Efficient Large Language Models: A Survey

Zhongwei Wan[*] *wan.512@osu.edu*

Xin Wang[*] *wang.15980@osu.edu*

Che Liu[†] *che.liu21@imperial.ac.uk*

Samiul Alam[*] *alam.140@osu.edu*

Yu Zheng[‡] *zhengy30@msu.edu*

Jiachen Liu[§] *amberljc@umich.edu*

Zhongnan Qu[¶] *znqu@amazon.com*

Shen Yan[‖] *shenyan@google.com*

Yi Zhu[††] *yi@boson.ai*

Quanlu Zhang[**] *quzha@microsoft.com*

Mosharaf Chowdhury[§] *mosharaf@umich.edu*

Mi Zhang[*] *mizhang.1@osu.edu*

[*] *The Ohio State University* [†] *Imperial College London* [‡] *Michigan State University* [§] *University of Michigan* [¶] *Amazon AWS AI* [‖] *Google Research* [**] *Microsoft Research Asia* [††] *Boson AI*

Reviewed on OpenReview: https://openreview.net/forum?id=bsCCJHbO8A

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in important tasks such as natural language understanding and language generation, and thus have the potential to make a substantial impact on our society. Such capabilities, however, come with the considerable resources they demand, highlighting the strong need to develop effective techniques for addressing their efficiency challenges. In this survey, we provide a systematic and comprehensive review of efficient LLMs research. We organize the literature in a taxonomy consisting of three main categories, covering distinct yet interconnected efficient LLMs topics from model-centric, data-centric, and framework-centric perspective, respectively. We have also created a GitHub repository where we organize the papers featured in this survey at https://github.com/AIoT-MLSys-Lab/Efficient-LLMs-Survey. We will actively maintain the repository and incorporate new research as it emerges. We hope our survey can serve as a valuable resource to help researchers and practitioners gain a systematic understanding of efficient LLMs research and inspire them to contribute to this important and exciting field.

1 Introduction

Large Language Models (LLMs) are a type of advanced AI models designed to understand and generate human languages. Recently, we have witnessed a surge in LLMs including those developed by Open AI (GPT-4 (Achiam et al., 2023) and GPT-3 (Brown et al., 2020)), Meta (LLaMA-3 (Meta, 2024), LLaMA-2 (Touvron et al., 2023b), LLaMA-1 (Touvron et al., 2023a)), and Google (Gemini (Team & Google, 2023), PaLM-2 (Anil et al., 2023), PaLM (Chowdhery et al., 2022), GLaM (Du et al., 2022)) as well as many other models such as BLOOM (Scao et al., 2023), PanGu-$\sum$ (Ren et al., 2023b), and GLM (Zeng et al., 2023). These models have demonstrated remarkable performance across a variety of tasks such as natural language understanding (NLU), language generation, complex reasoning (Yang et al., 2024), and domain-specific tasks related to biomedicine (He et al., 2023; Wan et al., 2023; 2022), law (Eliot, 2021) and code generation (Wei et al., 2022b; Chen et al., 2021b). Such performance breakthroughs can be attributed to their massive scales in model sizes and volumes of training data, as they contain billions or even trillions of parameters while being trained on a gigantic amount of data from diverse sources.

Although LLMs are leading the next wave of AI revolution, their remarkable capabilities come at substantial resource demands (Achiam et al., 2023; Du et al., 2022; Chowdhery et al., 2022; Ren et al., 2023b). Figure 1 illustrates the relationship between model performance and model training time in terms of GPU hours for LLaMA series, where the size of each circle is proportional to the number of model parameters. As shown, although larger models are able to achieve better performance, the amounts of GPU hours used for training them grow exponentially as model sizes scale up. In addition to training, inference also contributes quite significantly to the operational cost of LLMs. Figure 2 depicts the relationship between model performance and inference throughput. Similarly, scaling up the model size enables better performance but comes at the cost of lower inference throughput (higher inference latency), presenting challenges for these models in expanding their reach to a broader customer base and diverse applications in a cost-effective way.

The high resource demands of LLMs highlight the strong need to develop techniques to enhance the efficiency of LLMs. As shown in Figure 2, compared to LLaMA-1-33B, Mistral-7B (Jiang et al., 2023a), which uses grouped-query attention and sliding window attention to speed up inference, achieves comparable performance and much higher throughput. This superiority highlights the feasibility and significance of designing efficiency techniques for LLMs.

The overarching goal of this survey is to provide a holistic view of the technological advances in efficient LLMs. As illustrated in Figure 3, we organize the literature in a taxonomy consisting of three main categories, covering efficient LLMs topics from **model-centric**, **data-centric**, and **framework-centric** perspective, respectively. These three categories cover distinct yet interconnected research topics, collectively providing a systematic and comprehensive review of efficient LLMs research. Specifically,

- **Model-Centric Methods:** Model-centric methods focus on both **algorithm-level** and **system-level** efficient techniques where the model itself is the focal point. With billions or even trillions of parameters, LLMs exhibit distinct characteristics (Wei et al., 2022a) compared to smaller-scale models, necessitating the development of new techniques to enhance their efficiency. In §2, we survey efficient techniques that cover research directions related to model compression, efficient pre-training, efficient fine-tuning, efficient inference, and efficient architecture design.

- **Data-Centric Methods:** In the realm of LLMs, the importance of data is as crucial as that of the model itself. Data-centric methods focus on the role of the quality and structure of data in enhancing the efficiency of LLMs. In §3, we survey efficient techniques that cover research directions related to data selection and prompt engineering.

- **LLM Frameworks**: The advent of LLMs necessitates the development of specialized frameworks to efficiently handle their training, fine-tuning, inference, and serving. While mainstream AI frameworks such as TensorFlow and PyTorch provide the foundations, they lack built-in support for specific optimizations and features crucial for LLMs. In §4, we survey existing frameworks specifically designed for efficient LLMs, covering their unique features, underlying libraries, and specializations.

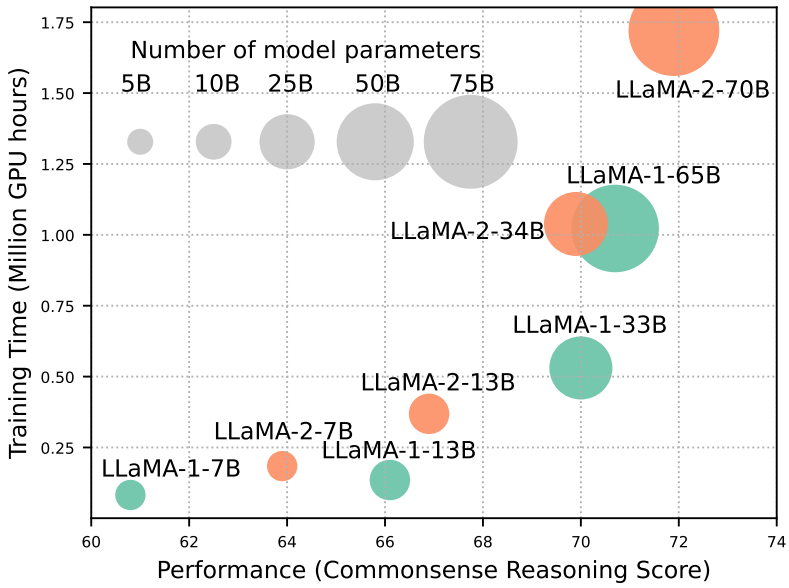

Figure 1: Illustration of model performance and model training time in GPU hours of LLaMA models at different scales. The reported performance is the average score of several commonsense reasoning benchmarks. The training time is based on Nvidia A100 80GB GPU. The size of each circle corresponds to the number of model parameters. The original data can be found in Touvron et al. (2023a;b).

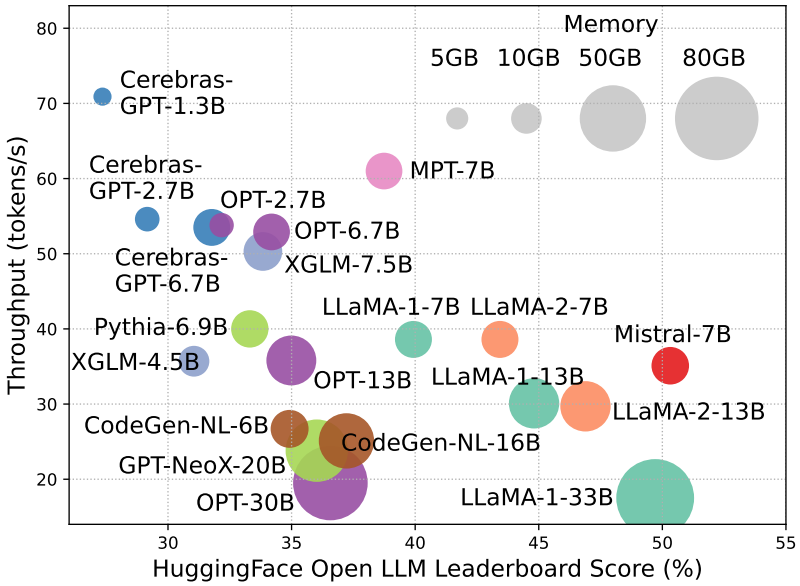

Figure 2: Performance score *vs.* inference throughput for various LLMs. The throughputs are measured on Nvidia A100 80GB GPU with 16-bit floating point quantization. The size of each circle corresponds to the memory footprint (in Gigabytes) of each model when running with a batch size of 1, prompt size of 256, and generating 1000 tokens. The original data can be found in Ilyas Moutawwakil (2023).

In addition to the survey, we have established a **GitHub repository** where we compile the papers featured in this survey at https://github.com/AIoT-MLSys-Lab/Efficient-LLMs-Survey. We will actively maintain it and incorporate new research as it emerges.

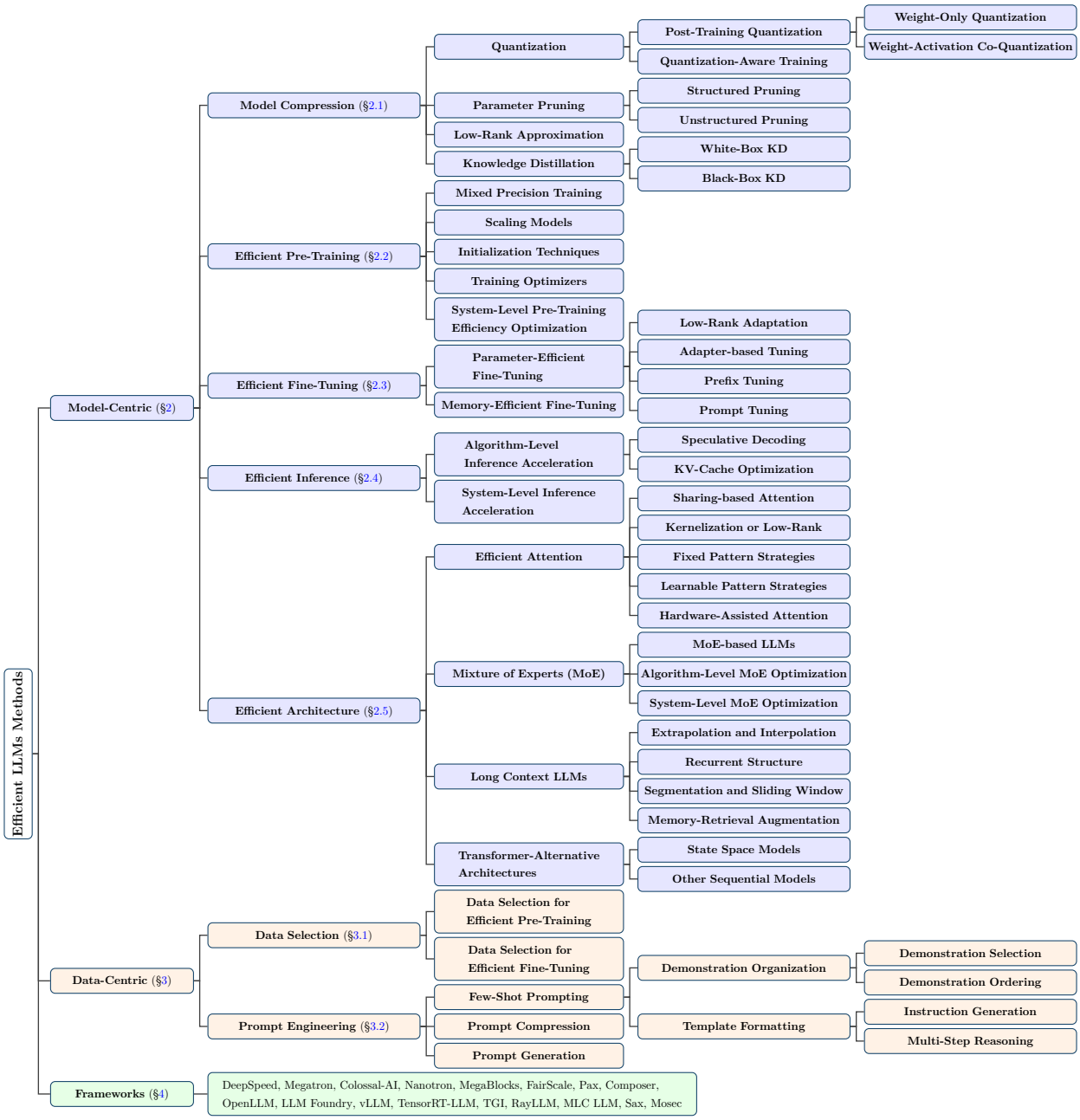

Figure 3: Taxonomy of efficient large language models (LLMs) literature.

Although there are a few surveys on LLMs (Zhao et al., 2023a; Chang et al., 2024; Wang et al., 2023i; Kaddour et al., 2023), this survey provides a focused review and discussion on the literature related to the efficiency aspect of LLMs. There are also surveys on efficient Transformers (Tay et al., 2022) and their training methods (Zhuang et al., 2023). In contrast, this survey specifically focuses on efficiency techniques designed for models of more than billions of parameters. We hope this survey together with the GitHub repository can help researchers and practitioners navigate through the literature and serve as a catalyst for inspiring further research on efficient LLMs.

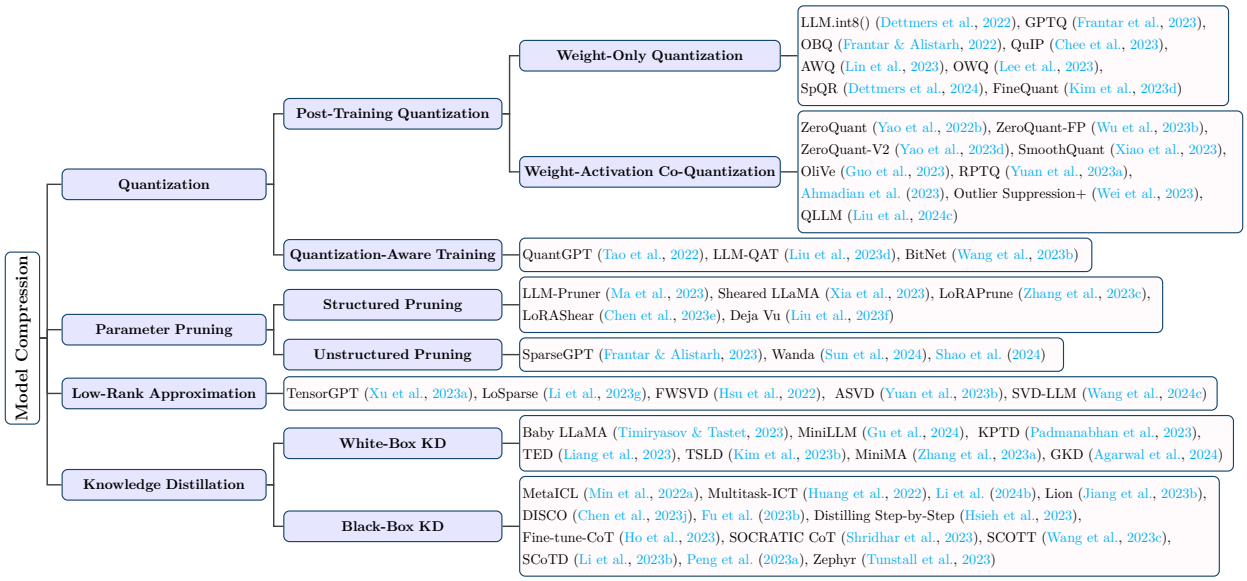

Figure 4: Summary of model compression techniques for LLMs.

2 Model-Centric Methods

2.1 Model Compression

Model compression enhances efficiency by reducing the sizes and the amount of arithmetic operations of LLMs. Unlike conventional model compression techniques, most LLM compression approaches are designed under the post-training setting to avoid resource-intensive retraining. As summarized in Figure 4, model compression techniques for LLMs can be grouped into four categories: quantization, parameter pruning, low-rank approximation, and knowledge distillation. These four categories are orthogonal to each other, and compress LLMs from different perspectives.

2.1.1 Quantization

Quantization compresses LLMs by converting model weights and/or activations of high-precision data types $\mathbf{X}^{\mathrm{H}}$ such as 32-bit floating point into low-precision data types $\mathbf{X}^{\mathrm{L}}$ such as 8-bit integer (Dettmers et al., 2023) as:

$$\mathbf{X}^{\mathrm{L}} = \mathrm{Round}\left(\frac{\mathrm{absmax}\left(\mathbf{X}^{\mathrm{L}}\right)}{\mathrm{absmax}\left(\mathbf{X}^{\mathrm{H}}\right)}\mathbf{X}^{\mathrm{H}}\right) = \mathrm{Round}\left(\mathcal{K}\cdot\mathbf{X}^{\mathrm{H}}\right), \tag{1}$$

where Round denotes mapping a floating point number into an approximate integer; absmax denotes the absolute maximum of the input elements; and $\mathcal{K}$ denotes the quantization constant. Quantization techniques for LLMs can be classified into post-training quantization (PTQ) and quantization-aware training (QAT). Compared to other LLM compression methods such as parameter pruning and low-rank approximation, quantization methods have been shown to achieve superior compression-accuracy trade-offs (Li et al., 2024c).

Post-Training Quantization (PTQ). PTQ quantizes LLMs after the model has been trained. To compensate for the accuracy drop, PTQ uses a small calibration dataset to update the quantized weights and/or activations. PTQ in general can be grouped into two categories: weight-only quantization, and weight-activation co-quantization.

- ***Weight-Only Quantization*** focuses on quantizing model weights only. For example, Dettmers et al. (2022) introduce the first multi-billion-scale 8-bit integers (or INT8) weight quantization method named LLM.int8() that significantly reduces memory usage during inference while being able

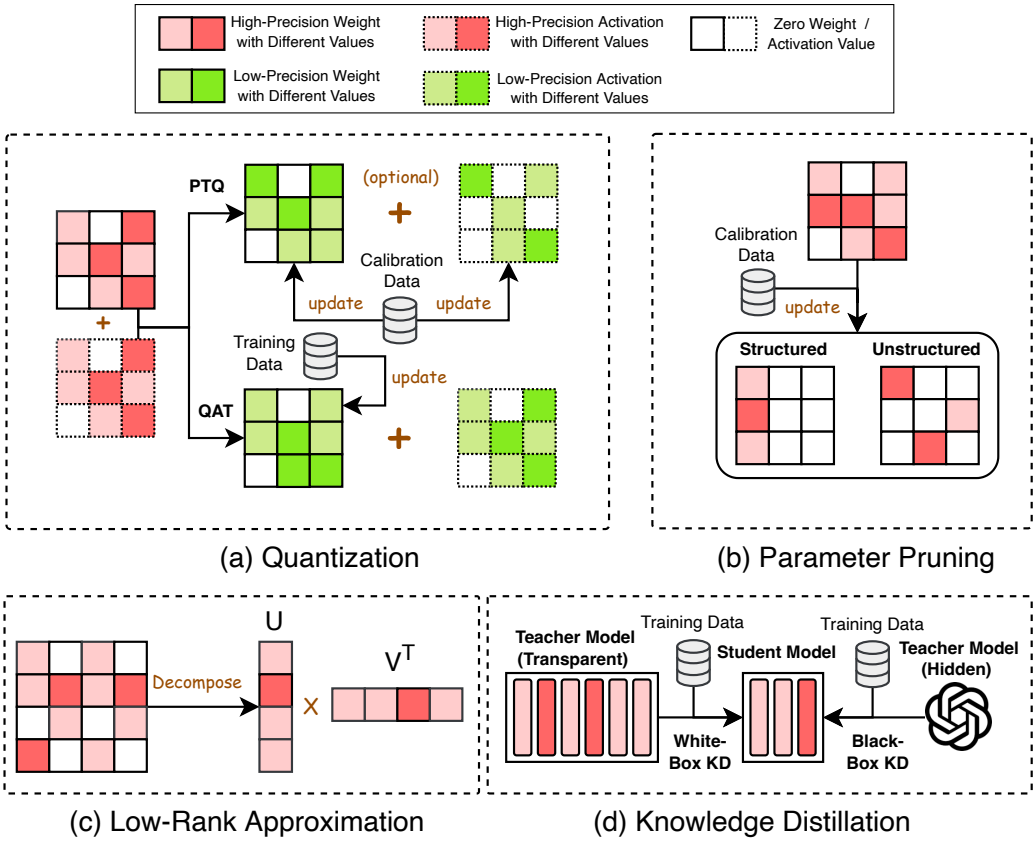

Figure 5: Illustrations of model compression techniques for LLMs.

to maintain the performance of the full-precision model. Frantar et al. (2023) push one step further and propose GPTQ, a post-training weight quantization method that compresses LLM weights to 3 or 4 bits instead of 8 bits. GPTQ employs layer-wise quantization with Optimal Brain Quantization (OBQ) (Frantar & Alistarh, 2022) to update weights with inverse Hessian information. This technique enables quantizing GPT models with 175 billion parameters in roughly four GPU hours with minimal accuracy drop compared to the original model. Furthermore, driven by the insights that quantization can be more effective when model weights and proxy Hessian matrices are incoherent, Chee et al. (2023) propose QuIP, a post-training quantization method that applies incoherence processing to quantize LLMs to 2 bits per weight. As another line of research under weight-only quantization, Lin et al. (2023) observe that there exists a small subset of model weights, characterized by larger activation magnitudes, known as salient weights, play a crucial role in determining the quantization loss. Based on this observation, they propose an approach named activation-aware weight quantization (AWQ) to quantize LLMs while preserving the salient weights in high precision, demonstrating superior performance over GPTQ. Similarly, Lee et al. (2023) observe that activation outliers amplify weight quantization loss. They propose outlier-aware weight quantization (OWQ) to identify those vulnerable weights with activation outliers and allocate high-precision to them. Dettmers et al. (2024) introduce Sparse-Quantized Representation (SpQR) to separate outlier weights that are prone to large quantization errors. These outlier weights are preserved at higher precision, while the remaining weights are compressed to 3-4 bits. Additionally, they introduce a decoding scheme tailored for the SpQR format that enhances the efficiency of inference on a token-by-token basis. Lastly, Kim et al. (2023d) aim to address the issue of outliers that distort the distribution of quantized weights, and propose FineQuant that employs an empirically crafted, heuristic-based approach to allocate varying levels of granularity to different weight matrices.

- ***Weight-Activation Co-Quantization*** differs from weight-only quantization in the sense that it quantizes both model weights and activations. For instance, Yao et al. (2022b) propose ZeroQuant, which combines group-wise quantization for model weights and token-wise quantization for activations. However, ZeroQuant falls short in maintaining accuracy for models with more than 175 billion parameters. To address this issue, Yao et al. (2023d) and Wu et al. (2023b) propose ZeroQuant-FP and ZeroQuant-V2 respectively, both of which utilize low-rank matrices to recover the accuracy drop. A key challenge of weight-activation co-quantization is that due to the existence of outliers, activations are more difficult to quantize than model weights (Bondarenko et al., 2021). To address this challenge, Xiao et al. (2023) propose SmoothQuant which introduces a per-channel scaling transformation that migrates the quantization difficulty from activations to weights to achieve lossless quantization of weights and activations to 8 bits for LLMs up to 530 billion parameters. Guo et al. (2023) pinpoint that outliers are critical in weight-activation co-quantization but their nearby normal values are not. Given that, they propose OliVe, which prunes normal values adjacent to the outliers so that the outliers can be encoded with higher precision. Yuan et al. (2023a) identify the challenge of quantizing activations when different channels have disparate ranges. They propose RPTQ, which groups channels in activations that have similar value ranges and applies uniform quantization parameters to the values in each group. Ahmadian et al. (2023) demonstrate that it is possible to suppress large activation outliers at scales as large as 52B. Given the right optimization choices during pre-training, they can quantize models ranging in size from 410M to 52B with minimal accuracy degradation. Wei et al. (2023) observe that the activation outliers in LLMs are asymmetric and tend to cluster in particular channels. Based on this observation, they propose Outlier Suppression+, which introduces operations that shift and scale channels individually to neutralize asymmetric outliers. Lastly, Liu et al. (2024c) propose QLLM, an adaptive channel reassembly method that tackles activation outliers and utilizes calibration data to offset the information loss incurred from quantization. Experimental result shows that QLLM achieves better compression performance than SmoothQuant and Outlier Suppression+ on LLaMA model family.

Quantization-Aware Training (QAT). Different from PTQ, QAT quantizes LLMs during the training process, allowing LLMs to learn quantization-friendly representations. Since QAT requires training using the complete training dataset, it is much more expensive and time consuming than PTQ. Tao et al. (2022) propose QuantGPT, which combines contrastive distillation from a full-precision teacher model and logit distillation to a quantized student model during autoregressive pretraining. QuantGPT achieves 14.4× and 13.4× compression rates on GPT-2 and BART with comparable performance with the full-precision models. LLM-QAT (Liu et al., 2023d) uses data generated by LLMs itself to distill knowledge with the objective of quantizing a student model. Specifically, LLM-QAT retains the original output distribution and is capable of quantizing a model irrespective of its initial training data. Besides quantizing weights and activations, LLM-QAT also quantizes the key-value cache, a crucial step for enhancing throughput and accommodating long sequence dependencies in LLMs. Experimental results show that LLM-QAT achieves better performance over training-free methods especially in low-bit settings. Lastly, BitNet (Wang et al., 2023b) pioneers QAT for 1-bit LLMs. It proposes to use low-precision binary weights and quantized activations while keeping optimizer states and gradients high-precision during training. Experimental results show that compared to FP16 Transformer baselines, BitNet is able to achieve competitive performance while substantially reducing memory footprint and energy consumption.

2.1.2 Parameter Pruning

Parameter pruning compresses LLMs by removing redundant or less important model weights. Parameter pruning methods for LLMs can be categorized into structured pruning and unstructured pruning.

Structured Pruning. Structured pruning focuses on pruning structured patterns such as groups of consecutive parameters or hierarchical structures such as rows, columns, or sub-blocks of the LLM weight matrices. For instance, LLM-Pruner (Ma et al., 2023) introduces a task-agnostic structured pruning strategy that selectively eliminates non-essential interconnected structures using gradient information. LLM-Pruner utilizes a small amount of data to obtain the weight, parameter, and group importance of the coupled structure for LLaMA (Touvron et al., 2023a), and uses LoRA (Hu et al., 2022) to recover accuracy after pruning,

showing competitive zero-shot performance. Sheared LLaMA (Xia et al., 2023), on the other hand, proposes two techniques to improve the performance of LLM-Pruner. The first technique, named targeted structured pruning, prunes a larger model to a designated target shape by eliminating layers, heads, intermediate and hidden dimensions in an end-to-end manner. The second technique, named dynamic batch loading, dynamically configures the composition of sampled data in each training batch based on losses in various domains. Through these two techniques, Sheared LLaMA is able to prune LLaMA2-7B down to 1.3B parameters, achieving superior compression ratio compared to LLM-Pruner. LoRAPrune (Zhang et al., 2023c) introduces a LoRA-based pruning criterion using LoRA's weights and gradients for importance estimation. By employing an iterative structure pruning process to eliminate excess channels and heads, LoRAPrune achieves better efficiency over LLM-Pruner at 50% compression rate. Lastly, given the input, Deja Vu (Liu et al., 2023f) predicts a small set of attention heads and MLP parameters, referred to as contextual sparsity, yields approximately the same output as the dense model. By exploiting such contextual sparsity, Deja Vu is able to achieve much lower latency compared to FasterTransformer without accuracy drop.

Unstructured Pruning. Unstructured pruning, on the other hand, focuses on pruning model weights individually. Compared to structured pruning, unstructured pruning has much more pruning flexibility and thus enjoys a lower accuracy drop. However, unstructured pruning incurs irregular sparsification, which in general makes the resulting pruned models difficult to be deployed on hardware except specific types of hardware such as Nvidia Ampere GPUs (Busato & Pool, 2020). For instance, Frantar & Alistarh (2023) introduce SparseGPT, an one-shot LLM unstructured pruning approach that does not require retraining. SparseGPT formulates pruning as a sparse regression problem and solves it by utilizing an approximate solver based on the inversion of the Hessian matrix. In doing so, SparseGPT reaches about 60% unstructured sparsity on models such as OPT-135B while experiencing only a slight performance drop. Sun et al. (2024) propose Wanda, which prunes weights based on the product values of weight magnitudes and their respective input activations. Compared to SparseGPT, Wanda neither relies on second-order information nor necessitates weight update, and is able to achieve competitive performance. Shao et al. (2024) improve the performance of SparseGPT in another way. Specifically, instead of performing the unstructured pruning with a unified ratio for every layer, they propose to utilize Hessian sensitivity-aware mixed sparsity pruning to achieve a minimum of 50% sparsity in LLMs without retraining. This method adaptively assigns sparsity based on sensitivity to minimize the error induced by pruning while preserving the overall level of sparsity.

2.1.3 Low-Rank Approximation

Low-rank approximation compresses LLMs by approximating the LLM weight matrix $\mathbf{W}^{m\times n}$ with smaller low-rank matrices $\mathbf{U}$ and $\mathbf{V}$ such that $\mathbf{W} \approx \mathbf{U}\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{m\times r}$, $\mathbf{V} \in \mathbb{R}^{n\times r}$, and r is typically much smaller than m, n. In doing so, low-rank approximation reduces the number of parameters and enhances efficiency. For example, Xu et al. (2023a) introduce TensorGPT which compresses the embedding layers of LLMs using Tensor-Train Decomposition (TTD). It transforms and breaks down each token embedding and creates an efficient embedding format named Matrix Product State (MPS) that can be efficiently computed in a distributed manner. LoSparse (Li et al., 2023g) improves the performance of TensorGPT by compressing the coherent and expressive components within neurons through low-rank approximation while eliminating the incoherent and non-expressive elements through pruning. As another line of research, FWSVD (Hsu et al., 2022) compresses the weight matrix of an LLM instead of the token embedding matrix via low-rank approximation. Specifically, instead of using vanilla singular value decomposition (SVD), FWSVD proposes a weighted SVD approach which uses Fisher information to weigh the importance of the weights for compression. While FWSVD demonstrates competitive compression results under low compression ratios, it requires calculating the gradients based on the training dataset of the target task to estimate the importance scores, which is task-specific and demands significant computation resources. In contrast, ASVD (Yuan et al., 2023b) proposes a training-free SVD-based approach. It scales the weight matrix based on the activation distribution that enhances the decomposition accuracy and efficiency for model compression. However, neither FWSVD nor ASVD directly correlate singular values with compression loss. Consequently, truncating the smaller singular values might result in increased compression loss. SVD-LLM (Wang et al., 2024c) addresses this drawback by incorporating a truncation-aware data whitening strategy that establishes a direct mapping between singular values and compression loss. Experimental results demonstrate the superiority of SVD-LLM over FWSVD and ASVD in terms of compression performance and speed.

2.1.4 Knowledge Distillation

Knowledge Distillation (KD) compresses LLMs by transferring knowledge from a large teacher LLM to a smaller student LLM. Though effective, compared to other LLM compression methods, knowledge distillation methods incur a resource-demanding distillation process. In general, KD for LLMs can be categorized into white-box KD methods and black-box KD methods.

White-Box Knowledge Distillation. White-box KD refers to KD techniques where the parameters or logits of the teacher LLM are used in the distillation process (Gou et al., 2021). For example, as a pioneering effort in this direction, Baby LLaMA (Timiryasov & Tastet, 2023) trains an ensemble of GPT-2 and a collection of smaller LLaMA models using the BabyLM dataset of 10M words. This ensemble is then distilled into a compact LLaMA model with 58 million parameters, which outperforms both its original teacher models as well as a comparable model that was trained without the use of distillation. Gu et al. (2024) observe that conventional KD objectives, such as Kullback-Leibler divergence (KLD), may not be suited for open text generation tasks due to their complex output spaces compared to classification tasks. To address this issue, they propose MiniLLM which minimizes reverse KLD using the gradient of the objective function through policy gradient techniques (Sutton et al., 1999). Experimental results show that MiniLLM achieves better accuracy than conventional KD or directly fine-tuning student models. KPTD (Padmanabhan et al., 2023) demonstrates that white-box KD can transfer and disseminate knowledge from entity definitions into the parameters of a pre-trained language model. Specifically, KPTD creates a transfer set by prompting the language model to generate text based on the definition of the entity. The model parameters are then updated to align the distribution of the student model with that of the teacher model. TED (Liang et al., 2023) introduces a technique for layer-specific task distillation. It uses specially designed filters to align the internal states of both student and teacher models in each layer. These filters extract relevant knowledge from the internal states that is beneficial for the specific task. TED shows considerable and steady gains in performance on both continual pre-training and fine-tuning. TSLD (Kim et al., 2023b) leverages token-level distillation to enhance QAT. It addresses the limitations of layer-to-layer KD in token prediction recovery by reforming intermediate representation and has successfully applied QAT to LLMs. MiniMA (Zhang et al., 2023a) proposes a viewport towards the capacity gap in distilling LLMs, converting it into a principle through analysis and introducing a 3B Language Model that sets a new benchmark for compute-performance pareto frontier. Experimental results show that MiniMA achieves the best accuracy compared to other 3B distilled LLMs. Lastly, Generalized knowledge distillation (GKD) (Agarwal et al., 2024) addresses the issue of distribution mismatch by drawing output sequences from the student model during training. GKD can be applied in combination with other distillation methods to improve their compression performance.

Black-Box Knowledge Distillation. Different from white-box KD, in black-box KD, only the outputs generated from the teacher LLM are used in the distillation process. Inspired by ICT (Chen et al., 2022c) and MetaICL (Min et al., 2022a), where the language model is meta-trained under a wide range of tasks using in-context learning objectives and then fine-tuned for unseen tasks through in-context learning, Multitask-ICT (Huang et al., 2022) introduces a concept known as in-context learning distillation to transfer the few-shot learning capabilities from the teacher model to the student model. Experimental results show that under Multitask-ICT, in-context learning objectives achieve the best performance when combined with language modeling objectives. Similarly, Li et al. (2024b) introduce a hybrid prompting technique that employs multi-task learning along with explanations generated by GPT-3 text-davinci-002 version (OpenAI, 2023). This method is used to distill explanations into smaller models, achieving consistent and significant improvements over strong single-task fine-tuning benchmarks in different scenarios. Experiments on multiple reasoning tasks show that this method even perform better than finetuning or prompting a 60x larger GPT-3 (175B) model by up to 9.5% in accuracy. Lion (Jiang et al., 2023b) introduces an adversarial distillation architecture aimed at enhancing the efficiency of knowledge transfer by incrementally improving the skill level of the student model. Specifically, it prompts LLMs to recognize challenging instructions and creates new complex instructions for the student model, thereby establishing a three-phase adversarial cycle involving imitation, discrimination, and generation. Experimental results show that Lion-13B not only achieves comparable open-ended generation capabilities to ChatGPT but surpasses conventional instruction-tuned models. DISCO (Chen et al., 2023j) prompts a general LLM to produce phrasal perturbations. These generated perturbations are then filtered by a specialized teacher model to distill high-quality counterfactual

Table 1: Pre-training costs of representative LLMs.

Model	Parameter Size	Data Scale	GPUs Cost	Training Time
GPT-3 (Brown et al., 2020)	175B	300B tokens	-	-
GPT-NeoX-20B (Black et al., 2022)	20B	825GB corpus	96 A100-40G	-
OPT (Zhang et al., 2022a)	175B	180B tokens	992 A100-80G	-
BLOOM (Scao et al., 2023)	176B	366B tokens	384 A100-80G	105 days
GLM (Zeng et al., 2023)	130B	400B tokens	786 A100-40G	60 days
LLaMA (Touvron et al., 2023a)	65B	1.4T tokens	2048 A100-80G	21 days
LLaMA-2 (Touvron et al., 2023b)	70B	2T tokens	A100-80G	71,680 GPU days
Gopher (Rae et al., 2022)	280B	300B tokens	1024 A100	13.4 days
LaMDA (Thoppilan et al., 2022)	137B	768B tokens	1024 TPU-v3	57.7 days
GLaM (Du et al., 2022)	1200B	280B tokens	1024 TPU-v4	574 hours
PanGu-α (Zeng et al., 2021)	13B	1.1TB corpus	2048 Ascend 910	-
PanGu-$\sum$ (Ren et al., 2023b)	1085B	329B tokens	512 Ascend 910	100 days
PaLM (Chowdhery et al., 2022)	540B	780B tokens	6144 TPU-v4	-
PaLM-2 (Anil et al., 2023)	-	3.6T tokens	TPUv4	-
WeLM (Su et al., 2023a)	10B	300B tokens	128 A100-40G	24 days
Flan-PaLM (Chung et al., 2022)	540B	-	512 TPU-v4	37 hours
AlexaTM (Soltan et al., 2022)	20B	1.3 tokens	128 A100	120 days
Codegeex (Zheng et al., 2023)	13B	850 tokens	1536 Ascend 910	60 days
MPT-7B (Team, 2023)	7B	1T tokens	-	-

data into smaller student models, allowing the smaller models to learn causal representations more reliably. As another line of research, some studies have shown that chain-of-thought (CoT) prompting can elicit language models to solve complex reasoning tasks step by step, with the aim to transfer such ability from large models into smaller ones through black-box KD. For example, to enhance the CoT math reasoning capabilities of smaller models, Fu et al. (2023b) propose a method for instruct-tuning a student model (FlanT5) by distilling the reasoning pathways found in the GSM8K dataset from a teacher model (GPT-3.5 code-davinci-002 (Chen et al., 2021b)). Fine-tuning and distilling smaller models require substantial amounts of training data to match the performance of the large model. To address this issue, Hsieh et al. (2023) propose Distilling Step-by-Step, a technique that uses CoT prompting to extract LLM rationales for extra guidance in training smaller models within a multi-task setting. Experimental results show that Distilling Step-by-Step achieves better performance with much fewer labeled or unlabeled training examples compared to both fine-tuning and standard distillation. Fine-tune-CoT (Ho et al., 2023) utilizes existing zero-shot CoT prompting techniques (Kojima et al., 2022) to create rationales from LLMs. These rationales are then used to fine-tune smaller student models. It also introduces diverse reasoning, a method that employs stochastic sampling to generate a variety of reasoning solutions from teacher models, which serves to enrich the training data for the student models. SOCRATIC CoT (Shridhar et al., 2023) breaks down the original problem into a series of smaller sub-problems and utilizes this decomposition to direct the intermediate steps of reasoning. It is used to train a pair of smaller, distilled models: one specializes in dissecting the problem and the other focuses on solving these sub-problems. SOCRATIC COT is shown to be an effective alternative to CoT, enabling a much smaller model (GPT-2 large) to outperform a 10x larger model (GPT-3 6B). SCOTT (Wang et al., 2023c) uses rationales generated by LLMs to train a student model under a counterfactual reasoning framework. It ensures that the student model does not overlook the provided rationales, thereby preventing it from making inconsistent predictions. Experimental results show that SCOTT can generate CoT rationales that are more faithful than original CoT prompting. Li et al. (2023b) present a method called symbolic CoT distillation (SCoTD) that draws CoT rationales from a LLM using unlabeled data instances. A smaller model is then trained to predict both the sampled rationales and the associated labels. Lastly, Peng et al. (2023a) utilize GPT-4 as a teacher model to generate English and Chinese instruction-based datasets to refine student LLMs. They show that the 52K data points generated by GPT-4 are able to improve zero-shot performance compared to instruction-following data generated from previous state-of-the-art models.

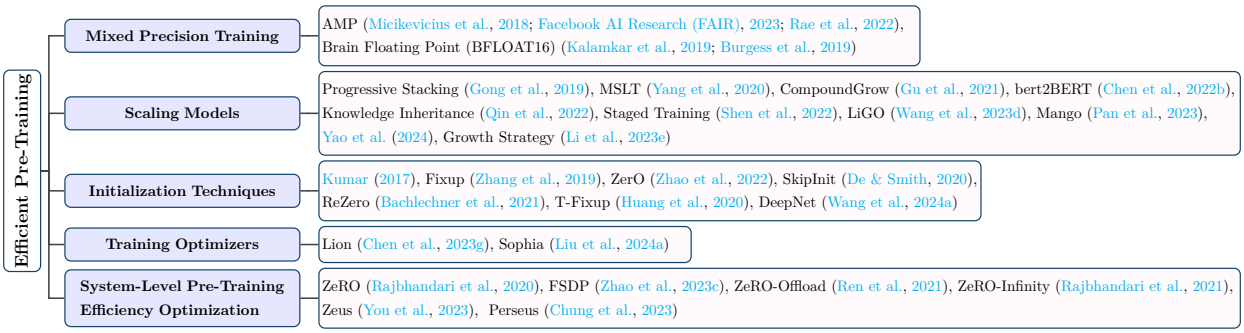

Figure 6: Summary of efficient pre-training techniques for LLMs.

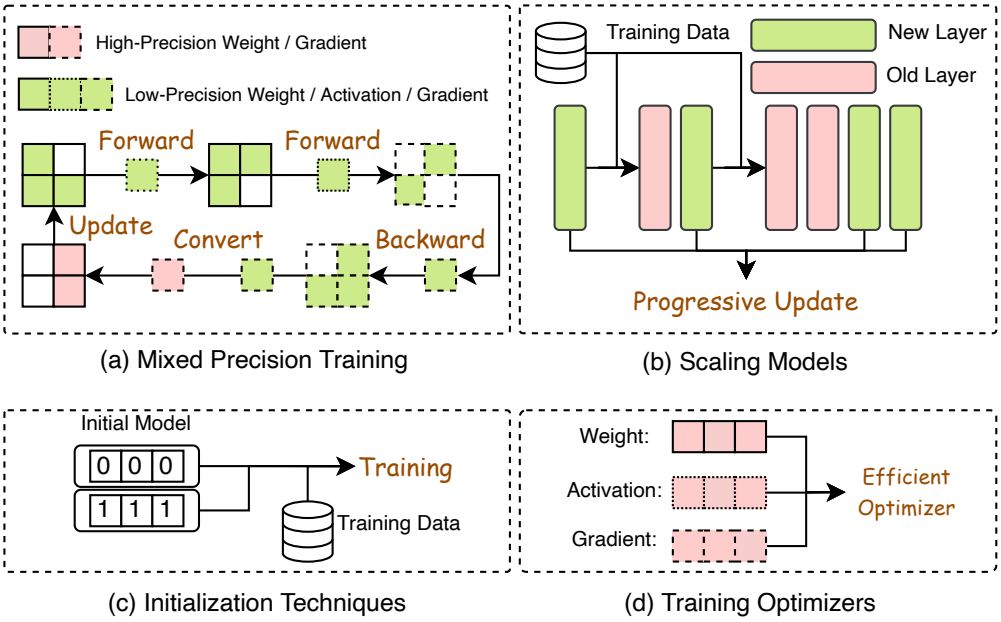

Figure 7: Illustrations of efficient pre-training techniques for LLMs.

2.2 Efficient Pre-Training

As shown in Table 1, pre-training LLMs incurs significant costs. Efficient pre-training techniques focus on reducing the costs of the LLM pre-training process in terms of compute resources, training time, memory and energy consumption. As summarized in Figure 6, enhancing the efficiency of pre-training can be achieved through different and complementary techniques, including mixed precision acceleration, scaling models, initialization techniques, training optimizers, and system-level pre-training efficiency optimization.

Mixed Precision Training. Mixed precision training enhances pre-training efficiency by using low-precision models for forward and backward propagation and then converting the calculated low-precision gradients to high-precision ones for updating the original high-precision weights. For example, Micikevicius et al. (2018) propose Automatic Mixed Precision (AMP) to keep a master copy of weights in full-precision (FP32) for updates, whereas weights, activations, and gradients are stored in FP16 for arithmetic operations. Notably, the improved version of AMP (Facebook AI Research (FAIR), 2023) has eliminated the copy of FP32 weights, but the optimizer (AdamW) still uses FP32 internally. Meanwhile, Rae et al. (2022) demonstrate that FP16 in AMP results in accuracy loss due to the restricted numerical range. To address this issue, Brain Floating Point (BFLOAT16), which has a greater dynamic range — i.e., number of exponent bits — than FP16, was proposed (Kalamkar et al., 2019; Burgess et al., 2019) to achieve better training performance.

Scaling Models. Techniques based on scaling models accelerate pre-training convergence and reduce training costs by leveraging the weights of a smaller model to upscale to a larger one. For example, Gong et al. (2019) introduce a technique named progressive stacking to transfer knowledge from a simpler model to a more complex one to enhance model training efficiency. Meanwhile, Yang et al. (2020) observe that as the depth of the model increases through progressive stacking, the training speed however decreases. To address this issue, they propose multi-stage layer training (MSLT), which only updates the output and newly introduced top encoder layers while keeping the previously trained layers unchanged. Once all the layers have been trained, MSLT fine-tunes the entire model by updating each layer with 20% of the total steps, making it more time-efficient than the traditional progressive stacking approach. Similarly, Gu et al. (2021) introduce CompoundGrow, which begins with training a small model and then incrementally expands it using a mix of model growth techniques, including increasing input length, model breadth and depth, leading to an acceleration in the pre-training process in wall-clock time compared to progressive stacking. Chen et al. (2022b) propose bert2BERT, which applies function-preserving initialization (FPI) and advanced knowledge initialization (AKI) to transfer the knowledge of a smaller pre-trained model to a large model to improve the pre-training efficiency of the large model. Specifically, FPI enforces the initialized larger model to closely mirror the behavior of the smaller model, laying a strong basis for later optimization; and AKI promotes faster convergence by replicating weights from higher layers. Experimental results show that bert2BERT is able to save a significant amount of training cost over MSLT. Qin et al. (2022) propose Knowledge Inheritance which employs knowledge distillation as an auxiliary supervision during pre-training. This facilitates training a larger model from a smaller teacher model, thereby enhancing both the pre-training speed and the generalization ability. Shen et al. (2022) introduce Staged Training that begins with a small model and progressively increases its depth and breadth through a growth operator. By starting each stage with the results from the previous one, it effectively reuses computation, leading to a more efficient training process compared to previous techniques like CompoundGrow and progressive stacking. Wang et al. (2023d) propose Linear Growth Operator (LiGO) that linearly maps the parameters of a smaller model to initiate a larger one. By using a composition of width-and depth-growth operators further enhanced with Kronecker factorization to capture architectural knowledge, LiGO outperforms bert2BERT which saves about 30% computational costs. Pan et al. (2023) introduce a technique named Mango which establishes a linear relationship between each weight of the target model and all weights of the pretrained model to boost acceleration capabilities. It also employs multi-linear operators to decrease computational and spatial complexity during pre-training, achieving 59.9% acceleration ratio compared to Chen et al. (2022b) and LiGO. Drawing from these scaling techniques and the progressive pre-training (Yao et al., 2024), recent LLMs like FLM-101B (Li et al., 2023e) introduce a growth strategy to cut LLM training costs by expanding model structures offline and resuming from the previous stage's smaller model checkpoint.

Initialization Techniques. Initialization plays a key role in enhancing the efficiency of LLM pre-training because a good initialization can accelerate the convergence of the model. Most LLMs employ initialization techniques that were adopted in training smaller-scale models. For example, initialization method introduced by Kumar (2017) balances input and output variances. Fixup (Zhang et al., 2019) and ZerO (Zhao et al., 2022) set the backbone to zero, preserving signal identity. SkipInit (De & Smith, 2020) substitutes batch normalization with a zero-value multiplier. ReZero (Bachlechner et al., 2021) adds zero-valued parameters to maintain identity which leads to faster convergence. T-Fixup (Huang et al., 2020) follows Fixup to adopt rescaling schemes for the initialization of the residual blocks of Transformer models. DeepNet (Wang et al., 2024a) adjusts the residual connection in deep Transformers using Post-LN-init, ensuring stable inputs to layer normalization and mitigating gradient vanishing for stable optimization.

Training Optimizers. Popular LLMs such as GPT-3 (Brown et al., 2020), OPT (Zhang et al., 2022a), BLOOM (Scao et al., 2023), and Chinchilla (Hoffmann et al., 2022) are predominately pre-trained using Adam (Kingma & Ba, 2017) or AdamW (Loshchilov & Hutter, 2019) as optimizers. However, both Adam and AdamW are memory hungry and computationally expensive. Some studies (Chen et al., 2023g; Liu et al., 2024a) propose new optimizers to accelerate LLM pre-training. Specifically, Chen et al. (2023g) propose to leverage search techniques to traverse a large and sparse program space to discover optimizers for model training. The discovered optimizer, named Lion (EvoLved Sign Momentum), is more memory-efficient than Adam as it only keeps track of the momentum. Liu et al. (2024a), on the other hand, propose Sophia as a lightweight second-order optimizer that outpaces Adam with doubling the pre-training speed. Sophia

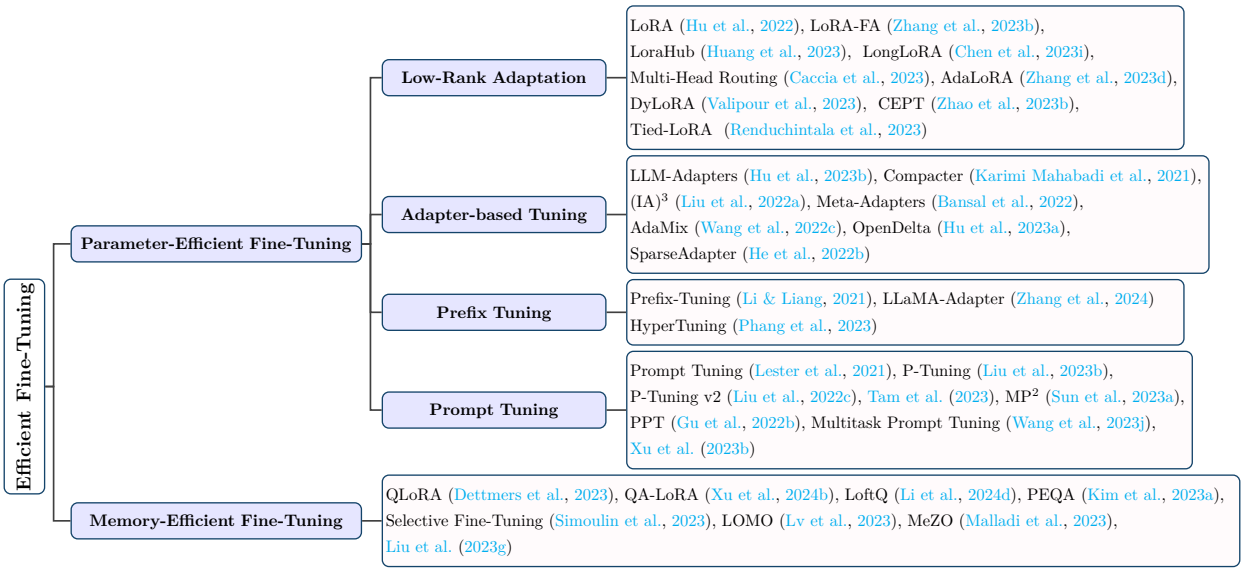

Figure 8: Summary of efficient fine-tuning methods for LLMs.

calculates the moving average of gradients and the estimated Hessian, dividing the former by the latter and applying element-wise clipping. It effectively moderates update sizes, addresses non-convexity and rapid hessian changes, enhancing both memory utilization and efficiency.

System-Level Pre-Training Efficiency Optimization. Due to high demand on memory and compute resources, LLMs are usually pre-trained across multiple compute nodes in a distributed manner. Therefore, most system-level optimization techniques are designed in the setting of large-scale distributed training. For instance, Zero Redundancy Data Parallelism (ZeRO) (Rajbhandari et al., 2020) provides three stages of optimization to partition various training states across different devices. Specifically, ZeRO-1 only partitions the optimizer states, whereas ZeRO-2 partitions both the optimizer states and the gradients. ZeRO-3 further partitions the model parameters across devices compared with ZeRO-1 and ZeRO-2. Although runtime memory is further reduced through ZeRO-3, there is about 50% increase in communication volume. Therefore, it is recommended to use ZeRO-3 within a node to minimize the communication time while using ZeRO-1 and ZeRO-2 across nodes. Fully Sharded Data Parallel (FSDP) (Zhao et al., 2023c) shares a similar idea for optimization, and designs a hybrid sharding strategy to allow users to define which nodes or processes to partition the gradients, model parameters, and optimizer states across different nodes. In the case when the weight memory exceeds the aggregated memory that can be provided by all of the compute nodes, ZeRO-Offload (Ren et al., 2021) enables offloading any stage of ZeRO to CPU memory, whereas ZeRO-Infinity (Rajbhandari et al., 2021) provides a mechanism to offload to NVMe drives in addition to CPU memory. However, it is quite difficult to maintain performance using these two alternatives, as the data movement between CPU and GPU is slow. Lastly, training LLMs on numerous GPUs consumes a massive amount of energy, Zeus (You et al., 2023) and Perseus (Chung et al., 2023) are proposed to optimize energy consumption by finding the best GPU-level configurations based on the unique LLM characteristics. The evaluation shows that Perseus reduces energy consumption of large model training by up to 30%.

2.3 Efficient Fine-Tuning

Efficient fine-tuning techniques focus on reducing the costs of the LLM fine-tuning process. As summarized in Figure 8, efficient fine-tuning techniques can be grouped into parameter-efficient fine-tuning (PEFT) and memory-efficient fine-tuning (MEFT).

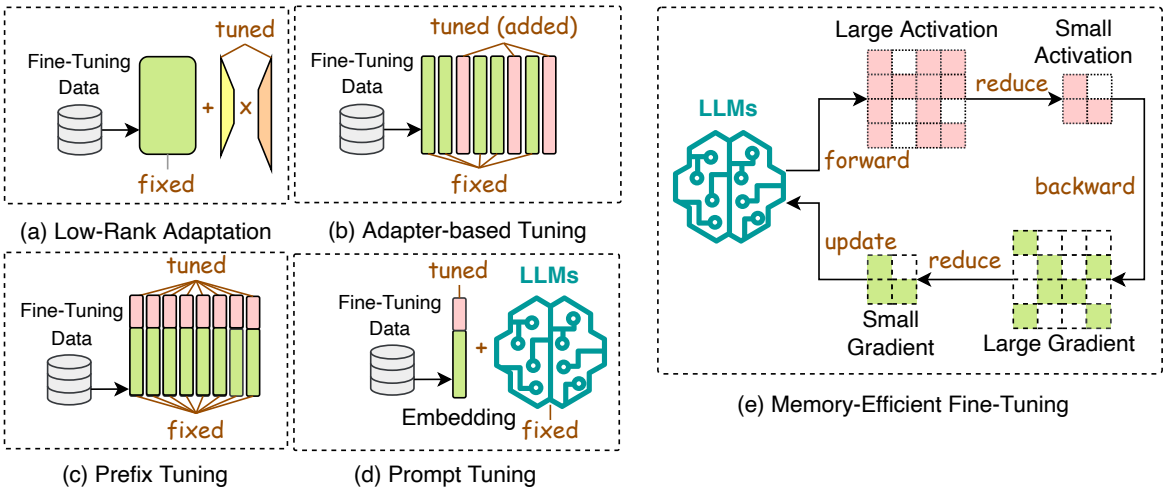

Figure 9: Illustrations of parameter-efficient fine-tuning (a)-(d) and memory-efficient fine-tuning (e).

2.3.1 Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) adapts an LLM to downstream tasks by freezing the whole LLM backbone and only updating a small set of newly added extra parameters. In general, PEFT methods can be grouped into four categories: low-rank adaptation, adapter-based tuning, prefix tuning, and prompt tuning.

Low-Rank Adaptation. Low-rank adaptation (LoRA) (Hu et al., 2022) is a widely used PEFT approach for LLMs. The hypothesis is that the change in weights during model adaptation has a low "intrinsic rank". Hence, LoRA introduces two trainable low-rank matrices $\mathbf{A} \in \mathbb{R}^{m \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times n}$ and adjusts the weight matrix by $\mathbf{W} \leftarrow \mathbf{W} + \Delta\mathbf{W} = \mathbf{W} + \mathbf{A} \cdot \mathbf{B}$. As such, only the small matrices $\mathbf{A}$ and $\mathbf{B}$ are updated during fine-tuning, while the original large weight matrix remains frozen, making the fine-tuning process more efficient. To enhance the efficiency of LoRA, LoRA-FA (Zhang et al., 2023b) keeps the projection-down weights of $\mathbf{A}$ fixed while only updating the projection-up weights of $\mathbf{B}$ in each LoRA adapter so that the weight modifications during fine-tuning are confined to a low-rank space, thereby eliminating the need to store the full-rank input activations. It achieves comparable accuracy related to full parameter fine-tuning and LoRA. Building on top of LoRA, LoraHub (Huang et al., 2023) explores the composability of LoRA for the purpose of generalizing across different tasks. It combines LoRA modules that have been trained on various tasks with the goal of attaining good performance on tasks that have not been seen before. LongLoRA (Chen et al., 2023i), on the other hand, extends LoRA to the long-context fine-tuning scenario. It introduces shift short attention (S^2-Attn), which effectively facilitates context expansion, showing that LoRA is effective for long context when utilizing trainable embedding and normalization. Multi-Head Routing (MHR) (Caccia et al., 2023) extends LoRA to Mixture-of-Experts (MoE) architectures. It outperforms Polytropon (Ponti et al., 2023) when operating with a similar parameter allocation. Notably, it achieves competitive performance while focusing on fine-tuning the routing function alone, without making adjustments to the adapters, demonstrating remarkable parameter efficiency. Zhang et al. (2023d) observe that many PEFT techniques neglect the differing significance of various weight parameters. To address this, they propose AdaLoRA which employs singular value decomposition to parameterize incremental updates and adaptively distributes the parameter budget based on the importance score of each weight matrix. The rank in LoRA is static and cannot be adaptively adjusted during fine-tuning. Valipour et al. (2023) propose DyLoRA to introduce a dynamic low-rank adaptation method that trains LoRA blocks across multiple ranks rather than just one by organizing the representations learned by the adapter module based on their ranks. Different from the above-mentioned methods that apply LoRA-based methods to full-size LLMs, CEPT (Zhao et al., 2023b) introduces a framework that utilizes compressed LLMs. Specifically, it assesses how prevalent LLM compression methods affect PEFT performance and subsequently implements strategies for knowledge retention and recovery to counteract the loss of knowledge induced by compression. Lastly, Tied-LoRA (Renduchintala et al., 2023) uses weight tying and selective training to further increase parameter efficiency of LoRA.

Adapter-based Tuning. Adapters are bottleneck-like trainable modules integrated into LLMs, which first down-project the input feature vector followed by a non-linear layer and then up-project back to the original size (Houlsby et al., 2019). Adapter-based tuning includes both series adapters and parallel adapters. In series adapters, each LLM layer has two adapter modules added after its attention and feed-forward modules; whereas parallel adapters position two adapter modules alongside the attention and feed-forward modules within each layer of the LLM. In particular, Hu et al. (2023b) propose LLM-Adapters, which integrates series or parallel adapters into LLMs for fine-tuning on different tasks. Karimi Mahabadi et al. (2021) propose Compacter, which unifies adapters, low-rank techniques, and the latest hyper-complex multiplication layers to achieve a balanced trade-off between the amount of trainable parameters and task performance compared to original Adapter method (Houlsby et al., 2019). Furthermore, $(IA)^3$ (Liu et al., 2022a) introduces a technique that scales activations using learned vectors. It outperforms Adapter (Houlsby et al., 2019) and Compacter on few-shot setting with better accuracy and computational efficiency. Following meta-learning principles, Meta-Adapters (Bansal et al., 2022) designs a resource-efficient fine-tuning technique for the few-shot scenario where it incorporates adapter layers that have been meta-learned into a pre-trained model, transforming the fixed pre-trained model into an efficient few-shot learning framework. Meta-Adapters outperforms Adapter (Houlsby et al., 2019) at few-shot fine-tuning with less parameters to fine-tune. AdaMix (Wang et al., 2022c) takes inspiration from sparsely-activated mixture-of-experts (MoE) models (Zuo et al., 2022) and proposes a mixture of adaptation modules to learn multiple views of the given task. Compared to Adapter, it demonstrates better results on both natural language understanding and generation tasks with less learnable parameters. Lastly, OpenDelta (Hu et al., 2023a) is an open-source software library that offers a versatile and plug-and-play framework for implementing a range of adapter-based techniques, and is designed to be compatible with various LLMs architectures.

Prefix Tuning. Prefix-Tuning (Li & Liang, 2021) adds a series of trainable vectors, known as prefix tokens, to each layer in an LLM. These prefix tokens are tailored to specific tasks and can be treated as virtual word embeddings. Building on top of Prefix-Tuning, LLaMA-Adapter (Zhang et al., 2024) incorporates a set of trainable adaptation embeddings and attaches them to the word embeddings in the upper layers of the LLMs. A zero-initialized attention scheme with zero gating is also introduced. It dynamically incorporates new guiding signals into LLaMA-1 while retaining its pre-trained knowledge. Different from conventional prefix tuning, HyperTuning (Phang et al., 2023) employs a hyper-model to produce task-specific parameters such as soft prefixes for a downstream model, showing improved performance through initialization from hypermodel-generated parameters for subsequent fine-tuning.

Prompt Tuning. Different from prefix tuning, prompt tuning incorporates trainable prompt tokens only at the input layer. These tokens can be inserted either as a prefix or anywhere within the input tokens. Prompt Tuning (Lester et al., 2021) keeps the entire pre-trained model fixed while adding an extra k trainable tokens at the beginning of the input text for each downstream task. It outperforms few-shot prompts and narrows the performance gap compared to full-model fine-tuning. P-Tuning (Liu et al., 2023b) utilizes a small number of parameters as prompts, which are processed by a prompt encoder before being used as input for pre-trained LLMs. Instead of searching for discrete prompts, P-Tuning fine-tunes these prompts through gradient descent and improves performance on a wide range of natural language understanding tasks compared to Prompt Tuning. Liu et al. (2022c) observe that earlier versions of prefix tuning struggle with complex sequence labeling tasks. To address this, they propose P-Tuning v2, which borrows the ideas from prefix tuning by introducing continuous prompts at each layer of the pre-trained model. This modification has proven effective in boosting performance across various parameter sizes for tasks related to natural language understanding. Tam et al. (2023) introduce efficient prompt tuning for text retrieval, updating just 0.1% of parameters and outperforming traditional full-parameter update methods in diverse domains. Sun et al. (2023a) claim that prompt tuning tends to struggle in few-shot learning scenarios, and thus propose MP^2 that pre-trains a collection of modular prompts using multitask learning. These prompts are then selectively triggered and assembled by a trainable routing mechanism for specific tasks. As a result, MP^2 can quickly adapt to downstream tasks by learning how to merge and reuse pretrained modular prompts. Different from MP^2, PPT (Gu et al., 2022b) attributes the performance degradation of prompt tuning in few-shot learning to the poor initialization of soft prompt, and thus proposes to add the soft prompt into the pre-training stage for a better initialization. Lastly, Multitask Prompt Tuning (Wang et al., 2023j) extends Prompt Tuning and harnesses the knowledge of the various tasks through the use of prompt vectors

in a multitask learning settings. Specifically, it initially learns a single, transferable prompt by extracting knowledge from various task-specific source prompts, and then applies multiplicative low-rank updates to this prompt to effectively tailor it for each downstream task. By doing this, Multitask Prompt Tuning is able to attain performance levels that are competitive compared to full-model fine-tuning methods.

2.3.2 Memory-Efficient Fine-Tuning

Different from PEFT methods which focus on parameter efficiency, MEFT methods focus on memory savings during the LLMs fine-tuning process. For instance, Dettmers et al. (2023) propose QLoRA which first quantizes the model into a 4-bit NormalFloat data type, and then fine-tunes this quantized model with added low-rank adapter (LoRA) weights (Hu et al., 2022). In doing so, QLoRA reduces memory usage during fine-tuning without performance degradation compared to standard full-model fine-tuning. QA-LoRA (Xu et al., 2024b) improves QLoRA by introducing group-wise operators that improve quantization flexibility (each group is quantized separately) while reducing adaptation parameters (each group utilizes shared adaptation parameters). Similarly, LoftQ (Li et al., 2024d) combines model quantization with singular value decomposition (SVD) to approximate the original high-precision pre-trained weights. As a result, it offers a favorable initialization point for subsequent LoRA fine-tuning, leading to enhancements over QLoRA on both natural language understanding and generation tasks. PEQA (Kim et al., 2023a) introduces a two-stage approach to quantization-aware fine-tuning. In the first stage, the parameter matrix for each fully connected layer is quantized into a matrix of low-bit integers along with a scalar vector. In the second stage, the low-bit matrix remains unchanged, while fine-tuning is focused solely on the scalar vector for each specific downstream task. Employing this two-stage approach, PEQA not only minimizes memory usage during fine-tuning but also speeds up inference time by maintaining weights in a low-bit quantized form, showing better perplexity than GPTQ (Frantar et al., 2023) with LoRA. Different from above-mentioned MEFT methods that combine LoRA with quantization to reduce fine-tuning memory footprints, as another line of research, some studies propose MEFT methods based on gradient optimization. Specifically, Simoulin et al. (2023) propose Selective Fine-Tuning which minimizes memory usage by specifically preserving a subset of intermediate activations from the forward pass for which the calculated gradients are nonzero. Notably, this approach delivers performance equivalent to full-model fine-tuning while using just up to one-third of the GPU memory required otherwise. Lv et al. (2023) introduce LOMO, which minimizes memory consumption during fine-tuning by combining gradient calculation and parameter updating into a single step. As such, LOMO eliminates all components of the optimizer state, lowering the memory requirements for gradient tensors to $O(1)$. Lastly, MeZO (Malladi et al., 2023) improves the zeroth-order method (Spall, 1992) for gradient estimation using only two forward passes. This enables efficient fine-tuning of LLMs with memory requirements similar to inference and supports both full-parameter and PEFT methods like LoRA (Hu et al., 2022) and prefix tuning (Li & Liang, 2021), enabling MeZO to train a 30-billion parameter model on a single A100 80GB GPU.

2.4 Efficient Inference

Efficient inference techniques focus on reducing the costs of the LLMs inference process. As summarized in Figure 10, efficient inference techniques can be grouped into techniques at algorithm level and system level.

Algorithm-Level Inference Efficiency Optimization. Techniques that enhance LLM inference efficiency at the algorithm level include speculative decoding and KV-cache optimization.

- ***Speculative Decoding.*** Speculative decoding (i.e., speculative sampling) (Leviathan et al., 2023) is a decoding strategy for autoregressive language models that speeds up the sampling process by computing tokens using a smaller draft model in parallel to create speculative prefixes for the large target model. Chen et al. (2023a) focus on the distributed serving setting for LLMs and propose to run a faster autoregressive model K times and then evaluate the preliminary output with the large target model. A tailored rejection sampling strategy is employed to approve a selection of the draft tokens in a left-to-right order, thereby recapturing the distribution of the large target model during the procedure. Staged Speculative (Spector & Re, 2023) transforms the speculative batch into a tree structure representing potential token sequences. This restructuring expedites the

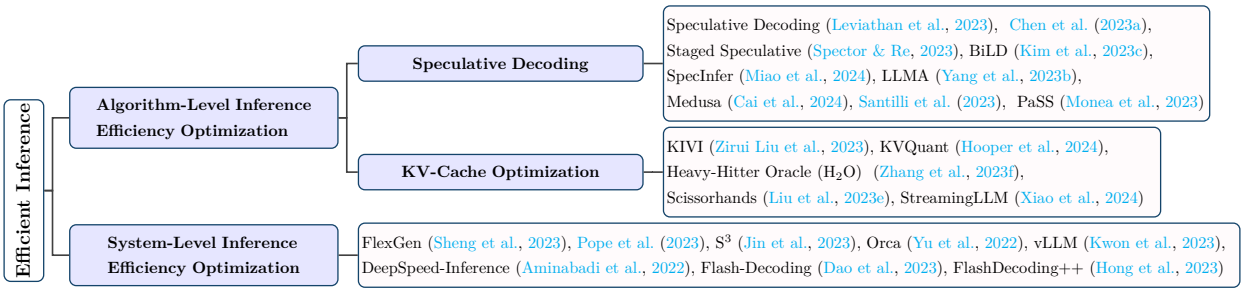

Figure 10: Summary of efficient inference techniques for LLMs.

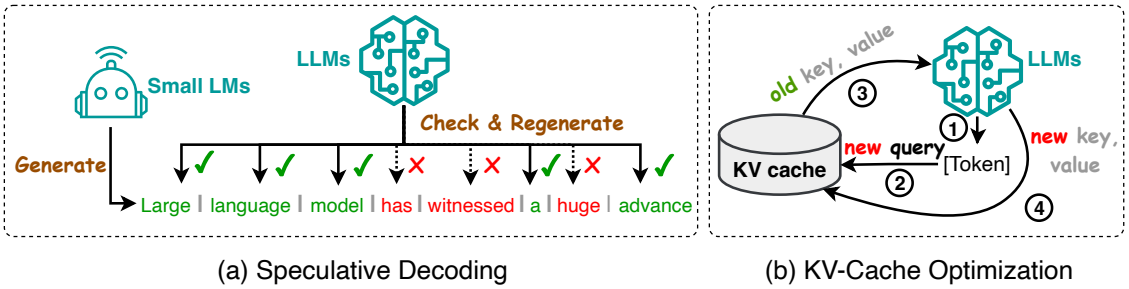

Figure 11: Illustrations of algorithm-level efficiency optimization techniques for LLM inference.

generation of larger and improved speculative batches. It also introduces an additional phase for speculative decoding of the initial model, thereby enhancing overall performance, showing 1.36x over standard speculative decoding. BiLD (Kim et al., 2023c) optimizes speculative decoding through two innovative techniques: the fallback policy that permits the smaller draft model to waive control to the larger target model when it lacks sufficient confidence; and the rollback policy that enables the target model to revisit and rectify any inaccurate predictions made by the smaller draft model. SpecInfer (Miao et al., 2024) extends speculative decoding and speeds up inference by employing speculative inference techniques and token tree validation. Its core idea involves merging a range of small speculative models that have been fine-tuned collectively to collaboratively forecast the output of the large target model, which is then used to validate all the predictions. Different from speculative decoding that needs to introduce an additional efficient drafter model to generate a draft for checking, LLMA (Yang et al., 2023b) chooses a text segment from a closely related reference and duplicates its tokens into the decoder. It then concurrently assesses the suitability of these tokens as the decoding output within a single decoding step. This approach results in a speed increase of more than two times while maintaining the same generated results as traditional greedy decoding. Similarly, instead of using a separate draft model to sequentially generate candidate output, Medusa (Cai et al., 2024) proposes to freeze the LLM backbone, fine-tune additional heads, and use a tree-based attention mechanism to process predictions in parallel to speed up the decoding process. Lastly, Santilli et al. (2023) propose parallel decoding including the Jacobi and Gauss-Seidel fixed-point iteration methods for speculative decoding. Among these methods, Jacobi decoding was extended into Lookahead decoding (Fu et al., 2023c) to further enhance the efficiency.

- ***KV-Cache Optimization.*** During inference, LLMs need to store the Key-Value (KV) pairs of the past tokens into the cache for future token generation. The size of KV cache needed enlarges massively with the increase of generated token length, resulting in considerable memory consumption and long inference latency. Therefore, reducing the size of KV cache is key to enhancing inference efficiency. Existing KV-cache optimization techniques can in general be grouped into two categories. The first category is to compress the KV cache. For example, Zirui Liu et al. (2023) propose KIVI, a tuning-free 2bit KV cache quantization algorithm which quantizes the key cache per-channel and the value cache per-token, achieving 2.6x less peak memory usage during inference. Similarly, Hooper

et al. (2024) conduct an empirical study on the impact of per-channel quantization and other types of quantization such as quantization before rotary positional embedding. Based on their findings, they propose KVQuant which combines these quantization methods to quantize the KV cache of LLaMA to 3-bit. The second category of KV-Cache optimization techniques is to evict some KVs from the cache. For instance, Zhang et al. (2023f) propose Heavy-Hitter Oracle (H_2O), a KV cache eviction strategy that formulates the KV cache eviction as a dynamic submodular problem and dynamically retains a balance between recent and performance-critical tokens, improving the throughput for LLMs inference. Similarly, Liu et al. (2023e) propose a hypothesis named the persistence of importance, suggesting that only tokens that were crucial at an earlier phase will have a significant impact on subsequent stages. Based on this hypothesis, they design Scissorhands which significantly reduces the KV cache without compromising model quality. Lastly, StreamingLLM (Xiao et al., 2024) incorporates window attention, where only the most recent KVs are cached into a fixed-size sliding window, into their algorithm design. Through evicting the outdated KVs, StreamingLLM ensures constant memory usage and decoding speed after the cache is initially filled.

System-Level Inference Efficiency Optimization. The efficiency of LLM inference can also be optimized at the system level under a specific hardware architecture. For example, FlexGen (Sheng et al., 2023) is a high-throughput inference engine that enables the execution of LLMs on GPUs with limited memory. It uses a linear programming-based search approach to coordinate various hardware, combining the memory and computation from GPU, CPU, and disk. Furthermore, FlexGen quantizes the weights and attention cache to 4 bits, which increases the inference speed of OPT-175B (Zhang et al., 2022a) on a single 16GB GPU. Pope et al. (2023) develop a simple analytical framework to partition a model in order to scale Transformer inference based on the application requirements. By combining it with scheduling and memory optimizations, they are able to achieve better efficiency on PaLM (Chowdhery et al., 2022) in comparison to FasterTransformer (NVIDIA, 2023a). Orca (Yu et al., 2022) employs iteration-level scheduling to serve batched sequences with variable output sequence length. When a sequence in a batch is completed, it is returned to the user so that a new sequence can be served immediately. As a result, Orca improves GPU utilization compared to static batching, showing 36.9× throughput improvement under the same level of latency compared to FasterTransformer. S^3 (Jin et al., 2023) creates a system that is aware of the output sequence beforehand. It can anticipate the length of the sequence and arrange generation requests accordingly, optimizing the utilization of device resources and increasing the rate of production, showing higher throughput than Orca with the same number of GPUs. However, both Orca and S^3 lead to memory fragmentation due to their inaccurate memory provisioning for each request. vLLM (Kwon et al., 2023) addresses the memory efficiency problem with PagedAttention, which enables the storage of continuous keys and values in non-contiguous memory space. Specifically, PagedAttention divides the KV cache of each sequence into blocks, each containing the keys and values for a fixed number of tokens. During attention computation, PagedAttention kernel manages these blocks efficiently by maintaining a block table to reduce memory fragmentation. Specifically, the contiguous logical blocks of a sequence are mapped to non-contiguous physical blocks via the table and the table automatically allocates a new physical block for every newly generated token. This reduces the amount of memory wasted when generating new tokens, thus improving its efficiency, showing that PagedAttention improves the throughput of popular LLMs by 2-4× with the same level of latency compared to FasterTransformer (NVIDIA, 2023a) and Orca (Yu et al., 2022). On the other hand, infinitely optimizing server-side aggregated metrics does not necessarily lead to good user experience or Quality of Experience (QoE) especially under high server load. Andes (Liu et al., 2024b) first defines QoE for the LLM-based text streaming services and proposes a QoE-aware serving systems to optimize QoE by prioritizing requests based on their resource demand and service acquired. DeepSpeed-Inference (Aminabadi et al., 2022) is a multi-GPU inference approach designed to enhance the efficiency of both dense and sparse Transformer models when they are contained within the collective GPU memory. Furthermore, it provides a mixed inference technique that utilizes CPU and NVMe memory, in addition to GPU memory and computation, enabling high-throughput inference even for models that are too large to fit in the combined GPU memory. This approach demonstrates lower latency than FasterTransformer under the same throughput. Flash-Decoding (Dao et al., 2023) boosts the speed of long-context inference by breaking down keys/values into smaller pieces, computing attention on these pieces in parallel, and then combining them to generate the final output. It outperforms FasterTransformer and FlashAttention (Dao

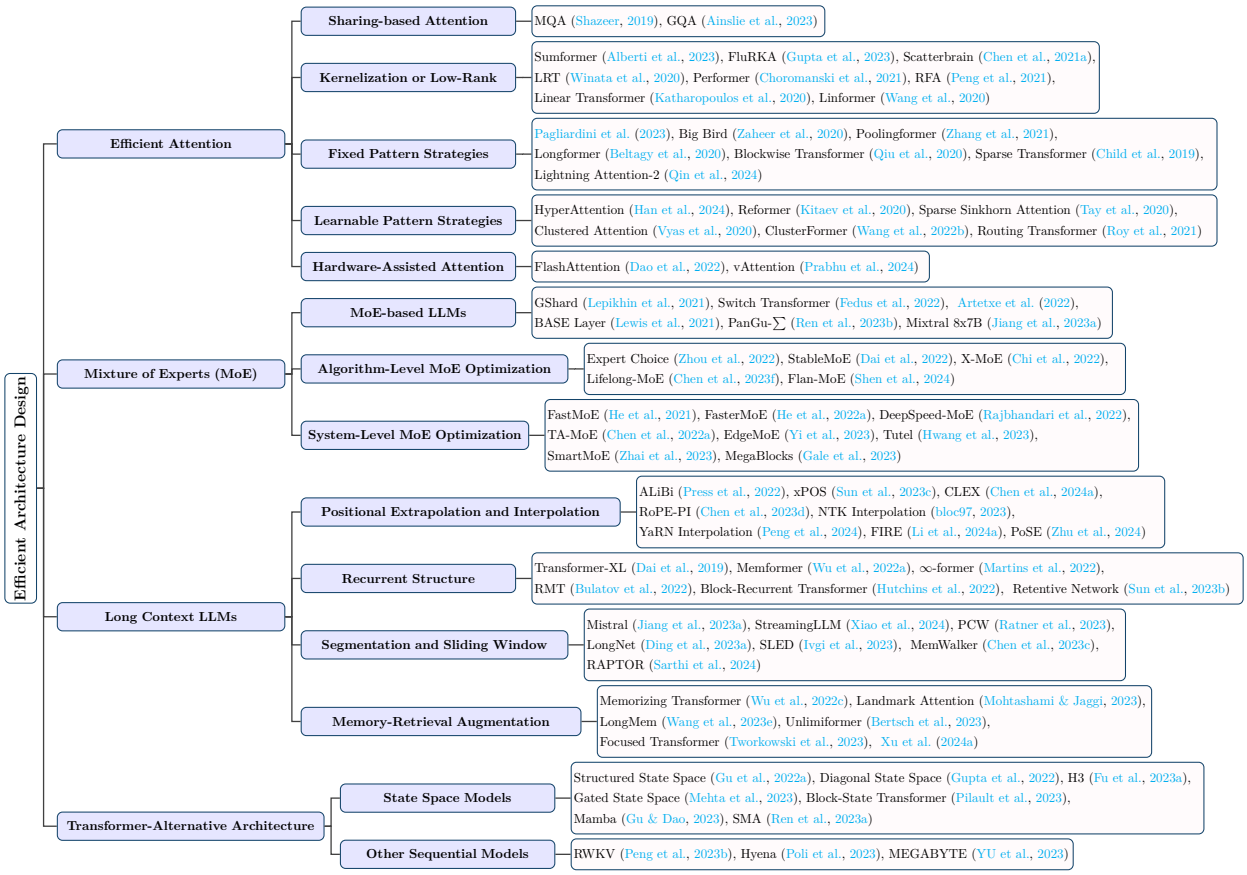

Figure 12: Summary of efficient architecture designs for LLMs.

et al., 2022) in decoding speed for very large sequences. FlashDecoding++ (Hong et al., 2023) supports mainstream language models and hardware backends through asynchronous softmax, double buffering for flat GEMM optimization, and heuristic dataflow, resulting in up to 4.86x and 2.18x acceleration on Nvidia and AMD GPUs respectively compared to HuggingFace implementations, showing higher speedup compared to Flash-Decoding under the same throughput.

2.5 Efficient Architecture Design

Efficient architecture design for LLMs refers to the strategic optimization of model architecture and computational processes to enhance performance and scalability while minimizing resource consumption. Figure 12 provides a summary of existing efforts on designing efficient architectures for LLMs.

2.5.1 Efficient Attention

The quadratic time and space complexity of attention modules considerably slows down the pre-training, inference, and fine-tuning of LLMs (Duman Keles et al., 2023). Many techniques have been proposed to make attention lightweight for more efficient execution. These techniques can be generally categorized as sharing-based attention, kernelization or low-rank, fixed pattern strategies, learnable pattern strategies, and hardware-assisted attention.

Sharing-based Attention. Sharing-based attention accelerates attention computation during inference through KV heads sharing. For example, LLaMA-2 (Touvron et al., 2023b) optimizes the autoregressive decoding process by using multi-query attention (MQA) (Shazeer, 2019) and grouped-query attention (GQA) (Ainslie et al., 2023). In traditional multi-head attention (MHA), each head has distinct linear

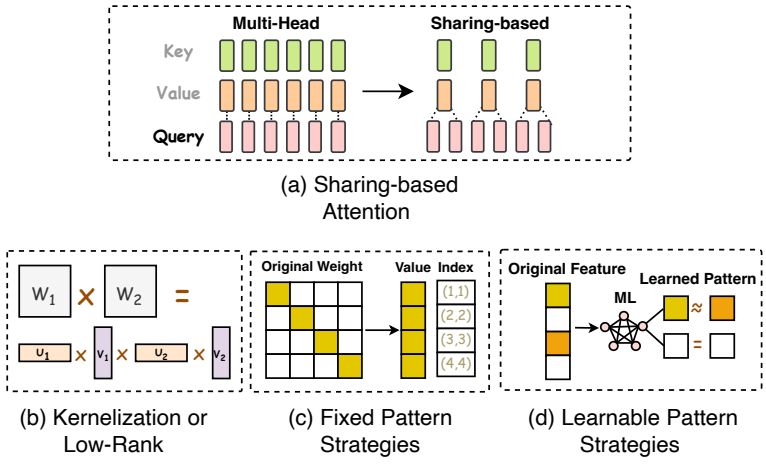

Figure 13: Illustrations of attention optimizations.

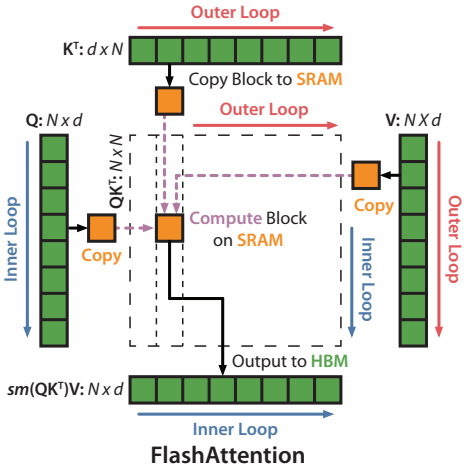

Figure 14: Design of FlashAttention (Dao et al., 2022).

transformations for input matrix queries (Q), keys (K) and values (V), allowing for diverse representations and attention mechanisms across different subspaces. In MQA, all heads share a single set of key and value weights across all query heads. Thus, MQA speeds up the inference but could compromise output quality. To address this drawback, GQA interpolates MQA and MHA by employing one key and value heads for each group of query heads to enhance inference quality.

Kernelization or Low-Rank. Kernelization or low-rank techniques adopted by models such as Sumformer (Alberti et al., 2023), FluRKA (Gupta et al., 2023), Scatterbrain (Chen et al., 2021a), Low-Rank Transformer (LRT) (Winata et al., 2020), Performer (Choromanski et al., 2021), Random Feature Attention (RFA) (Peng et al., 2021), Linear Transformer (Katharopoulos et al., 2020), and Linformer (Wang et al., 2020) enhance the efficiency by utilizing low-rank representations of the self-attention matrix or by adopting attention kernelization techniques. Specifically, low-rank methods focus on compacting the dimensions of attention keys and values. For example, Linformer (Wang et al., 2020) proposes to segment scaled dot-product attention into smaller units via linear projection. Kernelization, a variant of low-rank techniques, focuses on approximating the attention matrix (Choromanski et al., 2020). For example, Performer (Choromanski et al., 2021) condenses softmax attention-kernels using positive orthogonal random features, outperforming Reformer and Linformer on long protein sequence benchmark. Sumformer (Alberti et al., 2023) approximates the equivariant sequence-to-sequence function, offering a universal solution for Linformer and Performer.

Fixed Pattern Strategies. Fixed pattern strategies adopted by models such as (Pagliardini et al., 2023), Big Bird (Zaheer et al., 2020), Poolingformer (Zhang et al., 2021), Longformer (Beltagy et al., 2020), Blockwise Transformer (Qiu et al., 2020), and Sparse Transformer (Child et al., 2019) improve efficiency by sparsifying the attention matrix. This is achieved by confining the attention scope to predetermined patterns, such as local windows or fixed-stride block patterns. For instance, the attention mechanism adopted by Longformer (Beltagy et al., 2020), designed as an alternative to conventional self-attention, merges local windowed attention with globally oriented attention tailored to specific tasks. Pagliardini et al. (2023) expand FlashAttention (Dao et al., 2022) to support a broad spectrum of attention sparsity patterns, including key-query dropping and hashing-based attention techniques, achieving a multi-fold runtime speedup on top of FlashAttention on long text benchmark.

Learnable Pattern Strategies. Different from fixed pattern strategies, learnable pattern strategies adopted by models such as HyperAttention (Han et al., 2024), Reformer (Kitaev et al., 2020), Sparse Sinkhorn Attention (Tay et al., 2020), Clustered Attention (Vyas et al., 2020), ClusterFormer (Wang et al., 2022b), and Routing Transformer (Roy et al., 2021) improve efficiency by learning token relevance and subsequently grouping tokens into buckets or clusters. As an example, HyperAttention (Han et al., 2024) proposes a parameterization for spectral approximation and employs two key metrics: the maximal column norm in the

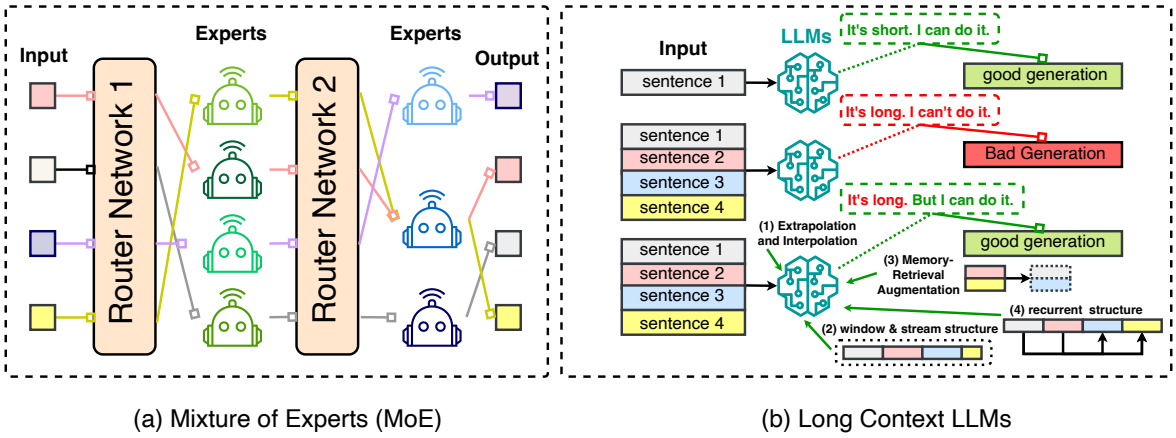

Figure 15: Illustrations of Mixture of Experts (MoE) and long context LLMs.

normalized attention matrix and the row norm ratio in the unnormalized matrix after large entry removal. It also utilizes the learnable sort locality-sensitive hashing (sortLSH) technique and fast matrix multiplication via row norm sampling. The experiment results show that HyperAttention enhances both inference and training speeds for LLMs with only minimal performance degradation, giving significant speed improvements compared to FlashAttention (Dao et al., 2022) on long contexts.

Hardware-Assisted Attention. Hardware-assisted attention focuses on developing hardware-specific techniques to enhance attention efficiency. For example, FlashAttention (Dao et al., 2022) reduces the number of memory access between GPU high-bandwidth memory (HBM) and GPU on-chip SRAM when calculating the attention module in LLMs. Instead of transmitting the values and results between HBM and SRAM multiple times as is done in the standard attention mechanism, FlashAttention combines all the attention operations into one kernel and tiles the weight matrices into smaller blocks to better fit the small SRAM as shown in Figure 14. As a result, only one communication is required to process each attention block, which significantly enhances the efficiency for processing the entire attention block. vAttention (Prabhu et al., 2024) is proposed to store KV cache in contiguous virtual memory without committing physical memory ahead-of-time. It avoids the software complexity to store KV cache by leveraging CUDA support of low-level virtual memory APIs.

2.5.2 Mixture of Experts (MoE)

Mixture of Experts (MoE) represents a sparse methodology utilized prominently in large-scale models like LLMs. It operates on the principle of segmenting a designated task into several sub-tasks, and then developing numerous smaller, specialized models, dubbed *experts*, with each honing in on a distinct sub-task. Subsequently, these experts collaborate to deliver a consolidated output. For pre-traning or fine-tuning, MoE requires developers to manage a huge number of parameters efficiently, enhancing the model's capacity and potentially its performance while keeping the computational and memory requirements relatively manageable. For inference, MoE decreases the inference time by not engaging all experts simultaneously, but rather activating only a select few. Additionally, MoE is capable of minimizing communication between devices in model-distributed scenarios by allocating each expert to an individual accelerator; communication is only necessary between the accelerators that host the router and the relevant expert model (Kaddour et al., 2023).

MoE-based LLMs. Several MoE-based LLMs have been proposed. For example, GShard (Lepikhin et al., 2021) is a MoE-based LLM that offers a refined method to articulate a variety of parallel computation frameworks with minor modifications to the existing model code. It also amplifies a multilingual neural machine translation Transformer model with Sparsely-Gated MoE beyond 600 billion parameters through automatic sharding. Switch Transformer (Fedus et al., 2022) brings forth a switch routing algorithm and crafts intuitively enhanced models, lowering communication and computational expenditures. It encompasses up to one trillion parameters, dividing tasks among up to 2,048 experts, thereby illustrating the scalability

and efficacy of the MoE framework. Artetxe et al. (2022) scale sparse language models to 1.1T parameters, discerning superior performance up to this scale in language modeling, zero-shot and few-shot learning in comparison to dense models. This suggests that sparse MoE models are a computationally efficient substitute for traditionally employed dense architectures. Its largest MoE model outperforms its dense counterpart where the latter requires twice as much computation. BASE Layer (Lewis et al., 2021) defines token-to-expert allocation as a linear assignment problem, allowing an optimal assignment where each expert acquires an equal number of tokens, achieving lower validation perplexity during training relative to Switch Transformer. PanGu-Σ (Ren et al., 2023b) is a MoE-based LLM with 1.085T parameters, transitioned from the dense Transformer model to a sparse one with Random Routed Experts (RRE), and effectively trains the model over 329B tokens utilizing Expert Computation and Storage Separation (ECSS). It outperforms dense model like ERNIE 3.0 Titan Wang et al. (2021) on zero-shot test of Chinese downstream task. Lastly, Mixtral 8x7B (Jiang et al., 2023a) is a MoE with 46.7B total parameters. By leveraging the advantage of MoE architecture, Mixtral 8x7B outperforms LLaMA-2 70B on most benchmarks such as MMLU, MBPP, and GSM-8K with 6x faster inference by only using 12.9B parameters of the model per token for inference.

Algorithm-Level MoE Optimization. The efficiency of MoE-based LLMs can be improved at the algorithm level. Expert Choice (Zhou et al., 2022) allows experts to pick the top-k tokens instead of having tokens choose the top-k experts, implying that each token can be directed to a variable number of experts while each expert maintains a fixed bucket size. This method demonstrates higher performance in the GLUE and SuperGLUE benchmarks, and outperforms T5 dense model in seven out of the 11 tasks. StableMoE (Dai et al., 2022) identifies the issue of altering target experts for identical input during training and addresses this by creating two training phases. Initially, it cultivates a balanced routing strategy, which is then distilled into a decoupled lightweight router. In the following phase, this distilled router is used for a fixed token-to-expert assignment, ensuring a stable routing strategy. StableMoE shows better results than Switch Transformer and BASE Layer with lower validation perplexity on language modeling. X-MoE (Chi et al., 2022) notes that earlier routing mechanisms foster token clustering around expert centroids, indicating a tendency toward representation collapse. It proposes to estimate the routing scores between tokens and experts on a low-dimensional hyper-sphere, showing improvements over Switch Transformer on multilingual multi-task benchmark. Lifelong-MoE (Chen et al., 2023f) observes that MoE increases the capacity of the model to adapt to different corpus distributions in online data streams without extra computational cost, simply by incorporating additional expert layers and suitable expert regularization. This facilitates continuous pre-training of a MoE-based LLM on sequential data distributions without losing previous knowledge. It outperforms other MoE models such as GShard on natural language generation and understanding tasks. Lastly, Shen et al. (2024) observe that compared to dense models, MoE gains more from instruction tuning. Based on this observation, they propose Flan-MoE, which combines MoE and instruction tuning to enlarge language models without increasing demands in memory and compute resources while showing better zero-shot and few-shot performance compared to FLAN-T5 dense model.

System-Level MoE Optimization. The efficiency of MoE-based LLMs can also be improved at the system level. For example, FastMoE (He et al., 2021) is a distributed MoE training system built on PyTorch, compatible with common accelerators. This system offers a hierarchical interface that allows both flexible model design and easy adaptation to various applications, such as Transformer-XL and Megatron-LM. FasterMoE (He et al., 2022a) tries to address the challenges of dynamic load imbalance, inefficient synchronous execution mode, and congested all-to-all communication during MoE training. It first introduces a performance model that predicts latency and analyzes end-to-end performance through a roofline-like methodology. Utilizing this model, it presents a dynamic shadowing technique for load balancing, a concurrent fine-grained schedule for operations, and a strategy to alleviate network congestion by adjusting expert selection for model training. It outperforms FastMoE and achieves 1.37× - 17.87× speedup compared to methods including ZeRO, GShard, and BASE Layer. Lina (Li et al., 2023a) also improves training efficiency and inference time by optimizing communication and load balancing in distributed MoE. Lina first prioritizes all-to-all over allreduce using tensor partitioning and pipelining to improve its bandwidth in training, and then dynamically balances the workload with token-level expert selection pattern in inference. DeepSpeed-MoE (Rajbhandari et al., 2022) has designed a Pyramid-Residual MoE (PR-MoE) architecture to enhance both the training and the inference efficiency of the MoE model parameter. PR-MoE is a dense-MoE hybrid that employs residual connections to optimally utilize experts, managing to reduce the parameter size by

up to 3x without sacrificing quality or compute requirements. It serves massive MoE models with up to 4.5x faster and 9x cheaper inference compared to quality-equivalent dense models. DeepSpeed-MoE also proposes a highly optimized MoE inference system which enables efficient scaling of inference workloads on hundreds of GPUs, providing up to 7.3x reduction in inference latency and cost when compared with existing MoE inference solutions. TA-MoE (Chen et al., 2022a) highlights that current MoE dispatch patterns do not fully leverage the underlying heterogeneous network environment and thus introduces a topology-aware routing strategy for large-scale MoE training that dynamically modifies the MoE dispatch pattern based on the network topology, making it outperform FastMoE, FasterMoE, and DeepSpeed-MoE. EdgeMoE (Yi et al., 2023) presents an on-device inference engine tailored for MoE-based LLMs. It optimizes memory and computation for inference by distributing the model across different storage levels. Specifically, non-expert model weights are stored directly on the edge device, while expert weights are kept externally and only loaded into the device's memory when necessary. Tutel (Hwang et al., 2023) proposes adaptive parallelism and pipelining features to adapt to the dynamic workload. It employs a consistent layout for MoE parameters and input data, supporting switchable parallelism and dynamic pipelining without any mathematical inconsistencies or tensor migration costs, thus enabling free run-time optimization, achieving up to 5.75× speedup for a single MoE layer. SmartMoE (Zhai et al., 2023) focuses on the automatic parallelization for MoE distributed training. In the offline stage, SmartMoE constructs a search space of hybrid parallelism strategies. In the online stage, it incorporates light-weight algorithms to identify the optimal parallel strategy. It achieves up to 1.88× speedup in end-to-end training over FasterMoE on a distributed training setting. Lastly, MegaBlocks (Gale et al., 2023) transforms MoE-oriented computation with block-sparse operations and creates block-sparse GPU kernels to optimize MoE computation on hardware. This leads to training time up to 40% shorter compared to Tutel and 2.4x shorter than dense models trained with Megatron-LM.

2.5.3 Long Context LLMs

In many real-world applications, such as multi-turn conversations and meeting summarization, existing LLMs are often required to comprehend or generate context sequences that are much longer than what they have been pre-trained with and may result in a degradation in accuracy due to the poor memorization for the long context. One direct way to address this issue is to fine-tune LLMs with similar long-sequence data, which, however, is time consuming and computation-intensive. To fill this gap, new methods have been developed to enable LLMs to adapt to longer context lengths in a more efficient way. These methods can be in general grouped into four categories: extrapolation and interpolation, recurrent structure, segmentation and sliding window, and memory-retrieval augmentation.

Positional Extrapolation and Interpolation. Standard positional encoding methods such as absolute positional embeddings (APE) (Vaswani et al., 2017), learned positional embeddings (LPE) (Wang et al., 2022a), relative positional embeddings (RPE) (Shaw et al., 2018), and rotary position embeddings (RoPE) (Su et al., 2023b) have advanced the integration of positional information in LLMs. For example, LPE has been used by GPT-3 and OPT, RPE was used by Gopher (Rae et al., 2022) and Chinchilla (Hoffmann et al., 2022), whereas RoPE was used by LLaMA-1 and GLM-130B. However, it is still challenging to train LLMs on sequences with a limited maximum length while still ensuring them to generalize well on significantly longer sequences during inference. Given that, techniques based on positional extrapolation (Press et al., 2022; Sun et al., 2023c; Chen et al., 2024a) and positional interpolation (Chen et al., 2023d; Peng et al., 2024; Li et al., 2024a) have been proposed.

Positional extrapolation strategies extend the encoding of positional information beyond what the model has explicitly learned during training. For example, ALiBi (Press et al., 2022) applies attention with linear biases to attain extrapolation for sequences that exceed the maximum length seen during training. By applying negatively biased attention scores with a linearly diminishing penalty based on the distance between the pertinent key and query, it facilitates efficient length extrapolation. Different from ALiBi, xPOS (Sun et al., 2023c) characterizes attention resolution as a marker for extrapolation and utilizes a relative position embedding to enhance attention resolution, thereby improving length extrapolation. However, these techniques have not been implemented in some of the recent LLMs such as GPT-4, LLaMA, and LLaMA-2. CLEX (Chen et al., 2024a) proposes to generalize position embedding scaling with ordinary differential

equations to model continuous dynamics over length scaling factors. In doing so, CLEX gets rid of the limitations of existing positional extrapolation scaling methods to enable long-sequence generation.

Positional interpolation strategies, on the other hand, reduce the scale of input position indices and extend the context window sizes, allowing LLMs to maintain their performance over longer text sequences. For example, Chen et al. (2023d) observe that extending beyond the trained context length could impair the self-attention mechanism. They propose RoPE-PI to reduce the position indices through linear interpolation, aligning the maximum position index with the prior context window limit encountered during pre-training. NTK interpolation (bloc97, 2023) modifies the base of the RoPE, effectively changing the rotational velocity of each RoPE dimension. YaRN interpolation (Peng et al., 2024) uses a ramp function to blend linear and NTK interpolation in varying proportions across dimensions and incorporates a temperature factor to counteract distribution shifts in the attention matrix caused by long inputs. Experimental results on long-text modeling shows that YaRN outperforms existing RoPE interpolation methods including RoPE-PI and NTK. FIRE (Li et al., 2024a) proposes a functional relative position encoding using learnable mapping of input positions to biases and progressive interpolation, ensuring bounded input for encoding functions across all sequence lengths to enable length generalization. It demonstrates competitive results compared to ALiBi, RoPE, and RoPE-PI on long text benchmarks. Lastly, PoSE (Zhu et al., 2024) proposes positional skip-wise training that simulates long inputs using a fixed context window and designs distinct skipping bias terms to manipulate the position indices of each chunk. This strategy reduces memory and time consumption compared to full-length fine-tuning.

Recurrent Structure. LLMs' ability to manage long sequences can also be enhanced through recurrence structure. For example, Transformer-XL (Dai et al., 2019) presents a segment-level recurrence mechanism and utilizes enhanced relative positional encoding to capture long-term dependencies and address the long-context fragmentation issue. Memformer (Wu et al., 2022a) leverages an external dynamic memory for encoding and retrieving past information, achieving linear time and constant memory space complexity for long sequences. It also proposes Memory Replay Back-Propagation (MRBP) to facilitate long-range back-propagation through time with significantly lower memory requirements, achieving better results than Transformer-XL on language modeling and image generation benchmarks. ∞-former (Martins et al., 2022) presents a Transformer model augmented with unbounded long-term memory (LTM). It employs a continuous space attention framework to balance the quantity of information units accommodated in memory against the granularity of their representations, showing better results than Transformer-XL on long text sorting and modeling. Recurrent Memory Transformer (RMT) (Bulatov et al., 2022) uses a recurrence mechanism to retain information from the past segment level by incorporating special memory tokens into the input or output sequence, and demonstrates superior performance compared to Transformer-XL in long context modeling. Block-Recurrent Transformer (BRT) (Hutchins et al., 2022) utilizes self-attention and cross-attention to execute a recurrent function across a broad set of state vectors and tokens so as to model long sequences through parallel computation. Lastly, Retentive Network (Sun et al., 2023b) introduces a multi-scale retention mechanism as an alternative to multi-head attention. By leveraging parallel and chunk-wise recurrent representations, it enables effective scaling, achieves training parallelization and constant inference cost, and offers linear long-sequence memory complexity compared to other Transformer models.

Segmentation and Sliding Window. Segmentation and sliding window techniques tackle the issue of long-context processing by dividing the input data into smaller segments, or applying a moving window to slide through the long sequence. For instance, Mistral (Jiang et al., 2023a) uses sliding window attention to handle sequences of arbitrary length with a reduced inference cost. StreamingLLM (Xiao et al., 2024) identifies an attention sink phenomenon, noting that retaining the Key-Value of initial tokens significantly restores the performance of window attention. Based on this observation, it suggests an efficient framework via merging window context and the first token, allowing LLMs trained with a finite length attention window, but have the ability to generalize to infinite sequence lengths without any fine-tuning. Parallel Context Windows (PCW) (Ratner et al., 2023) segments a long context into chunks, limiting the attention mechanism to function only within each window, and then re-deploys the positional embeddings across these windows. LongNet (Ding et al., 2023a) proposes dilated attention, which exponentially expands the attentive field as the distance increases, enabling the handling of sequence lengths of more than one billion tokens. SLED (Ivgi et al., 2023) handles long sequences by partitioning the long text into small chunks and leveraging

pretrained short-text language models for encoding and decoding. Different from the methods mentioned above, MemWalker (Chen et al., 2023c) transforms lengthy texts into segmented summaries within a tree structure, leveraging LLMs as an interactive entity for guided reading through iterative prompts. It outperforms traditional methods based on extended context, recurrence, and retrieval. Similar to MemWalker, RAPTOR (Sarthi et al., 2024) utilizes text chunks to construct a recursive tree to enable information integration from extensive documents across various abstraction levels during inference, achieving superior performance in multi-step reasoning.

Memory-Retrieval Augmentation. Lastly, several studies tackle the inference of extremely long text by employing memory-retrieval augmentation strategies. A notable example is the Memorizing Transformer (Wu et al., 2022c), which extends the attention context size by utilizing k-nearest-neighbor (KNN) lookup to fetch previously similar context embeddings. Additionally, Landmark Attention (Mohtashami & Jaggi, 2023) employs a landmark token to represent each block of input and trains the attention mechanism to utilize it for choosing relevant blocks. This allows the direct retrieval of blocks through the attention mechanism while maintaining the random access flexibility of the previous context, demonstrating comparable perplexity as Transformer-XL while reducing FLOPs for long-context modeling. LongMem (Wang et al., 2023e) proposes a decoupled network architecture with the original backbone LLM as a memory encoder and an adaptive residual side network as a memory retriever and reader, efficiently caching and updating long-term past contexts to prevent knowledge staleness. It outperforms Memorizing Transformer on long text modeling and natural language understanding tasks. Unlimiformer (Bertsch et al., 2023) enhances the KNN-augmented Transformer by outputting attention dot-product scores as KNN distances to enable indexing of virtually unlimited input sequences. Experimental results show that Unlimiformer outperforms Memorizing Transformer on long document summarization benchmarks. Tworkowski et al. (2023) observe that the ratio of relevant keys to irrelevant ones diminishes as context length increases. Based on this observation, they propose Focused Transformer (FoT), a contrastive learning-based technique to refine the structure of the Key-Value space. Unlike Memorizing Transformer and Transformer-XL, FoT does not require training on long sequences and shows better performance on both long context and short context tasks. Lastly, Xu et al. (2024a) discover that an LLM with a 4K context window, when augmented with simple retrieval during generation, can match the performance of a fine-tuned LLM with a 16K context window using positional interpolation (Chen et al., 2023d) on long context tasks while requiring significantly less computation.

2.5.4 Transformer-Alternate Architectures

While Transformer-based architectures are now at the forefront of LLMs, some studies propose new architectures to supplant Transformer-based architectures.

State Space Models. A promising approach that aims to substitute the attention mechanism is state space models (SSMs). SSM is formulated as $x'(t) = Ax(t) + Bu(t)$, $y(t) = Cx(t) + Du(t)$, which maps a single-dimension input signal $u(t)$ to an N-dimension latent state $x(t)$ before projecting to a single-dimension output signal $y(t)$, where A, B, C, D are parameters learned by gradient descent (Gu et al., 2022a). Compared to attention that has quadratic complexity, SSMs provide near-linear computational complexity related to the length of the sequence. Given such advantage, a series of techniques have been proposed to improve SSMs. For example, the Structured State Space (S4) sequence model (Gu et al., 2022a) refines SSMs by conditioning matrix A with a low-rank correction. This enables stable diagonalization and simplifies the SSM to the well-studied computation of a Cauchy kernel. Diagonal State Space (DSS) (Gupta et al., 2022) improves SSMs by proposing fully diagonal parameterization of state spaces instead of a diagonal plus low rank structure. To bridge the gap between SSMs and attention while adapting to modern hardware, H3 (Hungry Hungry Hippo) (Fu et al., 2023a) stacks two SSMs to interact with their output and input projection, allowing it to log tokens and facilitate sequence-wide comparisons. Mehta et al. (2023) introduce a more efficient layer called Gated State Space (GSS), which has been empirically shown to be 2 to 3 times faster than DSS while maintaining the perplexity on multiple language modeling benchmarks. Block-State Transformer (BST) (Pilault et al., 2023) designs a hybrid layer that combines an SSM sublayers for extended range contextualization with a Block Transformer sublayer for short-term sequence representation. Gu & Dao (2023) propose Mamba to enhance SSMs by designing a selection mechanism to eliminate irrelevant data and develop a hardware-aware parallel algorithm for recurrent operation, achieving 5x throughput than

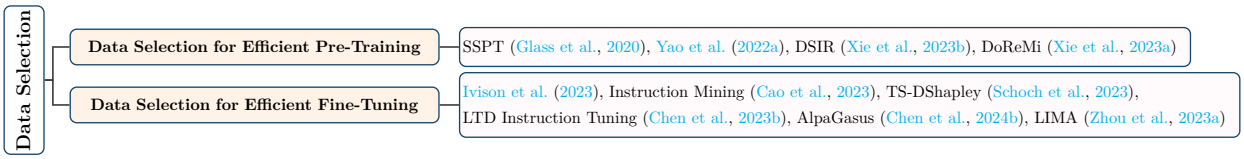

Figure 16: Summary of data selection techniques for LLMs.

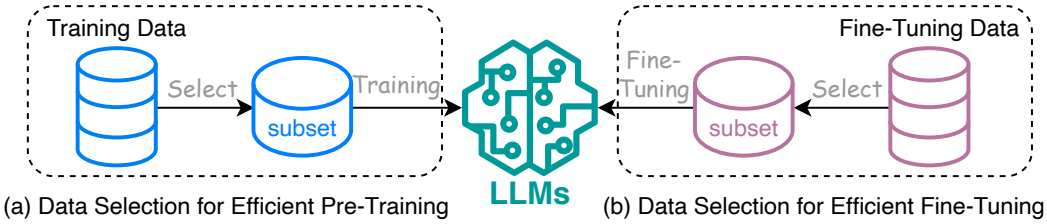

Figure 17: Illustrations of data selection techniques for LLMs.

Transformers. Ren et al. (2023a) extend SSMs and propose a general modular activation mechanism named Sparse Modular Activation (SMA), which unifies MoE, adaptive computation, dynamic routing and sparse attention, and further applies SMA to develop a novel architecture, SeqBoat, to achieve state-of-the-art quality-efficiency trade-off.

Other Sequential Models. Some other architectures have been proposed to replace the Transformer layer. For instance, Receptance Weighted Key Value (RWKV) model (Peng et al., 2023b) combines the advantages of recurring neural networks (RNN) and Transformers. Such combination utilizes the effective parallelizable training feature of Transformers coupled with the efficient inference ability of RNNs, thereby effectively tackling the challenges associated with long sequence processing, outperforming Transformer-based models such as BLOOM and OPT. Poli et al. (2023) propose Hyena, a sub-quadratic alternative to the attention mechanism to mitigate the quadratic cost in long sequences. This operator includes two efficient sub-quadratic primitives: an implicit long convolution and multiplicative element-wise gating of the input. Hyena facilitates the development of larger, more efficient convolutional language models for long sequences and outperforms RWKV and GPT-Neo on SuperGLUE tasks (Wang et al., 2019). Lastly, MEGABYTE (YU et al., 2023) breaks down long sequences into fixed-sized patches akin to tokens, comprising a patch embedder for encoding, a global module acting as a large autoregressive Transformer for patch representations, and a local module for predicting bytes within a patch.

3 Data-Centric Methods

3.1 Data Selection

Data selection is a fundamental technique for enhancing efficiency. As summarized in Figure 16, in the context of LLMs, data selection techniques have been primarily used for enhancing the efficiency of pre-training and fine-tuning.

3.1.1 Data Selection for Efficient Pre-Training

Data selection enhances LLMs pre-training efficiency by strategically selecting informative and diverse data samples during training. For example, SSPT (Glass et al., 2020) is a pre-training task based on the principles of reading comprehension. It involves selecting answers from contextually relevant text passages, which has shown notable improvements in performance across various Machine Reading Comprehension benchmarks. Yao et al. (2022a) propose a meta-learning-based method for selecting linguistically informative sentences which significantly elevates the quality of machine-generated translations. Xie et al. (2023b) propose DSIR,

a data selection method based on importance resampling for both general-purpose and specialized LLMs. It calculates how important different pieces of data are within a simpler set of features and chooses data based on these importance calculations. Experimental results demonstrate that DSIR achieves similar performance to expert curation across eight different target distributions. In the context of pre-training general-domain models, DSIR outperforms random selection and heuristic filtering baselines by 2–2.5% on the GLUE benchmark. Different from DSIR, Xie et al. (2023a) design DoReMi to address the distribution shift between pre-training and downstream tasks, which is also a critical problem for training data selection.

3.1.2 Data Selection for Efficient Fine-Tuning

Data selection can also boost LLM fine-tuning efficiency given that only a curated subset of examples is employed to refine the model. For example, Ivison et al. (2023) propose to use a few unlabeled data samples to retrieve similar labeled ones from a larger multitask dataset, improving task-specific model training. This method outperforms standard multitask data sampling for fine-tuning and enhances few-shot fine-tuning, yielding an 2-23% relative improvement. With the success of instruction tuning, many studies start focusing on the selection of high-quality instruction data to fine-tune LLMs. For example, Instruction Mining (Cao et al., 2023) presents a linear evaluation method to assess data quality in instruction-following tasks. It highlights the importance of high-quality data, showing that models trained with Instruction Mining-curated datasets outperform those trained on generic datasets in 42.5% of the considered cases. TS-DShapley (Schoch et al., 2023) is introduced to address the computational challenges of applying Shapley-based data valuation to fine-tuning LLMs. It employs an efficient sampling-based method that aggregates Shapley values computed from subsets to evaluate the entire training set. Low Training Data Instruction Tuning (LTD Instruction Tuning) (Chen et al., 2023b) challenges the need for large datasets in fine-tuning, showing that less than 0.5% of the original dataset is able to effectively train task-specific models without compromising performance. This approach enables more resource-efficient practices in data-scarce environments, combining selective data strategies with tailored training protocols for optimal data efficiency. AlpaGasus (Chen et al., 2024b) is a model fine-tuned on a mere 9K high-quality data samples, which are meticulously filtered from a larger dataset of 52K. It outperforms the original model trained on the full dataset and reduces the fine-tuning time by 5.7x, demonstrating the power of high-quality data in instruction-fine-tuning. Lastly, LIMA (Zhou et al., 2023a) fine-tunes LLMs with a small, selected set of examples, showing strong performance and challenging the need for extensive tuning. It generalizes well to new tasks, matching or exceeding GPT-4 in 43% of the considered cases.

3.2 Prompt Engineering

Prompt engineering (Liu et al., 2023a) focuses on designing effective inputs (i.e., prompts) to guide LLMs in generating desired outputs. It enhances inference efficiency by tailoring the input prompts or queries to better suit the capabilities of a specific language model. When used for some simple tasks, such as semantic classification, prompt engineering can even substitute fine-tuning to achieve high accuracy (Liu et al., 2022a). As summarized in Figure 18, prompt engineering techniques can in general be grouped into few-shot prompting, prompt compression, and prompt generation.

3.2.1 Few-Shot Prompting

Few-shot prompting aims to provide a LLM with a limited set of examples, referred to as demonstrations, to steer its understanding to a task it is required to execute (Wei et al., 2022a). These demonstrations are selected from the training corpus based on their similarity to the test example, and the LLM is expected to use the knowledge gained from these similar demonstrations to make the correct prediction (Dong et al., 2023). Few-shot prompting provides an efficient mechanism to use LLM by guiding the LLM to perform a wide variety of tasks without the need for additional training or fine-tuning. Furthermore, an effective few-shot prompting approach can make the created prompt concise enough to allow LLMs to quickly adjust to the task in high accuracy with only a slight increase of extra context, thus significantly improving inference efficiency. As illustrated in Figure 19, few-shot prompting techniques can in general be grouped into demonstration organization and template formatting.

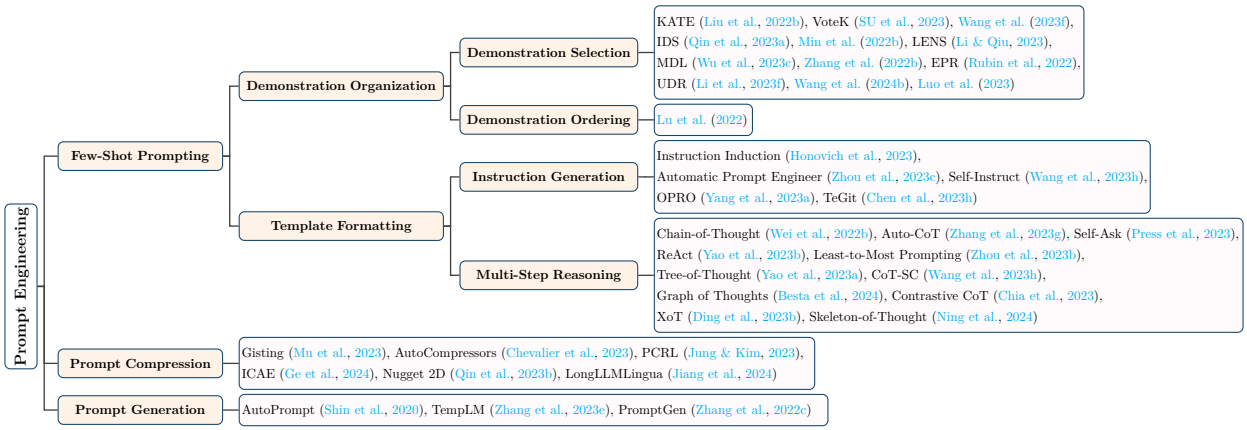

Figure 18: Summary of prompt engineering techniques for LLMs.

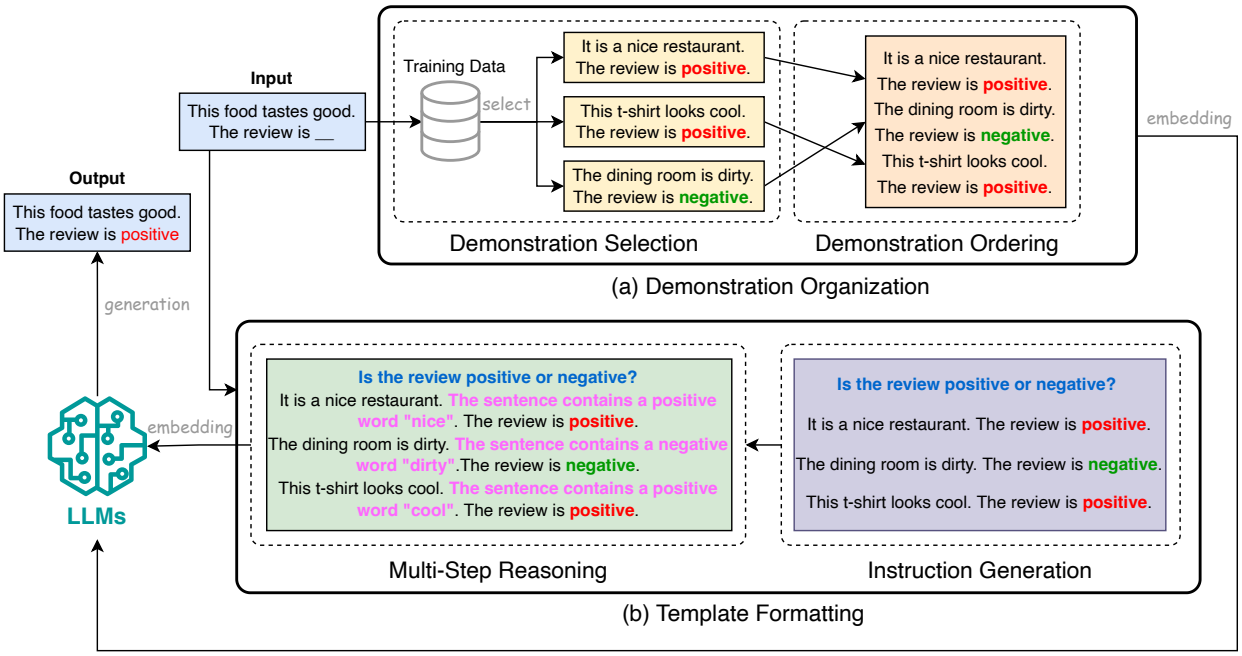

Figure 19: Illustrations of few-shot prompting techniques for LLMs.

Demonstration Organization. Demonstration organization refers to organizing the demonstrations in an appropriate way so as to form a suitable prompt for inference. Demonstration organization has a significant impact on the inference efficiency since improper organization may result in the processing of a considerable amount of unnecessary information, leading to significant slowdown. The optimization of demonstration organization comes from its two main steps: demonstration selection and demonstration ordering.

- ***Demonstration Selection.*** Demonstration selection aims to choose the good examples for few-shot prompting (Dong et al., 2023). In order to generate a satisfactory result, a good selection of demonstrations may only require a few number of demonstrations to be used for the prompt, thus making the prompt concise and straightforward for a more efficient inference. Existing demonstration selection techniques can be grouped into unsupervised methods (Liu et al., 2022b; SU et al., 2023; Wang et al., 2023f; Qin et al., 2023a; Min et al., 2022b; Li & Qiu, 2023; Wu et al., 2023c; Zhang et al., 2022b) and supervised methods (Rubin et al., 2022; Li et al., 2023f; Wang et al., 2024b; Luo

et al., 2023). Unsupervised methods aim to select the nearest examples from the training set using a predefined similarity function, such as L2 distance, cosine distance, and minimum description length (MDL) (Wu et al., 2023c). For example, KATE (Liu et al., 2022b) is an unsupervised selection method that directly uses the nearest neighbors of a given test sample as the corresponding demonstrations. VoteK (SU et al., 2023) is an improved version of KATE to resolve its limitation that requires a large set of examples to achieve good performance. Unlike KATE, VoteK increases the diversity of the demonstrations by penalizing examples similar to those already selected. In contrast, supervised methods require training a domain-specific retriever from the training set and using it for demonstration selection. For example, EPR (Rubin et al., 2022) is trained to select demonstrations from a small set of candidates initialized by the unsupervised retriever such as BM25 from the training corpus. UDR (Li et al., 2023f) further enhances EPR by adopting a unified demonstration retriever to unify the demonstration selection across different tasks. Compared to unsupervised methods, supervised methods often lead to a more satisfying generation result but require frequent adjustment of the retriever for handling the out-of-domain data, making them relatively less efficient for inference.

- ***Demonstration Ordering.*** After selecting representative samples from the training set, the next step is ordering these samples in the prompt. The order of the demonstrations also has a significant impact on the performance of the model. Therefore, selecting the right order of demonstrations can help the model quickly reach a good generation quality with fewer samples, thus improving the inference efficiency. To date, only a few studies have delved into this area. For example, Liu et al. (2022b) suggest arranging demonstrations based on their distance from the input, placing the closest demonstration furthest to the right. Lu et al. (2022) propose to develop both global and local entropy metrics and use the entropy metrics to set up the demonstration order.

Template Formatting. Template formatting aims to design a suitable template to form the prompt. A good template typically compiles all the information needed by LLMs into a brief statement, making the prompt and the entire input context as succinct as possible, thus leading to a higher inference efficiency. Template formatting can be divided into two parts: instruction generation and multi-step reasoning.

- ***Instruction Generation.*** The instruction of the template refers to a short description of the task. By adding instructions to the prompt, LLMs can quickly understand the context and the task they are currently performing, and thus may require fewer demonstrations to create a desirable prompt. The performance of a given task is highly affected by the quality of the instructions. The instructions vary not only between different datasets for the same task but also between different models. Unlike demonstrations that are usually included in traditional datasets, the generation of instructions is heavily dependent on human efforts. To enhance the efficiency of instruction generation, automatic instruction generation techniques have been proposed. For example, Instruction Induction (Honovich et al., 2023) and Automatic Prompt Engineer (Zhou et al., 2023c) demonstrate that LLMs can generate task instructions. Wang et al. (2023h) propose Self-Instruct, an approach that allows LLMs to align with self-generated instructions, highlighting their inherent adaptability. Experimental results show that it achieves an 33% improvement over the original model when applied on vanilla GPT3. Yang et al. (2023a) also discover that LLMs can be treated as an optimizer to iteratively generate better instructions for the target LLM and have applied this technique to various LLMs. Experiments demonstrate that the best prompts optimized by this method outperform human-designed prompts by up to 8% on GSM8K, and by up to 50% on Big-Bench Hard tasks. Chen et al. (2023h) develop TeGit for training language models as task designers, which can automatically generate inputs and outputs together with high-quality instructions to better filter the noise based on a given human-written text for fine-tuning LLMs. Despite the promise of automatic instruction generation methods, their complexity is still a major bottleneck for their real-world adoption.

- ***Multi-Step Reasoning.*** Multi-step reasoning (Huang & Chang, 2023) refers to techniques that guide the LLMs to produce a sequence of intermediate steps before outputting the final answer. Compared to fine-tuning, conducting specific task reasoning directly through this way is a more efficient approach. At the same time, its accuracy improvement is often not as good as fine-tuning. One

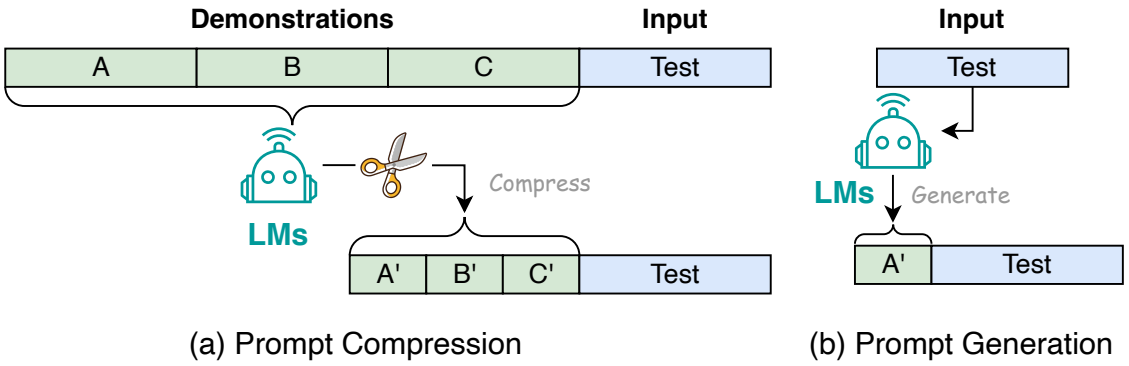

Figure 20: Illustrations of prompt compression (a) and prompt generation (b) for LLMs.

of the widely used techniques in multi-step reasoning is Chain-of-Thought (CoT) prompting (Wei et al., 2022b), which adds a series of reasoning steps into the prompt to make it more informative and comprehensive. Despite its advantages, it is still difficult for CoT to ensure the accuracy of every intermediate step (Dong et al., 2023). A number of techniques have been proposed to address this issue. For example, Auto-CoT (Zhang et al., 2023g) proposes to generate the CoT step by step from LLMs. Self-Ask (Press et al., 2023) incorporates the self-generated questions of each step into CoT. ReAct (Yao et al., 2023b) performs dynamic reasoning to create, maintain, and adjust high-level plans for acting, while interacting with external environments to incorporate additional information into reasoning. Least-to-Most Prompting (Zhou et al., 2023b) is a new milestone in CoT. It breaks down the complex question into smaller ones and answers them iteratively within the context of former questions and answers. Experimental results show that Least-to-Most Prompting can boost the accuracy of GPT-3 code-davinci-002 model with CoT prompting to 99.7% by using just 14 exemplars. Tree-of-Thought (ToT) (Yao et al., 2023a) expends CoT to include exploration over coherent units of text and deliberates decision-making processes. It outperforms GPT-4 with CoT prompting in the Game of 24 benchmark. CoT-SC (Wang et al., 2023h) introduces a novel decoding approach called self-consistency to replace the greedy decoding in CoT prompting. It starts by sampling various reasoning paths instead of just the greedy one and then determines the most consistent answer by considering all the sampled paths. Graph of Thoughts (GoT) (Besta et al., 2024) represents information produced by an LLM as a generic graph, with "LLM thoughts" as vertices and edges indicating dependencies between these vertices. Experimental results show that GoT increases the quality of sorting by 62% over ToT while reducing the cost by over 31%. Contrastive CoT (Chia et al., 2023) proposes to enhance language model reasoning by providing both valid and invalid reasoning demonstrations. XoT (Ding et al., 2023b) utilizes pretrained reinforcement learning and Monte Carlo Tree Search (MCTS) to integrate external domain knowledge into the thought processes of LLMs, thereby boosting their ability to efficiently generalize to new, unseen problems. Lastly, Skeleton of Thought (SoT) (Ning et al., 2024) proposes a method that first prompts the LLM to organize the output and then parallelizes the generation of different segments. Through splitting the serial generation into two different steps, it makes the generation workload more parallelizable. Therefore, it improves hardware utilization and provides speedups across LLM, revealing the possibility of exploiting data-level organization to enhance efficiency.

3.2.2 Prompt Compression

Prompt compression (Figure 20(a)) accelerates the processing of LLM inputs through either condensing lengthy prompt inputs or learning compact prompt representations. Mu et al. (2023) propose to train LLMs to distill prompts into a more concise set of tokens, referred to as gist tokens. These gist tokens encapsulate the knowledge of the original prompt and can be stored for future use. In doing so, it is able to compress prompts by up to 26x, leading to a reduction in floating-point operations per second (FLOPs) by up to

40%. Chevalier et al. (2023) propose AutoCompressors to condense long textual contexts into compact vectors, known as summary vectors, which can then be used as soft prompts for the language model. These summary vectors extend the model's context window, allowing it to handle longer documents with much less computational cost. In particular, AutoCompressors can utilize long contexts to improve perplexity of both fine-tuned OPT and LLaMA-2 on sequences of up to 30,720 tokens. Jung & Kim (2023) propose Prompt Compression with Reinforcement Learning (PCRL) that employs a policy network to directly edit prompts, aiming to reduce token count while preserving performance. It achieves an average reduction of 24.6% in token count across various instruction prompts. Ge et al. (2024) propose In-context Autoencoder (ICAE), which consists of a learnable encoder and a fixed decoder. The encoder compresses a long context into a limited number of memory slots, which the target language model can then condition on. With such design, ICAE is able to obtain 4x context compression. Nugget 2D (Qin et al., 2023b) represents the historical context as compact "nuggets" that are trained to enable reconstruction. Furthermore, it has the flexibility to be initialized using readily available models like LLaMA. Eperimental results show that Nugget 2D compresses context at a 20x compression ratio with a BLEU score of 98% for reconstruction, achieving nearly lossless encoding. Lastly, LongLLMLingua (Jiang et al., 2024) introduces a prompt compression technique containing question-aware coarse-to-fine compression, document reordering, dynamic compression ratios, and post-compression sub-sequence recovery to enhance LLMs' key information perception. Experimental results show that LongLLMLingua achieves 17.1% better performance over the original prompt with 4x fewer tokens as input to GPT-3.5-Turbo.

3.2.3 Prompt Generation

Prompt generation (Figure 20(b)) enhances efficiency by automatically creating effective prompts that guide the model in generating specific and relevant responses. For instance, AutoPrompt (Shin et al., 2020) proposes an automated method to generate prompts for a diverse set of tasks based on a gradient-guided search. It underscores the significance of human-written text in refining the quality and authenticity of data, emphasizing its pivotal role in optimizing LLM performance. Experimental results demonstrate that AutoPrompt outperforms manually created prompts on the LAMA benchmark in eliciting more precise factual knowledge from LLM. TempLM (Zhang et al., 2023e) proposes to combine generative and template-based methodologies to distill LLMs into template-based generators, offering a harmonized solution for data-to-text tasks. TempLM not only reduces the unfaithfulness rate of a fine-tuned BART model from 83% to 0%, but also substantially improves upon human-written ones in BERTScore. Lastly, PromptGen (Zhang et al., 2022c) considers dynamic prompt generation for knowledge probing based on pre-trained LLMs. It automatically generates prompts conditional on the input sentence and outperforms AutoPrompt on the LAMA benchmark.

4 LLM Frameworks

LLM frameworks can be in general grouped based on whether they support the tasks of training, fine-tuning, and inference. Specifically, frameworks that support training and/or fine-tuning aim to provide scalable, efficient, and flexible infrastructure that improves computation efficiency, reduces memory footprint, optimizes communication efficiency, and ensures reliability of the training/fine-tuning process. Frameworks that support inference focus on optimizing inference throughput and reducing memory footprint and latency. These frameworks offer a variety of deployment features to serve LLM requests. Table 2 provides a summary of existing LLM frameworks along with their key features.

DeepSpeed. Developed by Microsoft, DeepSpeed (Rasley et al., 2020) is an integrated framework for both training and serving LLMs. It has been used to train large models like Megatron-Turing NLG 530B (Smith et al., 2022) (in a joint effort with Nvidia Megatron framework) and BLOOM (Scao et al., 2023). Within this framework, DeepSpeed-Inference is the foundational library. A pivotal feature of DeepSpeed-Inference is ZeRO-Inference (Rajbhandari et al., 2020; 2021), an optimization technique created to address GPU memory constraints for large model inference. ZeRO-Inference distributes model states across multiple GPUs and CPUs, providing an approach to managing the memory constraints of GPUs.

Table 2: Comparison of LLM frameworks.

Framework	Training	Fine-Tuning	Inference	Key Features
DeepSpeed	✔	✔	✔	3D Parallelism with ZeRO (Rasley et al., 2020), ZeRO-2 (Rajbhandari et al., 2020), ZeRO Infinity (Rajbhandari et al., 2021) and ZeRO-Offload (Ren et al., 2021), Expert Parallelism (Rajbhandari et al., 2022), FlashAttention (Dao et al., 2022), PagedAttention (Kwon et al., 2023), Dynamic SplitFuse (Holmes et al., 2024), Continuous Batching (Yu et al., 2022), ZeroQuant (Wu et al., 2023b; Yao et al., 2023d), INT4 Quantization (Wu et al., 2023a), XTC (Ternary quantization) (Wu et al., 2022b), RLHF (Yao et al., 2023c), Kernel Optimizations, Diverse Hardware Support.
Megatron	✔	✔	✔	3D Parallelism (Shoeybi et al., 2020; Narayanan et al., 2021), Sequence Paralellism (Korthikanti et al., 2023; Li et al., 2023d), Expert Parallelism (Singh et al., 2023), FasterTransformer (NVIDIA, 2023a), FlashAttention (Dao et al., 2022), Selective Activation Recomputation (Korthikanti et al., 2023).
Colossal-AI	✔	✔	✔	3D Parallelism (Xu & You, 2023; Wang et al., 2023a; Bian et al., 2021), Sequence Parallelism (Li et al., 2023d), ZeRO Optimizer (Zhao et al., 2022), Auto-Parallelism (Liu et al., 2023c), Heterogeneous Memory Management (Fang et al., 2023), Expert Parallelism (Singh et al., 2023; Xue et al., 2023), PagedAttention (Kwon et al., 2023), FlashAttention-2 (Dao, 2024), Quantization (GPTQ (Frantar et al., 2023) and SmoothQuant (Xiao et al., 2023)), RLHF (Griffith et al., 2013; Singh et al., 2023).
Nanotron	✔	✔	✔	3D Parallelism (Narayanan et al., 2021; Huang et al., 2019), Expert Parallelism (Singh et al., 2023), ZeRO Optimizer (Zhao et al., 2022), SSM (Gu & Dao, 2023) Support, Spectral µTransfer Parametrization (Yang et al., 2022), DoReMi (Xie et al., 2023a).
MegaBlocks	✔	✔	✔	FSDP (Zhao et al., 2023c), dropless-MoE (Gale et al., 2023), Integration with Megatron and vLLM.
FairScale	✔	✔	✔	FSDP (Zhao et al., 2023c), Pipeline Parallelism (Huang et al., 2019), AdaScale optimizer (Johnson et al., 2020).
Pax	✔	✔	✔	Data Parallelism, Tensor Parallelism (Shoeybi et al., 2020).
Composer	✔	✔	✔	FSDP (Zhao et al., 2023c), Elastic Sharded Checkpointing.

Continued on next page

Table 2: Comparison of LLM frameworks. (Continued)

OpenLLM	✗	✓	✓	FSDP (Zhao et al., 2023c), Quantization (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023), SqueezeLLM (Kim et al., 2024), SpQR (Dettmers et al., 2024), LLM.int8 (Dettmers et al., 2022)), LangChain, Transformers Agents, Prometheus Metrics.
LLM Foundry	✗	✓	✓	FSDP (Zhao et al., 2023c), Continuous Batching (Yu et al., 2022).
vLLM	✗	✗	✓	Data Parallelism, Tensor Parallelism (Shoeybi et al., 2020), PagedAttention (Kwon et al., 2023), Continuous Batching (Yu et al., 2022), Quantization (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023), SqueezeLLM (Kim et al., 2024)), Multi-LoRa (Wang et al., 2023g).
TensorRT-LLM	✗	✗	✓	3D Parallelism, PagedAttention (Kwon et al., 2023), Continuous Batching (Yu et al., 2022), Quantization (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023), SmoothQuant (Xiao et al., 2023)).
TGI	✗	✗	✓	Data Parallelism, Tensor Parallelism (Shoeybi et al., 2020), PagedAttention (Kwon et al., 2023), Continuous Batching (Yu et al., 2022), Quantization (BitsAndBytes Dettmers (2023), GPTQ (Frantar et al., 2023)).
RayLLM	✗	✗	✓	Data Parallelism, Tensor Parallelism (Shoeybi et al., 2020), Continuous Batching (Yu et al., 2022), Quantization (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023), SqueezeLLM (Kim et al., 2024)), Prometheus Metrics.
MLC LLM	✗	✗	✓	TVM-based Compiler Acceleration (Feng et al., 2023; Chen et al., 2018), Continuous Batching (Yu et al., 2022), Quantized Model Support.
Sax	✗	✗	✓	Data Parallelism, Tensor Parallelism (Shoeybi et al., 2020), Serves Pax, JAX, and PyTorch models, Slice Serving, Prometheus Metrics.
Mosec	✗	✗	✓	Data Parallelism, Continuous Batching (Yu et al., 2022), Rust-based Task Coordinator, Prometheus Metrics.

Another important feature of DeepSpeed-Inference is its deep fusion mechanism, which allows for the fusion of operations without the necessity for global synchronization by tiling computations across iteration space dimensions (Ren et al., 2021; Tang et al., 2021; Li et al., 2022; Lu et al., 2023). Building on this, the DeepSpeed Model Implementations for Inference (DeepSpeed MII) module introduces Dynamic SplitFuse (Holmes et al., 2024), which leverages continuous batching (Yu et al., 2022) and non-contiguous KV caches to enable increased occupancy and higher responsivity for LLM serving. It also supports scaling beyond tensor, data, and pipeline parallelism (3D Parallelism) with Zero Infinity (Rajbhandari et al., 2021) and efficient post-training quantization to Int8 with ZeroQuant (Yao et al., 2022b), Int4 (W4A4) with Wu et al. (2023a) and ternary quantization with XTC (Wu et al., 2022b). Furthermore, the introduction of DeepSpeed-Chat (Yao et al., 2023c) adds chat support to the framework. This module focuses on training chatbot models across different scales, integrating techniques like Reinforcement Learning from Human Feedback (RLHF) (Griffith et al., 2013) with the DeepSpeed training system. Notably, its integration of the

ZeRO-Offload optimizer (Ren et al., 2021) facilitates training on both CPUs and GPUs, irrespective of their memory capacities.

Megatron. Megatron (Shoeybi et al., 2020) is Nvidia's efforts to streamline training and serving of LLMs such as GPT (Radford et al., 2019) and T5 (Raffel et al., 2020). It is the underlying framework used for Nvidia Megatron models (Shoeybi et al., 2020; Narayanan et al., 2021; Korthikanti et al., 2023). Central to Megatron's design is the strategic decomposition of the model's tensor operations, distributed across multiple GPUs, to optimize both processing speed and memory utilization, thus enhancing training throughput without compromising model quality (Shoeybi et al., 2020). In conjunction with 3D Parallelism, it also implements sequence parallelism and selective activation recomputation (Korthikanti et al., 2023), which enhances training efficiency. Megatron also uses FasterTransformer (NVIDIA, 2023a) for optimizing the inference process for large Transformer models and handling varying precision modes like FP16 and INT8, catering to diverse operational needs.

Colossal-AI. Colossal-AI (Li et al., 2023c) is a framework mainly designed for large-scale distributed training (Wang et al., 2023a). Colossal-AI unifies a wide range of parallelism techniques (Xu & You, 2023; Wang et al., 2023a; Bian et al., 2021) including sequence parallelism (Li et al., 2023d), auto-parallelism (Liu et al., 2023c), and Zero Redundancy Optimizer (Zhao et al., 2022). It also implements heterogeneous memory management (Fang et al., 2023) through a streamlined API. This integrated approach mitigates the steep learning curve often associated with orchestrating large-scale training in distributed environments. In addition, the framework integrates several other features like quantization (Frantar et al., 2023; Xiao et al., 2023), RLHF (Griffith et al., 2013), OpenMoE, and mixed-precision training.

Nanotron. Nanotron (HuggingFace, 2023a), introduced by Huggingface, is an LLM training framework with a primary focus on providing functionality with minimal overhead. As such, the library only subjectively incorporates the best optimizations required for modern LLM training requirements. Some highlights are its implementation of tensor, data, and pipeline parallelism with one-forward-one-backward pipeline engine, ZeRO-1 optimizer (Zhao et al., 2022), FP32 gradient accumulation, parameter tying/sharding and spectral µTransfer parametrization (Yang et al., 2022) for scaling up neural networks. Nanotron also incorporates DoReMi (Xie et al., 2023a) to further speed up training.

MegaBlocks. Developed by Databricks, MegaBlocks (Gale et al., 2023) is an LLM training framework for training Mixture-of-Experts (MoE) models. The core of MegaBlocks is dropless-MoE, a reformulation of MoE in terms of block-sparse operations, that avoids token dropping without sacrificing hardware efficiency. This design simplifies and accelerates training as it does not require the capacity factor as a hyper-parameter.

FairScale. Developed by Meta, FairScale (FairScale authors, 2021) is an extension library to PyTorch, dedicated to high-performance and large-scale training. As a highlight, FairScale uses Fully Sharded Data Parallel (FSDP) (Zhao et al., 2023c) as the preferred method for scaling the training operations of large neural networks. It uses AdaScale optimizer (Johnson et al., 2020) as its distributed optimizer.

Pax. Developed by Google, Pax (Google, 2023a) is a JAX-based distributed training framework. Pax has been used to train PaLM-2 (Anil et al., 2023) and Bard (Hsiao et al., 2023). It targets scalability and has reference examples for large model training, including across modalities (e.g., text, vision, speech). It supports data and tensor parallelism, and is heavily integrated with JAX and uses many libraries in the JAX ecosystem. Several key components Pax contains include SeqIO to handle sequential data processing, Optax for optimization, Fiddle for configuration, Orbax for checkpointing, PyGLove for automatic differentiation, and Flax for creating high-performance neural networks.

Composer. Composer (MosaicML, 2023a) is an LLM framework designed by Mosaic ML. It has been used to train Mosaic ML's MPT 7B and MPT 30B models and Replit's Code V-1.5 3B. The framework is built on top of PyTorch and provides a collection of acceleration methods that users can incorporate into their training loops or use with the Composer trainer. Composer supports FSDP for efficient parallelism, elastic shared checkpointing for robust intermittent training, and a dataset streaming implementation allowing the download of datasets from cloud blob storage on the fly during training. Composer also provides a functional API for integrating methods directly into its training loops, as well as a Trainer API which automatically implements a PyTorch-based training loop, reducing the workload for ML developers.

OpenLLM. OpenLLM (Pham et al., 2023) was made by BentoML to serve and fine-tune LLMs within production environments. Anchored within the BentoML ecosystem, OpenLLM makes it easy to self-host LLMs and integrate them with other cloud services. OpenLLM emphasizes on flexibility, SOTA LLM support, and streamlined APIs for self-hosting. Recognizing the diverse needs of production environments, OpenLLM supports the automatic generation of docker images. It also supports serving models as serverless endpoints using BentoML's cloud platform. OpenLLM further integrates advanced quantization techniques (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023), SqueezeLLM (Kim et al., 2024), SpQR (Dettmers et al., 2024), LLM.int8 (Dettmers et al., 2022)) for efficient inference. OpenLLM's design also incorporates robust monitoring with Prometheus metrics and logging tools, ensuring that operational insights are readily available for performance tuning and troubleshooting.

LLM Foundry. LLM Foundry (MosaicML, 2023b) is a library developed by MosaicML for fine-tuning, evaluating, and serving LLMs. It supports distributed inference via FSDP and continuous batching (Yu et al., 2022) for efficient serving. It also has cloud integration with the MosaicML platform.

vLLM. vLLM (Kwon et al., 2023) is an open-source library for LLM inference and serving. It adopts a different design in how KV cache is stored in memory. Central to this design is PagedAttention, a mechanism that segments the attention key and value (KV) cache for a set number of tokens. Unlike contiguous space storage, PagedAttention's blocks for the KV cache are stored flexibly, similar to the virtual memory management in operating systems. This facilitates memory sharing at a block level across various sequences tied to the same request or even different requests, thus enhancing memory management efficiency in handling attention mechanisms. Hence, vLLM can reduce the total memory usage when using complex decoding techniques such as parallel sampling and beam search as the memory blocks can be shared across different candidate samples. It also allows on-demand buffer allocation, while also eliminating external fragmentation as the blocks are uniformly sized. Furthermore, vLLM incorporates safeguards that prevent GPU memory overflow due to the increasing size of KV cache by evicting and recovering blocks as needed via swapping and recomputation. vLLM also supports Multi-LoRA (Wang et al., 2023g), continuous batching (Yu et al., 2022) and quantization (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023) and SqueezeLLM (Kim et al., 2024)). Lastly, it implements efficient kernels for both Nvidia and AMD GPUs.

TensorRT-LLM. TensorRT-LLM (NVIDIA, 2023b) is a streamlined library to serve LLMs. Built on top of the TensorRT engine, TensorRT-LLM integrates optimized kernels from FasterTransformer (Timonin et al., 2022) and employs tensor parallelism, facilitating efficient inference at scale across multiple GPUs and servers without necessitating developer intervention or model changes. It also integrates seamlessly with the Nvidia Triton Inference Server for serving LLMs. Additionally, it offers features like tensor parallelism, continuous batching (Yu et al., 2022), and PagedAttention (Kwon et al., 2023) as well as supports a wide range of quantization modes and techniques.

Text-Generation-Inference (TGI). Text-Generation-Inference (TGI) (HuggingFace, 2023b) is a high-performance LLM serving library developed by Huggingface and is used to power Hugging Chat. TGI supports a large variety of LLMs, and offers a wide range of features like tensor parallelism, continuous batching (Yu et al., 2022), efficient attention mechanisms like FlashAttention (Dao et al., 2022) and PagedAttention (Kwon et al., 2023), and quantization (BitsAndBytes (Dettmers, 2023), GPTQ (Frantar et al., 2023)) support.

RayLLM. RayLLM (Ray Project, 2023) is an LLM serving framework as part of the Ray ecosystem (Moritz et al., 2018). Built with Ray Serve and the Ray library developed by AnyScale, RayLLM eases LLM serving in multi-GPU and multi-node systems. At the core of RayLLM is the leveraging of Ray's inherent distributed computing capabilities. RayLLM integrates Ray's distributed task scheduling and execution mechanisms, ensuring that LLM tasks are efficiently distributed across available resources. RayLLM also simplifies adding new LLMs for custom use cases, and offers auto-scaling support and high-performance features like continuous batching (Yu et al., 2022), quantization (GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2023), SqueezeLLM (Kim et al., 2024)), and monitoring via Prometheus metrics. It also comes with advanced monitoring support as well which includes a CLI and a web frontend that can be used to compare and rank models and get cost and latency estimates.

MLC LLM. MLC LLM (Machine Learning Compilation for Large Language Models) (MLC-LLM, 2023) allows individuals to develop, optimize, and deploy LLMs on a wide range of platforms such as mobile phones and web browsers. Central to MLC LLM is its focus on machine learning compilation techniques. MLC-LLM compiles LLMs and deploys them in a process that is inherently tailored to the specific capabilities and constraints of each platform and hardware (Chen et al., 2018; Shao et al., 2022; Feng et al., 2023). This platform-native approach ensures that LLMs are not only efficient but also highly optimized for the platforms in which they operate.

Sax. Sax (Google, 2023b) is a platform designed by Google for serving Pax, JAX, and PyTorch models for inference tasks. It supports distributed inference with tensor and data parallelism. It also integrates easily with Google Cloud and cloud monitoring with Prometheus metrics.

Mosec. Mosec (Yang et al., 2021) is a serving framework built to streamline the serving of machine learning models into backend services and microservices. Mosec's key features include continuous batching (Yu et al., 2022), pipelined stages for handling mixed workloads, and essential cloud features like model warmup, graceful shutdown, and Prometheus monitoring metrics, making it easily manageable by Kubernetes or other container systems.

5 Concluding Remarks

In this survey, we provide a systematic review of efficient LLMs, an important area of research aimed at democratizing LLMs. We start with motivating the necessity for efficient LLMs. Guided by a taxonomy, we review algorithm-level and system-level efficient techniques for LLMs from model-centric and data-centric perspectives respectively. Furthermore, we review LLM frameworks with specific optimizations and features crucial for efficient LLMs. We believe that efficiency will play an increasingly important role in LLMs and LLMs-oriented systems. We hope this survey could enable researchers and practitioners to quickly get started in this field and act as a catalyst to inspire new research on efficient LLMs.

6 Acknowledgement

We would like to thank the action editor Greg Durrett and anonymous reviewers of Transactions on Machine Learning Research for their helpful and constructive comments.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael

Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2023, *arXiv preprint arXiv:2303.08774*. URL http://arxiv.org/abs/2303.08774.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. Generalized knowledge distillation for auto-regressive language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Arash Ahmadian, Saurabh Dash, Hongyu Chen, Bharat Venkitesh, Zhen Stephen Gou, Phil Blunsom, Ahmet Üstün, and Sara Hooker. Intriguing properties of quantization at scale. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=IYe8j7Gy8f.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023. URL https://aclanthology.org/2023.emnlp-main.298.

Silas Alberti, Niclas Dern, Laura Thesing, and Gitta Kutyniok. Sumformer: Universal approximation for efficient transformers. In *Proceedings of 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML)*, volume 221, 2023. URL https://proceedings.mlr.press/v221/alberti23a.html.

Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, and Yuxiong He. Deepspeed-inference: Enabling efficient inference of transformer models at unprecedented scale. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, Dallas, Texas, 2022. URL https://dl.acm.org/doi/abs/10.5555/3571885.3571946.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer,

Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. Palm 2 technical report, 2023, *arXiv preprint arXiv:2305.10403*. URL http://arxiv.org/abs/2305.10403.

Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, Giridharan Anantharaman, Xian Li, Shuohui Chen, Halil Akin, Mandeep Baines, Louis Martin, Xing Zhou, Punit Singh Koura, Brian O'Horo, Jeffrey Wang, Luke Zettlemoyer, Mona Diab, Zornitsa Kozareva, and Veselin Stoyanov. Efficient large scale language modeling with mixtures of experts. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11699–11732, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.804. URL https://aclanthology.org/2022.emnlp-main.804.

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: fast convergence at large depth. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161, 2021. URL https://proceedings.mlr.press/v161/bachlechner21a.html.

Trapit Bansal, Salaheddin Alzubi, Tong Wang, Jay-Yoon Lee, and Andrew McCallum. Meta-adapters: Parameter efficient few-shot fine-tuning through meta-learning. In *International Conference on Automated Machine Learning*, Baltimore, US, 2022. URL https://openreview.net/forum?id=BCGNf-prLg5.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020, *arXiv preprint arXiv:2004.05150*. URL http://arxiv.org/abs/2004.05150.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R. Gormley. Unlimiformer: Long-range transformers with unlimited length input. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=lJWUJWLCJo.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38, 2024. URL https://ojs.aaai.org/index.php/AAAI/article/view/29720.

Zhengda Bian, Qifan Xu, Boxiang Wang, and Yang You. Maximizing parallelism in distributed training for huge neural networks, 2021, *arXiv preprint arXiv:2105.14450*. URL http://arxiv.org/abs/2105.14450.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, virtual+Dublin, 2022. URL https://aclanthology.org/2022.bigscience-1.9.

bloc97. Ntk-aware scaled rope allows llama models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation. https://www.reddit.com/r/LocalLLaMA/comments/14lz7j5/ntkaware_scaled_rope_allows_llama_models_to_have/, 2023.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Understanding and overcoming the challenges of efficient transformer quantization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, 2021. URL https://aclanthology.org/2021.emnlp-main.627.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Aydar Bulatov, Yuri Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Uynr3iPhksa.

Neil Burgess, Jelena Milanovic, Nigel Stephens, Konstantinos Monachopoulos, and David Mansell. Bfloat16 processing for neural networks. In *IEEE Symposium on Computer Arithmetic*, Kyoto, 2019. URL https://ieeexplore.ieee.org/document/8877390.

Federico Busato and Jeff Pool. Exploiting nvidia ampere structured sparsity with cusparselt. https://developer.nvidia.com/blog/exploiting-ampere-structured-sparsity-with-cusparselt, 2020.

Lucas Caccia, Edoardo Ponti, Zhan Su, Matheus Pereira, Nicolas Le Roux, and Alessandro Sordoni. Multi-head adapter routing for cross-task generalization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=qcQhBli5Ho.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads, 2024, *arXiv preprint arXiv:2401.10774*. URL http://arxiv.org/abs/2401.10774.

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: When data mining meets large language model finetuning, 2023, *arXiv preprint arXiv:2307.06290*. URL http://arxiv.org/abs/2307.06290.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15, 2024. URL https://doi.org/10.1145/3641289.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. QuIP: 2-bit quantization of large language models with guarantees. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xrk9g5vcXR.

Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention. *Advances in Neural Information Processing Systems*, 34, 2021a. URL https://openreview.net/forum?id=SehIKudiIo1.

Chang Chen, Min Li, Zhihua Wu, Dianhai Yu, and Chao Yang. TA-moe: Topology-aware large scale mixture-of-expert training. In *Advances in Neural Information Processing Systems*, 2022a. URL https://openreview.net/forum?id=FRDiimH26Tr.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023a, *arXiv preprint arXiv:2302.01318*. URL http://arxiv.org/abs/2302.01318.

Cheng Chen, Yichun Yin, Lifeng Shang, Xin Jiang, Yujia Qin, Fengyu Wang, Zhi Wang, Xiao Chen, Zhiyuan Liu, and Qun Liu. bert2BERT: Towards reusable pretrained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022b. URL https://aclanthology.org/2022.acl-long.151.

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. CLEX: Continuous length extrapolation for large language models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=wXpSidPpc5.

Hao Chen, Yiming Zhang, Qi Zhang, Hantao Yang, Xiaomeng Hu, Xuetao Ma, Yifan Yanggong, and Junbo Zhao. Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning, 2023b, *arXiv preprint arXiv:2305.09246*. URL http://arxiv.org/abs/2305.09246.

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. Walking down the memory maze: Beyond context limit through interactive reading, 2023c, *arXiv preprint arXiv:2310.05029*. URL http://arxiv.org/abs/2310.05029.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=FdVXgSJhvz.

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

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation, 2023d, *arXiv preprint arXiv:2306.15595*. URL http://arxiv.org/abs/2306.15595.

Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. TVM: An automated End-to-End optimizing compiler for deep learning. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Carlsbad, CA, 2018. URL https://www.usenix.org/conference/osdi18/presentation/chen.

Tianyi Chen, Tianyu Ding, Badal Yadav, Ilya Zharkov, and Luming Liang. Lorashear: Efficient large language model structured pruning and knowledge recovery, 2023e, *arXiv preprint arXiv:2310.18356*. URL http://arxiv.org/abs/2310.18356.

Wuyang Chen, Yanqi Zhou, Nan Du, Yanping Huang, James Laudon, Zhifeng Chen, and Claire Cui. Lifelong language pretraining with distribution-specialized experts. In *Proceedings of the 40th International Conference on Machine Learning*, Honolulu, Hawaii, USA, 2023f. URL https://dl.acm.org/doi/10.5555/3618408.3618621.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V Le. Symbolic discovery of optimization algorithms. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023g. URL https://openreview.net/forum?id=ne6zeqLFCZ.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. Meta-learning via language model in-context tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022c. URL https://aclanthology.org/2022.acl-long.53.

Yongrui Chen, Haiyun Jiang, Xinting Huang, Shuming Shi, and Guilin Qi. Tegit: Generating high-quality instruction-tuning data with text-grounded task design, 2023h, *arXiv preprint arXiv:2309.05447*. URL http://arxiv.org/abs/2309.05447.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models, 2023i, *arXiv preprint arXiv:2309.12307*. URL http://arxiv.org/abs/2309.12307.

Zeming Chen, Qiyue Gao, Antoine Bosselut, Ashish Sabharwal, and Kyle Richardson. DISCO: Distilling counterfactuals with large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023j. URL https://aclanthology.org/2023.acl-long.302.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023. URL https://aclanthology.org/2023.emnlp-main.232.

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, Heyan Huang, and Furu Wei. On the representation collapse of sparse mixture of experts. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=mWaYC6CZf5.

Yew Ken Chia, Guizhen Chen, Luu Anh Tuan, Soujanya Poria, and Lidong Bing. Contrastive chain-of-thought prompting, 2023, *arXiv preprint arXiv:2311.09277*. URL http://arxiv.org/abs/2311.09277.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers, 2019, *arXiv preprint arXiv:1904.10509*. URL http://arxiv.org/abs/1904.10509.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, David Belanger, Lucy Colwell, and Adrian Weller. Masked language modeling for proteins via linearly scalable long-context transformers, 2020, *arXiv preprint arXiv:2006.03555*. URL http://arxiv.org/abs/2006.03555.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022, *arXiv preprint arXiv:2204.02311*. URL http://arxiv.org/abs/2204.02311.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha

Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022, *arXiv preprint arXiv:2210.11416*. URL http://arxiv.org/abs/2210.11416.

Jae-Won Chung, Yile Gu, Insu Jang, Luoxi Meng, Nikhil Bansal, and Mosharaf Chowdhury. Perseus: Removing energy bloat from large model training, 2023, *arXiv preprint arXiv:2312.06902*. URL http://arxiv.org/abs/2312.06902.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. StableMoE: Stable routing strategy for mixture of experts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022. URL https://aclanthology.org/2022.acl-long.489.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019. URL https://aclanthology.org/P19-1285.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35, 2022. URL https://openreview.net/forum?id=H4DqfPSibmx.

Tri Dao, Daniel Haziza, Francisco Massa, and Grigory Sizov. Flash-decoding for long-context inference. https://pytorch.org/blog/flash-decoding/, 2023.

Soham De and Samuel L. Smith. Batch normalization biases residual blocks towards the identity function in deep networks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020. URL https://dl.acm.org/doi/abs/10.5555/3495724.3497400.

Tim Dettmers. Bitsandbytes. https://github.com/TimDettmers/bitsandbytes, 2023.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. GPT3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=dXiGWqBoxaD.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=OUIFPHEgJU.

Tim Dettmers, Ruslan A. Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Q1u25ahSuy.

Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. Longnet: Scaling transformers to 1,000,000,000 tokens, 2023a, *arXiv preprint arXiv:2307.02486*. URL http://arxiv.org/abs/2307.02486.

Ruomeng Ding, Chaoyun Zhang, Lu Wang, Yong Xu, Minghua Ma, Wei Zhang, Si Qin, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. Everything of thoughts: Defying the law of penrose triangle for thought generation, 2023b, *arXiv preprint arXiv:2311.04254*. URL http://arxiv.org/abs/2311.04254.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A survey on in-context learning, 2023, *arXiv preprint arXiv:2301.00234*. URL http://arxiv.org/abs/2301.00234.

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 P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, 2022. URL https://proceedings.mlr.press/v162/du22c.html.

Feyza Duman Keles, Pruthuvi Mahesakya Wijewardena, and Chinmay Hegde. On the computational complexity of self-attention. In *Proceedings of The 34th International Conference on Algorithmic Learning Theory*, 2023. URL https://proceedings.mlr.press/v201/duman-keles23a.html.

Lance Eliot. Generative pre-trained transformers (gpt-3) pertain to ai in the law, 2021. URL http://dx.doi.org/10.2139/ssrn.3974887.

Facebook AI Research (FAIR). fairseq: Fp16 optimizer - line 468. https://github.com/facebookresearch/fairseq/blob/main/fairseq/optim/fp16_optimizer.py, 2023.

FairScale authors. Fairscale: A general purpose modular pytorch library for high performance and large scale training. https://github.com/facebookresearch/fairscale, 2021.

Jiarui Fang, Zilin Zhu, Shenggui Li, Hui Su, Yang Yu, Jie Zhou, and Yang You. Parallel training of pre-trained models via chunk-based dynamic memory management. *IEEE Transactions on Parallel and Distributed Systems*, 34(1):304–315, 2023. doi: 10.1109/TPDS.2022.3219819. URL https://ieeexplore.ieee.org/document/9940581.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23, 2022. URL https://dl.acm.org/doi/abs/10.5555/3586589.3586709.

Siyuan Feng, Bohan Hou, Hongyi Jin, Wuwei Lin, Junru Shao, Ruihang Lai, Zihao Ye, Lianmin Zheng, Cody Hao Yu, Yong Yu, and Tianqi Chen. Tensorir: An abstraction for automatic tensorized program optimization. In *ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2023. URL https://doi.org/10.1145/3575693.3576933.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *Advances in Neural Information Processing Systems*, New Orleans, Louisiana, 2022. URL https://openreview.net/forum?id=ksVGCOlOEba.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/frantar23a.html.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

Daniel Y Fu, Tri Dao, Khaled Kamal Saab, Armin W Thomas, Atri Rudra, and Christopher Re. Hungry hungry hippos: Towards language modeling with state space models. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=COZDy0WYGg.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. In *Proceedings of the 40th International Conference on Machine Learning*, Honolulu, Hawaii, 2023b. URL https://proceedings.mlr.press/v202/fu23d.html.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Breaking the sequential dependency of llm inference using lookahead decoding, 2023c. URL https://lmsys.org/blog/2023-11-21-lookahead-decoding/.

Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. Megablocks: Efficient sparse training with mixture-of-experts. *Proceedings of Machine Learning and Systems*, 5, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/hash/5a54f79333768effe7e8927bcccffe40-Abstract-mlsys2023.html.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uREj4ZuGJE.

Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, G P Shrivatsa Bhargav, Dinesh Garg, and Avi Sil. Span selection pre-training for question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020. URL https://aclanthology.org/2020.acl-main.247.

Linyuan Gong, Di He, Zhuohan Li, Tao Qin, Liwei Wang, and Tieyan Liu. Efficient training of BERT by progressively stacking. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, Long Beach, California, 2019. URL https://proceedings.mlr.press/v97/gong19a.html.

Google. Pax: A jax-based machine learning framework for large scale models. https://github.com/google/paxml, 2023a. URL https://github.com/google/paxml. GitHub repository.

Google. Sax. https://github.com/google/saxml, 2023b.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129, 2021. URL https://doi.org/10.1007/s11263-021-01453-z.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 26, 2013. URL https://proceedings.neurips.cc/paper%5Ffiles/paper/2013/file/e034fb6b66aacc1d48f445ddfb08da98-Paper.pdf.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023, *arXiv preprint arXiv:2312.00752*. URL http://arxiv.org/abs/2312.00752.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Xiaotao Gu, Liyuan Liu, Hongkun Yu, Jing Li, Chen Chen, and Jiawei Han. On the transformer growth for progressive BERT training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, 2021. URL https://aclanthology.org/2021.naacl-main.406.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. PPT: Pre-trained prompt tuning for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022b. URL https://aclanthology.org/2022.acl-long.576.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Cong Guo, Jiaming Tang, Weiming Hu, Jingwen Leng, Chen Zhang, Fan Yang, Yunxin Liu, Minyi Guo, and Yuhao Zhu. Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*, 2023. URL https://doi.org/10.1145/3579371.3589038.

Ahan Gupta, Yueming Yuan, Yanqi Zhou, and Charith Mendis. Flurka: Fast fused low-rank & kernel attention, 2023, *arXiv preprint arXiv:2306.15799*. URL http://arxiv.org/abs/2306.15799.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=RjS0j6tsSrf.

Insu Han, Rajesh Jayaram, Amin Karbasi, Vahab Mirrokni, David Woodruff, and Amir Zandieh. Hyperattention: Long-context attention in near-linear time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Eh0Od2BJIM.

Jiaao He, Jiezhong Qiu, Aohan Zeng, Zhilin Yang, Jidong Zhai, and Jie Tang. Fastmoe: A fast mixture-of-expert training system, 2021, *arXiv preprint arXiv:2103.13262*. URL http://arxiv.org/abs/2103.13262.

Jiaao He, Jidong Zhai, Tiago Antunes, Haojie Wang, Fuwen Luo, Shangfeng Shi, and Qin Li. Fastermoe: modeling and optimizing training of large-scale dynamic pre-trained models. In *Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2022a. URL https://dl.acm.org/doi/10.1145/3503221.3508418.

Kai He, Rui Mao, Qika Lin, Yucheng Ruan, Xiang Lan, Mengling Feng, and Erik Cambria. A Survey of Large Language Models for Healthcare: from Data, Technology, and Applications to Accountability and Ethics, 2023, *arXiv preprint arXiv:2310.05694*. URL http://arxiv.org/abs/2310.05694.

Shwai He, Liang Ding, Daize Dong, Miao Zhang, and Dacheng Tao. Sparseadapter: An easy approach for improving the parameter-efficiency of adapters. In *Findings of EMNLP*, 2022b. URL https://aclanthology.org/2022.findings-emnlp.160.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 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, 2023. URL https://aclanthology.org/2023.acl-long.830.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack William Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=iBBcRUlOAPR.

Connor Holmes, Masahiro Tanaka, Michael Wyatt, Ammar Ahmad Awan, Jeff Rasley, Samyam Rajbhandari, Reza Yazdani Aminabadi, Heyang Qin, Arash Bakhtiari, Lev Kurilenko, and Yuxiong He. Deepspeed-fastgen: High-throughput text generation for llms via mii and deepspeed-inference, 2024, *arXiv preprint arXiv:2401.08671*. URL http://arxiv.org/abs/2401.08671.

Ke Hong, Guohao Dai, Jiaming Xu, Qiuli Mao, Xiuhong Li, Jun Liu, Kangdi Chen, Yuhan Dong, and Yu Wang. Flashdecoding++: Faster large language model inference on gpus, 2023, *arXiv preprint arXiv:2311.01282*. URL http://arxiv.org/abs/2311.01282.

Or Honovich, Uri Shaham, Samuel R. Bowman, and Omer Levy. Instruction induction: From few examples to natural language task descriptions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.acl-long.108.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization, 2024, *arXiv preprint arXiv:2401.18079*. URL http://arxiv.org/abs/2401.18079.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Sissie Hsiao, Yury Pinsky, and Sundar Pichai. Bard: Google's generative language model. https://blog.google/products/search/bard-updates/, 2023.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.findings-acl.507.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language model compression with weighted low-rank factorization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uPv9Y3gmAI5.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Shengding Hu, Ning Ding, Weilin Zhao, Xingtai Lv, Zhen Zhang, Zhiyuan Liu, and Maosong Sun. OpenDelta: A plug-and-play library for parameter-efficient adaptation of pre-trained models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Toronto, Canada, 2023a. URL https://aclanthology.org/2023.acl-demo.26.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023b. URL https://aclanthology.org/2023.emnlp-main.319.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition, 2023, *arXiv preprint arXiv:2307.13269*. URL http://arxiv.org/abs/2307.13269.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.findings-acl.67.

Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. Improving transformer optimization through better initialization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, 2020. URL https://proceedings.mlr.press/v119/huang20f.html.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Mia Xu Chen, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2019.

Yukun Huang, Yanda Chen, Zhou Yu, and Kathleen McKeown. In-context learning distillation: Transferring few-shot learning ability of pre-trained language models, 2022, *arXiv preprint arXiv:2212.10670*. URL http://arxiv.org/abs/2212.10670.

HuggingFace. text-generation-inference. https://github.com/huggingface/nanotron, 2023a. Accessed: 2024-05-10.

HuggingFace. text-generation-inference. https://github.com/huggingface/text-generation-inference, 2023b. Accessed: 2024-05-10.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. Block-recurrent transformers. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=uloenYmLCAo.

Changho Hwang, Wei Cui, Yifan Xiong, Ziyue Yang, Ze Liu, Han Hu, Zilong Wang, Rafael Salas, Jithin Jose, Prabhat Ram, Joe Chau, Peng Cheng, Fan Yang, Mao Yang, and Yongqiang Xiong. Tutel: Adaptive mixture-of-experts at scale. *Proceedings of Machine Learning and Systems*, 5, 2023.

Régis Pierrard Ilyas Moutawwakil. Llm-perf leaderboard. https://huggingface.co/spaces/optimum/llm-perf-leaderboard, 2023.

Maor Ivgi, Uri Shaham, and Jonathan Berant. Efficient long-text understanding with short-text models. *Transactions of the Association for Computational Linguistics*, 11, 2023. URL https://aclanthology.org/2023.tacl-1.17.

Hamish Ivison, Noah A. Smith, Hannaneh Hajishirzi, and Pradeep Dasigi. Data-efficient finetuning using cross-task nearest neighbors. In *Findings of the Association for Computational Linguistics*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.findings-acl.576.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023a, *arXiv preprint arXiv:2310.06825.* URL http://arxiv.org/abs/2310.06825.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=9YvfRrpmyw.

Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. Lion: Adversarial distillation of proprietary large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023b. URL https://aclanthology.org/2023.emnlp-main.189.

Yunho Jin, Chun-Feng Wu, David Brooks, and Gu-Yeon Wei. s^3: Increasing GPU utilization during generative inference for higher throughput. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=zUYfbdNl1m.

Tyler Johnson, Pulkit Agrawal, Haijie Gu, and Carlos Guestrin. AdaScale SGD: A user-friendly algorithm for distributed training. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4911–4920. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/johnson20a.html.

Hoyoun Jung and Kyung-Joong Kim. Discrete prompt compression with reinforcement learning, 2023, *arXiv preprint arXiv:2308.08758.* URL http://arxiv.org/abs/2308.08758.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. Challenges and applications of large language models, 2023, *arXiv preprint arXiv:2307.10169.* URL http://arxiv.org/abs/2307.10169.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, Jiyan Yang, Jongsoo Park, Alexander Heinecke, Evangelos Georganas, Sudarshan Srinivasan, Abhisek Kundu, Misha Smelyanskiy, Bharat Kaul, and Pradeep Dubey. A study of bfloat16 for deep learning training, 2019, *arXiv preprint arXiv:1905.12322.* URL http://arxiv.org/abs/1905.12322.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. In *Advances in Neural Information Processing Systems*, volume 34, New Orleans, Louisiana, 2021. URL https://proceedings.neurips.cc/paper%5Ffiles/paper/2021/file/081be9fdff07f3bc808f935906ef70c0-Paper.pdf.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Jeonghoon Kim, Jung Hyun Lee, Sungdong Kim, Joonsuk Park, Kang Min Yoo, Se Jung Kwon, and Dongsoo Lee. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=2jUKhUrBxP.

Minsoo Kim, Sihwa Lee, Janghwan Lee, Sukjin Hong, Du-Seong Chang, Wonyong Sung, and Jungwook Choi. Token-scaled logit distillation for ternary weight generative language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=FUnEkOkodU.

Sehoon Kim, Karttikeya Mangalam, Suhong Moon, Jitendra Malik, Michael W. Mahoney, Amir Gholami, and Kurt Keutzer. Speculative decoding with big little decoder. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023c. URL https://openreview.net/forum?id=EfMyf9MC3t.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization, 2024, *arXiv preprint arXiv:2306.07629*. URL http://arxiv.org/abs/2306.07629.

Young Jin Kim, Rawn Henry, Raffy Fahim, and Hany Hassan Awadalla. Finequant: Unlocking efficiency with fine-grained weight-only quantization for llms, 2023d, *arXiv preprint arXiv:2308.09723*. URL http://arxiv.org/abs/2308.09723.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017, *arXiv preprint arXiv:1412.6980*. URL http://arxiv.org/abs/1412.6980.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgNKkHtvB.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=e2TBb5y0yFf.

Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeybi, and Bryan Catanzaro. Reducing activation recomputation in large transformer models. In *Proceedings of Machine Learning and Systems*, volume 5, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/hash/e851ca7b43815718fbbac8afb2246bf8-Abstract-mlsys2023.html.

Siddharth Krishna Kumar. On weight initialization in deep neural networks, 2017, *arXiv preprint arXiv:1704.08863*. URL http://arxiv.org/abs/1704.08863.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. URL https://doi.org/10.1145/3600006.3613165.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. Owq: Lessons learned from activation outliers for weight quantization in large language models, 2023, *arXiv preprint arXiv:2306.02272*. URL http://arxiv.org/abs/2306.02272.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qrwe7XHTmYb.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021. URL https://aclanthology.org/2021.emnlp-main.243.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://dl.acm.org/doi/10.5555/3618408.3619203.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. BASE Layers: Simplifying training of large, sparse models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, 2021. URL https://proceedings.mlr.press/v139/lewis21a.html.

Conglong Li, Ammar Ahmad Awan, Hanlin Tang, Samyam Rajbhandari, and Yuxiong He. 1-bit lamb: Communication efficient large-scale large-batch training with lamb's convergence speed. In *IEEE International Conference on High Performance Computing, Data, and Analytics (HiPC)*, 2022. URL https://ieeexplore.ieee.org/abstract/document/10106313.

Jiamin Li, Yimin Jiang, Yibo Zhu, Cong Wang, and Hong Xu. Accelerating distributed MoE training and inference with lina. In *USENIX Annual Technical Conference (USENIX ATC)*, Boston, MA, 2023a. URL https://www.usenix.org/conference/atc23/presentation/li-jiamin.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 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, 2023b. URL https://aclanthology.org/2023.acl-long.150.

Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. Functional interpolation for relative positions improves long context transformers. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=rR03qFesqk.

Shenggui Li, Hongxin Liu, Zhengda Bian, Jiarui Fang, Haichen Huang, Yuliang Liu, Boxiang Wang, and Yang You. Colossal-ai: A unified deep learning system for large-scale parallel training. In *Proceedings of the 52nd International Conference on Parallel Processing*, 2023c. URL https://doi.org/10.1145/3605573.3605613.

Shenggui Li, Fuzhao Xue, Chaitanya Baranwal, Yongbin Li, and Yang You. Sequence parallelism: Long sequence training from system perspective. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023d. URL https://aclanthology.org/2023.acl-long.134.

Shiyang Li, Jianshu Chen, yelong shen, Zhiyu Chen, Xinlu Zhang, Zekun Li, Hong Wang, Jing Qian, Baolin Peng, Yi Mao, Wenhu Chen, and Xifeng Yan. Explanations from large language models make small reasoners better. In *Workshop on Sustainable AI*, 2024b. URL https://openreview.net/forum?id=rH8ZUcfL9r.

Shiyao Li, Xuefei Ning, Luning Wang, Tengxuan Liu, Xiangsheng Shi, Shengen Yan, Guohao Dai, Huazhong Yang, and Yu Wang. Evaluating quantized large language models, 2024c, *arXiv preprint arXiv:2402.18158*. URL http://arxiv.org/abs/2402.18158.

Xiang Li, Yiqun Yao, Xin Jiang, Xuezhi Fang, Xuying Meng, Siqi Fan, Peng Han, Jing Li, Li Du, Bowen Qin, Zheng Zhang, Aixin Sun, and Yequan Wang. Flm-101b: An open llm and how to train it with $100k budget, 2023e, *arXiv preprint arXiv:2309.03852*. URL http://arxiv.org/abs/2309.03852.

Xiang Lisa Li and Percy Liang. 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)*, 2021. URL https://aclanthology.org/2021.acl-long.353.

Xiaonan Li and Xipeng Qiu. Finding support examples for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore, 2023. URL https://aclanthology.org/2023.findings-emnlp.411.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. Unified demonstration retriever for in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023f. URL https://aclanthology.org/2023.acl-long.256.

Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. LoSparse: Structured compression of large language models based on low-rank and sparse approximation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 20336–20350, 23–29 Jul 2023g. URL https://proceedings.mlr.press/v202/li23ap.html.

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. Loftq: LoRA-fine-tuning-aware quantization for large language models. In *The Twelfth International Conference on Learning Representations*, 2024d. URL https://openreview.net/forum?id=LzPWWPAdY4.

Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. Less is more: Task-aware layer-wise distillation for language model compression. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 2023. URL https://proceedings.mlr.press/v202/liang23j.html.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration, 2023, *arXiv preprint arXiv:2306.00978*. URL http://arxiv.org/abs/2306.00978.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022a. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/0cde695b83bd186c1fd456302888454c-Paper-Conference.pdf.

Hong Liu, Zhiyuan Li, David Leo Wright Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic second-order optimizer for language model pre-training. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=3xHDeA8Noi.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, 2022b. URL https://aclanthology.org/2022.deelio-1.10.

Jiachen Liu, Zhiyu Wu, Jae-Won Chung, Fan Lai, Myungjin Lee, and Mosharaf Chowdhury. Andes: Defining and enhancing quality-of-experience in llm-based text streaming services, 2024b, *arXiv preprint arXiv:2404.16283*. URL http://arxiv.org/abs/2404.16283.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. QLLM: Accurate and efficient low-bitwidth quantization for large language models. In *The Twelfth International Conference on Learning Representations*, 2024c. URL https://openreview.net/forum?id=FIplmUWdm3.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55, 2023a. URL https://doi.org/10.1145/3560815.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 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)*, Dublin, Ireland, 2022c. URL https://aclanthology.org/2022.acl-short.8.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too, 2023b, *arXiv preprint arXiv:2103.10385*. URL http://arxiv.org/abs/2103.10385.

Yuliang Liu, Shenggui Li, Jiarui Fang, Yanjun Shao, Boyuan Yao, and Yang You. Colossal-auto: Unified automation of parallelization and activation checkpoint for large-scale models, 2023c, *arXiv preprint arXiv:2302.02599*. URL http://arxiv.org/abs/2302.02599.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. Llm-qat: Data-free quantization aware training for large language models, 2023d, *arXiv preprint arXiv:2305.17888*. URL http://arxiv.org/abs/2305.17888.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023e. URL https://openreview.net/forum?id=JZfg6wGi6g.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, and Beidi Chen. Deja vu: Contextual sparsity for efficient LLMs at inference time. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 2023f. URL https://proceedings.mlr.press/v202/liu23am.html.

Zirui Liu, Guanchu Wang, Shaochen Zhong, Zhaozhuo Xu, Daochen Zha, Ruixiang Tang, Zhimeng Jiang, Kaixiong Zhou, Vipin Chaudhary, Shuai Xu, and Xia Hu. Winner-take-all column row sampling for memory efficient adaptation of language model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023g. URL https://openreview.net/forum?id=SquMNyrk1O.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022. URL https://aclanthology.org/2022.acl-long.556.

Yucheng Lu, Conglong Li, Minjia Zhang, Christopher De Sa, and Yuxiong He. Maximizing communication efficiency for large-scale training via 0/1 adam. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=-CefY2EOupj.

Man Luo, Xin Xu, Zhuyun Dai, Panupong Pasupat, Mehran Kazemi, Chitta Baral, Vaiva Imbrasaite, and Vincent Y Zhao. Dr.ICL: Demonstration-retrieved in-context learning. In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023. URL https://openreview.net/forum?id=NDNb6L5xjI.

Kai Lv, Yuqing Yang, Tengxiao Liu, Qinghui Gao, Qipeng Guo, and Xipeng Qiu. Full parameter fine-tuning for large language models with limited resources, 2023, *arXiv preprint arXiv:2306.09782*. URL http://arxiv.org/abs/2306.09782.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=J8Ajf9WfXP.

Sadhika Malladi, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D. Lee, Danqi Chen, and Sanjeev Arora. Fine-tuning language models with just forward passes. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Vota6rFhBQ.

Pedro Henrique Martins, Zita Marinho, and Andre Martins. ∞-former: Infinite memory transformer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022. URL https://aclanthology.org/2022.acl-long.375.

Harsh Mehta, Ankit Gupta, Ashok Cutkosky, and Behnam Neyshabur. Long range language modeling via gated state spaces. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=5MkYIYCbva.

Meta. Introducing meta llama 3: The most capable openly available llm to date, 2024. URL https://ai.meta.com/blog/meta-llama-3/.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. Specinfer: Accelerating generative large language model serving with tree-based speculative inference and verification, 2024, *arXiv preprint arXiv:2305.09781*. URL http://arxiv.org/abs/2305.09781.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1gs9JgRZ.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Seattle, United States, 2022a. URL https://aclanthology.org/2022.naacl-main.201.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, 2022b. URL https://aclanthology.org/2022.emnlp-main.759.

MLC-LLM, MLC-LLM, 2023, https://github.com/mlc-ai/mlc-llm.

Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers, 2023, *arXiv preprint arXiv:2305.16300*. URL http://arxiv.org/abs/2305.16300.

Giovanni Monea, Armand Joulin, and Edouard Grave. Pass: Parallel speculative sampling, 2023, *arXiv preprint arXiv:2311.13581*. URL http://arxiv.org/abs/2311.13581.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging AI applications. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Carlsbad, CA, 2018. URL https://www.usenix.org/conference/osdi18/presentation/moritz.

MosaicML. Composer. https://github.com/mosaicml/composer, 2023a. GitHub repository.

MosaicML. Llm foundry. https://github.com/mosaicml/llm-foundry, 2023b. GitHub repository.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=2DtxPCL3T5.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021. URL https://doi.org/10.1145/3458817.3476209.

Xuefei Ning, Zinan Lin, Zixuan Zhou, Zifu Wang, Huazhong Yang, and Yu Wang. Skeleton-of-thought: Prompting LLMs for efficient parallel generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mqVgBbNCm9.

NVIDIA. Fastertransformer: High performance transformer kernels. https://github.com/NVIDIA/FasterTransformer, 2023a. GitHub repository.

NVIDIA. Tensorrt-llm. https://github.com/NVIDIA/TensorRT-LLM, 2023b. Accessed: 2024-05-10.

OpenAI. Gpt base model. https://platform.openai.com/docs/models/gpt-base, 2023.

Shankar Padmanabhan, Yasumasa Onoe, Michael JQ Zhang, Greg Durrett, and Eunsol Choi. Propagating knowledge updates to LMs through distillation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=DFaGf3O7jf.

Matteo Pagliardini, Daniele Paliotta, Martin Jaggi, and François Fleuret. Fast attention over long sequences with dynamic sparse flash attention. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=UINHuKeWUa.

Yu Pan, Ye Yuan, Yichun Yin, Zenglin Xu, Lifeng Shang, Xin Jiang, and Qun Liu. Reusing pretrained models by multi-linear operators for efficient training. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=RgNXKIrWyU.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4, 2023a, *arXiv preprint arXiv:2304.03277*. URL http://arxiv.org/abs/2304.03277.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore, 2023b. URL https://aclanthology.org/2023.findings-emnlp.936.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=wHBfxhZu1u.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=QtTKTdVrFBB.

Aaron Pham, Chaoyu Yang, Sean Sheng, Shenyang Zhao, Sauyon Lee, Bo Jiang, Fog Dong, Xipeng Guan, and Frost Ming, OpenLLM: Operating LLMs in production, 2023, https://github.com/bentoml/OpenLLM.

Jason Phang, Yi Mao, Pengcheng He, and Weizhu Chen. Hypertuning: toward adapting large language models without back-propagation. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://dl.acm.org/doi/10.5555/3618408.3619566.

Jonathan Pilault, Mahan Fathi, Orhan Firat, Christopher Pal, Pierre-Luc Bacon, and Ross Goroshin. Block-state transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, New Orleans, Louisiana, 2023. URL https://openreview.net/forum?id=XRTxIBs2eu.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Re. Hyena hierarchy: Towards larger convolutional language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 2023. URL https://proceedings.mlr.press/v202/poli23a.html.

Edoardo Maria Ponti, Alessandro Sordoni, Yoshua Bengio, and Siva Reddy. Combining parameter-efficient modules for task-level generalisation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Dubrovnik, Croatia, 2023. URL https://aclanthology.org/2023.eacl-main.49.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5, 2023.

Ramya Prabhu, Ajay Nayak, Jayashree Mohan, Ramachandran Ramjee, and Ashish Panwar. vattention: Dynamic memory management for serving llms without pagedattention, 2024, *arXiv preprint arXiv:2405.04437*. URL http://arxiv.org/abs/2405.04437.

Ofir Press, Noah Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=R8sQPpGCv0.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP*, Singapore, 2023. URL https://aclanthology.org/2023.findings-emnlp.378.

Chengwei Qin, Aston Zhang, Anirudh Dagar, and Wenming Ye. In-context learning with iterative demonstration selection, 2023a, *arXiv preprint arXiv:2310.09881*. URL http://arxiv.org/abs/2310.09881.

Guanghui Qin, Corby Rosset, Ethan C. Chau, Nikhil Rao, and Benjamin Van Durme. Nugget 2d: Dynamic contextual compression for scaling decoder-only language models, 2023b, *arXiv preprint arXiv:2310.02409*. URL http://arxiv.org/abs/2310.02409.

Yujia Qin, Yankai Lin, Jing Yi, Jiajie Zhang, Xu Han, Zhengyan Zhang, Yusheng Su, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. Knowledge inheritance for pre-trained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Seattle, United States, 2022. URL https://aclanthology.org/2022.naacl-main.288.

Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. Lightning attention-2: A free lunch for handling unlimited sequence lengths in large language models, 2024, *arXiv preprint arXiv:2401.04658*. URL http://arxiv.org/abs/2401.04658.

Jiezhong Qiu, Hao Ma, Omer Levy, Wen-tau Yih, Sinong Wang, and Jie Tang. Blockwise self-attention for long document understanding. In *Findings of the Association for Computational Linguistics: EMNLP*, 2020. URL https://aclanthology.org/2020.findings-emnlp.232.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI blog, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21, 2020. URL https://dl.acm.org/doi/abs/10.5555/3455716.3455856.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020. URL https://dl.acm.org/doi/10.5555/3433701.3433727.

Samyam Rajbhandari, Olatunji Ruwase, Jeff Rasley, Shaden Smith, and Yuxiong He. Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2021. URL https://doi.org/10.1145/3458817.3476205.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. DeepSpeed-MoE: Advancing mixture-of-experts inference and training to power next-generation AI scale. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, 2022. URL https://proceedings.mlr.press/v162/rajbhandari22a.html.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020. URL https://doi.org/10.1145/3394486.3406703.

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. Parallel context windows for large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.acl-long.352.

Ray Project, RayLLM - LLMs on Ray, GitHub repository, 2023, https://github.com/ray-project/ray-llm, Accessed on: 2023-10-02.

Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, Olatunji Ruwase, Shuangyan Yang, Minjia Zhang, Dong Li, and Yuxiong He. ZeRO-Offload: Democratizing Billion-Scale model training. In *USENIX Annual Technical Conference (USENIX ATC*, 2021. URL https://www.usenix.org/conference/atc21/presentation/ren-jie.

Liliang Ren, Yang Liu, Shuohang Wang, Yichong Xu, Chenguang Zhu, and ChengXiang Zhai. Sparse modular activation for efficient sequence modeling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=TfbzX6I14i.

Xiaozhe Ren, Pingyi Zhou, Xinfan Meng, Xinjing Huang, Yadao Wang, Weichao Wang, Pengfei Li, Xiaoda Zhang, Alexander Podolskiy, Grigory Arshinov, Andrey Bout, Irina Piontkovskaya, Jiansheng Wei, Xin Jiang, Teng Su, Qun Liu, and Jun Yao. Pangu-Σ: Towards trillion parameter language model with sparse heterogeneous computing, 2023b, *arXiv preprint arXiv:2303.10845.* URL http://arxiv.org/abs/2303.10845.

Adithya Renduchintala, Tugrul Konuk, and Oleksii Kuchaiev. Tied-lora: Enhacing parameter efficiency of lora with weight tying, 2023, *arXiv preprint arXiv:2311.09578.* URL http://arxiv.org/abs/2311.09578.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9, 2021. URL https://aclanthology.org/2021.tacl-1.4.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Seattle, United States, 2022. URL https://aclanthology.org/2022.naacl-main.191.

Andrea Santilli, Silvio Severino, Emilian Postolache, Valentino Maiorca, Michele Mancusi, Riccardo Marin, and Emanuele Rodola. Accelerating transformer inference for translation via parallel decoding. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.acl-long.689.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. RAPTOR: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=GN921JHCRw.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurençon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo González Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jörg Frohberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, María Grandury, Mario Šaško, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto Luis López, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Taşar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Fevry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre François Lavallée, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aurélie Névéol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdeněk Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Ajibade, Bharat Saxena, Carlos Muñoz Ferrandis, Daniel McDuff, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri,

Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyebade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel León Periñán, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, Maria A Castillo, Marianna Nezhurina, Mario Sänger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Théo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. Bloom: A 176b-parameter open-access multilingual language model, 2023, *arXiv preprint arXiv:2211.05100*. URL http://arxiv.org/abs/2211.05100.

Stephanie Schoch, Ritwick Mishra, and Yangfeng Ji. Data selection for fine-tuning large language models using transferred shapley values. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.acl-srw.37.

Hang Shao, Bei Liu, and Yanmin Qian. One-shot sensitivity-aware mixed sparsity pruning for large language models, 2024, *arXiv preprint arXiv:2310.09499*. URL http://arxiv.org/abs/2310.09499.

Junru Shao, Xiyou Zhou, Siyuan Feng, Bohan Hou, Ruihang Lai, Hongyi Jin, Wuwei Lin, Masahiro Masuda, Cody Hao Yu, and Tianqi Chen. Tensor program optimization with probabilistic programs. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper%5Ffiles/paper/2022/file/e894eafae43e68b4c8dfdacf742bcbf3-Paper-Conference.pdf.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, 2018. URL https://aclanthology.org/N18-2074.

Noam Shazeer. Fast transformer decoding: One write-head is all you need, 2019, *arXiv preprint arXiv:1911.02150*. URL http://arxiv.org/abs/1911.02150.

Sheng Shen, Pete Walsh, Kurt Keutzer, Jesse Dodge, Matthew Peters, and Iz Beltagy. Staged training for transformer language models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, 2022. URL https://proceedings.mlr.press/v162/shen22f.html.

Sheng Shen, Le Hou, Yanqi Zhou, Nan Du, Shayne Longpre, Jason Wei, Hyung Won Chung, Barret Zoph, William Fedus, Xinyun Chen, Tu Vu, Yuexin Wu, Wuyang Chen, Albert Webson, Yunxuan Li, Vincent Y Zhao, Hongkun Yu, Kurt Keutzer, Trevor Darrell, and Denny Zhou. Mixture-of-experts meets instruction tuning: A winning combination for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=6mLjDwYte5.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Daniel Y. Fu, Zhiqiang Xie, Beidi Chen, Clark W. Barrett, Joseph Gonzalez, Percy Liang, Christopher Ré, Ioan Cristian Stoica, and Ce Zhang. High-throughput generative inference of large language models with a single gpu. In *International Conference on Machine Learning*, 2023. URL https://dl.acm.org/doi/10.5555/3618408.3619696.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, 2020. URL https://aclanthology.org/2020.emnlp-main.346.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2020, *arXiv preprint arXiv:1909.08053*. URL http://arxiv.org/abs/1909.08053.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada, 2023. URL https://aclanthology.org/2023.findings-acl.441.

Antoine Simoulin, Namyong Park, Xiaoyi Liu, and Grey Yang. Memory-efficient selective fine-tuning. In *Workshop on Efficient Systems for Foundation Models*, 2023. URL https://openreview.net/forum?id=zaNbLceVwm.

Siddharth Singh, Olatunji Ruwase, Ammar Ahmad Awan, Samyam Rajbhandari, Yuxiong He, and Abhinav Bhatele. A hybrid tensor-expert-data parallelism approach to optimize mixture-of-experts training. In *ICS 2023*, 2023.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, Elton Zhang, Rewon Child, Reza Yazdani Aminabadi, Julie Bernauer, Xia Song, Mohammad Shoeybi, Yuxiong He, Michael Houston, Saurabh Tiwary, and Bryan Catanzaro. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model, 2022, *arXiv preprint arXiv:2201.11990*. URL http://arxiv.org/abs/2201.11990.

Saleh Soltan, Shankar Ananthakrishnan, Jack FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith Peris, Stephen Rawls, Andy Rosenbaum, Anna Rumshisky, Chandana Satya Prakash, Mukund Sridhar, Fabian Triefenbach, Apurv Verma, Gokhan Tur, and Prem Natarajan. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model, 2022, *arXiv preprint arXiv:2208.01448*. URL http://arxiv.org/abs/2208.01448.

J.C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37, 1992. URL https://ieeexplore.ieee.org/document/119632.

Benjamin Spector and Chris Re. Accelerating llm inference with staged speculative decoding, 2023, *arXiv preprint arXiv:2308.04623*. URL http://arxiv.org/abs/2308.04623.

Hongjin SU, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=qY1hlv7gwg.

Hui Su, Xiao Zhou, Houjin Yu, Xiaoyu Shen, Yuwen Chen, Zilin Zhu, Yang Yu, and Jie Zhou. Welm: A well-read pre-trained language model for chinese, 2023a, *arXiv preprint arXiv:2209.10372*. URL http://arxiv.org/abs/2209.10372.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023b, *arXiv preprint arXiv:2104.09864*. URL http://arxiv.org/abs/2104.09864.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PxoFut3dWW.

Tianxiang Sun, Zhengfu He, Qin Zhu, Xipeng Qiu, and Xuanjing Huang. Multitask pre-training of modular prompt for Chinese few-shot learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023a. URL https://aclanthology.org/2023.acl-long.625.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023b, *arXiv preprint arXiv:2307.08621*. URL http://arxiv.org/abs/2307.08621.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. A length-extrapolatable transformer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023c. URL https://aclanthology.org/2023.acl-long.816.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Weng Tam, Xiao Liu, Kaixuan Ji, Lilong Xue, Jiahua Liu, Tao Li, Yuxiao Dong, and Jie Tang. Parameter-efficient prompt tuning makes generalized and calibrated neural text retrievers. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore, 2023. URL https://aclanthology.org/2023.findings-emnlp.874.

Hanlin Tang, Shaoduo Gan, Ammar Ahmad Awan, Samyam Rajbhandari, Conglong Li, Xiangru Lian, Ji Liu, Ce Zhang, and Yuxiong He. 1-bit adam: Communication efficient large-scale training with adam's convergence speed. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, 2021. URL https://proceedings.mlr.press/v139/tang21a.html.

Chaofan Tao, Lu Hou, Wei Zhang, Lifeng Shang, Xin Jiang, Qun Liu, Ping Luo, and Ngai Wong. Compression of generative pre-trained language models via quantization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022. URL https://aclanthology.org/2022.acl-long.331.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. URL https://dl.acm.org/doi/abs/10.5555/3524938.3525813.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Computing Surveys*, 55, 2022. URL https://doi.org/10.1145/3530811.

Gemini Team and Google. Gemini: A family of highly capable multimodal models. https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf, 2023.

The MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms. https://www.mosaicml.com/blog/mpt-7b, 2023.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. Lamda: Language models for dialog applications, 2022, *arXiv preprint arXiv:2201.08239*. URL http://arxiv.org/abs/2201.08239.

Inar Timiryasov and Jean-Loup Tastet. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, Singapore, 2023. URL https://aclanthology.org/2023.conll-babylm.24.

Denis Timonin, Bo Yang Hsueh, and Vinh Nguyen. Accelerated inference for large transformer models using nvidia triton inference server, 2022. URL https://developer.nvidia.com/blog/accelerated-inference-for-large-transformer-models-using-nvidia-fastertransformer-and-nvidia-triton-inference-server.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023a. URL https://api.semanticscholar.org/CorpusID:257219404.

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

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment, 2023, *arXiv preprint arXiv:2310.16944*. URL http://arxiv.org/abs/2310.16944.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. Focused transformer: Contrastive training for context scaling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=s1FjXzJ0jy.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. DyLoRA: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Dubrovnik, Croatia, 2023. URL https://aclanthology.org/2023.eacl-main.239.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Apoorv Vyas, Angelos Katharopoulos, and François Fleuret. Fast transformers with clustered attention. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f6a8dd1c954c8506aadc764cc32b895e-Paper.pdf.

Zhongwei Wan, Yichun Yin, Wei Zhang, Jiaxin Shi, Lifeng Shang, Guangyong Chen, Xin Jiang, and Qun Liu. G-MAP: General memory-augmented pre-trained language model for domain tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, 2022. URL https://aclanthology.org/2022.emnlp-main.441.

Zhongwei Wan, Che Liu, Mi Zhang, Jie Fu, Benyou Wang, Sibo Cheng, Lei Ma, César Quilodrán-Casas, and Rossella Arcucci. Med-unic: Unifying cross-lingual medical vision-language pre-training by diminishing bias. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/af38fb8e90d586f209235c94119ba193-Paper-Conference.pdf.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019. URL https://dl.acm.org/doi/10.5555/3454287.3454581.

Boxiang Wang, Qifan Xu, Zhengda Bian, and Yang You. Tesseract: Parallelize the tensor parallelism efficiently. In *Proceedings of the 51st International Conference on Parallel Processing*, 2023a. URL https://doi.org/10.1145/3545008.3545087.

Guoxin Wang, Yijuan Lu, Lei Cui, Tengchao Lv, Dinei Florencio, and Cha Zhang. A simple yet effective learnable positional encoding method for improving document transformer model. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, 2022a. URL https://aclanthology.org/2022.findings-aacl.42.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. Bitnet: Scaling 1-bit transformers for large language models, 2023b, *arXiv preprint arXiv:2310.11453*. URL http://arxiv.org/abs/2310.11453.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024a. URL https://ieeexplore.ieee.org/document/10496231.

Liang Wang, Nan Yang, and Furu Wei. Learning to retrieve in-context examples for large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, St. Julian's, Malta, 2024b. URL https://aclanthology.org/2024.eacl-long.105.

Ningning Wang, Guobing Gan, Peng Zhang, Shuai Zhang, Junqiu Wei, Qun Liu, and Xin Jiang. ClusterFormer: Neural clustering attention for efficient and effective transformer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, 2022b. URL https://aclanthology.org/2022.acl-long.170.

Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. SCOTT: Self-consistent chain-of-thought distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023c. URL https://aclanthology.org/2023.acl-long.304.

Peihao Wang, Rameswar Panda, Lucas Torroba Hennigen, Philip Greengard, Leonid Karlinsky, Rogerio Feris, David Daniel Cox, Zhangyang Wang, and Yoon Kim. Learning to grow pretrained models for efficient transformer training. In *The Eleventh International Conference on Learning Representations*, 2023d. URL https://openreview.net/forum?id=cDYRS5iZ16f.

Shuohuan Wang, Yu Sun, Yang Xiang, Zhihua Wu, Siyu Ding, Weibao Gong, Shikun Feng, Junyuan Shang, Yanbin Zhao, Chao Pang, Jiaxiang Liu, Xuyi Chen, Yuxiang Lu, Weixin Liu, Xi Wang, Yangfan Bai, Qiuliang Chen, Li Zhao, Shiyong Li, Peng Sun, Dianhai Yu, Yanjun Ma, Hao Tian, Hua Wu, Tian Wu, Wei Zeng, Ge Li, Wen Gao, and Haifeng Wang. Ernie 3.0 titan: Exploring larger-scale knowledge enhanced pre-training for language understanding and generation, 2021, *arXiv preprint arXiv:2112.12731*. URL http://arxiv.org/abs/2112.12731.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity, 2020, *arXiv preprint arXiv:2006.04768*. URL http://arxiv.org/abs/2006.04768.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023e. URL https://openreview.net/forum?id=BryMFPQ4L6.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression, 2024c, *arXiv preprint arXiv:2403.07378*. URL http://arxiv.org/abs/2403.07378.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023f. URL https://openreview.net/forum?id=BGvkwZEGt7.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 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, 2022c. URL https://aclanthology.org/2022.emnlp-main.388.

Yiming Wang, Yu Lin, Xiaodong Zeng, and Guannan Zhang. Multilora: Democratizing lora for better multi-task learning, 2023g, *arXiv preprint arXiv:2311.11501*. URL http://arxiv.org/abs/2311.11501.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023h. URL https://aclanthology.org/2023.acl-long.754.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. Aligning large language models with human: A survey, 2023i, *arXiv preprint arXiv:2307.12966*. URL http://arxiv.org/abs/2307.12966.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. Multitask prompt tuning enables parameter-efficient transfer learning. In *The Eleventh International Conference on Learning Representations*, 2023j. URL https://openreview.net/forum?id=Nk2pDtuhTq.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022a. URL https://openreview.net/forum?id=yzkSU5zdwD.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023. URL https://aclanthology.org/2023.emnlp-main.102.

Genta Indra Winata, Samuel Cahyawijaya, Zhaojiang Lin, Zihan Liu, and Pascale Fung. Lightweight and efficient end-to-end speech recognition using low-rank transformer. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020. URL https://ieeexplore.ieee.org/document/9053878.

Qingyang Wu, Zhenzhong Lan, Kun Qian, Jing Gu, Alborz Geramifard, and Zhou Yu. Memformer: A memory-augmented transformer for sequence modeling. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, 2022a. URL https://aclanthology.org/2022.findings-aacl.29.

Xiaoxia Wu, Zhewei Yao, Minjia Zhang, Conglong Li, and Yuxiong He. Extreme compression for pre-trained transformers made simple and efficient. In *NeurIPS 2022*, 2022b.

Xiaoxia Wu, Cheng Li, Reza Yazdani Aminabadi, Zhewei Yao, and Yuxiong He. Understanding int4 quantization for language models: latency speedup, composability, and failure cases. In *Proceedings of the 40th International Conference on Machine Learning*, 2023a. URL https://dl.acm.org/doi/10.5555/3618408.3619970.

Xiaoxia Wu, Zhewei Yao, and Yuxiong He. Zeroquant-fp: A leap forward in llms post-training w4a8 quantization using floating-point formats, 2023b, *arXiv preprint arXiv:2307.09782*. URL http://arxiv.org/abs/2307.09782.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022c. URL https://openreview.net/forum?id=TrjbxzRcnf-.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada, 2023c. URL https://aclanthology.org/2023.acl-long.79.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization*, 2023. URL https://openreview.net/forum?id=6s77hjBNfS.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up language model pretraining. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=lXuByUeHhd.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. Data selection for language models via importance resampling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023b. URL https://openreview.net/forum?id=uPSQv0leAu.

Mingxue Xu, Yao Lei Xu, and Danilo P. Mandic. Tensorgpt: Efficient compression of the embedding layer in llms based on the tensor-train decomposition, 2023a, *arXiv preprint arXiv:2307.00526*. URL http://arxiv.org/abs/2307.00526.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets long context large language models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=xw5nxFWMlo.

Qifan Xu and Yang You. An efficient 2d method for training super-large deep learning models. In *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 222–232, 2023. doi: 10.1109/IPDPS54959.2023.00031.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, XIAOPENG ZHANG, and Qi Tian. QA-loRA: Quantization-aware low-rank adaptation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=WvFoJccpo8.

Zhaozhuo Xu, Zirui Liu, Beidi Chen, Yuxin Tang, Jue Wang, Kaixiong Zhou, Xia Hu, and Anshumali Shrivastava. Compress, then prompt: Improving accuracy-efficiency trade-off of llm inference with transferable prompt, 2023b, *arXiv preprint arXiv:2305.11186*. URL http://arxiv.org/abs/2305.11186.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. Openmoe: Open mixture-of-experts language models, 2023. URL https://github.com/XueFuzhao/OpenMoE.

Cheng Yang, Shengnan Wang, Chao Yang, Yuechuan Li, Ru He, and Jingqiao Zhang. Progressively stacking 2.0: A multi-stage layerwise training method for bert training speedup, 2020, *arXiv preprint arXiv:2011.13635*. URL http://arxiv.org/abs/2011.13635.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers, 2023a, *arXiv preprint arXiv:2309.03409*. URL http://arxiv.org/abs/2309.03409.

Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, David Farhi, Jakub Pachocki, Xiaodong Liu, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. In *NeurIPS 2021*, March 2022. URL https://www.microsoft.com/en-us/research/publication/tuning-large-neural-networks-via-zero-shot-hyperparameter-transfer/.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*, 2024. URL https://doi.org/10.1145/3649506.

Keming Yang, Zichen Liu, and Philip Cheng, MOSEC: Model Serving made Efficient in the Cloud, 2021, https://github.com/mosecorg/mosec.

Nan Yang, Tao Ge, Liang Wang, Binxing Jiao, Daxin Jiang, Linjun Yang, Rangan Majumder, and Furu Wei. Inference with reference: Lossless acceleration of large language models, 2023b, *arXiv preprint arXiv:2304.04487*. URL http://arxiv.org/abs/2304.04487.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=5Xc1ecxO1h.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.

Xingcheng Yao, Yanan Zheng, Xiaocong Yang, and Zhilin Yang. NLP from scratch without large-scale pretraining: A simple and efficient framework. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, 2022a. URL https://proceedings.mlr.press/v162/yao22c.html.

Yiqun Yao, Zheng Zhang, Jing Li, and Yequan Wang. Masked structural growth for 2x faster language model pre-training. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rL7xsg1aRn.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In *Advances in Neural Information Processing Systems*, volume 35, 2022b. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/adf7fa39d65e2983d724ff7da57f00ac-Paper-Conference.pdf.

Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, Zhongzhu Zhou, Michael Wyatt, Molly Smith, Lev Kurilenko, Heyang Qin, Masahiro Tanaka, Shuai Che, Shuaiwen Leon Song, and Yuxiong He. Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales, 2023c, *arXiv preprint arXiv:2308.01320*. URL http://arxiv.org/abs/2308.01320.

Zhewei Yao, Xiaoxia Wu, Cheng Li, Stephen Youn, and Yuxiong He. Zeroquant-v2: Exploring post-training quantization in llms from comprehensive study to low rank compensation, 2023d, *arXiv preprint arXiv:2303.08302*. URL http://arxiv.org/abs/2303.08302.

Rongjie Yi, Liwei Guo, Shiyun Wei, Ao Zhou, Shangguang Wang, and Mengwei Xu. Edgemoe: Fast on-device inference of moe-based large language models, 2023, *arXiv preprint arXiv:2308.14352*. URL http://arxiv.org/abs/2308.14352.

Jie You, Jae-Won Chung, and Mosharaf Chowdhury. Zeus: Understanding and optimizing GPU energy consumption of DNN training. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, 2023. URL https://www.usenix.org/conference/nsdi23/presentation/you.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for Transformer-Based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Carlsbad, CA, 2022. URL https://www.usenix.org/conference/osdi22/presentation/yu.

LILI YU, Daniel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. MEGABYTE: Predicting million-byte sequences with multiscale transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=JTmO2V9Xpz.

Zhihang Yuan, Lin Niu, Jiawei Liu, Wenyu Liu, Xinggang Wang, Yuzhang Shang, Guangyu Sun, Qiang Wu, Jiaxiang Wu, and Bingzhe Wu. Rptq: Reorder-based post-training quantization for large language models, 2023a, *arXiv preprint arXiv:2304.01089*. URL http://arxiv.org/abs/2304.01089.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. Asvd: Activation-aware singular value decomposition for compressing large language models, 2023b, *arXiv preprint arXiv:2312.05821*. URL http://arxiv.org/abs/2312.05821.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: transformers for longer sequences. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020. URL https://dl.acm.org/doi/abs/10.5555/3495724.3497174.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. GLM-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=-AwOrrrPUF.

Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, Chen Li, Ziyan Gong, Yifan Yao, Xinjing Huang, Jun Wang, Jianfeng Yu, Qi Guo, Yue Yu, Yan Zhang, Jin Wang, Hengtao Tao, Dasen Yan, Zexuan Yi, Fang Peng, Fangqing Jiang, Han Zhang, Lingfeng Deng, Yehong Zhang, Zhe Lin, Chao Zhang, Shaojie Zhang, Mingyue Guo, Shanzhi Gu, Gaojun Fan, Yaowei Wang, Xuefeng Jin, Qun Liu, and Yonghong Tian. Pangu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation, 2021, *arXiv preprint arXiv:2104.12369*. URL http://arxiv.org/abs/2104.12369.

Mingshu Zhai, Jiaao He, Zixuan Ma, Zan Zong, Runqing Zhang, and Jidong Zhai. SmartMoE: Efficiently training Sparsely-Activated models through combining offline and online parallelization. In *2023 USENIX Annual Technical Conference (USENIX ATC)*, Boston, MA, 2023. URL https://www.usenix.org/conference/atc23/presentation/zhai.

Chen Zhang, Dawei Song, Zheyu Ye, and Yan Gao. Towards the law of capacity gap in distilling language models, 2023a, *arXiv preprint arXiv:2311.07052*. URL http://arxiv.org/abs/2311.07052.

Hang Zhang, Yeyun Gong, Yelong Shen, Weisheng Li, Jiancheng Lv, Nan Duan, and Weizhu Chen. Poolingformer: Long document modeling with pooling attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, 2021. URL https://proceedings.mlr.press/v139/zhang21h.html.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. Residual learning without normalization via better initialization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1gsz30cKX.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning, 2023b, *arXiv preprint arXiv:2308.03303*. URL http://arxiv.org/abs/2308.03303.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. Loraprune: Pruning meets low-rank parameter-efficient fine-tuning, 2023c, *arXiv preprint arXiv:2305.18403*. URL http://arxiv.org/abs/2305.18403.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023d. URL https://openreview.net/forum?id=lq62uWRJjiY.

Renrui Zhang, Jiaming Han, Chris Liu, Aojun Zhou, Pan Lu, Yu Qiao, Hongsheng Li, and Peng Gao. LLaMA-adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=d4UiXAHN2W.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona 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. Opt: Open pre-trained transformer language models, 2022a, *arXiv preprint arXiv:2205.01068*. URL http://arxiv.org/abs/2205.01068.

Tianyi Zhang, Mina Lee, Xiang Lisa Li, Ende Shen, and Tatsunori Hashimoto. TempLM: Distilling language models into template-based generators. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada, 2023e. URL https://aclanthology.org/2023.findings-acl.124.

Yiming Zhang, Shi Feng, and Chenhao Tan. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, 2022b. URL https://aclanthology.org/2022.emnlp-main.622.

Yue Zhang, Hongliang Fei, Dingcheng Li, and Ping Li. PromptGen: Automatically generate prompts using generative models. In *Findings of the Association for Computational Linguistics: NAACL 2022*, Seattle, United States, 2022c. URL https://aclanthology.org/2022.findings-naacl.3.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Re, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2o: Heavy-hitter oracle for efficient generative inference of large language models. In *Workshop on Efficient Systems for Foundation Models*, 2023f. URL https://openreview.net/forum?id=ctPizehA9D.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*, 2023g. URL https://openreview.net/forum?id=5NTt8GFjUHkr.

Jiawei Zhao, Florian Tobias Schaefer, and Anima Anandkumar. ZerO Initialization: Initializing neural networks with only zeros and ones. *Transactions on Machine Learning Research*, 2022. URL https://openreview.net/forum?id=1AxQpKmiTc.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A Survey of Large Language Models, 2023a, *arXiv preprint arXiv:2303.18223*. URL http://arxiv.org/abs/2303.18223.

Weilin Zhao, Yuxiang Huang, Xu Han, Zhiyuan Liu, Zhengyan Zhang, and Maosong Sun. CPET: Effective parameter-efficient tuning for compressed large language models, 2023b, *arXiv preprint arXiv:2307.07705*. URL http://arxiv.org/abs/2307.07705.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel. *Proceedings of the VLDB Endowment*, 16, 2023c. URL https://doi.org/10.14778/3611540.3611569.

Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Lei Shen, Zihan Wang, Andi Wang, Yang Li, Teng Su, Zhilin Yang, and Jie Tang. Codegeex: A pre-trained model for code generation with multilingual benchmarking on humaneval-x. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023. URL https://doi.org/10.1145/3580305.3599790.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less is more for alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=KBMOKmX2he.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=WZH7099tgfM.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, zhifeng Chen, Quoc V Le, and James Laudon. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/2f00ecd787b432c1d36f3de9800728eb-Paper-Conference.pdf.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*, 2023c. URL https://openreview.net/forum?id=92gvk82DE-.

Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. PoSE: Efficient context window extension of LLMs via positional skip-wise training. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3Z1gxuAQrA.

Bohan Zhuang, Jing Liu, Zizheng Pan, Haoyu He, Yuetian Weng, and Chunhua Shen. A survey on efficient training of transformers, 2023, *arXiv preprint arXiv:2302.01107*. URL http://arxiv.org/abs/2302.01107.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi : Plug-and-play 2bit kv cache quantization with streaming asymmetric quantization. *Arxiv*, 2023. doi: 10.13140/RG.2.2.28167.37282. URL https://rgdoi.net/10.13140/RG.2.2.28167.37282.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Jianfeng Gao, and Tuo Zhao. Taming sparsely activated transformer with stochastic experts. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=B72HXs80q4.