Published: 06 May 2025
Online AM: 08 April 2025
Accepted: 01 April 2025
Revised: 28 February 2025
Received: 22 March 2024

SURVEY

Efficient Compressing and Tuning Methods for Large Language Models: A Systematic Literature Review

GUN-IL KIM, Yonsei University, Seoul, South Korea

SUNGA HWANG, Yonsei University, Seoul, South Korea

BEAKCHEOL JANG, Yonsei University, Seoul, South Korea

Efficient Compressing and Tuning Methods for Large Language Models: A Systematic Literature Review

GUN IL KIM, Graduate School of Information, Yonsei University, Seoul, Korea (the Republic of)
SUNGA HWANG, Graduate School of Information, Yonsei University, Seoul, Korea (the Republic of)
BEAKCHEOL JANG, Graduate School of Information, Yonsei University, Seoul, Korea (the Republic of)

Efficient compression and tuning techniques have become indispensable in addressing the increasing computational and memory demands of large language models (LLMs). While these models have demonstrated exceptional performance across a wide range of natural language processing tasks, their growing size and resource requirements pose significant challenges to accessibility and sustainability. This survey systematically reviews state-of-the-art methods in model compression, including compression techniques such as knowledge distillation, low-rank approximation, parameter pruning, and quantization, as well as tuning techniques such as parameter-efficient fine-tuning and inference optimization. Compression techniques, though well-established in traditional deep learning, require updated methodologies tailored to the scale and dynamics of LLMs. Simultaneously, parameter-efficient fine-tuning, exemplified by techniques like Low-Rank Adaptation (LoRA) and query tuning, emerges as a promising solution for adapting models with minimal resource overhead. This study provides a detailed taxonomy of these methods, examining their practical applications, strengths, and limitations. Critical gaps are identified in scalability, and the integration of compression and tuning strategies, signaling the need for unified frameworks and hybrid approaches to maximize efficiency and performance. By addressing these challenges, this survey aims at guiding researchers toward sustainable, efficient, and accessible LLM development, ensuring their broader applicability across diverse domains while mitigating resource constraints.

CCS Concepts: • **General and reference → Surveys and overviews**; • **Theory of computation → Design and analysis of algorithms**; • **Computer systems organization → Architectures**; • **Computing methodologies → Natural language processing;**

Additional Key Words and Phrases: Large language models, knowledge distillation, low-rank strategy, quantization, parameter pruning, LoRA, PEFT, inference tuning

ACM Reference Format:
Gun Il Kim, Sunga Hwang, and Beakcheol Jang. 2025. Efficient Compressing and Tuning Methods for Large Language Models: A Systematic Literature Review. *ACM Comput. Surv.* 57, 10, Article 253 (May 2025), 39 pages. https://doi.org/10.1145/3728636

All authors contributed equally to this research.
This work was supported by the National Research Foundation of Korea Fund of RS-2023-00273751.
Authors' Contact Information: Gun Il Kim, Graduate School of Information, Yonsei University, Seoul, Korea (the Republic of); e-mail: kim_gunil_94@yonsei.ac.kr; Sunga Hwang, Graduate School of Information, Yonsei University, Seoul, Korea (the Republic of); e-mail: sungahwang@yonsei.ac.kr; Beakcheol Jang (Corresponding author), Graduate School of Information, Yonsei University, Seoul, Korea (the Republic of); e-mail: bjang@yonsei.ac.kr.

ACM 0360-0300/2025/05-ART253
https://doi.org/10.1145/3728636

1 Introduction

With the advent of **large language models** (**LLMs**), a significant shift has occurred in the research community, with many scholars focusing on the intricate mechanisms that underpin language models at scale within the realm of **natural language processing** (**NLP**). Meanwhile, a diverse group of researchers, multinational corporations, and organizations have turned their efforts toward developing practical, real-world applications spanning various sectors such as healthcare, finance, education, and engineering for end-users. Models like OpenAI's ChatGPT and Google's Gemini leverage massive datasets and billions of parameters, enabling exceptional performance in tasks such as machine translation, question answering, and text generation. Figure 1 illustrates the rapid growth of LLMs over the past three years, marked by increased computational demands measured in **floating point operations per second** (**FLOPS**). The scale and complexity of these models pose substantial challenges for training and deployment, highlighting the need for efficient methods to optimize resource usage while sustaining or enhancing NLP task performance.

LLMs, such as GPT-3 [11], have showcased remarkable capabilities. However, their exponential growth in size has raised concerns about energy efficiency, cost, and accessibility, particularly for **small- and medium-sized enterprises** (**SMEs**) and individual researchers. These challenges highlight the pressing need for innovative approaches to balance computational efficiency with model performance, thereby promoting broader accessibility and sustainability.

Researchers have investigated two primary approaches to address these concerns: compression and tuning. Compression techniques such as knowledge distillation, low-rank approximation, parameter pruning, and quantization aim at reducing the memory and computational demands of LLMs with minimal impact on performance. Tuning methods, including parameter-efficient fine-tuning, query tuning, and inference tuning, focus on optimizing models for specific tasks or environments, often leveraging hardware-aware optimizations to enhance efficiency during inference or fine-tuning. Together, these strategies have significantly improved the efficiency of LLMs, facilitating their deployment in resource-limited settings while preserving their capability for advanced NLP tasks.

Despite these advancements, achieving efficient and scalable LLMs remains a significant challenge. Balancing the tradeoffs between compression and model performance, ensuring the adaptability of tuning methods across diverse tasks, and integrating these techniques into existing infrastructures are key areas of ongoing research. Additionally, as the computational demand for LLMs grows, the development of more effective strategies to combine compression and tuning techniques is increasingly vital.

The primary contributions of our survey are summarized as follows:

- We provide a systematic literature review that focuses on subcategories of compression and tuning, accompanied by a detailed taxonomy of these frameworks.
- We present and compare existing surveys, highlighting the topics they covered and identifying gaps.
- Practical applications are listed with a comparative analysis, including cross-domain and cross-lingual applicability of compression and tuning techniques.
- We offer insightful future directions, such as the scalability of compression strategies, developing user-centric optimization methods, ensuring long-term performance stability, and advancing hierarchical efficiency techniques for LLMs.

2 Related Publications

As LLMs have garnered significant attention, numerous surveys on LLMs and articles have been published on efficient techniques, reflecting the resource-intensive nature of these models. Table 1

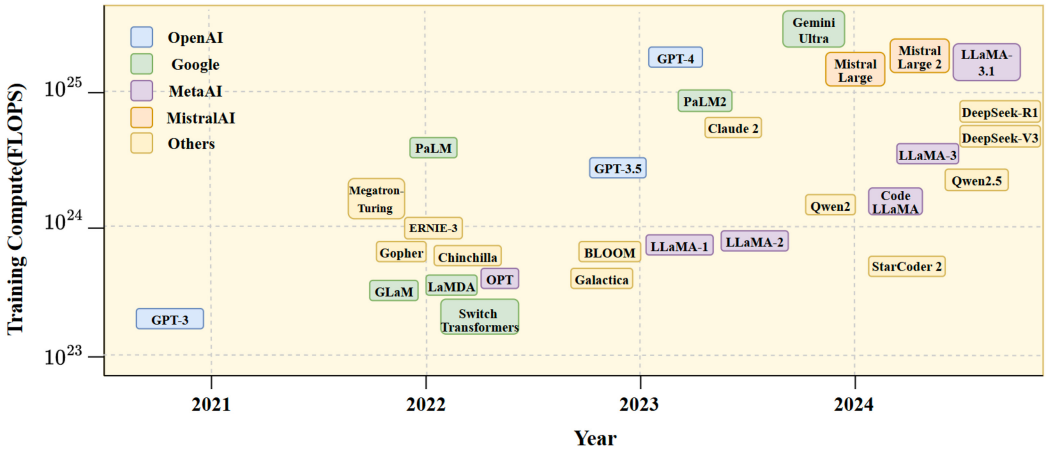

Fig. 1. Graph showing the training computation requirement over time for LLMs.

Table 1. Existing Surveys on Model Compressions and Tuning Techniques on Foundation LLMs

Survey	Year	Scope	Topics covered									
			KD	LRS	PR	QN	FT	QT	IT	FL	AP	DT
[173]	2023	Model Compressions	✓	✓	✓	✓						✓
[137]	2024	Model-centric, Data-centric, Frameworks Methods	✓	✓	✓	✓		✓	✓			✓
[155]	2024	LLMs and Multimodal Foundation Models	✓	✓	✓	✓	✓		✓	✓		
[147]	2024	Efficient Federated Learning Methods			✓	✓	✓	✓		✓		
Ours	2024	Model Compressions, Tuning, Federated Learning, Applications, Data	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

KD: Knowledge Distillation; LRS: Low-Rank Strategies; PR: Pruning; QN: Quantization; FT: Parameter-Efficient Fine-Tuning; QT: Query Tuning; IT: Inference Tuning; FL: Federated Learning; AP: Applications; and DT: Data

summarizes existing literature reviews of surveys related to model compression and tuning methods for foundational LLMs, outlining the key topics covered.

Several authors have surveyed techniques for improving the efficiency of LLMs. Zhu et al. [173] classified model compression techniques into four categories: quantization, pruning, knowledge distillation, and low-rank factorization, focusing on articles published before 2023. They also examined metrics, benchmark datasets, and challenges related to more advanced methods, like AutoML and explainability. Wan et al. [137] organized their survey into model-, data-, and framework-centric methods, providing insights into model compression, efficient pretraining and fine-tuning, data selection, prompt engineering, and specialized frameworks for LLMs. Their work highlights innovative strategies for reducing the computational and memory demands of LLMs, rendering them more accessible and sustainable for widespread use. Xu et al. [155] addressed multimodal models alongside LLMs and categorized them into three major areas: resource-efficient architectures, algorithms, and systems. They discussed efficient attention, dynamic neural networks, and diffusion-specific optimization. Algorithms included model compression and fine-tuning algorithms as well as categories related to inference. Additionally, they explored distributed learning, **federated learning** (**FL**), and serving models, such as cloud and edge systems, demonstrating a

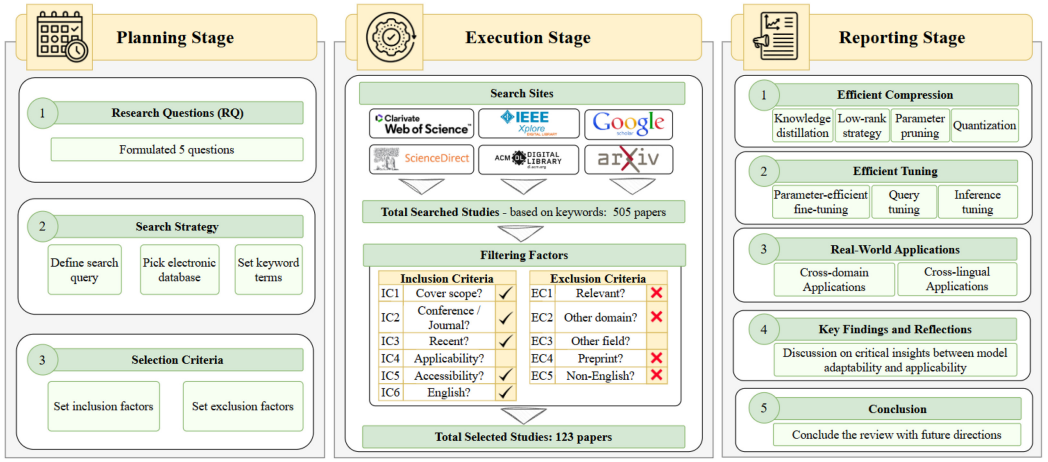

Fig. 2. Overall processing stages of our Systematic Literature Review (SLR) scheme [71, 72].

very broad scope. We also investigated the benefits of FL by integrating it into efficient compression and tuning techniques. Woisetschlger et al. [147] demonstrated that FL can reduce the number of parameters that must be transmitted, while efficient tuning can ensure the stability of model training on each device or server.

While recent surveys [137, 147, 155, 173] have made significant strides in exploring techniques for enhancing the efficiency of large language models, such as model compression, tuning, and resource-optimized architectures, a critical gap remains in understanding how these methods interact and synergize, particularly in real-world, cross-domain, and cross-lingual applications. Despite advancements in individual techniques, there is a lack of comprehensive studies that examine both the theoretical and technical aspects of these methods as well as their practical applicability and scalability in diverse scenarios. This underscores the need for an integrated survey that addresses how such techniques can be combined and adapted to meet the demands of dynamic, real-world environments.

3 SLR Methodology

We adopt a **Systematic Literature Review** (**SLR**) guided by Kitchenham et al. [71] and Kitchenham et al. [72] authors that offer a structured methodology to address the need for our comprehensive study on the efficient compression and tuning techniques for LLMs. By systematically collecting, analyzing, and synthesizing relevant research, an SLR can provide deeper insights into how model compression, tuning strategies, and resource-optimized architectures interact in both theoretical and practical contexts. In our survey, we carried out three primary stages to provide how our selected articles were searched, organized, and discussed. The planning stage establishes the primary research questions, defines the search strategy, and sets the criteria for selecting relevant state-of-the-art studies. The execution stage identifies and selects the necessary studies based on the criteria established in the planning stage. The reporting stage discusses the selected studies, highlights existing gaps, and outlines potential future research directions. Figure 2 provides the overall processing stages adopted from the SLR [71, 72].

3.1 Research Question

Our primary research question is: What are the latest state-of-the-art techniques in model compression and tuning methods for LLMs? To address this question comprehensively, secondary research

questions were identified to recognize and address gaps in the existing literature and explore the techniques and technologies shaping the future of LLMs.

RQ1: What are the existing techniques for model compression in LLMs?

RQ2: What are the current trends in tuning methods for LLMs?

RQ3: What are the real-world applications of compression and tuning techniques in LLMs?

RQ4: What are the key findings and implications of applying compression or tuning techniques to LLMs?

RQ1 investigates model compression techniques for LLMs, focusing on reducing size and computational costs while preserving performance, addressing challenges unique to their exponential growth. RQ2 explores recent tuning advancements in LLMs, emphasizing innovative strategies that enhance adaptability and efficiency while minimizing additional parameters for computational and memory savings. RQ3 examines real-world applications of compression and tuning techniques in LLMs, showcasing their impact on optimizing performance across diverse domains. RQ4 synthesizes key findings from compression and tuning applications, identifying lessons learned and best practices for optimizing LLMs.

3.2 Search Strategy

Search Sources. Our primary sources of literature comprised peer-reviewed publications, including conferences and journal venues, that align with the scope of our criteria. Within the boundaries of our research scope, which included ACM Digital Library, IEEE Xplore, Web of Science Master Journal, Science Direct, Google Scholar, and ArXiv- we identified numerous studies conducted over the past five years -from 2020 to 2024- focusing on compressing LLMs using efficient techniques.

Search Terms. We compiled a list of prominent conferences and journals in AI, computer science, and natural language processing fields. To substantiate the classification, we incorporated relevant references from prior studies [137, 147, 155, 173] and explained the rationale behind each category based on their topics. Subsequently, we carefully selected and used these representative compression-related keywords to form the foundation of our taxonomy for our search queries, such as "efficient large language modeling", "pruning large language models", "distilling large language models", "efficient quantization on large language models", and "efficient inference for large language models". Figure 3 presents a taxonomy of efficient methods for LLMs, which are categorized into two types: compression and tuning.

3.3 Selection Criteria

To produce and discuss the final list of selected articles, we carefully evaluated and finalized which articles to include and exclude based on our evaluation criteria checklist. The following are the **inclusion criteria** (**IC**) taken into consideration:

IC1: A article that covers and satisfies the search terms listed in the aforementioned section within the title and its main content.

IC2: Published in major AI and computer science domain conferences or journals.

IC3: Published within the past 5 years' range of 2020 and 2024.

IC4: A article that demonstrates real-world applications of efficient techniques.

IC5: A article that can be accessed publicly or downloaded via a provided link.

IC6: A article written in English.

Also, the following are the **exclusion criteria** (**EC**) taken into account:

EC1: A article that mainly focuses on other than the search terms listed in the aforementioned section within its main content.

EC2: Published in different domains other than AI and computer science conferences or journals.

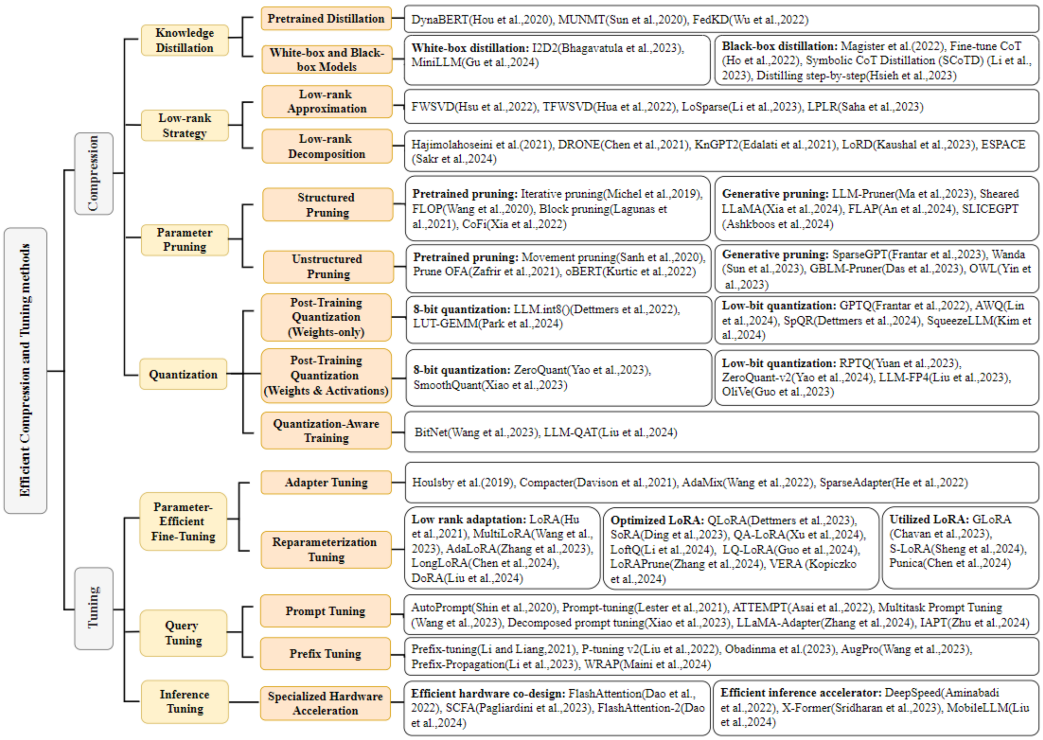

Fig. 3. Taxonomy of efficient compression and tuning methods for LLMs.

EC3: Published in different fields (i.e., robotics, cybersecurity) other than natural language processing, natural language generation, and natural language understanding.
EC4: A article that is either a preprint or unpublished articles that are not yet officially indexed.
EC5: A article that is written in languages other than English.

3.4 Article Selection

A systematic search was conducted across the selected electronic databases following a predefined search strategy, yielding a total of 505 articles. To accommodate the varying search functionalities of different database engines, minor adjustments were made to the search string. The automated search was then performed on each database, targeting the title and abstract fields. Following this, a screening process was conducted to filter out duplicate and irrelevant articles based on their titles and abstracts. This initial screening reduced the number of relevant studies to 257. The exclusion criteria were effective in eliminating non-pertinent studies from the review. A further in-depth analysis of the full texts led to the exclusion of an additional 134 articles, leaving 123 studies that were ultimately included in our survey.

4 Efficient Compression

RQ1: What are the existing techniques for model compressions with LLMs? The model compression algorithm aims at reducing the size of existing models by eliminating unnecessary parameters, sharing common parameter values, and reducing the size of existing models without losing the expressiveness of the parameters.

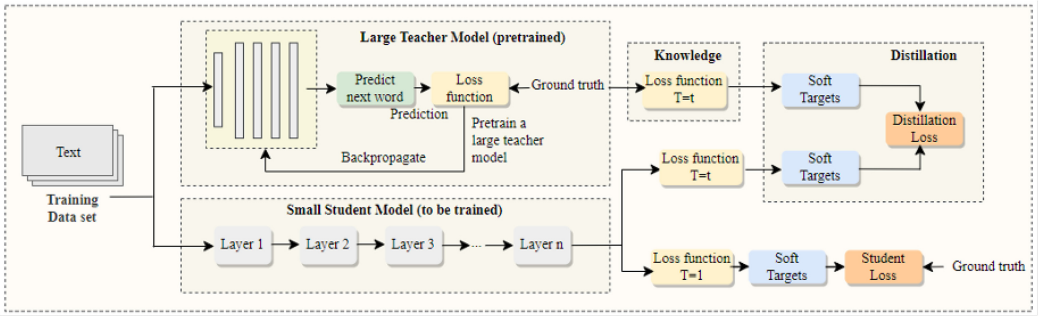

Fig. 4. Visualization of the detailed process for the distillation [52] and evaluation of models [39].

4.1 Knowledge Distillation

Knowledge distillation enhances model performance by transferring the outputs of a pretrained teacher model to a smaller student model, efficiently compressing knowledge from multiple models into one deployable model [52]. Figure 4 shows the detailed process through which the models were distilled and evaluated [39, 52]. First, we pretrain the large teacher model and then distill its knowledge to the smaller student model by calculating the cost function, which is the sum of the distillation and student losses; the difference should be minimized.

4.1.1 Pretrained Distillation. In recent years, distilled PLMs have been introduced by implementing intermediate layers into considerably more efficient distillation changes without significantly compromising performance. For example, DynaBERT [55] is designed in a two-stage procedure of adaptive width and depth, wherein the width adapts the multi-head attention and feed-forward network to be performed in parallel by rewiring the connections between the two intermediate layers. This facilitates the model in transferring the most important attention heads and neurons, while the depth is controlled using depth multipliers that avoid the catastrophic forgetting [104] problem during the training stage. They experimented with the GLUE benchmark [138] and SQuAD dataset [118] to mitigate the data biases by evaluating multiple sub-network configurations, leveraging knowledge distillation for generalization, and transparently reporting performance under varying computational limits from ARM-based smartphones. Sun et al. [130] trained and developed a distilled **multilingual unsupervised neural machine translation** (**MUNMT**) model capable of translating cross-lingual languages from a single encoder and decoder, and experimented on WMT monolingual news crawl datasets [6]. It uses two distinctive knowledge distillation techniques by using self-knowledge distillation to reduce the KL-divergence loss between one language (i.e., English) and branch knowledge distillation (i.e., 12 other considered languages besides English) paths to be similar. Specifically, self-knowledge distillation ensures enhanced cross-lingual knowledge sharing by integrating additional languages in training, while language branch knowledge distillation leverages linguistic similarities within language families to reduce disparities in performance, particularly for low-resource and zero-shot translation scenarios. FedKD [148] is a federated learning approach based on knowledge distillation, where each client has a large teacher model and a shared, smaller student model, reducing communication by sharing only the student model across clients. Locally, the teacher and student models are trained on client data and distilled from each other, with the strength of their knowledge transfer adjusted based on prediction accuracy. While the local teacher model is updated independently on each client, the student model updates from all clients are sent to a central server, aggregated, and redistributed to maintain a synchronized global model. To further reduce communication costs,

FedKD uses a dynamic gradient approximation method with **singular value decomposition** (**SVD**), compressing the exchanged gradients through adaptive precision.

4.1.2 White-box and Black-box Models. There are two primary categories of knowledge distillation for LLMs: white-box methods, which use teacher model parameters during distillation, and black-box methods, which do not rely on teacher model parameters but can only be accessed through the API interface.

White-box distillation. Several researchers explored methods for distilling open-source LLMs. For example, I2D2 [9], a new commonsense distillation framework model, deviates from traditional symbolic knowledge distillation [146] by not relying on large-scale teacher models. It incorporates two key innovations. First, NeuroLogic Decoding [97] is used to improve the generation quality of smaller, readily available language models. Second, self-imitation learning enables the model to iteratively learn and enhance its own commonsense understanding capabilities. Moreover, to mitigate the potential biases that may arise from the large-scale generation process using GPT-3 [11], the authors focused on narrow, commonsense-specific scenarios with careful prompting to constrain outputs and employed a critic model trained on human annotations to filter and correct low-quality or biased generations, thereby ensuring the data remains aligned with ethical standards and is free of harmful or biased content. Also, the MiniLLM [40] model minimizes reverse KL-divergence by using a policy gradient that helps avoid problems associated with biased low-probability distributions that can occur from overestimation of the teacher's distribution when distilling to the student model. The authors proposed solutions to combat high variance, reward hacking, and generation-length bias encountered in fine-tuning language models, including single-step regularization, teacher-mixed sampling, and length normalization.

Building on these advancements, DeepSeek-R1 [42], developed by DeepSeek, is a reinforcement learning-based LLM without the need of supervised fine-tuning that uses the **group relative policy optimization** (**GRPO**) technique, which is a novel training algorithm that was first introduced in DeepSeekMath [125], designed to enhance reasoning capabilities through **chain-of-thought** (**CoT**) prompting. Unlike the traditional **proximal policy optimization** (**PPO**) [124], which maximizes absolute rewards for individual responses, GRPO employs a group-based ranking mechanism, ensuring that multi-step reasoning outputs are explicitly incentivized rather than solely optimizing for final-task accuracy. This structured approach improves hierarchical step-by-step reasoning, reducing hallucinations and increasing logical consistency, particularly in domains such as mathematical problem-solving, commonsense inference, and symbolic reasoning. To further extend these improvements while maintaining computational efficiency, DeepSeek-R1 distills its structured reasoning capabilities to LLaMA-3 [32] and Qwen-2.5 [157] models through layer-wise distillation. Unlike conventional distillation methods that focus on soft label matching, this approach preserves intermediate reasoning steps, ensuring that the distilled models retain CoT-based structured reasoning while significantly reducing computational costs.

Black-box distillation. Magister et al. [99] explored enhancing smaller models' reasoning capabilities through knowledge distillation. This involves training a compact model using the output from a more extensive model's reasoning processes. Their findings suggest that this approach significantly boosts smaller models' performance across various tasks by transferring complex reasoning skills from larger to smaller models. In comparison to the Magister et al. [99] authors, in which it requires fine-tuning on larger language models, guided by LLMs CoT-based text generation, "Distilling step-by-step" mechanism [57] outperforms LLMs using less training data than traditional fine-tuning or distillation methods. This approach involves extracting rationales from the LLMs as additional supervision within a multi-task framework which leads to superior performance compared to both fine-tuning and distillation techniques. It enables smaller models to

outperform few-shot prompted LLMs by requiring less training data and therefore, it mitigates inherited biases from the teacher LLM by extracting and leveraging high-quality rationales, which serve as structured and interpretable guidance for smaller models, while avoiding over-reliance on potentially biased raw labels.

In addition to the distilled reasoning models, other researchers have explored distilling LLMs using the CoT method. For example, Fine-tune CoT [53] fine-tunes the GPT-3 [11] model and generates a sample of reasoning CoT-based queries that enable reasoning teacher models to facilitate complex reasoning onto smaller models and teach these smaller models to solve complex tasks, including arithmetic, symbolic, and common sense reasoning. Similar to the method of Ho et al. authors [53], **Symbolic CoT Distillation** (**SCoTD**) [80] uses the CoT prompting method on smaller large models to improve its accuracy based on the smaller (student) model. This is achieved by curating a set of labeled CoTs during few-shot prompting, and the student model is trained using standard language modeling loss.

4.2 Low-rank Strategy

The low-rank strategy is a mathematical technique used to analyze and extract essential information from a large data matrix by decomposing it into smaller matrices with lower dimensions. The fundamental idea governing the low-rank strategy involves determining a factorization of a large weight matrix W into two orthogonal matrices U and V, such that W is approximately equal to UV. The products of U and V approximate the original weight matrix, resulting in a substantial reduction in both the number of parameters and the computational overhead.

4.2.1 Low-rank Approximation. Low-rank approximation, though primarily focused on minimizing reconstruction error with limited direct impact on model performance, is being adapted to improve LLM efficiency. By using SVD to reduce dimensionality, this approach uncovers latent semantic structures, efficiently capturing relationships between terms and documents. In the SVD method, there was a greater impact on the accuracy of the original matrix during its decomposition and reconstruction of the original matrix in relation to the errors caused by the parameters. To address this issue, Fisher information was introduced to weigh the importance of the parameters affecting the model prediction, resulting in **Fisher-weighted SVD** (**FWSVD**) [58] and reducing the performance drop after truncation. Furthermore, **true-weighted SVD** (**TFWSVD**) [60] can provide a better solution than FWSVD by reverting to numerical optimization methods. Specifically, implementing an **alternating least squares** (**ALS**) optimization based on a switching point calculated by a hybrid of ADAM [70] and SGD optimizers can effectively compress the model by associating each parameter with its unique importance. In addition, a **low-rank and sparse approximation** (**LoSparse**) [83] combines the advantages of both low-rank approximations and pruning, while avoiding limitations such as low-rank approximations. This renders the accurate approximation of parts that may be important to the model performance challenging because it is difficult to consider all the various weights. Approximating a weight matrix using the sum of the low-rank approximation, indicated by the coherent parts in the weight matrix, and a sparse matrix, indicated by the incoherent parts in the weight matrix, is more efficient and stable. Before decomposition, **low-precision low-rank factorization** (**LPLR**) [120] randomly selects a column from an existing matrix, calculates its basis, and then performs quantization on this basis. Subsequently, it was projected onto a randomly selected column of the existing matrix, centered on the basis to which this quantization was applied with a small number of bits to minimize the Frobenius norm error, which was applied to the LLaMA-7B [136] model.

4.2.2 Low-rank Decomposition. Matrix decomposition techniques have been applied to model layers, with methods like triangular decomposition enabling more efficient computation and

Table 2. Categorization of the Parameter Pruning Methods into Two Types: Structured and Unstructured Pruning, with Descriptions of the Main Source Model, and Key Techniques

Type	Methods	Year	Baseline	Key techniques
Structured	Iterative pruning [105]	2019	BERT	Taylor expansion
	FLOP [143]	2020	BERT	Decomposition, Augmented Lagrangian
	Block pruning [75]	2021	BERT	Block set of structure
	CoFi [150]	2022	BERT	Coarse-grained, Fine-grained
	LLM-Pruner [98]	2023	LLaMA-2	Full gradient information, Approximation
	Sheared-LLaMA [151]	2024	LLaMA-2	Target algorithm, Dynamic batch loading
	FLAP [3]	2024	LLaMA	Adaptive structure search
	SLICEGPT [5]	2024	LLaMA-2	Orthogonal-matrix transformation, PCA
Unstructured	Movement pruning [122]	2020	BERT	Gradient form, Straight-through estimator
	Prune OFA [163]	2021	BERT	Learning rate rewinding
	oBERT [74]	2022	BERT	Inverse Hessian, WoodFisher approach
	SparseGPT [36]	2023	OPT	Inverse Hessian, Optimal partial updates
	GBLM-pruner [24]	2023	LLaMA	Gradient matrix, Element-wise product
	Wanda [132]	2024	LLaMA	Element-wise product
	OWL [160]	2024	LLaMA	Layer-wise outlier distribution

reducing computational cost. One such study is progressive low-rank decomposition [46]. It decomposes the models into multiple intermediate steps such that it can apply decomposition to all layers instead of one-shot compression and is fine-tuned to recover its accuracy. Data-aware low-rank compression (DRONE) [15] demonstrated that low-rank decomposition can be realized when the data distribution is in a lower intrinsic dimensional space while maintaining the key input factors, even if the metrics in the model cannot be approximated. By minimizing the approximation error of the output, it has been proven to have a significant impact that can perform much better than the conventional SVD approach. Another approach involves applying the Kronecker decomposition method such that all the weight matrices across different heads and layers of the transformer-based models, such as the GPT-2 model [117], can be decomposed into Kronecker factors, such as KnGPT2 [33]. Moreover, methods for compressing code generation LLMs such as StarCoder [81], called **low-rank decomposition** (**LoRD**) [67], calculate the aspect ratio of the smaller dimensional matrix to the larger dimensional matrix based on the parity point across the rank for reduction. This reduces the ranks by 39.58% in a one-shot setting and impacts less than 1% perplexity. Recently, unlike traditional approaches that focused on decomposing weights in layers, the ESPACE [121] method, which reduces the dimensionality of activations, has been developed. ESPACE preserves weight matrices and applies eigenvalue decomposition to the activation tensors' autocorrelation matrix, using principal eigenvectors to form a static orthogonal matrix that projects activations into a lower-dimensional space. This process minimizes the noise introduced during the dimensionality reduction of activations, enabling up to 50% compression across various LLMs with minimal performance degradation. Additionally, since the weight matrices are not directly modified, all weights remain trainable, improving the model's convergence during retraining.

4.3 Parameter Pruning

Parameter pruning removes or freezes low-saliency weights based on a threshold or second-derivative objective, improving generalization, reducing data needs, and accelerating learning. Table 2 presents the classification of parameter pruning into either structured or unstructured schemes. We also provided the primary source model with key techniques for each method.

4.3.1 Structured Pruning. Structured pruning removes all edges connected to a specific node, which improves speed by eliminating entire sets of computations; however, it has the drawback of achieving high pruning rates and generally performs less effectively than unstructured pruning.

Pretrained pruning. Studies in structured pruning reveal that using fewer attention heads can sometimes outperform parallelizing with multiple heads. One approach, called iterative pruning of attention heads, calculates a proxy score via Taylor expansion based on data distribution and loss to identify impactful heads, then reduces multi-heads across layers accordingly [105]. This approach can be implemented in various ways to achieve model compression. One of these methods is structured block pruning [75], which uses blocks of any size and extends to movement pruning [122]. Furthermore, FLOP [143], which selects rank-one components in a reparameterized matrix using a diagonal mask, significantly outperforms structured block pruning. Because pruning in a structured manner has the drawback of flexibility, several researchers have introduced a new approach that can prune with smaller units by jointly pruning the coarse- and fine-grained pruning, such as layers and hidden dimensions for each, called **coarse- and fine-grained pruning** (**CoFi**) [150].

Generative pruning. LLM-Pruner [98] was the first attempt at pruning generative LLMs structurally, and it offers the calculation of group importance scores for ranking removal structures. Typically, this score is calculated based on the full-gradient information, which can prevent resource consumption when dealing with full models. The authors employed dependency-based structural pruning, which ensures interdependent components are pruned together, preventing misaligned intermediate representations that could skew performance across tasks. Additionally, by utilizing small, publicly available datasets for post-pruning fine-tuning, the method minimizes reliance on proprietary training corpora, enhancing the generalizability and fairness of the pruned model across diverse tasks and datasets. Furthermore, Sheared-LLaMA [151] exhibited target-structured pruning, which is an extension of CoFi that facilitates pruning by determining the target shape in addition to the Lagrangian constraints while maintaining the structure of the target model and uses the dynamic batch loading algorithm to utilize data more efficiently. In comparison with LLM-Pruner [98] and the structured-pruning version of Wanda [132] methods, **Fluctuation-based Adaptive Structured Pruning** (**FLAP**) [3] introduces a retraining-free structured pruning method that selects low-variance input channels based on feature fluctuation to minimize pruning impact. Additionally, they generate a bias compensation mechanism that restores pruned output feature maps using baseline values calculated from calibration data. This approach ensures that pruning-induced performance degradation is minimized without requiring retraining, enhancing the generalizability and fairness of the pruned model across various tasks, as is similarly observed from the LLM-Pruner method [98]. SLICEGPT [5] introduces a technique for model compression by leveraging computational invariance in transformers, applying orthogonal transformations to modify weight matrices without altering model outputs. Using RMSNorm connections, orthogonal matrices are integrated, and **principal component analysis** (**PCA**) projects each block's signal matrix onto its main components. Unimportant components are then removed by deleting specific rows and columns, effectively reducing the model's embedding dimension. It has demonstrated the capability to compress LLMs, including LLaMA-2 [135] and OPT [170] models by up to 25%, all while preserving 99% of their original performance and improving inference speed.

4.3.2 Unstructured Pruning. Unstructured pruning removes individual weights based on specific criteria, setting unnecessary weights to zero without considering the structure. While it allows for a higher pruning rate than structured pruning, it doesn't significantly enhance computational speed since the pruned weights remain as zero values within the matrix, still contributing to matrix operations.

Pretrained pruning. One such unstructured pruning method is the movement pruning [122], which diverges from the conventional approach of evaluating weights based on their magnitudes after fine-tuning. It guides the directional changes made during the tuning process using a gradient-based formulation called first-order pruning. Pruning parameters exhibiting positive divergence are indicative of lower importance. Similarly, Prune OFA [163] was designed to preserve its sparsity pattern in the transfer learning step by combining the knowledge distillation method with unstructured pruning. The teacher model corresponds to a dense model trained for a specific task, whereas the student model represents a model exhibiting either a predetermined level of sparsity or undergoing a pruning process using the concept of learning rate rewinding into gradual magnitude pruning by resetting the learning rate scheduler. Meanwhile, in comparison with the movement pruning method [122], the **optimal BERT** (**oBERT**) surgeon [74] is a second-order pruning framework that uses an inverse Hessian approximation, offering a more precise pruning method than first-order approaches. Although computationally intensive due to row-by-row matrix iteration, oBERT improves efficiency through the WoodFisher [128] method by using a set of basis vectors and ignoring correlations between blocks. This approach enables oBERT to outperform at both pretraining and fine-tuning stages by removing parameters with low importance scores that have been experimented with on the GLUE benchmark [138] and SQuAD dataset [118] to demonstrate its generalizability across different domains.

Generative pruning. SparseGPT [36], proposed by Frantar et al.(2023), introduces a one-shot unstructured pruning method that avoids the need for retraining. It offers an iterative approach to applying masks by reconstructing weights individually rather than applying a full mask at once. SparseGPT also employs iterative blocking to set different sparsities for each column by selectively masking columns with the largest errors and updating weights through repeated steps. This method integrates pruning and quantization into a single process, enabling mutual consideration of pruned and quantized weights, enhancing computation speed and reducing memory usage. Thus, it achieves a significant unstructured sparsity of up to 60% in GPT-based models while minimizing its perplexity. Since updating weights in SparseGPT remains a computationally demanding process, Wanda [132], which calculates weight importance based on the element-wise product between weights and their respective input activations, can prune in a one-shot setting without an iterative weight update or Hessian inverse computation, thereby lowering its time complexity. In a recent study, a **gradient-based language model pruner** (**GBLM-Pruner**) [24] proposed that previous unstructured pruning methods (i.e., SparseGPT and Wanda) disregarded the informative gradients originating from pretrained LLMs. By utilizing a gradient matrix derived from pretrained LLMs, the weight importance can be determined by element-wise multiplication of the weight magnitude and either L1 or L2 normalization of the gradients. This allows the unstructured pruning method to reveal structured patterns and surpass previous unstructured pruning approaches in terms of zero-shot performance and perplexity. Another technique is the **outlier weighted layer-wise** (**OWL**) sparsity [160] which emphasizes that the **layer-wise outlier distribution** (**LOD**) should be considered rather than being uniformly distributed to all layers for pruning LLMs, as preserving outliers is important. This method assigned greater weights to layers with more outliers based on the calculated LOD and facilitated subtle alignment between the sparsity of the weight matrix and outliers.

4.4 Quantization

Quantization reduces the precision of neural network values, like weights and activations, to make networks more computationally efficient and memory-friendly [20, 47], especially for resource-limited devices. This process, often involving conversion from higher-bit to lower-bit formats (e.g., FP32 to INT8/INT4), can lead to information loss and reduced accuracy. Various methods,

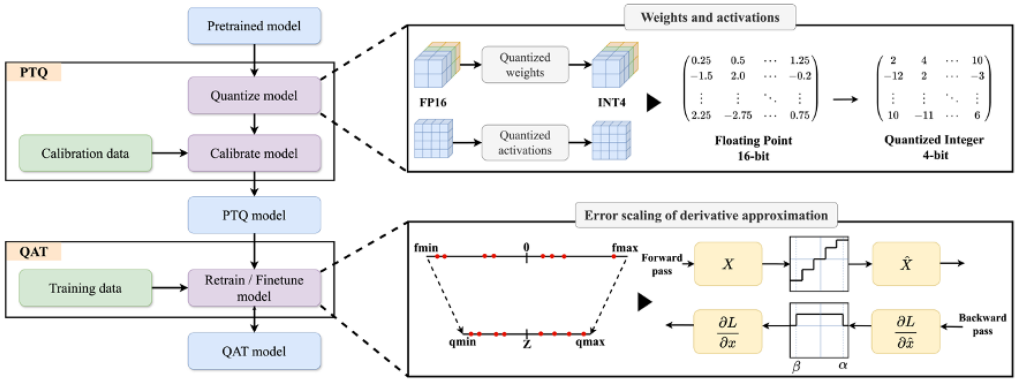

Fig. 5. Overall framework of quantization categorized into post-training quantization and quantization-aware training, illustrating how weights and activations are quantized and how error scaling is reduced through the STE [8] derivative approximation.

Table 3. Quantization can be Categorized into PTQ and QAT in which Each Technique Varying in Bit-size and Methods

Authors	Year	PTQ	QAT	Bits	Key techniques
Dettmers et al. [25]	2022	WO		1	Vector-wise quantization, Mixed-precision decomposition
Frantar et al. [34]	2022	WO		2~4	Cholesky kernel
Yao et al. [158]	2023	WA		3	Token-level quantization on activations, Row-level quantization on weights
Xiao et al. [152]	2023	WA		3~4	Offline migration
Yuan et al. [161]	2023	WA		3	Rearrangement and clustering on channels
Liu et al. [89]	2023	WA		4~8	Joint format, Max value search
Guo et al. [41]	2023	WA		4	Outlier-Victim Pair(OVP) function
Wang et al. [140]	2023		QAT	1	Low-precision binarization quantized activations
Lin et al. [85]	2024	WO		8	Activation-aware scaling
Dettmers et al. [27]	2024	WO		4	Group-wise quantization function
Kim et al. [69]	2024	WO		3	Sensitivity-based non-uniform function, Dense-and-sparse function
Park et al. [113]	2024	WO		2~32	Bias term of binary coding quantization, Group-wise quantization
Liu et al. [95]	2024		QAT	4	Data-free distillation function
Yao et al. [159]	2024	WA		8	Low-rank matrix factorization

PTQ: Post-Training Quantization; QAT: Quantization-Aware Training; WO: Weights-Only; WA: Weights and Activations

including **post-training quantization** (**PTQ**) and **quantization-aware training** (**QAT**), aim at minimizing this accuracy drop [37]. Figure 5 and Table 3 show the overall framework of quantization, with details on quantizing the weights and activations as well as scaling down the errors through a **straight-through estimation** (**STE**) [8] derivative approximation formula.

4.4.1 Post-training Weights-only Quantization. Post-training weight quantization in language models aims at achieving memory optimization by converting operations for already trained models based only on the weights of the LLMs.

8-bit quantization. LLM.int8() [25] was the first INT8 post-training weight quantization method introduced with up to 175 billion parameters without compromising the full-precision model performance while reducing memory use. This method applies vector-wise quantization to view matrix multiplication as a sequence of independent inner products and mixed-precision decomposition to handle high-precision multiplication of outliers that prevents information loss and performance degradation caused by biased distribution of outlier features. LUT-GEMM [113] leverages **binary coding quantization** (**BCQ**) [119] to balance compression rate and

quantization error, supporting both uniform and non-uniform quantization formats. It achieves energy efficiency by utilizing resources more effectively than cuBLAS with tensor parallelism. Unlike cuBLAS, which distributes tasks across multiple GPUs and often results in idle GPUs and increased latency, LUT-GEMM maximizes the use of a single GPU, minimizing synchronization delays. This optimized design enhances processing speed, reduces the number of GPUs needed, and significantly lowers energy consumption. Also, it ensures uniform and non-uniform quantization compatibility, preserving critical features during weight quantization while maintaining generalizability across diverse language model tasks and datasets.

Low-bit quantization. One of the methods of post-training weight-only quantization to lower than 8-bit is the GPTQ [34] algorithm. They showed that even LLMs such as OPT-175B [170] and BLOOM-176B [123] can be quantized and compressed into three or four bits per parameter by quantizing all the rows of weights in the same order, which is different from the OBQ [35] method, and by updating only the final column of its weight itself. Often, updating the remaining weights in incorrect directions can induce arbitrarily bad quantization problems; therefore, the authors optimized and used the Cholesky kernel for a better speedup performance to update only the necessary weights. Another technique is AWQ [85], wherein the authors argued that not all weights should be treated with the same importance; therefore, protecting only a few prominent weights can significantly reduce quantization errors. To reduce the quantization error of the prominent weights, the authors used the per-channel scaling equation and search space to determine the optimal scale determined by the activation scale (i.e., activation awareness). **Sparse-quantized representation** (**SpQR**) [27] is a method of quantizing weights, particularly focusing on isolating outlier weights, depending on the level of sensitivity by capturing small relative groups (i.e., group-wise quantization equation) and individual outliers (i.e., finding and isolating outliers as 16-bit weights, compressing other non-outliers into 3-4 bit representation and transferring them into 16-bit outlier weights). By leveraging these techniques, SpQR reduces memory usage by more than 4 times compared to traditional 16-bit models including LLaMA-variants [136] and OPT [170] models. Similar to the SpQR method, Kim et al.(2024) proposed SqueezeLLM [69] in which it has two key concepts for compressing LLMs: the implementation of a sensitivity-based non-uniform quantization scheme that improves the perplexity at 3-bit precision, and the dense-and-sparse quantization method that decomposes the weights such that the outliers and sensitive weights can be stored effectively and later removed to achieve a significant boost in quantization resolution.

4.4.2 Post-training Weights and Activation Quantization. Post-training quantization on LLMs with weights and activation targets the reduction of the differences in weight distribution between the quantized model's parameters and the float parameters, and the quantization of the activation distribution on the outputs of layers into a lower precision format.

8-bit quantization. ZeroQuant [158] proposed an 8-bit quantization scheme to solve the problems of significant activation-range variability in tokens of the same layer and large range differences among weight rows. This is achieved by token-level quantization of activations and row-level quantization of weights to provide an effective improvement in accuracy over traditional layer-wise quantization methods. SmoothQuant [152], developed by Xiao et al.(2023), innovatively addressed the challenge of quantizing neural network activations, which are typically more variable and harder to handle than weights. By transferring the outliers from the activations to the weights, the activation distribution is smoothed, thus simplifying the quantization while maintaining the activations as flat.

Low-bit quantization. Yuan et al.(2023) proposed RPTQ [161], which is the first method to use a three-bit quantization process combined with an activation function. Researchers have found that using the same quantization parameters across different channels results in a wide variation

in values. To overcome this issue, they proposed a clustering approach to group similar-range activation channels, thereby allowing the uniform application of quantization parameters within each cluster. Moreover, Liu et al.(2023) proposed the LLM-FP4 [89] method that quantizes LLaMA-13B [136] in both weights and activations to 4-bits by introducing two key concepts of joint format and maximum value search. Specifically, the joint formation minimizes the quantization error between the two distances of the intermediate output of the quantized layer and the output of the original layer, whereas the maximum value search stores the intermediate raw output of each layer through forward propagation and then iteratively updates the optimal format and biases for each layer by minimizing the reconstruction metric. The authors then demonstrated an efficient preshifted exponent bias to overcome the catastrophic high inter-channel activation variance in LLMs. The OliVe [41] algorithm improves LLM quantization using the **outlier-victim pair** (**OVP**) method, reducing hardware overhead and enhancing performance with localization, while its memory-aligned encoding ensures compatibility with hardware accelerators like the systolic array and tensor core. This approach results in up to 4.5 times faster acceleration and up to 4 times lower energy consumption compared to outlier-aware accelerators like OLAccel [112] and GOBO [162]. Additionally, tests on dynamic power consumption (DRAM, L1/L2 cache, register file) demonstrated OliVe's energy efficiency due to its 4-bit architecture. Initially introduced by ZeroQuant [158], the same authors introduced a **low-rank compensation** (**LRC**) [159] technique that applied a low-rank matrix factorization approach of SVD and formulated a new error approximation using two low-rank matrices to optimize and reduce the existing quantization error, thereby improving the performance while minimizing the impact of an increase in the model parameter size.

4.4.3 Quantization-aware Training. QAT optimizes neural network models by adjusting the quantization values during retraining following the initial quantization upon completion of training. BitNet [140] is the first QAT approach that trains Transformer-based models with 1-bit weights from scratch. The model's "BitLinear" layers use 1-bit weights, which simplifies computations to binary operations and reduces energy consumption significantly. Specifically, the energy efficiency in BitNet minimizes the arithmetic operations needed for these binary computations, as multiplying or adding 1-bit values requires much less power compared to 16-bit or 32-bit operations. Additionally, BitNet maintains high precision only in essential components such as the optimizer states and gradients during training, which ensures model stability and convergence while minimizing unnecessary power usage in non-critical areas. LLM-QAT [95], introduced by Liu et al.(2024), enhances the QAT process for LLMs to maintain their performance during 8-bit quantization. This approach incorporates innovative techniques such as data-free distillation, where the model generates its own distillation data and quantizes **key-value** (**KV**) caches. These methods improve the model's efficiency and better handle the long-term dependency characteristics of LLMs.

Expanding on these principles, DeepSeek-V3 [87] employs a fine-grained mixed-precision quantization framework centered around the FP8 quantization scheme, optimizing precision levels dynamically during training rather than relying on a static PTQ strategy. Unlike conventional low-bit quantization, which applies uniform bit-width reduction across all layers, the model selectively applies FP8 precision to compute-intensive operations, particularly in **General Matrix Multiplication** (**GEMM**) kernels during forward propagation, activation backpropagation, and weight backpropagation. This approach doubles computational speed compared to BF16-based training while reducing memory usage by caching activations in FP8. However, recognizing that certain components are more sensitive to quantization, DeepSeek-V3 preserves BF16 or FP32 precision for critical model elements such as the embedding module, output head, MoE gating mechanisms, LayerNorm, and attention operators, ensuring numerical stability during training.

5 Efficient Tuning

RQ2: What are the current trends tuning methods for LLMs? To optimize a specific task, save resources and time, and adapt to new data, the weights of the pretrained LLMs must be fine-tuned with new data to improve performance and reduce the learning time. However, this process can be computationally expensive. Therefore, research on efficient tuning methods is another significant field of research. Specifically, as presented in Table 4, we provide the primary aspects of various tuning techniques including parameter-efficient fine-tuning, query tuning and inference tuning, and provide a more detailed categorization of key techniques and attributes.

5.1 Parameter-efficient Fine-tuning

Parameter-efficient fine-tuning (**PEFT**) adapts pretrained language models by updating only a small subset of parameters, reducing training time, deployment costs, and computation demands while preserving model performance. There are two main tuning techniques, including adaptation tuning and reparameterization tuning.

5.1.1 Adapter Tuning. Adapter tuning incorporates additional adapters with a bottleneck structure into each transformer layer of BERT [28], limiting the number of parameters by focusing only on learning the parameters of these added adapters [56]. Similarly, Davison et al.(2021) developed a shared task-specific weight matrix, called Compacter [100] using a low-rank approach that combines fast, general information with slow, adapter-specific weights for each layer. This method balances performance with parameter efficiency during training, optimizing the tradeoff between the two. In comparison to [56] authors, AdaMix [141] utilizes a mix of adapter modules and trains them with stochastic routing, which randomly selects a feedforward up-projection layer and feedforward down-projection layer to perform the same modules in a given batch in all steps. This method outperforms the full fine-tuning method by using only 0.1–0.2% of the parameters. Similarly, SparseAdapter [50] prunes the lowest effective weights from the adapter modules with the highest scores based on its gradients in a single iteration from the initialization phase (before fine-tuning) up to 80% sparsity. This results in better performance compared to the full fine-tuning technique on BERT [28] and RoBERTa [93]. [50, 56, 100, 141] authors have all validated their methods across diverse general NLP tasks using the GLUE [138] benchmark as their basis and, in addition to that, they also tested on other general tasks such as SuperGLUE [139] benchmark, XSUM datasets for text summarization [108], and DART [106] for text generation ensuring that performance improvements are consistent across tasks like sentiment classification, natural language inference, and question-answering, which helps reduce bias toward any specific task type or dataset configuration. FedLLM [154] introduces a federated learning approach that uses **backpropagation** (**BP**)-free training with perturbed inferences to enhance model convergence, where devices apply self-generated small perturbations to model weights and compare output deviations from ground truth to approximate gradients. Specifically, FwdLLM integrates PEFT techniques of LoRA [59] and Adapter [115] to limit trainable parameters, employs a variance-controlled pacing mechanism to manage when to aggregate gradients based on convergence proximity, and uses discriminative perturbation sampling to filter out low-value perturbations. This approach achieves up to 3 times faster convergence and 14.6 times lower memory usage compared to conventional federated learning.

5.1.2 Reparameterization Tuning. In the reparameterization tuning section, LoRA [59] is the primary focus of research, with various advanced techniques based on LoRA forming integral parts of PEFT. These techniques include LoRA-variants, optimized LoRAs, and utilized LoRAs.

Table 4. Overview of the Efficient Tuning Techniques

Methods	Year	PEFT	QT	IT	Key techniques	Key attributes
AutoPrompt [127]	2020		✓		Prompt search, automating label	
Compacter [100]	2021	✓			Task-specific, low-rank approximation	Joint techniques
Prompt-tuning [78]	2021		✓		Task-specific, mixed-task batch	
Prefix-tuning [82]	2021		✓		Continuous task-specific vectors	
LoRA [59]	2021	✓			Decomposition	Baseline of LoRA
AdaMix [141]	2022	✓			Stochastic routing projection, mixed-task batch	For adaptive-task
SparseAdapter [50]	2022	✓			Sparse-adaptive pruning	For adaptive-task
P-tuning [91]	2022		✓		Deep prompt tuning, optimization	
ATTEMPT [4]	2022		✓		Attention, mixture-of-prompts	For multi-task
AugPro [144]	2022		✓		Distilled word embedding, parallelism optimization	Joint techniques
FlashAttention [23]	2022			✓	Algorithmic reordering, kernel fusion, parallelism optimization	For specialized speculative decoding
DeepSpeed [2]	2022			✓	Data parallelism on optimizer, gradients and parameters	For paralleled data-distributed training
MultiLoRA [142]	2023	✓			Decomposition	For multi-task
AdaLoRA [168]	2023	✓			Decomposition, budget scheduler	
QLoRA [26]	2023	✓			Decomposition, quantization	Joint techniques
SoRA [29]	2023	✓			Decomposition, post-pruning	Joint techniques
GLoRA [12]	2023	✓			Generalized prompt module	Unified framework
Multitask Pt [145]	2023		✓		Decomposition, distillation	For multi-task
Decomposed Pt [153]	2023		✓		Decomposition, intrinsic rank	
Prefix-Propagation [79]	2023		✓		Calibration, decomposition	For long sequences
FlashAttention-2 [22]	2023			✓	Support for multi-query and grouped query-key attention, backward pass optimizations	For specialized speculative decoding
SCFA [110]	2023			✓	Dynamic sparsity pattern tuning	Joint techniques
X-Former [129]	2023			✓	Data parallelism on optimizer, gradients and parameters	For paralleled data-distributed training
LongLoRA [17]	2024	✓			Decomposition, S^2-attention	For long sequences
QA-LoRA [156]	2024	✓			Decomposition, quantization	Joint techniques
LoftQ [84]	2024	✓			Decomposition, quantization	Joint techniques
LQ-LoRA [43]	2024	✓			Decomposition, quantization	Joint techniques
LoRAPrune [166]	2024	✓			Decomposition, pruning	Joint techniques
VeRA [73]	2024	✓			Decomposition, scaling vectors	
Punica [13]	2024	✓			Kernel design	For LoRA serving
S-LoRA [126]	2024	✓			Unified paging, kernel design	For LoRA serving
DoRA [90]	2024	✓			Decomposition, weight normalization	For multimodal task
FedLLM [154]	2024	✓			Backpropagation-free gradient, perturbation inferencing	Joint techniques
LLaMA-Adapter [169]	2024		✓		Zero-initialized attention mechanism	For instruction-tuning
IAPT [172]	2024		✓		Soft prompt generators	For multi-task
WRAP [102]	2024		✓		Synthetic text generation	For instruction-tuning
MobileLLM [96]	2024			✓	Deployment for resource-constrained smartphones and edges	Joint techniques

PEFT: Parameter-Efficient Fine-Tuning, QT: Query Tuning, IT: Inference Tuning, Pt: Prompt tuning.
We provide detailed taxonomy of PEFT, query tuning and inference tuning with key techniques and key attributes for each method.

Low rank adaptation. Following the introduction of low-rank adaptation fine-tuning, LoRA [59] enhances the performance by leveraging new parameters instead of the original pretrained weights. Hu et al. [59] introduced a pair of rank decomposition matrices that could be trained in parallel with the existing weight matrix. Unlike the adapter method, which sequentially appends external modules, this approach reparameterizes the model into a more compact parameter set, simplifying backpropagation and resolving parameter calculation challenges associated with the original log-likelihood function. To enable a single model to handle multiple tasks, some studies have developed multitasking models; however, these models face performance limitations when

using LoRA techniques. For example, Wang et al.(2023) proposed a multi-parallel LoRA architecture, known as MultiLoRA [142], which reduces parameter sharing and enhances multitask adaptation by eliminating top singular vectors through optimized parameter initialization. To ensure consistency and generalizability, the authors of the MultiLoRA article followed the data verbalization strategies of QLoRA [26] and MeZO [103] to standardize input formats across diverse tasks, including instruction following Stanford Alpaca [134], world knowledge MMLU [51], arithmetic reasoning GSM8K [19], and natural language understanding SuperGLUE [139], while incorporating random shuffling during training to mitigate order-dependent biases and enhance robustness across benchmarks. Zhang et al.(2023) developed an algorithm called AdaLoRA [168] that can flexibly allocate parameterized settings to distribute more parameters based on weight matrices that are more important during the fine-tuning stage. Thus, it regulates the orthogonality of the component matrix of SVD by adding an additional penalty to the training loss, and iteratively removes singular values based on their importance score during training to control the budget (i.e., total singular value). Chen et al.(2024) introduced LongLoRA [17], which enhances fine-tuning for long-context models by combining the original LoRA approach with a shifted sparse attention (S^2-Attn) mechanism. The S^2-Attn efficiently approximates short-range attention in extended contexts, while LongLoRA addresses increasing variance in fine-tuning performance as sentence length grows. This approach improves attention across long contexts, making fine-tuning more efficient and stable for lengthy inputs. DoRA [90] refines fine-tuning by decomposing pretrained weights into magnitude and direction, applying LoRA [59] only to the direction component to stabilize gradients, improving efficiency and performance with about half the parameters of traditional LoRA [59]. Tested on multimodal tasks like visual question answering and image/video-text understanding, DoRA outperformed both full fine-tuning and standard LoRA [59] techniques.

Optimized LoRA. The various methodologies stemming from LoRA [59], which are combined with the model optimization method, are a key area of reparameterization tuning research, referred to as the optimized LoRA. One of these studies is the quantized LoRA, which augments LoRA by employing the quantization of the weight matrix into lower-bit integers (e.g., INT4), thereby minimizing memory consumption. Specifically, Dettmers et al.(2023) introduced QLoRA [26], a technique that incorporates a 4-bit normal float to circumvent the computational burden of the quantile quantization technique by fixing the input tensor distribution within the range of -1 to 1. Additionally, they proposed a double quantization strategy for quantization constants and introduced a paging optimization technique to manage memory overflow issues by alternating data between the GPU and CPU. QA-LoRA [156], an advanced version of QLoRA, addresses parameter imbalance by using group-wise operators, increasing quantized parameters while reducing adaptation parameters for more efficient computation. A key aspect of LoftQ [84] is its strategic initialization by applying both quantization and low-rank approximation techniques before fine-tuning. Following the quantization method in QLoRA [26], it approximates before and after quantization to accurately follow the weights of pretrained models. Figure 6 provides a visualized comparison between LoRA [59], QLoRA [26] and QA-LoRA [156] in terms of how it is updated efficiently. Moreover, to mitigate quantization errors, the LQ-LoRA [43] model was developed, which segregates the weight matrix into quantized and low-rank components, utilizing only low-rank components for adjustment purposes. In another study, Zhang et al.(2024) presented a combination of pruning and LoRA called LoRAPrune [166]. Pruning methods for large pretrained models are not suited to LoRA because the use of unstructured pruning of large pretrained models relies on the gradients of pretrained weights, which can cause significant memory overhead. They proposed a structured iterative pruning mechanism based on gradients, removing about 5% of unnecessary weights per iteration while leveraging LoRA's weight and gradient computations for importance estimation instead of pretrained weight gradients. Compared to AdaLoRA [168]

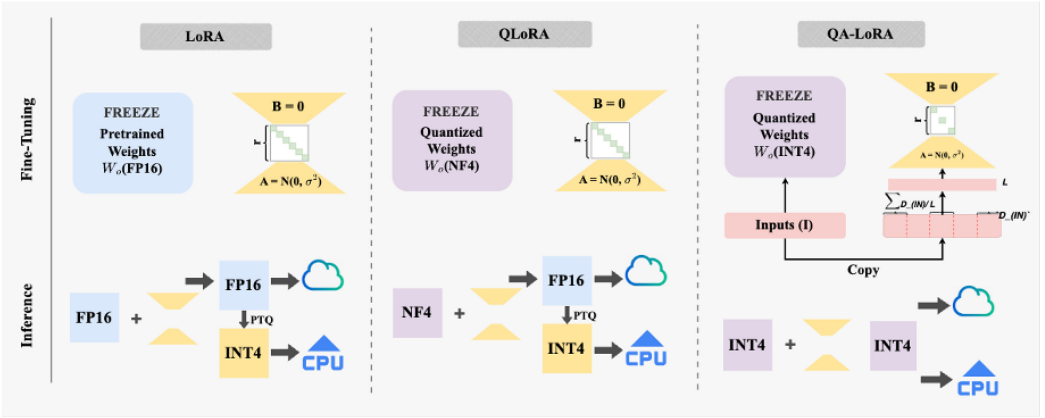

Fig. 6. Comparison between LoRA(FloatingPoint 16-bit) [59], QLoRA(NormalFloat 4-bit) [26], and QA-LoRA(Integer 4-bit) [156]. Compared to prior adaptation methods, QA-LoRA [156] can be computationally efficient in both the fine-tuning and inference phases. This enhances the representations of the LoRA models through high-rank initialization and efficiently manages the increased parameters by updating them in a sparse manner.

and LoRA [59], **Sparse LoRA** (**SoRA**) [29] assumes adaptation is inherently low-dimensional, using optimized gate units to control rank under gate sparsity and dynamically removing zero-rank parameters via proximal gradient iteration theory. VeRA [73] improves parameter efficiency in LoRA by replacing trainable low-rank matrices in each layer with two fixed random matrices shared across all layers, along with small, trainable scaling vectors that adapt the model. These scaling vectors selectively activate matrix elements, enabling fine-tuning with minimal parameters per layer and memory efficiency since only scaling vectors and seed information are stored. VeRA achieved enhanced performance in instruction tuning tasks on the LLaMA-2 [135] model while using less than 1/100th of the parameters required by standard LoRA [59].

Utilized LoRA. GLoRA [12] employs a generalized prompt framework to optimize pretrained models, enhancing task flexibility by controlling activations, setting multi-path networks, and unifying tunable space formulations across multiple dimensions. Furthermore, it supports effective parameter adaptation through the implementation of a scalable layer-wise search, which acquires individual adapters for each layer and involves no extra inference costs owing to structural reparameterization. Moreover, Sheng et al.(2024) and Chen et al.(2024) introduced the S-LoRA [126] and Punica [13] techniques, respectively, which significantly enhance GPU efficiency by optimizing memory and computation through tensor parallelism for multi-GPU inference, unified paging for adapter weights in KV caches, and CUDA kernel optimizations for efficient batch processing across LoRA models.

5.2 Query Tuning

The primary concept of query tuning is to tune prompts or prefixes such that the model responds in the desired manner. This makes it more flexible and responsive to user input.

5.2.1 Prompt Tuning. Prompt tuning involves guiding the response of the model to align it with the desired output style or content. A pioneering study in this field is the proposal of AutoPrompt [127], which automatically generates prompt templates using a gradient-based search mechanism. This mechanism streamlines development by automating high-quality prompt generation without manual crafting, enhancing generalizability through adaptive prompts, randomized data splits, and

balanced training to prevent overfitting. Furthermore, prompt tuning [78] maintains the integrity of the entire language model while adopting soft prompts [116] that are conceptualized as vector representations to tailor the model for specific tasks. This was achieved by appending a few trainable tokens to the input text. Compared to prompt tuning [78], ATTEMPT [4] uses a mixture of soft prompts to transfer knowledge across tasks, integrating multiple source prompts to generate instance-level prompts for each target task. Throughout the training process, only the prompt for the target task and the attention weights, which are shared across tasks, are updated, while keeping the remainder unchanged. Wang et al. [145] introduced a multitask learning technique that consolidates diverse task-specific prompts into a unified prompt, followed by low-rank adaptations to customize prompts for each downstream task. This approach enables efficient and rapid prompt tuning without modifying the core model architecture. Furthermore, Xiao et al.(2023) introduced decomposed prompt tuning [153], which uses a low-rank strategy to re-parameterize soft prompts as a product of two compact matrices rather than relying on random initialization. This approach reduces the number of trainable parameters by adjusting bottlenecks through control of the intermediate dimension size. The LLaMA-Adapter [169] enhances LLaMA fine-tuning by adding trainable adaptation prompts as prefixes in the input layer, addressing the challenge of random initialization through a zero-initialized attention mechanism. The inclusion of zero-initialized attention mechanisms ensures that the original pretrained knowledge is preserved while progressively injecting new instructional signals, providing a robust and consistent performance across varied tasks and domains. It employs a gating factor with independent SoftMax functions to separate pretrained knowledge from adaptation prompts, enabling efficient multi-head utilization and rapid fine-tuning within an hour using 1.2M parameters. Zhu et al.(2024) proposed IAPT [172], a method that places prompt generators in each model layer to produce soft prompts based on input instructions. Each prompt generator includes a down-projection for dimensionality reduction, self-attention pooling to highlight important information, a learnable activation function for task adaptability, and an up-projection to restore original dimensions. While self-attention pooling remains layer-specific to ensure performance, the other components are shared across layers to enhance parameter efficiency.

5.2.2 Prefix Tuning. Prefix tuning [82], introduced by Li and Liang (2021), adds continuous task-specific vectors (prefixes) that function like virtual tokens, influencing subsequent tokens while being the only trained parameters. To overcome these limitations regarding the universality across layers, a p-tuning framework [91] was proposed. This framework integrates prefix tokens across all model layers, deeply influencing predictions, and demonstrates consistent fine-tuning-level performance across models from 330M to 10B parameters using only 0.1% task-specific parameters. Based on the p-tuning framework [91], Obadinma et al. [109] authors examined data augmentation's effectiveness with p-tuning [91] and LoRA [59] techniques, especially in settings where data is limited. Their findings show that while data augmentation can enhance LLM performance, inconsistencies arise, with some techniques reducing performance in complex tasks or larger models, emphasizing the need for careful technique selection to maintain sentence representation quality. However, by incorporating a contrastive loss function, this mitigates inconsistencies by enhancing robustness to noisy data and improving augmented data handling, though it may introduce significant computational overhead. To address such problem, **Augmentation with Projection** (**AugPro**) [144] was introduced for an effective data augmentation method, backboned by MixUp [165] and **Fast Gradient Signed Method** (**FGSM**) [38] algorithms that distill on the word embeddings and labels, and robustness of its model respectively. In terms of its efficiency, such method lowers its complexity and can be computed in parallel, saving much of high computational overheads. To overcome the challenge of performance drop when modeling long documents,

Prefix-Propagation [79], which is an approach of conditional prefixing on previous hidden states through attention computation, was developed. Unlike prefix tuning [82], this method appends supplementary hidden states before projection through pretrained weight matrices, allowing prefixes to attend globally to other hidden states and enhancing the model's representation capacity. Through these methods, the performance is enhanced compared to the approach of prefix tuning [82], while employing 50% fewer parameters. Meanwhile, Maini et al. (2024) introduced **Web Rephrase Augmented Pretraining** (**WRAP**) [102], which uses an instruction-tuned model to paraphrase web documents into specific styles, leveraging real and synthetic data to enhance training efficiency, mitigate data noisiness, and improve downstream task performance.

5.3 Inference Tuning

Inference tuning involves optimizing the parameters and configurations of a machine learning model to improve its performance during inference. This process often leverages hardware accelerations, such as GPUs or TPUs, to speed up computation and reduce latency.

5.3.1 Specialized Hardware Acceleration. Specialized hardware acceleration involves the use of dedicated components, such as GPUs for training and testing models and NVMe for data loading and storage, to expedite specific computational tasks. This approach enhances the performance and efficiency of processes, like data transfer, storage operations, and neural network execution.

Efficient hardware co-design. One recent method, FlashAttention [23], enhances attention computation by using block-sparse attention to optimize data movement between the GPU high-bandwidth memory and SRAM, resulting in a 15% training speedup over baselines like BERT [28]. However, despite being faster than prior methods, FlashAttention still lags behind the optimized GEMM in terms of FLOPs utilization. FlashAttention-2 [22] addresses this issue by doubling speed and increasing forward and backward pass rates by 73% and 63%, respectively, through enhanced parallelism. It distributes work across GPU warps, reducing shared memory use and boosting throughput, thus enabling efficient handling of longer sequences with high contextual quality. To further optimize FlashAttention, **Sparse Causal FlashAttention** (**SCFA**) [110] was introduced, incorporating flexible sparsity patterns. It expands beyond the triangular causal masks of FlashAttention, enabling the efficient processing of models requiring nonstandard masking, such as those that dynamically drop keys/queries or use hashing-based attention. The SCFA utilizes QK-sparse and hash-sparse kernels, which dynamically prune attention computations based on relevance, reducing overhead and memory use while retaining exact attention. With these kernels, the SCFA achieves significant speedups on long sequences, outperforming FlashAttention without compromising model perplexity.

To further improve inference efficiency, DeepSeek-V2 [86] introduces the **multi-head latent attention** (**MLA**) mechanism, a technique designed to reduce KV-cache memory overhead while maintaining high attention fidelity in long-sequence processing. Unlike the standard **multi-head self-attention** (**MHA**) mechanism, which stores full-precision KV caches for all tokens during auto-regressive decoding, MLA applies low-rank joint compression to both keys and values that improve the cache locality and reduce the redundant computations by sharing a compressed latent representation across multiple attention heads, allowing for faster retrieval and reduced storage requirements. Specifically, MLA ensures that only the most relevant KV-pairs are stored at full precision while lower-rank approximations are used for less critical entries, effectively balancing memory efficiency with computational performance. Compared to prior KV-caching optimizations, MLA achieves notable reductions in inference latency while enabling the model to scale efficiently for long-context reasoning and generative tasks. DeepSeek-V3 [87] extends these principles further by refining latent compression techniques to optimize contextual memory usage, but the core MLA

mechanism remains consistent across both versions, emphasizing joint compression strategies to enhance throughput and reduce memory bottlenecks in high-performance inference.

Efficient inference accelerator. Another hardware accelerator method is DeepSpeed [2], a GPU-only inference solution designed to lower LLM computational costs by combining a GPU, CPU, and NVMe in a heterogeneous approach for limited GPU memory environments. DeepSpeed employs several parallelism techniques, including tensor parallelism, which distributes layers across multiple GPUs to improve memory efficiency, and pipeline parallelism, which splits the models into stages to address memory limitations. Expert parallelism was used for sparse models with mixtures of expert layers, assigning different experts to different **graphics processing units** (**GPUs**) for efficient processing. Using these methods, DeepSpeed can reduce inference latency by up to 7.3 times while increasing throughput by up to 1.5 times. Another method that has shown novelty in fast inference is the X-Former [129] architecture. X-Former utilizes a hybrid in-memory approach by combining NVMe, called **resistive RAM** (**ReRAM**) crossbars, highly optimized for executing massively parallelized **matrix-vector multiplications** (**MVM**) to be performed within the memory arrays, and a CMOS-based attention engine, called 8T-SRAM cells, that performs dynamic matrix multiplications for the attention mechanism. The projection engine handles static MVM operations in which one of the operands (query, key, or value) remains the same for each input, allowing the projection engine to be stored without the need for constant rewriting, and the attention engine handles dynamic NVM operations, where both operands change, making it more efficient than NVM cells in lowering writing latency and energy consumption. The authors tested X-Former's mechanism on GLUE tasks with BERT [28], achieving energy reductions of approximately 52.1x for the base model and 36.8x for the large model compared with conventional SRAM accelerators. Highlighting the practical importance of the use of AI on devices with LLMs, MobileLLM [96] employs a deep-and-thin structure for abstraction, embedding sharing to reduce parameters without accuracy loss, and grouped-query attention to minimize redundancy, enabling efficient deployment while maintaining strong performance on mobile hardware. Moreover, MobileLLM employs a blockwise layer-sharing approach to reuse weights within the memory hierarchy, increasing layer depth without additional memory overhead, enabling state-of-the-art performance in reasoning, reading comprehension, chat, and API tasks for on-device applications.

6 Real-world Applications

RQ3: What are the real-world applications of compression and tuning techniques with LLMs? This section explores the real-world applications of efficient compression and tuning techniques in LLMs, highlighting how these advancements are transforming various fields into two key categories: cross-domain applications, where models are adapted to perform effectively across diverse tasks or industries, and cross-lingual applications, where they bridge linguistic barriers to enable seamless multilingual communication and understanding.

6.1 Cross-domain Applicability

The efficient compression and tuning methods we discussed have proven to be practically effective in various domains, such as biomedicine, healthcare, finance, and legal. Table 5 provides a comprehensive analysis comparing various real-world cross-domain and cross-lingual applications for efficient compression and tuning techniques.

Medical. In the literature, the primary method for evaluating medical language models is multiple-choice question-answering, with accuracy serving as the primary metric. Following this convention, we adopted three prominent medical QA benchmarks for evaluation: PubMedQA [65], MedMCQA [111], and MedQA [64]. For instance, Wu et al.(2023) developed the PMC-LLaMA [149] for medical applications and demonstrated that LoRA [59] outperformed p-tuning [92] in these

Table 5. Comparison Analysis of Practical Cross-domain and Cross-lingual Applications with Efficient Compression and Tuning Techniques

Model	Year	Method Types							Baselines	Domains	Benchmarks
		KD	LRS	PR	QN	PEFT	QT	IT			
MUNMT [130]	2020	✓							Unified Encoder-Decoder	Multilingual Machine Translation	WMT monolingual news
PMC-LLaMA [149]	2023					✓	✓		LLaMA LLaMA-2 ChatGPT Med-Alpaca Chat-Doctor	Medical	PubMedQA MedMCQA MedQA
LoRDCoder [67]	2023		✓		✓	✓		✓	CodeGen StarCoder	Programming Codes	HumanEval
TinyChatEngine [85]	2023				✓			✓	LLaMA LLaMA-2 LLaVA-7B OPT	General	WikiText-2 MS-COCO
DA [63]	2024	✓			✓	✓			LLaMA	Biomedicine Finance	PubMedQA USMLE BioMMLU FiQA Headline FPB
FinLLM [44]	2024				✓	✓	✓		GPT-3.5 LLaMA-3	Finance	News (Abstracts)
D-Pruner [167]	2024	✓		✓	✓				LLaMA-2 BLOOM	Healthcare Laws	MedNLI PubMedQA HQS CaseHOLD BillSum
MBS [164]	2024			✓	✓				BLOOM	Multilingual Summarization	CC-100

KD: Knowledge Distillation; LRS: Low-Rank Strategies; PR: Pruning; QN: Quantization; PEFT: Parameter-Efficient Fine-Tuning; QT: Query Tuning; IT: Inference Tuning.

medical QA tasks. Notably, LoRA [59] achieved nearly four times the time efficiency and three times the memory efficiency compared to full fine-tuning with only a slight tradeoff in accuracy, highlighting its effectiveness for medical QA tasks.

Biomedicine and Finance. Inspired by regex-based AdaptLLM [18] for domain adaptation through reading comprehension, Jiang et al. [63] extended this approach to improve domain-specific language modeling, particularly for biomedicine and finance benchmarks. Their method integrated LLM-based preprocessing to generate high-quality question-answer pairs, enhance comprehension, and enable efficient domain adaptation. By applying **parameter-efficient fine-tuning (PEFT)** techniques such as LoRA [59], which optimizes self-attention layers with minimal memory usage, and knowledge distillation to fine-tune a compact LLaMA-7B [136] model, the approach significantly reduces the computational overhead. To overcome the challenge of limited context in datasets such as PubMed abstracts and financial documents, length-based clustering was used to extend the input context by grouping similar documents. This clustering enriches the corpus and improves the domain-specific question-answering performance on benchmarks such as PubMedQA [65], MedQA [111], and FiQA [101]. Overall, the combination of the extended context, LoRA, and efficient knowledge distillation ensures high-quality domain adaptation while maintaining computational efficiency.

Finance. Guo et al. [44] developed an LLM for the abstract summarization of financial news by applying three methods: CoT, instruction tuning, and reinforcement learning. Based on query tuning, the authors used QLoRA [26] to quantize the LLaMA-3 [32] model using the BitsandBytes 4-bit quantization library [34], and the resulting quantized FinLLM achieved the highest performance

for all ROUGE scores. These applications highlight the potential for cross-domain applications because the efficiency and modularity of LoRAs [59] can be adapted to optimize their performance in various specialized fields beyond medicine and finance, such as legal documents and math problem-solving tasks.

Healthcare and Laws. D-Pruner [167] employs a dual-pruning methodology to effectively adapt LLMs to domain-specific benchmarks, such as PubMedQA [65], **Health Question Summarization** (**HQS**) [7], MultiLegalPile [66], and CaseHOLD [171]. This technique preserves crucial weights for both general language processing and domain-specific knowledge by incorporating weight importance into training via a regularization term, balancing linguistic proficiency and multi-task versatility. D-Pruner evaluates benchmarks using a domain-specific calibration dataset, identifying essential weights through gradient-based importance scoring and leveraging empirical Fisher approximations [133] for computational efficiency. By pruning the LLaMA-2 7B [135] model to 50% sparsity, D-Pruner achieved competitive performance on healthcare and legal datasets, maintaining high accuracy in question-answering tasks, such as PubMedQA [65] and CaseHOLD [171], as well as summarization tasks in the legal and healthcare domains. This efficiency is bolstered by iterative blocking and the integration of task-agnostic pretraining objectives, demonstrating the robustness of the model across diverse domain-specific tasks.

Programming. Kaushal et al.(2023) developed LoRDCoder [67], a compressed version of the existing StarCoder-16B [81] code generation LLM, by applying LoRD to linear layers such as those used in multi-headed attention (i.e., query, key, and value projection matrices) and MLP layers. The rank reduction allowed compression to 12.3B, with a slight decrease of approximately 2 points in the HumanEval [14] score compared to the base model, whereas the inference speed increased by up to 22.35%.

6.2 Cross-lingual Applicability

Efficient compression techniques, originally designed and optimized for high-resource languages such as English, have also demonstrated adaptability and effectiveness in low-resource languages, enabling these methods to maintain robust performance across diverse linguistic contexts with appropriate calibration and fine-tuning strategies.

Machine translation. Sun et al. [130] demonstrated strong cross-lingual applicability using a unified encoder-decoder architecture capable of translating 13 languages from three language families and six language branches. It achieves this by leveraging multilingual pretraining, shared latent representations, and advanced knowledge distillation techniques (self-knowledge and language branch knowledge distillation) to enhance zero-shot translation, improve the performance for low-resource languages, and effectively generalize across diverse linguistic contexts. This technique can be extended to other cross-lingual applications, such as cross-lingual information retrieval, sentiment analysis, and question answering, where shared latent representations and multilingual pretraining can enable effective knowledge transfer across languages, particularly for low-resource scenarios.

Chatbot. The authors of the AWQ technique, Lin et al. [85], introduced TinyChatEngine, an on-device inference library for LLMs. This library applies both SmoothQuant [152] and AWQ techniques to various LLMs in real-time, achieving over three times the speed of the FP16 implementation. The TinyChat engine integrates hardware optimization methods, such as SIMD-aware weight packing and kernel fusion, making it compatible with desktops and mobile GPUs.

Multilingual calibration. Zeng et al. [164] introduced the **Multilingual Brain Surgeon** (**MBS**) method and extended the **Optimal Brain Surgeon** (**OBS**) [76] framework to enhance multilingual LLM compression. Unlike traditional methods that focus on monolingual calibration data, typically in English, MBS employs proportional calibration data sampling aligned with the

language distribution in the training data of the model. This innovation ensures performance retention across both high- and low-resource languages by carefully pruning fewer critical connections while maintaining multilingual parameters. Specifically, MBS demonstrated robust performance on benchmarks such as XL-Sum [48] and BLOOM [123] by minimizing perplexity increases across diverse languages, including underrepresented languages. This method leverages Hessian matrix approximations for informed parameter pruning, balancing computational costs and model integrity. By aligning calibration data with language prevalence, MBS not only preserves cross-lingual and spoken language understanding capabilities but also provides a scalable framework applicable across multilingual LLM compression tasks.

7 Key Findings and Reflections

RQ4: What are the key findings and reflections when applying compression or tuning techniques? Our exploration of efficient compression and tuning techniques for LLMs revealed critical insights into optimizing performance, balancing resource constraints, and enhancing the synergy between model adaptability and applicability.

Efficient model updating. While efficient tuning methods enable model updates without full retraining, they typically assume initial training only, necessitating retraining for new data. As data demands grow, efficient tuning alone becomes insufficient for minimizing resource use. Continual learning offers a promising solution by enabling models to adapt to new data and tasks incrementally without full retraining, making it a resource-efficient approach to evolving requirements [45]. One study applying continual learning was by Chen et al. [16], which focused on newly incoming data from diverse distributions. Specifically, they leverage **mixture-of-experts** (**MoE**) architectures such as GShard [77] and GLaM [31], which allow the scaling model capacity to match different data distributions without incurring additional computational costs. Specifically, when new data with different distributions is introduced, existing experts are frozen to retain the previously learned knowledge, new experts are added, and their outputs are normalized to ensure that the model adapts to the new data. This lifelong MoE technique allows the model to retain the benefits of prior learning while incorporating new data and maintaining a stable computational cost.

Building on this idea, DeepSeek-V3 [87] refines the MoE-based routing mechanism by optimizing expert selection and workload balancing, ensuring that only the most relevant experts are activated per inference step. Unlike GShard [77], which statically partitions experts based on non-overlapping tokenized routing, DeepSeek-V3 dynamically selects a specialized subset of experts on a per-query basis that was initially introduced in DeepSeekMoE [21] architecture, allowing for finer-grained expert activation and reducing unnecessary computations. This adaptive gating function minimizes load imbalance, a common issue in traditional MoE implementations, by dynamically distributing computational resources across different expert pathways. This efficient routing mechanism ensures that DeepSeek-V3 retains the scalability benefits of MoE, while simultaneously reducing computational overhead per query, making it highly suitable for large-scale, inference-heavy applications. By integrating continual learning principles with a refined MoE framework, DeepSeek-V3 achieves greater efficiency in handling evolving data distributions, maintaining high adaptability without the instability issues found in earlier GShard-based MoE architectures.

Interaction between compression and tuning. Combining compression techniques such as pruning and LoRA can introduce challenges such as underfitting and reduced generalization. Pruning introduces sparsity by removing less significant parameters, and when paired with LoRA (e.g., LoRAPrune [166]), it can significantly reduce the representational capacity of the model. This reduced capacity may lead the model to underfit the training data, fail to capture complex patterns effectively, and require additional fine-tuning to restore performance. Similarly, when quantization

is combined with LoRA (e.g., LoftQ [84]), the restricted tuned parameters may lack the flexibility required for task-specific nuances. PEFT methods such as LoRA [59] may be susceptible to such precision loss, impairing the expressiveness of the model. Altogether, these combined compression techniques can lead to models that generalize poorly to unseen data, owing to their limited adaptability and capacity.

However, a careful combination of these techniques can mitigate some issues. For instance, knowledge distillation can be used to retain the generalization ability of larger models, even when they are compressed. A distilled model trained on the output of a larger teacher model can maintain its performance of the larger model despite aggressive compression. By aligning the output distributions of the student and teacher models, knowledge distillation can help the student model retain rich, generalized knowledge, even when quantization is applied [68]. Additionally, integrating techniques such as LoRA [59] with structured pruning methods such as Wanda [132] allows for more flexible and task-specific parameter tuning without sacrificing model capacity. By selectively pruning parts of the model that are redundant or less important (i.e., weights and activations) for a specific task while allowing the LoRA to tune key parameters, the model can maintain both efficiency and generalization capacity. Although combining compression and tuning methods can lead to underfitting and reduced generalization, careful selection and balancing of the techniques that we introduced can help overcome these issues.

Applicability across model architectures. Compression and tuning techniques apply differently to model architectures, with transformers being particularly well suited for compression methods such as quantization and pruning because of their self-attention mechanism and parallel processing capability. RNNs, such as LSTMs and GRUs, excel in handling temporal dependencies but face scalability challenges and reduced effectiveness when applying compression techniques. Quantization effectiveness differs significantly between Transformers and RNNs because of their distinct architectures. Transformers handle lower-bit precision (e.g., 8-bit or 4-bit) effectively, leveraging parallel processing to improve speed and memory efficiency without substantial performance degradation. Conversely, RNNs that rely on sequential processing are more vulnerable to error accumulation with aggressive quantization, which can compromise the long-term dependency [1, 49]. Pruning affects Transformers and RNNs differently. For Transformers, structured and unstructured pruning can reduce the model size by removing redundant parameters, such as attention heads, without significantly affecting performance, owing to the modular nature of attention mechanisms. RNNs, which reuse parameters over time steps, are less suited to aggressive pruning. However, structured pruning, such as removing neurons, gates, or layers, can improve the efficiency of models such as LSTMs and GRUs with built-in redundancy [88, 107]. However, the sequential nature of RNNs limits the extent to which pruning can reduce computational complexity without compromising accuracy. Consequently, compression techniques such as quantization and pruning are less common in RNNs owing to less favorable parameter efficiency and performance tradeoffs.

8 Conclusion and Future Directions

Rapid advancements in LLMs and growing computational demands for their development and deployment have made it both urgent and crucial to research efficient compression and tuning techniques. In this survey, we present a systematic literature review of recent state-of-the-art methods for enhancing the efficiency of LLMs using compression and tuning techniques. By comprehensively analyzing these approaches, we highlighted how they address the pressing issues of high resource consumption, difficulty in accessibility, and applicability for individual researchers and general consumers while preserving the remarkable performance of these powerful LLMs. This encompasses knowledge distillation to reduce model size, low-rank strategies to optimize

parameter utilization, parameter pruning to eliminate redundancy, quantization to enhance computational efficiency, and parameter-efficient fine-tuning alongside query and inference tuning to streamline task-specific performance. Finally, we provide potential directions for future research as follows:

Scalability of compression strategies. In-depth research to enhance the scalability of efficient compression and tuning techniques for LLMs is a critical and promising avenue for future research. Recent advancements in efficient compression and tuning techniques for LLMs have laid the foundation for long-context scalability; however, several challenges remain in this dynamic and evolving field. One promising direction is developing methods that mitigate the computational overhead associated with increasing context lengths. For instance, LongLoRA [17] addresses the exponential growth in computational costs by segmenting contexts and enabling efficient cross-attention across token groups, thereby achieving a context length of 100k tokens on a single GPU. Similarly, LongRoPE [30] extends the context lengths up to eight times by optimizing the rotary position embeddings to minimize information loss. Another significant advancement is KVQuant [54], which uses 3–4-bit quantization techniques, including GPTQ [34] and SqueezeLLM [69], to scale context lengths to one million tokens on a single A100 GPU (80GB). These methods demonstrate the potential of combining innovative approaches to address the computational and memory challenges associated with long-context processing. Future work should explore hybrid strategies that integrate context partitioning, embedding optimizations, and advanced quantization techniques while also investigating their integration with hardware-aware designs to ensure scalability and efficiency in fast inference response throughput.

Tailoring user-centric optimization strategies. Tradeoffs between latency, accuracy, and resource efficiency often require careful consideration; however, significant challenges remain in tailoring these methods to user-centric optimization strategies. For instance, techniques such as LLM.int8() [25] prioritize accuracy preservation, and the LLM-Pruner [98] focuses on reducing latency. Experiments with the LLM-KICK benchmark [62] highlight that although compression techniques such as SparseGPT [36], SqueezeLLM [132], and GPTQ [34] can significantly improve performance metrics, exceeding certain compression thresholds often leads to substantial performance degradation. To address this issue, future research should explore adaptive optimization strategies that allow users to balance the compression levels with their specific latency and accuracy requirements. For instance, optimizing real-time response tasks may involve setting latency thresholds, as demonstrated by experiments showing the impact of compression on models, such as LLaMA3-8B [32], when deployed on resource-constrained hardware. In addition to compression, integrating hardware-aware approaches and exploring hybrid techniques that combine compression with advanced tuning methods are potential directions for creating scalable and user-focused LLM solutions that satisfy diverse application demands.

Long-term performance stability. Ensuring the long-term performance stability of LLMs when applying efficient compression and tuning techniques remains a critical challenge. Chen et al. [16] demonstrated that warm-up techniques with incremental learning rates applied to a small fraction of data could preserve long-term stability and mitigate performance degradation over time. By fine-tuning and rewarming models, such as Pythia 410M [10], they achieved stable performance on datasets, such as Pile and SlimPajama, with minimal increases in validation loss and perplexity, highlighting the importance of gradual adaptation. Similarly, Huang et al. [61] proposed the **self-synthesized rehearsal** (**SSR**) method, which generates synthetic data to retain prior knowledge, particularly in scenarios where access to the original datasets is limited. SSR enables continual learning while countering knowledge erosion, making it particularly valuable in resource-constrained settings. Moreover, integrating SSR with compression techniques can offset the information loss typically associated with model compression, helping to stabilize long-term

performance. These approaches suggest that future research should focus on adaptive methods that combine fine-tuning, synthetic data generation, and compression strategies to ensure that LLMs maintain their capabilities over time while addressing the challenges of continual learning and efficient resource utilization.

Hierarchical efficiency methodology. The exploration of hierarchical efficiency methodologies presents a promising direction for improving the scalability and resource utilization of LLMs. For instance, TRIFORCE [131] employs a hierarchical speculative decoding strategy, where an initial lightweight model, such as LLaMA-68M, generates rapid preliminary predictions that are subsequently verified and refined using a larger model, such as LLaMA2-7B-128k [114]. This hierarchical two-layered approach significantly reduces the resource requirements in the early prediction stages while maintaining quality output, leading to improved computational efficiency. Similarly, the HiFT method [94] introduces a blockwise hierarchical update mechanism in which the model layers are divided into distinct blocks that can be activated or deactivated sequentially. By updating blocks in top-to-bottom, bottom-to-up, or random order, HiFT ensures that only active blocks are loaded into the GPU memory during a given training step, thereby reducing memory consumption by an average of 89%. These hierarchical approaches demonstrate the potential for integrating lightweight and modular design principles into compression and tuning techniques to optimize memory and computational efficiency. Future research should focus on investigating how these hierarchical methodologies can be extended and adapted to different architectures and application domains to enhance the efficiency and scalability of LLMs.

References

[1] Md. Zahangir Alom, Adam T. Moody, Naoya Maruyama, Brian C. Van Essen, and Tarek M. Taha. 2018. Effective quantization approaches for recurrent neural networks. In *International Joint Conference on Neural Networks, Rio de Janeiro, Brazil, July 8–13, 2018*. 1–8.

[2] Reza Yazdani Aminabadi, Samyam Rajbhandari, Minjia Zhang, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Jeff Rasley, Shaden Smith, Olatunji Ruwase, and Yuxiong He. 2022. DeepSpeed-Inference: Enabling efficient inference of transformer models at unprecedented scale. In *International Conference for High Performance Computing, Networking, Storage and Analysis, Dallas, TX, USA, November 13–18, 2022*.

[3] Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. 2024. Fluctuation-based adaptive structured pruning for large language models. In *38th AAAI Conference on Artificial Intelligence, 36th Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, 14th Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20–27, 2024, Vancouver, Canada*.

[4] Akari Asai, Mohammadreza Salehi, Matthew E. Peters, and Hannaneh Hajishirzi. 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, Abu Dhabi, United Arab Emirates, December 7–11, 2022*.

[5] Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. SliceGPT: Compress large language models by deleting rows and columns. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024*.

[6] Loic Barrault, Ondrej Bojar, Marta Ruiz Costa-jussa, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Muller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the 14th Conference on Machine Translation, WMT 2019, Florence, Italy, 2019 - Volume 2: Shared Task Papers*.

[7] Asma Ben Abacha, Yassine Mrabet, Yuhao Zhang, Chaitanya P. Shivade, Curt P. Langlotz, and Dina Demner-Fushman. 2021. Overview of the MEDIQA 2021 shared task on summarization in the medical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*.

[8] Yoshua Bengio, Nicholas Leonard and Aaron C. Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. https://arxiv.org/abs/1308.3432

[9] Chandra Bhagavatula, Jena D. Hwang, Doug Downey, Ronan Le Bras, Ximing Lu, Keisuke Sakaguchi, Swabha Swayamdipta, Peter West, and Yejin Choi. 2023. I2D2: Inductive knowledge distillation with NeuroLogic and self-imitation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, Canada, July 9–14, 2023*.

[10] Stella Biderman, Hailey Schoelkopf, Quentin G. Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the International Conference on Machine Learning, 23–29 July 2023, Honolulu, Hawaii, USA.*

[11] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Ma-teusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020.*

[12] Arnav Chavan, Zhuang Liu, Deepak K. Gupta, Eric P. Xing, and Zhiqiang Shen. 2023. One-for-All: Generalized LoRA for Parameter-Efficient Fine-tuning. https://arxiv.org/abs/2306.07967

[13] Lequn Chen, Zihao Ye, Yongji Wu, Danyang Zhuo, Luis Ceze, Arvind Krishnamurthy, and Duke University. 2024. Punica: Multi-Tenant LoRA serving. In *Proceedings of the 7th Annual Conference on Machine Learning and Systems, Santa Clara, CA, USA, May 13–16, 2024.*

[14] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mo Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W. Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, Suchir Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever and Wojciech Zaremba. 2021. Evaluating large language models trained on code. https://arxiv.org/abs/2107.03374

[15] Patrick H. Chen, Hsiang-Fu Yu, Inderjit S. Dhillon, and Cho-Jui Hsieh. 2021. DRONE: Data-aware low-rank compression for large NLP models. In *Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, December 6–14, 2021, virtual.*

[16] Wuyang Chen, Yan-Quan Zhou, Nan Du, Yanping Huang, James Laudon, Z. Chen, and Claire Cu. 2023. Lifelong language pretraining with distribution-specialized experts. In *Proceedings of the International Conference on Machine Learning, 23–29 July 2023, Honolulu, Hawaii, USA.*

[17] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2024. LongLoRA: Efficient fine-tuning of long-context large language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[18] Daixuan Cheng, Shaohan Huang, and Furu Wei. 2024. Adapting large language models via reading comprehension. In *Proceedings of the 12th International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7–11, 2024.*

[19] Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse and John Schulman. 2021. Training verifiers to solve math word problems. https://arxiv.org/abs/2110.14168

[20] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. 2015. Low precision arithmetic for deep learning. In *Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, May 7–9, 2015, Workshop Track Proceedings.*

[21] Damai Dai, Chengqi Deng, Chenggang Zhao, Runxin Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. 2024. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Bangkok, Thailand, August 11–16, 2024.* Association for Computational Linguistics, 1280–1297.

[22] Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[23] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-Awareness. In *Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, New Orleans, LA, USA, November 28–December 9, 2022.*

[24] Rocktim Jyoti Das, Liqun Ma, and Zhiqiang Shen. 2023. Beyond size: How gradients shape pruning decisions in large language models. https://arxiv.org/abs/2407.21783

[25] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, New Orleans, LA, USA, November 28–December 9, 2022.*

[26] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, New Orleans, LA, USA, December 10–16, 2023.*

[27] Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. 2024. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[28] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers).*

[29] Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. 2023. Sparse low-rank adaptation of pre-trained language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Singapore, December 6–10, 2023.*

[30] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. LongRoPE: Extending LLM context window beyond 2 million tokens. In *Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria, July 21–27, 2024.*

[31] Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen S. Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V. Le, Yonghui Wu, Z. Chen, and Claire Cui. 2022. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the International Conference on Machine Learning, 17–23 July 2022, Baltimore, Maryland, USA.*

[32] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, and Artem Korenev. 2024. The Llama 3 herd of models. Retrieved from https://arxiv.org/abs/2407.21783

[33] Ali Edalati, Marzieh S. Tahaei, Ahmad Rashid, V. Nia, James J. Clark, and Mehdi Rezagholizadeh. 2022. Kronecker decomposition for GPT compression. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Dublin, Ireland, May 22–27, 2022.*

[34] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations 2023. Kigali Rwanda, May 1 - May 5, 2023.* 1–16. https://arxiv.org/abs/2210.17323

[35] Elias Frantar and Dan Alistarh. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, New Orleans, LA, USA, November 28–December 9, 2022.*

[36] Elias Frantar and Dan Alistarh. 2023. SparseGPT: Massive language models can be accurately pruned in one-shot. In *Proceedings of the International Conference on Machine Learning, 23–29 July, 2023, Honolulu, Hawaii, USA.*

[37] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. A survey of quantization methods for efficient neural network inference. https://arxiv.org/abs/2103.13630

[38] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.* https://arxiv.org/abs/1412.6572

[39] Jianping Gou, B. Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision* 129, 6 (2021), 1789–1819.

[40] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, 2024.*

[41] Cong Guo, Jiaming Tang, Weiming Hu, Jingwen Leng, Chen Zhang, Fan Yang, Yun-Bo Liu, Minyi Guo, and Yuhao Zhu. 2023. OliVe: Accelerating large language models via hardware-friendly outlier-victim pair quantization. In *Proceedings of the 50th Annual International Symposium on Computer Architecture, Orlando, FL, USA, June 17–21, 2023.*

[42] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruizhe Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaojin Zhang, Xingkai Yu, Yuxuan Wu, Zhenfeng Wu, Zhe Gou, Zhihong Shao, Zilin Li, Zhenda Gao, Aixin Liu, Bei Xue, Bin Wang, Bingxuan Wang, Bo Liu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, Jian Liang, Jiaqi Ni, Jianzhong Guo, Jia Li, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang

Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, Tian Pei, Tian Yuan, Tianyu Sun, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Liu, Xin Xie, Xingkai Yu, Xinnan Song, Xinyi Zhou, Xinyu Yang, Xuan Lu, Xuecheng Su, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Zheng, Yichao Zhang, Yiliang Xiong, Yilong Zhao, Ying He, Ying Tang, Yishi Piao, Yixin Dong, Yixuan Tan, Yiyuan Liu, Yongji Wang, Yongqiang Guo, Yuchen Zhu, Yuduan Wang, Yuheng Zou, Yukun Zha, Yunxian Ma, Yuting Yan, Yuxiang You, Yuxuan Liu, Zehui Ren, Zhangli Sha, Zhe Fu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhewen Hao, Zhihong Shao, Zhiniu Wen, Zhipeng Xu, Zhongyu Zhang, Zhuoshu Li, Zihan Wang, Zihui Gu, Zilin Li, Ziwei Xie, and Ziheng Pan. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. https://arxiv.org/abs/2501.12948

[43] Han Guo, Philip Greengard, Eric P. Xing, and Yoon Kim. 2024. LQ-LoRA: Low-rank plus quantized matrix decomposition for efficient language model finetuning. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024*.

[44] Lubingzhi Guo, Javier Sanz-Cruzado, and Richard McCreadie. 2024. University of glasgow at the FinLLM challenge task: Adapting Llama for financial news abstractive summarization. In *Proceedings of the 8th Financial Technology and Natural Language Processing and the 1st Agent AI for Scenario Planning*. 127–132.

[45] Kshitij Gupta, Benjamin Therien, Adam Ibrahim, Mats L. Richter, Quentin G. Anthony, Eugene Belilovsky, Irina Rish, and Timothee Lesort. 2023. Continual pre-training of large language models: How to (re)warm your model?. In *Proceedings of the Workshop on Efficient Systems for Foundation Models @ ICML2023*.

[46] Habib Hajimolahoseini, Mehdi Rezagholizadeh, Vahid Partovinia, Marzieh S. Tahaei, Omar Mohamed Awad, and Yang Liu. 2021. Compressing pre-trained language models using progressive low rank decomposition. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2021, December 6–14, 2021, Virtual*.

[47] Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. https://arxiv.org/abs/1510.00149

[48] Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Samin Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XL-Sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics, Online Event, August 1–6, 2021*.

[49] Qinyao He, He Wen, Shuchang Zhou, Yuxin Wu, Cong Yao, Xinyu Zhou, and Yuheng Zou. 2016. Effective quantization methods for recurrent neural networks. https://arxiv.org/abs/1611.10176

[50] Shwai He, Liang Ding, Daize Dong, Miao Zhang, and Dacheng Tao. 2022. SparseAdapter: An easy approach for improving the parameter-efficiency of adapters. In *Findings of the Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, December 7–11, 2022*.

[51] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of the 9th International Conference on Learning Representations, Virtual Event, Austria, 2021*.

[52] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. https://arxiv.org/abs/1503.02531

[53] Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, Canada, July 9–14, 2023*.

[54] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. 2024. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *Proceedings of the Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems, Vancouver, BC, Canada, December 10–15, 2024*.

[55] Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2020. DynaBERT: Dynamic BERT with adaptive width and depth. In *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, 2020*.

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

[57] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander J. Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! Outperforming larger language models with less

training data and smaller model sizes. In *Findings of the Association for Computational Linguistics, Toronto, Canada, July 9–14, 2023*.

[58] Yen-Chang Hsu, Ting Hua, Sung-En Chang, Qiang Lou, Yilin Shen, and Hongxia Jin. 2022. Language model compression with weighted low-rank factorization. In *Proceedings of the 10th International Conference on Learning Representations, Virtual Event, April 25–29, 2022*.

[59] J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of the 10th International Conference on Learning Representations, Virtual Event, April 25–29, 2022*.

[60] Ting Hua, Yen-Chang Hsu, Felicity Wang, Qiang Lou, Yilin Shen, and Hongxia Jin. 2022. Numerical optimizations for weighted low-rank estimation on language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, December 7–11, 2022*.

[61] Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Bangkok, Thailand, August 11–16, 2024*.

[62] Ajay Kumar Jaiswal, Zhe Gan, Xianzhi Du, Bowen Zhang, Zhangyang Wang, and Yinfei Yang. 2024. Compressing LLMs: The truth is rarely pure and never simple. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024*.

[63] Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. 2024. Improving domain adaptation through extended-text reading comprehension. https://arxiv.org/abs/2401.07284

[64] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *Applied Sciences* 11, 14 (2021), 6421. https://doi.org/10.3390/app11146421

[65] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, November 3–7, 2019*.

[66] Joel Niklaus, Veton Matoshi, Matthias Stürmer, Ilias Chalkidis, and Daniel E. Ho. 2024. MultiLegalPile: A 689GB multilingual legal corpus. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, L.-W. Ku, A. Martins, and V. Srikumar (Eds.). 15077–15094.

[67] Ayush Kaushal, Tejas Vaidhya, and Irina Rish. 2023. LORD: Low rank decomposition of monolingual code LLMs for one-shot compression. In *Proceedings of the International Conference on Machine Learning, 23–29 July 2023, Honolulu, Hawaii, USA*.

[68] Minsoo Kim, Sihwa Lee, Suk Joon Hong, Duhyeuk Chang, and Jungwook Choi. 2022. Understanding and improving knowledge distillation for quantization aware training of large transformer encoders. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, December 7–11, 2022*.

[69] Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. 2024. SqueezeLLM: Dense-and-sparse quantization. In *Proceedings of the 42st International Conference on Machine Learning, Vienna, Austria, July 21–27, 2024*.

[70] D. P. Kingma and J. Ba. 2014. Adam: A Method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1412.6980

[71] Barbara Ann Kitchenham. 2004. Procedures for performing systematic reviews. In *Joint Technical Report TR/SE0401 and 0400011T.1. Computer Science Department, Keele University and National ICT Australia Ltd.*

[72] Barbara Ann Kitchenham, Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen G. Linkman. 2009. Systematic literature reviews in software engineering - A systematic literature review. *Information and Software Technology* 51, 1 (2009), 7–15.

[73] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki Markus Asano. 2024. VeRA: Vector-based random matrix adaptation. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024*.

[74] Eldar Kurtic, Daniel Fernando Campos, Tuan Nguyen, Elias Frantar, Mark Kurtz, Ben Fineran, Michael Goin, and Dan Alistarh. 2022. The optimal BERT surgeon: Scalable and accurate second-order pruning for large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, December 7–11, 2022*.

[75] François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M. Rush. 2021. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Virtual Event / Punta Cana, Dominican Republic, 7–11 November, 2021*.

[76] Yann LeCun, John S. Denker, and Sara A. Solla. 1989. Optimal brain damage. In *Proceedings of the Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27–30, 1989]*.

[77] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam M. Shazeer, and Z. Chen. 2021. GShard: Scaling giant models with conditional computation and automatic sharding. In *Proceedings of the 9th International Conference on Learning Representations, Virtual Event, Austria, May 3–7, 2021*.

[78] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Virtual Event / Punta Cana, Dominican Republic, 7–11 November, 2021*.

[79] Jonathan Li, Will Aitken, Rohan Bhambhoria, and Xiao-Dan Zhu. 2023. Prefix propagation: Parameter-efficient tuning for long sequences. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Toronto, Canada, July 9–14, 2023*.

[80] Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023. Symbolic chain-of-thought distillation: Small models can also "Think" step-by-step. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, Canada, July 9–14, 2023*.

[81] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy V, Jason T. Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nour Fahmy, Urvashi Bhattacharyya, Wenhao Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2023. StarCoder: May the source be with you! *Transactions on Machine Learning Research*. 1–55.

[82] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, (Volume 1: Long Papers), Virtual Event, August 1–6, 2021*.

[83] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy V, Jason T. Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nour Fahmy, Urvashi Bhattacharyya, Wenhao Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. 2023. StarCoder: May the source be with you! *Transactions on Machine Learning Research*. 1–55.

[84] Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. 2024. LoftQ: LoRA-fine-tuning-aware quantization for large language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024*.

[85] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. 2024. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of the 7th Annual Conference on Machine Learning and Systems, Santa Clara, CA, USA, May 13–16, 2024*.

[86] DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, Hao Zhang, Hanwei Xu, Hao Yang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, Tao Wang, Tian Pei, Tian Yuan, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, and Xiaowen Sun. 2024. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. https://arxiv.org/abs/2405.04434

[87] DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng , Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024. DeepSeek-V3 Technical Report. https://arxiv.org/abs/2412.19437

[88] Shiwei Liu, Decebal Constantin Mocanu, Yulong Pei, and Mykola Pechenizkiy. 2021. Selfish sparse RNN training. In *Proceedings of the 38th International Conference on Machine Learning, 18–24 July 2021, Virtual Event.*

[89] Shih-Yang Liu, Zechun Liu, Xijie Huang, Pingcheng Dong, and Kwang-Ting Cheng. 2023. LLM-FP4: 4-Bit floating-point quantized transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Singapore, December 6–10, 2023.*

[90] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria, July 21–27, 2024.*

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

[92] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2024. GPT understands, too. *AI Open* 5 (2024), 208–215. https://doi.org/10.1016/j.aiopen.2023.08.012

[93] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. In *Proceedings of the 8th International Conference on Learning Representations, Virtual Event, April 26–May 1, 2020.*

[94] Yongkang Liu, Yiqun Zhang, Qian Li, Shi Feng, Daling Wang, Yifei Zhang, and Hinrich Schutze. 2024. HiFT: A hierarchical full parameter fine-tuning strategy. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing.*

[95] Zechun Liu, Barlas O?uz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2024. LLM-QAT: Data-free quantization aware training for large language models. In *Findings of the Association for Computational Linguistics, Bangkok, Thailand and Virtual Meeting, August 11–16, 2024.*

[96] Zechun Liu, Changsheng Zhao, Forrest N. Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, Liangzhen Lai, and Vikas Chandra. 2024. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. In *Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria, July 21–27, 2024.*

[97] Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. NeuroLogic decoding: (Un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, June 6–11, 2021.*

[98] Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-Pruner: On the structural pruning of large language models. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, New Orleans, LA, USA, December 10–16, 2023.*

[99] Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2023. Teaching small language models to reason. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Toronto, Canada, July 9–14, 2023.*

[100] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. In *Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, December 6–14, 2021, Virtual.*

[101] Macedo Maia, Siegfried Handschuh, Andre Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. WWW'18 open challenge: Financial opinion mining and question answering. In *Companion of the The Web Conference 2018 on The Web Conference 2018, Lyon , France, April 23–27, 2018.*

[102] Pratyush Maini, Skyler Seto, Richard He Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. 2024. Rephrasing the web: A recipe for compute and data-efficient language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Bangkok, Thailand, August 11–16, 2024.*

[103] Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alexandru Damian, Jason D. Lee, Danqi Chen, and Sanjeev Arora. 2023. Fine-tuning language models with just forward passes. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, New Orleans, LA, USA, December 10–16, 2023.*

[104] Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation* 24 (1989), 109–165.

[105] Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one?. In *Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, December 8–14, 2019, Vancouver, BC, Canada.*

[106] Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

[107] Sharan Narang, Gregory Frederick Diamos, Shubho Sengupta, and Erich Elsen. 2017. Exploring sparsity in recurrent neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26.* 110. https://arxiv.org/abs/1704.05119

[108] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.*

[109] Stephen Obadinma, Hongyu Guo, and Xiao-Dan Zhu. 2023. Effectiveness of data augmentation for parameter efficient tuning with limited data. In *Proceedings of the 8th Workshop on Representation Learning for NLP, Toronto, Canada, July 13, 2023.*

[110] Matteo Pagliardini, Daniele Paliotta, Martin Jaggi, and François Fleuret. 2023. Fast attention over long sequences with dynamic sparse flash attention. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, New Orleans, LA, USA, December 10–16, 2023.*

[111] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning, 7–8 April 2022, Virtual Event.*

[112] Eunhyeok Park, Dongyoung Kim, and Sungjoo Yoo. 2018. Energy-efficient neural network accelerator based on outlier-aware low-precision computation. In *Proceedings of the 45th ACM/IEEE Annual International Symposium on Computer Architecture, Los Angeles, CA, USA, June 1–6, 2018.* 688–698.

[113] Gunho Park, Baeseong Park, Minsub Kim, Sungjae Lee, Jeonghoon Kim, Beomseok Kwon, Se Jung Kwon, Byeongwook Kim, Youngjoo Lee, and Dongsoo Lee. 2024. LUT-GEMM: Quantized matrix multiplication based on LUTs for efficient inference in large-scale generative language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[114] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. YaRN: Efficient context window extension of large language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[115] Jonas Pfeiffer, Andreas Ruckle, Clifton A. Poth, Aishwarya Kamath, Ivan Vulic, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16–20, 2020.*

[116] Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, June 6–11, 2021.*

[117] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019), 9.

[118] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, USA, November 1–4, 2016.*

[119] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. XNOR-Net: ImageNet classification using binary convolutional neural networks. In *Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, October 11–14, 2016.*

[120] Rajarshi Saha, Varun Srivastava, and Mert Pilanci. 2023. Matrix compression via randomized low rank and low precision factorization. In *Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, New Orleans, LA, USA, December 10–16, 2023.*

[121] Charbel Sakr and Brucek Khailany. 2024. ESPACE: Dimensionality reduction of activations for model compression. In *Proceedings of the Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems, Vancouver, BC, Canada, December 10–15, 2024.*

[122] Victor Sanh, Thomas Wolf, and Alexander M. Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. In *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, December 6–12, 2020, virtual.*

[123] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ili'c, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, François Yvon, Matthias Galle, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, and Pawan Sasanka Ammanamanchi. 2022. BLOOM: A 176B-parameter open-access multilingual language model. arXiv:2211.05100. Retrieved from https://arxiv.org/abs/2211.05100

[124] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. https://arxiv.org/abs/1707.06347

[125] Zihan Shao, Peng Wang, Qiang Zhu, Rui Xu, Jiawei Song, Ming Zhang, Yikang Li, Yiming Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. https://arxiv.org/abs/2402.03300

[126] Ying Sheng, Shiyi Cao, Dacheng Li, Coleman Hooper, Nicholas Lee, Shuo Yang, Christopher Chou, Banghua Zhu, Lianmin Zheng, Kurt Keutzer, Joseph Gonzalez, and Ion Stoica. 2024. S-LoRA: Serving thousands of concurrent LoRA adapters. In *Proceedings of the 5th MLSys Conference, Santa Clara, CA, USA, 2024.*

[127] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Online, November 16–20, 2020.*

[128] Sidak Pal Singh and Dan Alistarh. 2020. WoodFisher: Efficient second-order approximations for model compression. In *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, December 6–12, 2020, virtual.*

[129] Srinjoy Sridharan, John R. Stevens, Karthik Roy, and Anand Raghunathan. 2023. X-Former: In-memory acceleration of transformers. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 31 (2023), 1223–1233.

[130] Haipeng Sun, Rui Wang, Kehai Chen, Masao Utiyama, Eiichiro Sumita, and Tiejun Zhao. 2020. Knowledge distillation for multilingual unsupervised neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020.*

[131] Hanshi Sun, Zhuoming Chen, Xinyu Yang, Yuandong Tian, and Beidi Chen. 2024. TriForce: Lossless acceleration of long sequence generation with hierarchical speculative decoding. In *Proceedings of the 1st Conference on Language Modeling, Philadelphia, PA, USA, October 7–9, 2024.*

[132] Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[133] Yi-Lin Sung, Varun Nair, and Colin Raffel. 2021. Training neural networks with fixed sparse masks. In *Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, December 6–14, 2021, Virtual.*

[134] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori Hashimoto. 2023. Stanford alpaca: An instruction-following LLaMA model. GitHub. https://github.com/tatsu-lab/stanford_alpaca. (Accessed on October 16, 2024).

[135] Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Niko-lay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen,

Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melissa Hall Melanie Kambadur, Sharan Narang, Aur'elien Rodriguez, Robert Stojnic, Sergey Edunov and Thomas Scialom. 2023. Llama 2: Open foundation and fine-Tuned chat models. https://arxiv.org/abs/2307.09288

[136] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Roziere, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: open and efficient foundation language models. https://arxiv.org/abs/2302.13971

[137] Zhongwei Wan, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang. 2024. Efficient large language models: A survey. *Transactions on Machine Learning Research* (2024). 1–67.

[138] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the Workshop: Analyzing and Interpreting Neural Networks for NLP, Brussels, Belgium, 2018.*

[139] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Proceedings of the Advances in Neural Information Processing Systems 32.*

[140] Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. BitNet: Scaling 1-bit transformers for large language models. https://arxiv.org/abs/2310.11453

[141] Yaqing Wang, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, and Jianfeng Gao. 2022. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, December 7–11, 2022.*

[142] Yiming Wang, Yu Lin, Xiaodong Zeng, and Guannan Zhang. 2023. MultiLoRA: Democratizing LoRA for better multi-task learning. https://arxiv.org/abs/2311.11501

[143] Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Online, November 16–20, 2020.*

[144] Ziqi Wang, Yuexin Wu, Frederick Liu, Daogao Liu, Le Hou, Hongkun Yu, Jing Li, and Heng Ji. 2023. Augmentation with projection: Towards an effective and efficient data augmentation paradigm for distillation. In *Proceedings of the 11th International Conference on Learning Representations, Kigali, Rwanda, May 1–5, 2023.*

[145] Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Schmidt Feris, Huan Sun, and Yoon Kim. 2023. Multitask prompt tuning enables parameter-efficient transfer learning. In *Proceedings of the 11th International Conference on Learning Representations, Kigali, Rwanda, May 1–5, 2023.*

[146] Peter West, Chandrasekhar Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: From general language models to commonsense models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, WA, United States, July 10–15, 2022.*

[147] Herbert Woisetschlager, Alexander Isenko, Shiqiang Wang, Ruben Mayer, and Hans-Arno Jacobsen. 2024. A survey on efficient federated learning methods for foundation model training. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence, Jeju, South Korea, August 3–9, 2024.*

[148] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. 2022. Communication-efficient federated learning via knowledge distillation. *Nature Communications* (2022).

[149] Chaoyi Wu, Weixiong Lin, Xiaoman Zhang, Ya Zhang, Weidi Xie, and Yanfeng Wang. 2024. PMC-LLaMA: Toward building open-source language models for medicine. *Journal of the American Medical Informatics Association* 31, 9 (2024), 1833–1843.

[150] Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, May 22–27, 2022.*

[151] Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024.*

[152] Guangxuan Xiao, Ji Lin, Mickael Seznec, Julien Demouth, and Song Han. 2023. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the International Conference on Machine Learning, 23–29 July 2023, Honolulu, Hawaii, USA.*

[153] Yao Xiao, Lu Xu, Jiaxi Li, Wei Lu, and Xiaoli Li. 2023. Decomposed prompt tuning via low-rank reparameterization. In *Findings of the Association for Computational Linguistics, Singapore, December 6–10, 2023*.

[154] Mengwei Xu, Dongqi Cai, Yaozong Wu, Xiang Li, and Shangguang Wang. 2024. FwdLLM: Efficient federated fine-tuning of large language models with perturbed inferences. In *Proceedings of the 2024 USENIX Annual Technical Conference, Santa Clara, CA, USA, July 10–12, 2024*.

[155] Mengwei Xu, Wangsong Yin, Dongqi Cai, Rongjie Yi, Daliang Xu, Qipeng Wang, Bingyang Wu, Yihao Zhao, Chen Yang, Shihe Wang, Qiyang Zhang, Zhenyan Lu, Li Zhang, Shangguang Wang, Yuanchun Li, Yunxin Liu, Xin Jin, and Xuanzhe Liu. 2024. A survey of resource-efficient LLM and multimodal foundation models. https://arxiv.org/abs/2401.08092

[156] Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhensu Chen, Xiaopeng Zhang, and Qi Tian. 2024. QA-LoRA: Quantization-aware low-rank adaptation of large language models. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, May 7–11, 2024*.

[157] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan and Zekun Wang. 2025. Qwen2.5 Technical Report. ArXiv, https://arxiv.org/abs/2412.15115

[158] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, New Orleans, LA, USA, November 28–December 9, 2022*.

[159] Zhewei Yao, Xiaoxia Wu, Cheng Li, Stephen Youn, and Yuxiong He. 2024. Exploring post-training quantization in LLMs from comprehensive study to low rank compensation. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*.

[160] Lu Yin, You Wu, Zhenyu (Allen) Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Mykola Pechenizkiy, Yi Liang, Zhangyang Wang, and Shiwei Liu. 2024. Outlier weighed layerwise sparsity (OWL): A missing secret sauce for pruning LLMs to high sparsity. In *Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria, July 21–27, 2024*.

[161] Zhihang Yuan, Lin Niu, Jia-Wen Liu, Wenyu Liu, Xinggang Wang, Yuzhang Shang, Guangyu Sun, Qiang Wu, Jiaxiang Wu, and Bingzhe Wu. 2023. RPTQ: Reorder-based Post-training quantization for large language models. https://arxiv.org/abs/2304.01089

[162] Ali Hadi Zadeh and Andreas Moshovos. 2020. GOBO: Quantizing attention-based NLP models for low latency and energy efficient inference. In *Proceedings of the 53rd Annual IEEE/ACM International Symposium on Microarchitecture, Athens, Greece, October 17–21, 2020*. 811–824.

[163] Ofir Zafrir, Ariel Larey, Guy Boudoukh, Haihao Shen, and Moshe Wasserblat. 2021. Prune once for all: Sparse pre-trained language models. In *Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, December 6–14, 2021, Workshop*.

[164] Hongchuan Zeng, Hongshen Xu, Lu Chen, and Kai Yu. 2024. Multilingual brain surgeon: Large language models can be compressed leaving no language behind. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, 20–25 May, 2024, Torino, Italy*.

[165] Hongyi Zhang, Moustapha Cisse, Yann Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical Risk Minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018*. https://arxiv.org/abs/1710.09412

[166] Mingyang Zhang, Hao Chen, Chunhua Shen, Zhenyi Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. 2024. LoRAPrune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics, Bangkok, Thailand and Virtual Meeting, August 11–16, 2024*.

[167] Nan Zhang, Yanchi Liu, Xujiang Zhao, Wei Cheng, Runxue Bao, Rui Zhang, Prasenjit Mitra, and Haifeng Chen. 2024. Pruning as a domain-specific LLM extractor. In *Findings of the Association for Computational Linguistics, Mexico City, Mexico, June 16–21, 2024*.

[168] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *Proceedings of the 11th International Conference on Learning Representations, Kigali, Rwanda, May 1–5, 2023*.

[169] Renrui Zhang, Jiaming Han, Chris Liu, Aojun Zhou, Pan Lu, Yu Jiao Qiao, Hongsheng Li, and Peng Gao. 2024. LLaMA-Adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *Proceedings of the 12th International Conference on Learning Representations, Vienna, Austria, 2024*.

[170] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang and Luke Zettlemoyer. 2022. OPT: Open Pre-trained transformer language models. https://arxiv.org/abs/2205.01068

[171] Lucia Zheng, Neel Guha, Brandon R. Anderson, Peter Henderson, and Daniel E. Ho. 2021. When does pretraining help?: Assessing self-supervised learning for law and the CaseHOLD dataset of 53, 000+ legal holdings. In *Proceedings of the 18th International Conference for Artificial Intelligence and Law*.

[172] Wei Zhu, Aaron Xuxiang Tian, Congrui Yin, Yuan Ni, Xiaoling Wang, and Guo Tong Xie. 2024. IAPT: Instance-aware prompt tuning for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[173] Xunyu Zhu, Jian Li, Yong Liu, Can Ma and Weiping Wang. 2023. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics* 12 (2023), 1556–1577. https://doi.org/10.1162/tacl_a_00704

Received 22 March 2024; revised 28 February 2025; accepted 1 April 2025