foundation models, such as BERT, GPT, CLIP, and ViT, trained on massive datasets with billions of parameters, have not only redefined the state-of-the-art in natural language processing (NLP), computer vision (CV), and multimodal learning, but also raised critical questions about the future of model training, fine-tuning, and deployment [2]. These models are characterized by their scale, versatility, and the promise of "few-shot" or "zero-shot" generalization, which have opened up unprecedented opportunities and challenges in the design and customization of machine learning systems [3]. However, the increasing computational and storage demands associated with training and deploying foundation models pose significant practical barriers. Full fine-tuning of such models is prohibitively expensive for most practitioners and organizations, requiring access to high-end hardware infrastructure, considerable energy consumption, and complex engineering pipelines [4]. Moreover, fine-tuning the entire parameter set of a foundation model for each downstream task leads to significant redundancy and storage inefficiency, especially when maintaining separate models for multiple tasks or clients. These issues have spurred interest in more efficient adaptation techniques that can preserve the benefits of large-scale pretraining while mitigating the costs associated with traditional fine-tuning. One of the most promising approaches to address this challenge is Low-Rank Adaptation (LoRA), a parameter-efficient fine-tuning method that injects low-rank trainable matrices into pre-trained models while freezing the original weights [5]. Originally introduced as a way to adapt large language models with minimal computational overhead, LoRA has rapidly gained traction across various modalities and architectures due to its simplicity, effectiveness, and versatility [6]. By constraining the update space to low-rank subspaces, LoRA enables the model to adapt to new tasks using only a small number of additional parameters—often orders of magnitude fewer than full fine-tuning—while retaining performance comparable to or even surpassing traditional methods [7]. The conceptual foundation of LoRA builds upon well-established principles in linear algebra and matrix decomposition, notably the idea that the differences between pre-trained and fine-tuned weights often lie in low-rank subspaces. This insight has profound implications for model design: it suggests that meaningful adaptations can be achieved without perturbing the full parameter space, allowing for modular, interpretable, and efficient fine-tuning mechanisms. The LoRA paradigm also naturally supports scenarios such as multi-task learning, continual learning, and personalization, where separate low-rank adapters can be trained and swapped in without modifying the shared base model [8]. This decoupling of the task-specific and shared components has inspired a

*Corresponding author.

broader family of approaches often referred to as parameter-efficient fine-tuning (PEFT), with LoRA emerging as a central and unifying framework [9]. In the age of foundation models, LoRA occupies a crucial position at the intersection of scalability, adaptability, and accessibility. Its low barrier to entry democratizes the use of foundation models, enabling researchers and practitioners without access to large-scale compute to still benefit from state-of-the-art performance. At the same time, its technical underpinnings have stimulated new lines of inquiry into the geometry of weight updates, the structure of neural network representations, and the theoretical limits of low-rank approximation in deep learning. The impact of LoRA extends beyond academic research into real-world applications ranging from generative AI and robotics to medical imaging and edge computing, where efficiency and modularity are not merely desirable but often essential [10]. Despite its widespread adoption and growing ecosystem of variants, a comprehensive understanding of LoRA—its design principles, empirical behaviors, theoretical foundations, and practical trade-offs—remains elusive. The rapidly evolving literature has produced a diverse set of methods inspired by LoRA, including variants that explore different decomposition strategies, adaptive rank selection, integration with attention mechanisms, and combinations with other efficient adaptation techniques such as adapters, prompt tuning, and pruning. These developments have not only broadened the applicability of LoRA but also introduced new dimensions of complexity that warrant careful analysis and systematization [11]. This survey aims to provide a thorough and structured overview of LoRA in the context of foundation models [12]. We begin by contextualizing the need for parameter-efficient fine-tuning and tracing the evolution of LoRA from its original formulation to its modern variants [13]. We then delve into the theoretical motivations and design choices underlying LoRA, including rank selection, initialization strategies, and integration with various neural architectures [14]. Next, we present an in-depth empirical analysis, synthesizing results from multiple domains and benchmarks to highlight the strengths and limitations of LoRA in practice. We also examine the broader ecosystem of LoRA-inspired methods, drawing connections to adjacent paradigms such as adapters, bitfit, prompt tuning, and dynamic reparameterization. Finally, we discuss open challenges, future directions, and the role of LoRA in the evolving landscape of scalable and responsible AI [15]. By offering a comprehensive synthesis of the current state of LoRA and its applications to foundation models, this survey aspires to serve both as a reference for researchers and a practical guide for practitioners. In doing so, we hope to illuminate the path toward more efficient, flexible, and inclusive adaptation strategies in the era of foundation-scale artificial intelligence [16].

II. Background and Preliminaries

A. The Rise of Foundation Models

The term *foundation models* refers to a new class of large-scale, pre-trained models that serve as general-purpose backbones for a wide range of downstream tasks. These models are trained on broad datasets using self-supervised or weakly supervised objectives and demonstrate remarkable capabilities in transfer learning. Seminal examples include BERT [17] for natural language understanding, GPT-series [**?**], [**?**], [**?**] for text generation, CLIP [18] for vision-language understanding, and Vision Transformers (ViT) [19] for image classification. Foundation models have exhibited emergent behaviors such as in-context learning, long-range reasoning, and multi-modal grounding, leading to their adoption as core components in both academic and industrial AI systems. These models are often parameterized by billions (or even trillions) of weights, trained over massive corpora on specialized hardware such as TPUs or GPU clusters [20], [21]. As a result, the cost of developing foundation models is largely centralized, while their deployment across tasks is increasingly global [22]. This centralized pretraining and decentralized adaptation paradigm has created a dichotomy: while few institutions can train foundation models from scratch, many users can benefit from them—provided that efficient methods for adapting these models exist.

B. Challenges in Fine-Tuning Foundation Models

Traditional fine-tuning involves adjusting all the parameters of a pre-trained model to adapt it to a new task or domain [23]. While effective in small- to medium-sized models, this approach becomes infeasible at the scale of foundation models due to several key limitations:

- **Computational Cost:** Updating billions of parameters during training requires significant memory, compute resources, and training time [24].
- **Storage Overhead:** Fine-tuning a model separately for each downstream task leads to multiple full copies of the model, resulting in redundant storage.
- **Catastrophic Forgetting:** Without careful regularization, full fine-tuning may cause the model to forget information learned during pretraining [25].
- **Limited Personalization and Multi-Task Scalability:** It is challenging to maintain separate, personalized fine-tuned models at scale for multiple users or tasks.

These challenges have sparked the development of *parameter-efficient fine-tuning* (PEFT) techniques, which aim to adapt large models using only a small number of trainable parameters [26]. Among these, LoRA has emerged as a particularly attractive method due to its simplicity, strong empirical performance, and compatibility with a wide range of architectures.

C. Overview of Parameter-Efficient Fine-Tuning (PEFT)

Parameter-efficient fine-tuning refers to strategies that adapt large pre-trained models with minimal changes to their parameters [27]. The general goals of PEFT methods include:

- Reducing the number of trainable parameters.
- Preserving the pre-trained knowledge of the model [28].
- Enabling scalable and modular adaptation.
- Lowering the memory and compute footprint during training and inference [29].

Common PEFT methods include:

Adapters Introduced by Houlsby et al. [30], adapters are small bottleneck modules inserted between layers of a pre-trained model. Only the adapters are trained, leaving the base model unchanged [31].

Prompt Tuning Methods like prefix tuning [32] or P-Tuning [**?**] train task-specific soft prompts that guide the frozen model's behavior, particularly in NLP.

BitFit A minimalist approach that fine-tunes only the bias terms of the model [**?**] [33].

LoRA A linear adaptation strategy that injects low-rank updates into the model weights [34], detailed in the next section [35].

LoRA distinguishes itself by approximating the weight updates as a product of two low-rank matrices, significantly reducing the number of trainable parameters while maintaining performance. This strategy not only facilitates efficient adaptation but also yields insights into the structure of model updates and their implications on downstream task learning.

D. Mathematical Foundations of Low-Rank Approximation

To fully appreciate the LoRA methodology, it is essential to understand the mathematical basis of low-rank matrix approximation. Given a full-rank matrix $W \in \mathbb{R}^{d\times k}$, LoRA aims to approximate updates ΔW to this matrix using the product of two low-rank matrices $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$ such that:

$$\Delta W \approx AB, \quad \text{with } r \ll \min(d, k)$$

The total number of trainable parameters in AB is $r(d + k)$, which is substantially smaller than dk when r is small [36]. This technique leverages the empirical observation that the changes induced by task-specific fine-tuning often reside in a low-dimensional subspace of the weight space [37]. LoRA utilizes this insight to constrain the optimization process, yielding efficient updates that are both memory- and computation-efficient.

E. LoRA in the Broader Landscape of Model Adaptation

LoRA can be interpreted both as a practical fine-tuning strategy and as a lens into the geometric and statistical structure of deep learning models [38]. Its formulation invites comparisons with classical techniques in numerical linear algebra (e.g., SVD), statistical modeling (e.g., factor analysis), and neural network compression (e.g., low-rank pruning) [39]. Furthermore, LoRA is often compatible with other PEFT methods, leading to hybrid approaches such as LoRA+Prompt Tuning or LoRA+Adapters that combine their respective benefits [40]. Given its conceptual clarity and empirical robustness, LoRA has been widely adopted in open-source tools (e.g., Hugging Face PEFT library, LoRA implementations in Stable Diffusion and LLaMA) and production-grade systems. Its impact spans language modeling, computer vision, audio processing, and multi-modal applications. In the following section, we delve into the technical design of LoRA, analyze its architectural integration, and explore its key hyperparameters and implementation details [41].

III. LoRA: Methodology and Design

A. Core Idea

The fundamental motivation behind Low-Rank Adaptation (LoRA) is based on the empirical observation that the weight updates required for adapting large-scale models often lie in a low-dimensional subspace of the full parameter space. Instead of updating the full weight matrix of a pre-trained model, LoRA proposes to represent the update as a low-rank decomposition, dramatically reducing the number of trainable parameters [42]. Formally, consider a weight matrix $W_0 \in \mathbb{R}^{d\times k}$ in a pre-trained neural network layer. In full fine-tuning, we would learn an updated matrix $W = W_0 + \Delta W$, where $\Delta W \in \mathbb{R}^{d\times k}$ is the learned update. In LoRA, the update is constrained to a low-rank form:

$$\Delta W = AB, \quad A \in \mathbb{R}^{d\times r}, \quad B \in \mathbb{R}^{r\times k}, \quad r \ll \min(d, k)$$

Instead of learning ΔW directly, we learn the matrices A and B, while keeping W_0 frozen. The updated weight is then:

$$W = W_0 + \alpha \cdot AB$$

Here, $\alpha \in \mathbb{R}$ is a scaling factor, often chosen as $\alpha = \frac{r}{s}$ for some scalar s to stabilize training [43].

B. LoRA in Transformer Architectures

LoRA is particularly well-suited for transformer-based architectures, such as BERT and GPT, where the majority of parameters reside in the attention and feed-forward layers. For example, in a typical self-attention mechanism, the output of the attention head is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

where the query, key, and value matrices are computed as:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V$$

Applying LoRA to this setting involves introducing low-rank updates to W^Q, W^K, or W^V (or all of them). For instance:

$$W^Q = W_0^Q + \Delta W^Q = W_0^Q + A^Q B^Q$$

The original matrix W_0^Q is frozen, and only $A^Q \in \mathbb{R}^{d\times r}$ and $B^Q \in \mathbb{R}^{r\times d}$ are trained. This mechanism is lightweight and modular—each modified layer adds only $2rd$ trainable parameters per adaptation [44].

C. Parameter Efficiency Analysis

The parameter savings of LoRA are significant, particularly when r is small [45]. Table I illustrates a comparative analysis between full fine-tuning and LoRA-based adaptation for a transformer layer with input/output dimension $d = 1024$ and hidden size $k = 4096$ [46].

TABLE I
TRAINABLE PARAMETERS FOR FULL FINE-TUNING VS. LORA WITH VARIOUS RANKS.

Method	Trainable Params	Reduction Ratio
Full Fine-Tuning	$1024 \times 4096 = 4.19 \times 10^6$	$1\times$
LoRA ($r = 8$)	$8 \cdot (1024 + 4096) = 40,960$	$\sim 102\times$
LoRA ($r = 16$)	$16 \cdot (1024 + 4096) = 81,920$	$\sim 51\times$
LoRA ($r = 32$)	$32 \cdot (1024 + 4096) = 163,840$	$\sim 25.6\times$

As evident from Table I, LoRA can reduce the number of trainable parameters by more than two orders of magnitude while achieving competitive performance on downstream tasks, especially when combined with pre-trained model knowledge.

D. Initialization and Training Details

LoRA typically initializes the low-rank matrices A and B such that the initial update $\Delta W = AB$ is close to zero. A common choice is:

$$A \sim \mathcal{N}(0, \sigma^2), \quad B \sim 0$$

This ensures that at the start of training, the LoRA-injected model behaves identically to the base pre-trained model. The scale α is often absorbed into A or B during training for implementation convenience [47].

E. Deployment and Inference Efficiency

One notable advantage of LoRA is that during inference, the low-rank components AB can be merged with the frozen weight W_0 into a single matrix:

$$W_{\text{merged}} = W_0 + AB$$

This allows LoRA-adapted models to run at the same speed and memory footprint as fully fine-tuned models, making them attractive for deployment on resource-constrained devices.

F. LoRA Design Choices and Hyperparameters

Key design choices in LoRA include:

- **Rank r:** Determines the dimensionality of the low-rank update [48]. Higher r increases capacity but reduces parameter efficiency.
- **Target Layers:** LoRA can be selectively applied to attention, feedforward, or even normalization layers.
- **Dropout:** Regularization can be applied to A or B during training to improve generalization [49].
- **Merge Behavior:** During training, LoRA can operate in a decomposed form; during inference, it can be merged.

In practice, LoRA has shown robust performance with small ranks (e.g., $r = 4$ or $r = 8$) on tasks ranging from natural language understanding to image generation, often with minimal tuning.

G. Variants and Extensions

Recent extensions to LoRA include:

- **Dynamic LoRA:** Learns or adjusts the rank r during training.
- **Grouped LoRA:** Applies different low-rank adapters to different attention heads.
- **LoRA + Prompt Tuning:** Combines low-rank updates with soft prompts for enhanced flexibility.
- **Multi-LoRA:** Uses a mixture-of-experts design with multiple LoRA modules activated conditionally.

These variants expand LoRA's expressiveness and adapt it to increasingly complex tasks and architectures [50]. We explore these extensions in detail in a later section.

IV. Applications and Empirical Results

A. Natural Language Processing

LoRA has found widespread success in Natural Language Processing (NLP), where foundation models such as GPT, BERT, and T5 dominate [51]. By inserting low-rank updates into attention and feedforward layers, LoRA enables fast and memory-efficient task-specific fine-tuning. A representative set of benchmarks is summarized in Table II [52]. LoRA achieves competitive or state-of-the-art performance across tasks with a tiny fraction of trainable parameters compared to full fine-tuning.

TABLE II
PERFORMANCE COMPARISON ON NLP BENCHMARKS USING A ROBERTA-LARGE MODEL.

Method	Avg. GLUE Score	Trainable Params	Relative Size (%)
Full Fine-Tuning	89.8	355M	100.0%
Adapter (Houlsby)	89.2	9M	2.5%
Prefix Tuning	88.7	1.2M	0.3%
LoRA ($r = 8$)	89.6	2.4M	0.7%
LoRA ($r = 16$)	**89.9**	4.8M	1.4%

In open-ended text generation, LoRA has also been used to fine-tune large language models like GPT-2/3 and LLaMA on domain-specific corpora, instruction-following datasets, and dialogue agents. Notably, LoRA has become a staple in open-source implementations of instruction-tuned models (e.g., Alpaca, Vicuna, and OpenAssistant) [53].

B. Computer Vision

In vision tasks, LoRA is commonly applied to Vision Transformers (ViTs), which share architectural similarities with NLP transformers. Fine-tuning a ViT for new image classification tasks, domain shifts, or even style transfer using LoRA has demonstrated both parameter efficiency and robustness [54].

TABLE III
TOP-1 ACCURACY ON CIFAR-100 WITH VIT-B/16 PRE-TRAINED ON IMAGENET-21K.

Method	Top-1 Accuracy (%)	Trainable Params
Full Fine-Tuning	91.2	85M
Adapter	90.3	5.6M
LoRA ($r = 8$)	90.9	3.1M
LoRA ($r = 16$)	**91.3**	6.2M

Recent work also explores LoRA in vision-language models such as CLIP, enabling adaptation to new multi-modal retrieval, captioning, or VQA datasets with minimal computational overhead.

C. Multi-Modal and Cross-Domain Applications

LoRA excels in multi-modal models that combine vision, language, and audio modalities. Models like Flamingo, BLIP, and LLaVA have incorporated LoRA to train task-specific heads or modify attention flows between modalities [55]. Applications include:

- **Visual Question Answering (VQA):** LoRA modules fine-tune cross-attention layers for language-image alignment.
- **Image Captioning:** LoRA adapts language decoders conditioned on vision embeddings.
- **Text-to-Image Generation:** In models like Stable Diffusion, LoRA modules are inserted into UNet and CLIP components to personalize generation to user-defined concepts (e.g., LoRA DreamBooth).

D. Resource-Constrained and On-Device Adaptation

Because LoRA minimizes trainable parameters and allows post-training merging, it is well-suited to edge and mobile deployment. It enables:

- **On-device personalization:** Users can fine-tune small LoRA modules without needing to update the full model.
- **Model composition:** Multiple LoRA modules can be loaded and switched dynamically to handle task-switching efficiently [56].

E. Performance Under Low-Data Regimes

LoRA has proven effective in few-shot and low-resource settings [57]. Its strong inductive bias—via low-rank constraint—prevents overfitting and facilitates generalization when data is limited. For example, fine-tuning GPT-2 on a few thousand domain-specific instructions with LoRA often outperforms full fine-tuning and even larger models fine-tuned on larger datasets.

F. Summary of Empirical Findings

Overall, empirical studies consistently demonstrate that LoRA:

- Matches or exceeds full fine-tuning performance in many cases.
- Drastically reduces memory usage and compute cost.
- Enables modularity, scalability, and deployment flexibility.
- Generalizes well in low-data and zero-shot scenarios.

In the next section, we explore theoretical interpretations of LoRA's effectiveness, including its connections to linear subspace learning, optimization geometry, and generalization theory [58].

V. Theoretical Perspectives and Foundations

A. LoRA as Low-Rank Subspace Learning

The core premise of LoRA hinges on the hypothesis that fine-tuning a large-scale model often requires updates confined to a low-dimensional subspace of the parameter space. Let $\mathcal{W} \subset \mathbb{R}^{d\times k}$ denote the full space of possible weight updates. LoRA assumes the existence of a subspace $\mathcal{S}_r \subset \mathcal{W}$ of dimension $r \ll \min(d, k)$ such that:

$$\Delta W^* = \arg\min_{\Delta W \in \mathcal{S}_r} \mathcal{L}(W_0 + \Delta W)$$

where $\mathcal{L}$ is the task-specific loss function [59]. By optimizing within $\mathcal{S}_r$, parameter updates remain expressive yet efficient. The structure imposed by the low-rank factorization AB restricts ΔW to a specific family of linear transformations, similar in spirit to PCA-like subspace approximations [60].

B. Optimization Geometry and Inductive Bias

From an optimization geometry standpoint, the low-rank constraint in LoRA can be interpreted as imposing a structured manifold over the parameter space [61]. Consider the Riemannian manifold of rank-r matrices:

$$\mathcal{M}_r = \left\{W \in \mathbb{R}^{d\times k} : \operatorname{rank}(W) \leq r\right\}$$

Optimization on this manifold, as in LoRA, introduces an inductive bias toward compact, low-complexity representations. This bias acts as a form of regularization, which may explain LoRA's favorable generalization, especially in low-data regimes. Further, the constrained optimization problem:

$$\min_{A,B} \mathcal{L}(W_0 + AB)$$

defines a non-convex objective. However, empirical evidence shows that standard optimizers such as Adam often converge to effective solutions, suggesting that the loss landscape within $\mathcal{M}_r$ is relatively benign [62].

C. Gradient Projection Interpretation

An alternative perspective views LoRA as projecting the full gradient onto a low-rank subspace. Given the gradient $\nabla_W \mathcal{L}$ at W_0, LoRA updates W via:

$$W \leftarrow W_0 + \eta \cdot \mathcal{P}_r\left(\nabla_W \mathcal{L}\right)$$

where $\mathcal{P}_r$ denotes a rank-r projection operator. In practice, training LoRA corresponds to learning the best projection direction implicitly through A and B. This interpretation aligns with classical ideas in optimization where low-rank projections are used to reduce the variance of updates or to enforce constraints (e.g., in trust region methods or natural gradient descent).

D. Generalization Bounds

Theoretical work on low-rank matrix learning provides useful insights into LoRA's generalization ability [63]. Under the assumption that W lies in a bounded trace-norm ball, classical Rademacher complexity bounds imply that the generalization error scales with the rank r, not with the full number of parameters:

$$\mathfrak{R}_n(\mathcal{F}) \leq \frac{C \cdot \|W\|_*}{\sqrt{n}} \quad \Rightarrow \quad \mathcal{E}_{\text{gen}} = \mathcal{O}\left(\frac{r(d+k)}{\sqrt{n}}\right)$$

where $\|W\|_*$ denotes the nuclear norm (sum of singular values), and n is the number of samples. LoRA's constrained parameterization effectively limits $\|W\|_*$, thereby controlling overfitting and improving generalization in small-n regimes.

E. Connections to Matrix Factorization and Meta-Learning

LoRA's low-rank design is closely related to classical matrix factorization problems, such as:

- **PCA:** Projects high-dimensional data to a low-dimensional subspace [64].
- **SVD-Based Compression:** Decomposes a weight matrix W into singular vectors to approximate it with a rank-r structure.

Moreover, LoRA can be interpreted through the lens of meta-learning, where A and B act as meta-parameters that specialize the base model W_0 to new tasks. This interpretation is especially compelling in scenarios where multiple LoRA modules are trained for different tasks, and a meta-controller selects or composes them at inference time [65].

F. Theoretical Limits and Open Questions

Despite its empirical success, the theoretical limits of LoRA remain an open area of investigation. Key questions include:

- **Expressivity:** What class of transformations can be effectively captured by rank-r LoRA updates [66]?
- **Identifiability:** Given W_0 and training data, can the optimal A, B decomposition be uniquely recovered [67]?
- **Adaptivity:** Can rank r be learned or adjusted dynamically based on task complexity?
- **Transferability:** Do LoRA modules trained on one task generalize or transfer well to related tasks?

Preliminary research suggests that LoRA strikes a favorable balance between expressivity and regularization, but further theoretical study is needed to formalize these properties [68].

VI. Variants and Extensions of LoRA

Since its inception, LoRA has inspired a rich set of variants aimed at improving its expressiveness, modularity, and applicability to diverse architectures and tasks. These extensions preserve LoRA's core principles—parameter-efficient adaptation via low-rank decomposition—while enhancing flexibility and scope.

A. Grouped and Per-Head LoRA

Standard LoRA applies the same low-rank update across all attention heads [69]. However, different heads may encode distinct functionalities. Grouped LoRA (or Head-wise LoRA) assigns separate (A_i, B_i) modules per head i in multi-head attention:

$$W_i^Q = W_{0,i}^Q + A_i^Q B_i^Q, \quad i = 1, \ldots, h$$

where h is the number of heads. This formulation increases expressiveness by enabling heterogeneous adaptation across heads. Although it increases the number of trainable parameters by a factor of h, the resulting model may achieve higher task performance, particularly on multi-task and multi-domain settings.

B. Layer-Wise and Selective LoRA

Layer-wise LoRA involves applying LoRA modules only to a subset of layers in the transformer stack. This can be done based on:

- **Fixed schedules:** E.g., only apply LoRA to the last k layers.
- **Importance scores:** Use gradient-based metrics to identify layers with high sensitivity to adaptation.

Formally, let $\mathcal{L}$ denote the set of layers selected for LoRA injection. Then for each $l \in \mathcal{L}$:

$$W_l = W_{0,l} + A_l B_l$$

This strategy reduces memory and compute further while retaining adaptation capability where it matters most.

C. Dynamic Rank LoRA

Dynamic LoRA removes the need to predefine the rank r [70]. Instead, it learns r or prunes unused directions during training. One method is to define:

$$\Delta W = A \cdot \text{diag}(s) \cdot B$$

where $s \in \mathbb{R}^r$ is a learned scale vector [71]. Sparsity constraints or regularization (e.g., ℓ_1 penalty) on s encourage automatic rank selection. This formulation balances capacity and efficiency dynamically [72].

D. LoRA Fusion and Composition

When multiple LoRA modules are trained on different tasks or domains, fusion techniques aim to combine them. Let $\{A_i, B_i\}_{i=1}^n$ be LoRA modules for n tasks. Fusion strategies include:

- **Linear fusion:**

$$\Delta W = \sum_{i=1}^{n} \lambda_i A_i B_i, \quad \sum \lambda_i = 1$$

- **Gated or conditional fusion:** Use learned gates or controllers to activate subsets of LoRA modules based on input.
- **Mixture-of-LoRA:** Maintain a bank of LoRA experts and route inputs via a learned policy.

Such compositional schemes allow for efficient multi-task adaptation without retraining the base model.

E. LoRA + Prompt Tuning Hybrids

Combining LoRA with prompt-based methods leverages complementary strengths: LoRA modifies the model weights, while soft prompts alter the input embeddings or hidden states [73]. A typical hybrid setup might include:

- **Soft Prompt Prefix:** Injected at the input to each transformer block [74].
- **LoRA Module:** Applied to attention projection matrices [75].

This combination has been shown to outperform either method alone, especially in few-shot settings.

F. DropLoRA and Regularization Extensions

To prevent overfitting and improve robustness, several regularization schemes for LoRA have been proposed:

- **DropLoRA:** Randomly zeroes out rank components or entire LoRA modules during training.
- **Orthogonal LoRA:** Constrains A and B to be approximately orthogonal to prevent redundancy.
- **Stochastic LoRA:** Samples r dynamically from a distribution each step, encouraging robustness.

These approaches draw inspiration from dropout, ensemble learning, and stochastic regularization techniques.

G. Domain-Specific and Architecture-Aware Variants

LoRA has also been tailored for specific domains and architectures [76]. Examples include:

- **ConvLoRA:** Adapts LoRA to convolutional kernels in CNN-based vision models [77].
- **LoRA-ViT:** Designs LoRA modules that respect spatial locality in vision transformers [78].
- **TimeLoRA:** Specialized for temporal modeling in sequence models (e.g., transformers for time-series).
- **LoRA-RNN:** Applies LoRA to gate matrices in recurrent networks like LSTMs [79].

These variations illustrate the broad applicability of LoRA's core idea across deep learning architectures.

H. Emerging Directions

Finally, ongoing research continues to push the boundaries of LoRA:

- **Neural Architecture Search for LoRA placement.**
- **LoRA-aware optimizers** that adapt learning rates or weight decay based on low-rank structure [80].
- **LoRA distillation** to compress learned adapters into a single monolithic model [81].

These developments reflect the growing maturity and ecosystem around LoRA, suggesting it may become a fundamental building block in modular and adaptive AI systems.

VII. Comparison with Other Parameter-Efficient Fine-Tuning (PEFT) Methods

The emergence of LoRA has sparked renewed interest in parameter-efficient fine-tuning (PEFT), which aims to adapt large pre-trained models to new tasks while updating only a small subset of parameters [82]. In this section, we position LoRA within the broader PEFT taxonomy, comparing it with prominent alternatives in terms of design philosophy, efficiency, generalization, and practical deployment [83].

A. PEFT Design Space

The space of PEFT methods can be roughly categorized into the following paradigms:

- **Adapter-based methods:** Insert small feedforward layers (adapters) into the model while keeping base weights frozen [30] [84].
- **Prompt-based tuning:** Optimize task-specific prompts that influence the model without modifying internal weights [85].
- **Bias-only tuning:** Only update bias terms in the network [**?**].
- **LoRA-based methods:** Add trainable low-rank updates to weight matrices [34] [86].

Each approach offers a different balance between flexibility, interpretability, and efficiency [87].

B. Comparative Table

We summarize core characteristics of prominent PEFT methods in Table IV.

TABLE IV
COMPARISON OF PEFT METHODS. TRAIN [88]. PARAMS REFERS TO THE FRACTION OF TRAINABLE PARAMETERS. ΔPERF INDICATES TYPICAL DROP OR GAIN RELATIVE TO FULL FINE-TUNING.

Method	Train [89]. Params	Inference Overhead	Mergeable	Task Transfer	ΔPerf
Full Fine-Tuning	100%	None	N/A	Low	0.0
BitFit [**?**]	∼0.1%	None	Yes	Low	-1.0 to -3.0
Adapters [30]	1–4%	High	No	Medium	-0.5 to -2.0
Prompt Tuning [85]	∼0.1%	Low	No	Medium	-1.0 to -3.0
Prefix Tuning [32]	∼1%	Moderate	No	High	-0.5 to -2.5
LoRA ($r = 8$) [34]	0.5–2%	Low	Yes	High	-0.1 to +0.2

C. Mergeability and Deployment Considerations

A key advantage of LoRA over other PEFT methods (e.g., adapters, prefix tuning) is that the trained updates can be *merged* into the base weights for inference:

$$W_{\text{inferred}} = W_0 + AB$$

This avoids extra forward-pass computations and memory overhead at inference time, making LoRA highly favorable for real-time and edge deployment [90]. In contrast, prompt- and adapter-based methods require maintaining auxiliary parameters during inference, which may increase latency and resource usage [91].

D. Task Transfer and Modularity

LoRA modules are compositional and often exhibit strong task modularity. When multiple LoRA modules are trained independently, they can be selectively activated or combined for multi-task inference, which is non-trivial for most adapter-based or prompt-tuning methods. Moreover, LoRA supports dynamic swapping and conditional routing, enabling efficient task switching. This property has catalyzed its adoption in open-source instruction-tuned models, where community-contributed LoRA modules are hot-swappable [92].

E. Empirical Trade-Offs

While LoRA often achieves near-parity with full fine-tuning, its advantage depends on several factors:

- **Model architecture:** LoRA performs better on transformers than on CNNs or RNNs.
- **Target task:** It excels in text classification, generation, and instruction tuning; results may vary in complex structured prediction tasks.
- **Data availability:** LoRA shines in low-resource regimes due to its regularization effect from low-rank constraints.

Figure 1 illustrates the Pareto frontier of PEFT methods in terms of performance vs. trainable parameters.

F. Summary

LoRA distinguishes itself from other PEFT approaches by offering:

- Competitive performance with minimal parameter footprint.
- Mergeability into base weights for zero-inference overhead.
- High modularity and support for task composition.
- Strong generalization in few-shot and low-data scenarios [93].

These advantages have led to its widespread adoption in open-source and industrial settings alike, and position it as a cornerstone technique in the PEFT toolkit.

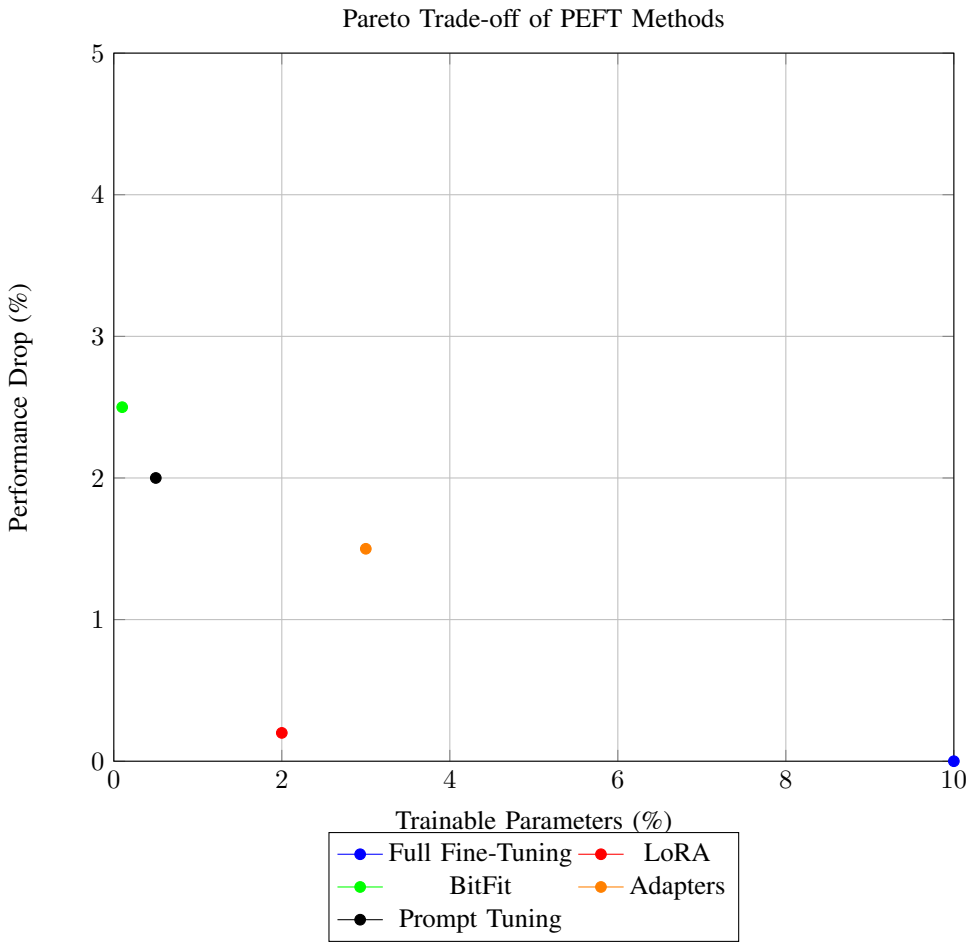

Fig. 1. Pareto trade-off between model performance and number of trainable parameters across PEFT techniques. LoRA achieves an optimal balance in most settings.

VIII. Open Challenges and Future Directions

While LoRA has established itself as a powerful and versatile paradigm for parameter-efficient fine-tuning, several open challenges and unresolved questions remain. These limitations point to fertile directions for theoretical analysis, algorithmic innovation, and practical refinement [94].

A. Theoretical Understanding and Expressivity Bounds

Despite its empirical success, a formal understanding of LoRA's expressivity remains incomplete. Key questions include:

- What is the precise class of transformations or tasks that a rank-r LoRA update can approximate [95]?
- Can we characterize the approximation gap between full-rank fine-tuning and LoRA-constrained optimization [96]?
- How does the interaction between LoRA modules and the frozen backbone affect convergence properties and generalization bounds?

Answering these questions requires deeper integration of LoRA analysis with tools from matrix theory, optimization geometry, and statistical learning theory [97].

B. Automatic Rank Selection and Capacity Control

Current LoRA implementations require manually setting the rank r, a hyperparameter that significantly impacts performance and memory footprint[98]. While heuristic grid searches are common, this approach is inefficient and brittle [99]. Future work should explore:

- Bayesian or differentiable methods for rank selection [100].
- Adaptive LoRA schemes where the model learns to expand or prune ranks during training.
- Task- and data-dependent rank modulation, possibly guided by learned importance scores [101].

Formally, we seek mechanisms that solve:

$$\min_{r \in \mathbb{Z}_+} \mathcal{L}(W_0 + A_r B_r) + \lambda \cdot \mathcal{C}(r)$$

where $\mathcal{C}(r)$ penalizes complexity, storage, or inference cost.

C. LoRA for Structured and Non-Textual Modalities

While LoRA has shown promise in language models, its adaptation to other modalities and architectures remains underexplored. Future work may focus on:

- **Vision Transformers:** How should spatial structure and inductive biases be preserved in LoRA modules?
- **Speech and Audio:** Can LoRA capture domain-specific correlations such as temporal smoothness [102]?
- **Graph Neural Networks:** Is there an analog of low-rank adaptation that respects graph topology?
- **Multimodal Fusion:** How can LoRA be designed to align and adapt across modalities?

Cross-modal LoRA architectures, including shared or hybrid low-rank spaces, offer a compelling avenue for unifying adaptation strategies [103].

D. Security, Privacy, and Robustness Considerations

As LoRA modules become widely distributed and integrated into production models, concerns about safety and security arise [104]. Open challenges include:

- **Trojan LoRA modules:** How can we detect maliciously trained low-rank adapters that behave normally on validation data but trigger specific behaviors under certain prompts [105]?
- **Privacy leakage:** Do LoRA updates trained on private data risk memorizing or leaking sensitive information?
- **Adversarial robustness:** How does low-rank adaptation affect a model's susceptibility to perturbations or adversarial attacks [106]?

Auditing and certifying LoRA modules may require novel forms of verification, trust calibration, or differentially private LoRA training techniques [107].

E. Interoperability and Standards

The rapid growth of community-contributed LoRA modules has outpaced the development of consistent standards for:

- Interface design (e.g., where and how LoRA is injected) [108].
- Metadata (e.g., dataset, rank, seed, and hyperparameter tracking) [109].
- Evaluation protocols (e.g., zero-shot vs. few-shot, task metrics, domain coverage) [110].

Developing robust interoperability standards will be essential to support reproducibility, composability, and secure reuse of LoRA modules across diverse ecosystems.

F. Continual Learning and Lifelong Adaptation

LoRA offers a promising foundation for continual learning, where the goal is to adapt a model incrementally without forgetting previous tasks. Open challenges include:

- How to design LoRA modules that accumulate knowledge across time without catastrophic interference [111]?
- How to compose LoRA modules learned over disjoint datasets or domains?
- How to manage memory and inference cost as the number of LoRA modules grows [112]?

One direction involves sparsifying or distilling historical LoRA modules into compact representations while preserving functionality.

G. LoRA Beyond Transformers

Lastly, extending LoRA principles beyond transformer architectures remains largely unexplored. Can similar low-rank adaptation principles be fruitfully applied to:

- Convolutional neural networks [113]?
- Diffusion models [114]?
- Reinforcement learning policies?
- Meta-learning systems?

Emerging work suggests that LoRA-style adaptation may serve as a general recipe for modular and efficient specialization in deep learning.

H. Summary and Outlook

As large foundation models become the norm across AI disciplines, the need for lightweight, scalable adaptation methods will only intensify. LoRA has emerged as a critical primitive in this space, offering strong empirical performance with elegant theoretical motivations. However, realizing its full potential will require addressing foundational questions in theory, architecture, security, and systems [115]. The future of LoRA lies not just in refining its components, but in reimagining its role as a modular interface between massive pre-trained models and the rapidly evolving landscape of downstream applications [116].

IX. Conclusion

In the era of foundation models, where pretraining scales to hundreds of billions of parameters and downstream applications span diverse domains, the need for efficient, modular, and adaptable fine-tuning strategies has become paramount. Low-Rank Adaptation (LoRA) has emerged as one of the most practical and theoretically grounded solutions to this challenge.

This survey has traced the arc of LoRA from its core mathematical formulation—injecting low-rank matrices into frozen linear layers—to its ecosystem of variants, extensions, and hybrid models. We have shown how LoRA excels across multiple dimensions:

- **Parameter Efficiency:** LoRA enables tuning with as little as 0.1% of the full model parameters.
- **Performance Robustness:** Empirically competitive with full fine-tuning across a wide range of NLP and multimodal tasks.
- **Modularity and Composability:** Supports plug-and-play adaptation, task fusion, and dynamic routing.
- **Scalability and Deployment:** Mergeable at inference time, making it ideal for edge and production use.

Through comparative analysis, we highlighted LoRA's position within the broader PEFT landscape and discussed trade-offs relative to prompt tuning, adapters, and bias-only methods. We also explored its adoption in open-source ecosystems, such as HuggingFace and OpenLLaMA, where community-contributed LoRA modules are accelerating the pace of innovation.

At the same time, we identified key open challenges—from automatic rank selection and multimodal generalization to robust security protocols and theoretical foundations. These represent compelling frontiers for future research, and we posit that LoRA is not merely an optimization trick, but a paradigm shift in how large models can be adapted, shared, and composed.

As foundation models grow more universal and compute budgets more constrained, techniques like LoRA offer a critical blueprint for achieving scalable intelligence. By enabling efficient specialization without retraining the world, LoRA is paving the way toward truly democratized and modular AI.

References

[1] S. Jie and Z.-H. Deng, "Fact: Factor-tuning for lightweight adaptation on vision transformer," in *Proceedings of the AAAI conference on artificial intelligence*, 2023.

[2] G. Kim, S. Kim, and S. Lee, "Aapl: Adding attributes to prompt learning for vision-language models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[3] Y. Yan, S. Tang, Z. Shi, and Q. Yang, "FeDeRA: Efficient fine-tuning of language models in federated learning leveraging weight decomposition," *arXiv preprint arXiv:2404.18848*, 2024.

[4] J. Xie, W. Mao, Z. Bai, D. J. Zhang, W. Wang, K. Q. Lin, Y. Gu, Z. Chen, Z. Yang, and M. Z. Shou, "Show-o: One single transformer to unify multimodal understanding and generation," *arXiv preprint arXiv:2408.12528*, 2024.

[5] A. Bhatti, S. Parmar, and S. Lee, "SM70: A large language model for medical devices," *arXiv preprint arXiv:2312.06974*, 2023.

[6] K. Zhang and D. Liu, "Customized segment anything model for medical image segmentation," *arXiv preprint arXiv:2304.13785*, 2023.

[7] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[8] M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.-N. Lim, "Visual prompt tuning," in *European Conference on Computer Vision*, 2022.

[9] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[10] X. Wang, Y. Yang, M. Zhu, K. Zheng, S. Liu, and W. Chen, "Mept: Multi-representation guided prompt tuning for vision-language model," *arXiv preprint arXiv:2408.09706*, 2024.

[11] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023.

[12] Y. Lin, X. Ma, X. Chu, Y. Jin, Z. Yang, Y. Wang, and H. Mei, "Lora dropout as a sparsity regularizer for overfitting control," *arXiv preprint arXiv:2404.09610*, 2024.

[13] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, pp. 1 – 35, 2021.

[14] J. Pan, Z. Lin, X. Zhu, J. Shao, and H. Li, "St-adapter: Parameter-efficient image-to-video transfer learning," *Advances in Neural Information Processing Systems*, 2022.

[15] R. Feng, W. Weng, Y. Wang, Y. Yuan, J. Bao, C. Luo, Z. Chen, and B. Guo, "Ccedit: Creative and controllable video editing via diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[16] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," *arXiv preprint arXiv:2303.05499*, 2023.

[17] J. Devlin, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[18] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[19] A. DOSOVITSKIY, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[20] Y. Wu, Y. Xiang, S. Huo, Y. Gong, and P. Liang, "Lora-sp: streamlined partial parameter adaptation for resource efficient fine-tuning of large language models," in *Third International Conference on Algorithms, Microchips, and Network Applications*, 2024, pp. 488–496.

[21] Y. Zniyed, T. P. Nguyen *et al.*, "Efficient tensor decomposition-based filter pruning," *Neural Networks*, vol. 178, p. 106393, 2024.

[22] S. Liu, C. Wang, H. Yin, P. Molchanov, Y. F. Wang, K. Cheng, and M. Chen, "Dora: Weight-decomposed low-rank adaptation," *arXiv preprint arXiv.2402.09353*, 2024.

[23] H. Yuan, Z. Yuan, R. Gan, J. Zhang, Y. Xie, and S. Yu, "Biobart: Pretraining and evaluation of a biomedical generative language model," in *Workshop on Biomedical Natural Language Processing*, 2022.

[24] X. Xia, D. Zhang, Z. Liao, Z. Hou, T. Sun, J. Li, L. Fu, and Y. Dong, "Scenegenagent: Precise industrial scene generation with coding agent," *arXiv preprint arXiv:2410.21909*, 2024.

[25] Z. Tang, Z. Yang, M. Khademi, Y. Liu, C. Zhu, and M. Bansal, "Codi-2: In-context interleaved and interactive any-to-any generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[26] Z. Jin and Z. Song, "Generating coherent comic with rich story using chatgpt and stable diffusion," *arXiv preprint arXiv:2305.11067*, 2023.

[27] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Neural Information Processing Systems*, 2019.

[28] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "Boolq: Exploring the surprising difficulty of natural yes/no questions," *ArXiv*, vol. abs/1905.10044, 2019.

[29] Z. Li, B. Yang, Q. Liu, Z. Ma, S. Zhang, J. Yang, Y. Sun, Y. Liu, and X. Bai, "Monkey: Image resolution and text label are important things for large multi-modal models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[30] N. Houlsby, A. Giurgiu, X. Hu, N. Goyal, A. Rashid, A. Vaswani, and N. Shazeer, "Parameter-efficient transfer learning for nlp," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

[31] W. G. C. Bandara and V. M. Patel, "Attention prompt tuning: Parameter-efficient adaptation of pre-trained models for spatiotemporal modeling," *arXiv preprint arXiv:2403.06978*, 2024.

[32] X. Li, P. Liang, Z. Zhang, D. Xie, Y. Li, Y. Zhang, H. Zhang, Y. Wang, and R. Wang, "Prefix-tuning: Optimizing continuous prompts for generation," in *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2021.

[33] H. Lin, J. Cho, A. Zala, and M. Bansal, "Ctrl-adapter: An efficient and versatile framework for adapting diverse controls to any diffusion model," *arXiv preprint arXiv:2404.09967*, 2024.

[34] E. Hu, Y. Tsai, K. Hsu, A. Sussman, K. Eleni, B. Martinez, F. Martinez, and R. Charle, "Lora: Low-rank adaptation of large language models," in *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

[35] S. Chen, Z. Jie, and L. Ma, "Llava-mole: Sparse mixture of lora experts for mitigating data conflicts in instruction finetuning mllms," *arXiv preprint arXiv.2401.16160*, 2024.

[36] M. Santacroce, Y. Lu, H. Yu, Y. Li, and Y. Shen, "Efficient RLHF: reducing the memory usage of PPO," *arXiv preprint arXiv:2309.00754*, 2023.

[37] W. Jiang, B. Lin, H. Shi, Y. Zhang, Z. Li, and J. T. Kwok, "Effective and parameter-efficient reusing fine-tuned models," *arXiv preprint arXiv.2310.01886*, 2023.

[38] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-Or, "An image is worth one word: Personalizing text-to-image generation using textual inversion," *arXiv preprint arXiv:2208.01618*, 2022.

[39] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," https://github.com/tatsu-lab/stanford$_alpaca$, 2023.

[40] T. Jiang, S. Huang, S. Luo, Z. Zhang, H. Huang, F. Wei, W. Deng, F. Sun, Q. Zhang, D. Wang *et al.*, "Mora: High-rank updating for parameter-efficient fine-tuning," *arXiv preprint arXiv:2405.12130*, 2024.

[41] J. Zhang, S. Chen, J. Liu, and J. He, "Composing parameter-efficient modules with arithmetic operations," *arXiv preprint arXiv.2306.14870*, 2023.

[42] L. Yi, H. Yu, G. Wang, X. Liu, and X. Li, "pFedLoRA: Model-Heterogeneous Personalized Federated Learning with LoRA Tuning," *arXiv preprint arXiv:2310.13283*, 2023.

[43] R. Liu, R. Wu, B. V. Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, "Zero-1-to-3: Zero-shot one image to 3d object," 2023.

[44] L. Han, Y. Li, H. Zhang, P. Milanfar, D. Metaxas, and F. Yang, "Svdiff: Compact parameter space for diffusion fine-tuning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[45] D. Guo, A. M. Rush, and Y. Kim, "Parameter-efficient transfer learning with diff pruning," in *Annual Meeting of the Association for Computational Linguistics*, 2020.

[46] S. Quan, "Dmoerm: Recipes of mixture-of-experts for effective reward modeling," *arXiv preprint arXiv:2403.01197*, 2024.

[47] X. Liu, Q. Chen, C. Deng, H.-J. Zeng, J. Chen, D. Li, and B. Tang, "Lcqmc:a large-scale chinese question matching corpus," in *International Conference on Computational Linguistics*, 2018.

[48] Y. Hu, Y. Xie, T. Wang, M. Chen, and Z. Pan, "Structure-aware low-rank adaptation for parameter-efficient fine-tuning," *Mathematics*, vol. 11, no. 20, p. 4317, 2023.

[49] S. Yoo, K. Kim, V. G. Kim, and M. Sung, "As-plausible-as-possible: Plausibility-aware mesh deformation using 2d diffusion priors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4315–4324.

[50] S. Li, H. Lu, T. Wu, M. Yu, Q. Weng, X. Chen, Y. Shan, B. Yuan, and W. Wang, "Caraserve: Cpu-assisted and rank-aware lora serving for generative llm inference," *arXiv preprint arXiv:2401.11240*, 2024.

[51] Y. Ren, Y. Zhou, J. Yang, J. Shi, D. Liu, F. Liu, M. Kwon, and A. Shrivastava, "Customize-a-video: One-shot motion customization of text-to-video diffusion models," *ECCV*, 2024.

[52] A. X. Yang, M. Robeyns, T. Coste, J. Wang, H. Bou-Ammar, and L. Aitchison, "Bayesian reward models for LLM alignment," *arXiv preprint arXiv:2402.13210*, 2024.

[53] H. Zhao, B. Ni, H. Wang, J. Fan, F. Zhu, Y. Wang, Y. Chen, G. Meng, and Z. Zhang, "Continual forgetting for pre-trained vision models," *arXiv preprint arXiv.2403.11530*, 2024.

[54] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[55] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, "Palm: Scaling language modeling with pathways," *J. Mach. Learn. Res.*, vol. 24, pp. 240:1–240:113, 2023.

[56] Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, and T. Zhao, "Adaptive budget allocation for parameter-efficient fine-tuning," in *The Eleventh International Conference on Learning Representations*, 2023.

[57] C. Wang, Z. Duan, B. Liu, X. Zou, C. Chen, K. Jia, and J. Huang, "Pai-diffusion: Constructing and serving a family of open chinese diffusion models for text-to-image synthesis on the cloud," *arXiv preprint arXiv:2309.05534*, 2023.

[58] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[59] P. Liu, Z.-F. Gao, W. X. Zhao, Z. Y. Xie, Z.-Y. Lu, and J. rong Wen, "Enabling lightweight fine-tuning for pre-trained language model compression based on matrix product operators," in *Annual Meeting of the Association for Computational Linguistics*, 2021.

[60] V. Lialin, S. Muckatira, N. Shivagunde, and A. Rumshisky, "Relora: High-rank training through low-rank updates," in *The Twelfth International Conference on Learning Representations*, 2023.

[61] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020.

[62] K. Simonyan, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[63] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*, 2016.

[64] C. Liu, K. Sun, Q. Zhou, Y. Duan, J. Shu, H. Kan, Z. Gu, and J. Hu, "Cpmi-chatglm: Parameter-efficient fine-tuning chatglm with chinese patent medicine instructions," *Scientific Reports*, vol. 14, no. 1, p. 6403, 2024.

[65] Y. Zha, J. Wang, T. Dai, B. Chen, Z. Wang, and S.-T. Xia, "Instance-aware dynamic prompt tuning for pre-trained point cloud models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[66] W. Zhong, R. Cui, Y. Guo, Y. Liang, S. Lu, Y. Wang, A. Saied, W. Chen, and N. Duan, "Agieval: A human-centric benchmark for evaluating foundation models," *arXiv preprint arXiv:2304.06364*, 2023.

[67] Z. Gao, Q. Wang, A. Chen, Z. Liu, B. Wu, L. Chen, and J. Li, "Parameter-efficient fine-tuning with discrete fourier transform," *arXiv preprint arXiv:2405.03003*, 2024.

[68] Y. Ge, S. Zhao, J. Zhu, Y. Ge, K. Yi, L. Song, C. Li, X. Ding, and Y. Shan, "Seed-x: Multimodal models with unified multi-granularity comprehension and generation," *arXiv preprint arXiv:2404.14396*, 2024.

[69] D. Chen, "Aggregate, decompose, and fine-tune: A simple yet effective factor-tuning method for vision transformer," *arXiv preprint arXiv:2311.06749*, 2023.

[70] L. Ran, X. Cun, J.-W. Liu, R. Zhao, S. Zijie, X. Wang, J. Keppo, and M. Z. Shou, "X-adapter: Adding universal compatibility of plugins for upgraded diffusion model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[71] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *International conference on machine learning*. PMLR, 2022.

[72] Y. Guo, X. Gao, and B. Jiang, "An empirical study on jit defect prediction based on bert-style model," *arXiv preprint arXiv:2403.11158*, 2024.

[73] A. P. Gema, L. Daines, P. Minervini, and B. Alex, "Parameter-efficient fine-tuning of llama for the clinical domain," *arXiv preprint arXiv:2307.03042*, 2023.

[74] W. Tan, W. Zhang, S. Liu, L. Zheng, X. Wang, and B. An, "True knowledge comes from practice: Aligning llms with embodied environments via reinforcement learning," *arXiv preprint arXiv:2401.14151*, 2024.

[75] M. Cai, H. Liu, S. K. Mustikovela, G. P. Meyer, Y. Chai, D. Park, and Y. J. Lee, "Vip-llava: Making large multimodal models understand arbitrary visual prompts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[76] C. Louizos, M. Welling, and D. P. Kingma, "Learning sparse neural networks through l_0 regularization," *arXiv preprint arXiv.1712.01312*, 2017.

[77] S.-Y. Yeh, Y.-G. Hsieh, Z. Gao, B. B. Yang, G. Oh, and Y. Gong, "Navigating text-to-image customization: From lycoris fine-tuning to model evaluation," in *The Twelfth International Conference on Learning Representations*, 2023.

[78] Y. Ren, Y. Zhou, J. Yang, J. Shi, D. Liu, F. Liu, M. Kwon, and A. Shrivastava, "Customize-a-video: One-shot motion customization of text-to-video diffusion models," *arXiv preprint arXiv:2402.14780*, 2024.

[79] W. Dong, S. Xue, X. Duan, and S. Han, "Prompt tuning inversion for text-driven image editing using diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[80] M. Huh, B. Cheung, J. Bernstein, P. Isola, and P. Agrawal, "Training neural networks from scratch with parallel low-rank adapters," *arXiv preprint arXiv.2402.16828*, 2024.

[81] E. Xie, L. Yao, H. Shi, Z. Liu, D. Zhou, Z. Liu, J. Li, and Z. Li, "Difffit: Unlocking transferability of large diffusion models via simple parameter-efficient fine-tuning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[82] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen, J. Yi, W. Zhao, X. Wang, Z. Liu, H. Zheng, J. Chen, Y. Liu, J. Tang, J. Li, and M. Sun, "Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models," *ArXiv*, vol. abs/2203.06904, 2022.

[83] D. Biderman, J. J. G. Ortiz, J. Portes, M. Paul, P. Greengard, C. Jennings, D. King, S. Havens, V. Chiley, J. Frankle, C. Blakeney, and J. P. Cunningham, "Lora learns less and forgets less," *arXiv preprint arXiv.2405.09673*, 2024.

[84] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, L. Li, and Z. Sui, "A survey for in-context learning," *arXiv preprint arXiv.2301.00234*, 2023.

[85] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[86] R. Xu, F. Luo, Z. Zhang, C. Tan, B. Chang, S. Huang, and F. Huang, "Raise a child in large language model: Towards effective and generalizable fine-tuning," *ArXiv*, vol. abs/2109.05687, 2021.

[87] Y. Wang, S. Mukherjee, X. Liu, J. Gao, A. H. Awadallah, and J. Gao, "Adamix: Mixture-of-adapter for parameter-efficient tuning of large language models," *ArXiv*, vol. abs/2205.12410, 2022.

[88] M. Gheini, X. Ma, and J. May, "Know where you're going: Meta-learning for parameter-efficient fine-tuning," in *Annual Meeting of the Association for Computational Linguistics*, 2022.

[89] Z. Ye, L. Lovell, A. Faramarzi, and J. Ninic, "Sam-based instance segmentation models for the automation of structural damage detection," *arXiv preprint arXiv:2401.15266*, 2024.

[90] H. Chefer, S. Zada, R. Paiss, A. Ephrat, O. Tov, M. Rubinstein, L. Wolf, T. Dekel, T. Michaeli, and I. Mosseri, "Still-moving: Customized video generation without customized video data," *arXiv preprint arXiv:2407.08674*, 2024.

[91] D. Bershatsky, D. Cherniuk, T. Daulbaev, A. Mikhalev, and I. V. Oseledets, "Lotr: Low tensor rank weight adaptation," *arXiv preprint arXiv.2402.01376*, 2024.

[92] J. Zhao, Z. Zhang, B. Chen, Z. Wang, A. Anandkumar, and Y. Tian, "Galore: Memory-efficient LLM training by gradient low-rank projection," *arXiv preprint arXiv.2403.03507*, 2024.

[93] D. Zhang, Z. Hu, S. Zhoubian, Z. Du, K. Yang, Z. Wang, Y. Yue, Y. Dong, and J. Tang, "Sciglm: Training scientific language models with self-reflective instruction annotation and tuning," *arXiv preprint arXiv:2401.07950*, 2024.

[94] Z. Kong, Y. Zhang, T. Yang, T. Wang, K. Zhang, B. Wu, G. Chen, W. Liu, and W. Luo, "OMG: occlusion-friendly personalized multi-concept generation in diffusion models," *arXiv preprint arXiv:2403.10983*, 2024.

[95] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image captioning with semantic attention," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[96] T. Huang, B. Dong, Y. Yang, X. Huang, R. W. Lau, W. Ouyang, and W. Zuo, "Clip2point: Transfer clip to point cloud classification with image-depth pre-training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[97] Y. Yang, S. Xiong, A. Payani, E. Shareghi, and F. Fekri, "Harnessing the power of large language models for natural language to first-order logic translation," *arXiv preprint arXiv:2305.15541*, 2023.

[98] Y. Zniyed, T. P. Nguyen *et al.*, "Enhanced network compression through tensor decompositions and pruning," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[99] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang, J. Ji, Z. Yang, L. Zhao, X. Song *et al.*, "Cogvlm: Visual expert for pretrained language models," *arXiv preprint arXiv:2311.03079*, 2023.

[100] Y.-L. Sung, V. Nair, and C. Raffel, "Training neural networks with fixed sparse masks," in *Neural Information Processing Systems*, 2021.

[101] J. Park, J. Lee, and K. Sohn, "Dual-path adaptation from image to video transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[102] F. Liang, B. Wu, X. Dai, K. Li, Y. Zhao, H. Zhang, P. Zhang, P. Vajda, and D. Marculescu, "Open-vocabulary semantic segmentation with mask-adapted clip," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
[103] V. Lialin, V. Deshpande, and A. Rumshisky, "Scaling down to scale up: A guide to parameter-efficient fine-tuning," *ArXiv*, vol. abs/2303.15647, 2023.
[104] S. Ayupov and N. Chirkova, "Parameter-efficient finetuning of transformers for source code," *arXiv preprint arXiv:2212.05901*, 2022.
[105] J. H. Yeo, S. Han, M. Kim, and Y. M. Ro, "Where visual speech meets language: VSP-LLM framework for efficient and context-aware visual speech processing," *arXiv preprint arXiv:2402.15151*, 2024.
[106] S. Basu, S. Hu, D. Massiceti, and S. Feizi, "Strong baselines for parameter-efficient few-shot fine-tuning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.
[107] K. Zhang, Y. Zhou, X. Xu, X. Pan, and B. Dai, "Diffmorpher: Unleashing the capability of diffusion models for image morphing," *arXiv preprint arXiv:2312.07409*, 2023.
[108] T. GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Rojas, G. Feng, H. Zhao, H. Lai *et al.*, "Chatglm: A family of large language models from glm-130b to glm-4 all tools," *arXiv preprint arXiv:2406.12793*, 2024.
[109] E. T. K. Sang and F. D. Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," in *Conference on Computational Natural Language Learning*, 2003.
[110] S. Liu, J. Keung, Z. Yang, F. Liu, Q. Zhou, and Y. Liao, "Delving into parameter-efficient fine-tuning in code change learning: An empirical study," *arXiv preprint arXiv:2402.06247*, 2024.
[111] Y. Li, T. Ma, and H. Zhang, "Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations," in *Annual Conference Computational Learning Theory*, 2017.
[112] J. Philip, A. Berard, M. Gallé, and L. Besacier, "Monolingual adapters for zero-shot neural machine translation," in *Conference on Empirical Methods in Natural Language Processing*, 2020.
[113] Y. Wang, X. Jiang, D. Cheng, D. Li, and C. Zhao, "Actprompt: In-domain feature adaptation via action cues for video temporal grounding," *arXiv preprint arXiv:2408.06622*, 2024.
[114] B. Lee, B. Park, C. W. Kim, and Y. M. Ro, "Collavo: Crayon large language and vision model," *arXiv preprint arXiv:2402.11248*, 2024.
[115] J. Cheng, P. Xie, X. Xia, J. Li, J. Wu, Y. Ren, H. Li, X. Xiao, M. Zheng, and L. Fu, "Resadapter: Domain consistent resolution adapter for diffusion models," *arXiv preprint arXiv:2403.02084*, 2024.
[116] X. He, C. Li, P. Zhang, J. Yang, and X. E. Wang, "Parameter-efficient model adaptation for vision transformers," in *Thirty-Seventh AAAI Conference on Artificial Intelligence*, 2023, pp. 817–825.