# A Survey on Model Compression for Large Language Models

**Xunyu Zhu**[1,2], **Jian Li**[4*], **Yong Liu**[3], **Can Ma**[1,2], **Weiping Wang**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, China
[3]Gaoling School of Artificial Intelligence, Renmin University of China, China
[4]School of Artificial Intelligence, Beijing Normal University, China
{zhuxunyu, lijian9026, macan, wangweiping}@iie.ac.cn,
liuyonggsai@ruc.edu.cn jli@bnu.edu.cn

## Abstract

Large Language Models (LLMs) have transformed natural language processing tasks successfully. Yet, their large size and high computational needs pose challenges for practical use, especially in resource-limited settings. Model compression has emerged as a key research area to address these challenges. This paper presents a survey of model compression techniques for LLMs. We cover methods like quantization, pruning, and knowledge distillation, highlighting recent advancements. We also discuss benchmarking strategies and evaluation metrics crucial for assessing compressed LLMs. This survey offers valuable insights for researchers and practitioners, aiming to enhance efficiency and real-world applicability of LLMs while laying a foundation for future advancements.

## 1 Introduction

Large Language Models (LLMs) (Touvron et al., 2023a,b; Zhang et al., 2022; Scao et al., 2022; Wang and Komatsuzaki, 2021; OpenAI, 2024) refer to Transformer language models that contain billions (or more) of parameters, which are trained on massive text data. LLMs consistently exhibit remarkable performance across various tasks, but their exceptional capabilities come with significant challenges stemming from their extensive size and computational requirements. For instance, the GPT-175B model (Brown et al., 2020), with an impressive 175 billion parameters, demands a minimum of 350GB of memory in half-precision (FP16) format. Furthermore, deploying this model for inference necessitates at least five A100 GPUs, each featuring 80GB of memory, to efficiently manage operations. To tackle these issues, a prevalent approach known as model compression (Han et al., 2016) offers a solution. Model compression involves transforming a large, resource-intensive model into a compact version suitable for deployment on resource-constrained devices. Additionally, model compression can enhance LLM inference speed and optimizes resource efficiency.

In our paper, our primary objective is to illuminate the recent strides made in the domain of model compression techniques tailored specifically for LLMs. Our work conducts an exhaustive survey of methodologies, metrics, and benchmarks of model compression for LLMs. Figure 1 shows the taxonomy of model compression methods for LLMs, including quantization, pruning, knowledge distillation, and low-rank factorization. Figure 2 further shows basic flow of these model compression methods for LLMs. Furthermore, our study sheds light on prevailing challenges and offers a glimpse into potential future research trajectories in this evolving field. We advocate for collaborative efforts within the community to pave the way for an ecologically conscious, all-encompassing, and sustainable future for LLMs. While there were previous surveys on neural networks model compression (Li et al., 2023c) and it has been lightly discussed in prior surveys on LMs (Rogers et al., 2020) and LLMs (Zhao et al., 2023), our work is the inaugural survey dedicated solely to model compression for LLMs.

## 2 Metrics and Benchmarks

### 2.1 Metrics

Model compression of LLMs can be measured using various metrics, which capture different aspects of performance. These metrics are commonly presented alongside accuracy and zero-shot ability to comprehensively evaluate the LLM.

**Model Size** in a LLM typically is measured by the number of total parameters of the LLM.

*Corresponding author.



Transactions of the Association for Computational Linguistics, vol. 12, pp. 1556–1577, 2024. https://doi.org/10.1162/tacl_a_00704
Action Editor: Luke Zettlemoyer. Submission batch: 4/2024; Revision batch: 4/2024; Published 11/2024.

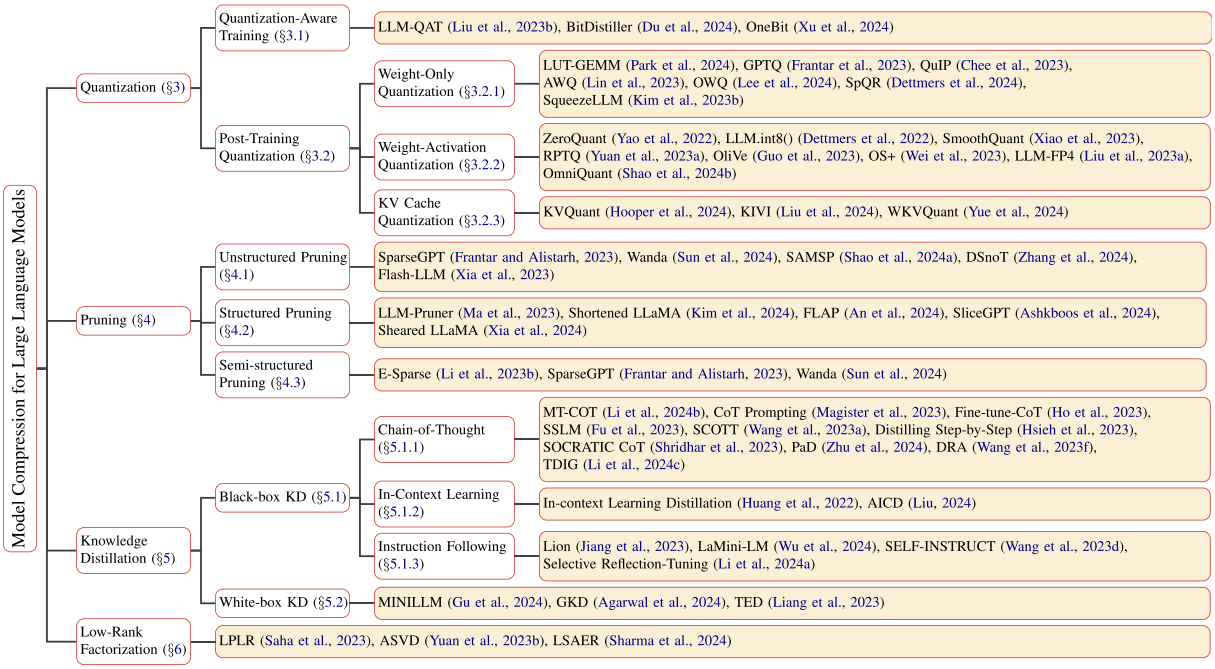


Figure 1: Taxonomy of model compression methods for large language models.

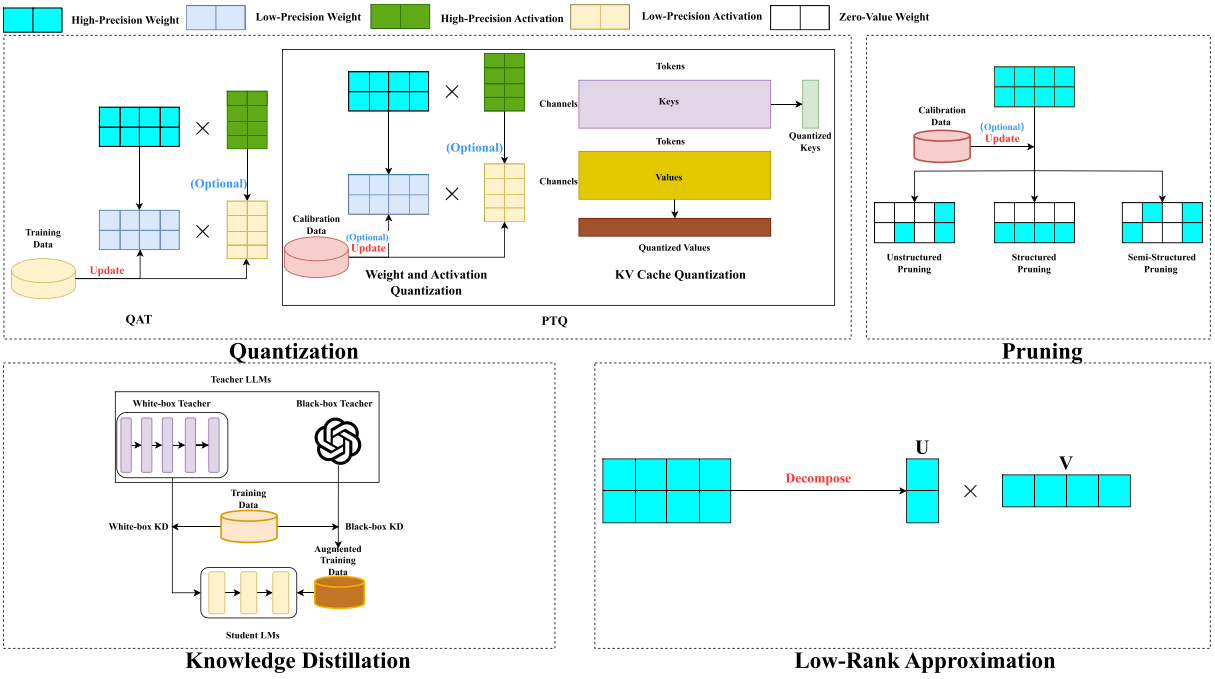


Figure 2: Illustrations of model compression methods for LLMs. In these methods, Quantization-Aware Training (QAT) and Knowledge Distillation (KD) stand out as task-based model compression techniques, tailored for specific tasks. Conversely, other model compression methods are task-agnostic, designed to operate independently of specific tasks.

In general, LLMs with more parameters often requires more computational resources and memory for both training and inference.

**Floating Point Operations (FLOPs)** is an indicator that measures the computational efficiency of LLMs, representing the number of floating-point operations required for the LLM to perform an instance. In model compression, reducing FLOPs helps to make the LLM run faster and more efficiently.

**Mean FLOPS Utilization (MFU)** quantifies the practical efficiency of computational resource

utilization by LLMs during tasks. MFU measures the ratio of actual FLOPS utilized by the LLM to the maximum theoretical FLOPS of a device. Unlike FLOPs, which estimates the maximum operations an LLM might perform, MFU assesses the actual effectiveness of resource use in operation. Essentially, while FLOPs measures a LLM's theoretical compute needs, MFU shows how effectively these computations are utilized in practice.

**Inference Time** (i.e., latency) measures the time taken by the LLM to process and generate responses for input data during inference. Inference time is particularly crucial for real-world applications where the LLM needs to respond for user queries or process large amounts of data in real-time.

**Speedup Ratio** measures how much faster a compressed LLM performs tasks compared to the original LLM. Specifically, it measures the ratio of the inference time of the uncompressed model over the inference time of the compressed model. Higher ratios mean greater efficiency and reduced computation time, highlighting effective compression.

**Compression Ratio** measures how much a LLM's size is reduced through compression, calculated as the original size divided by the compressed size. Higher ratios mean greater size reduction, showing the compression's effectiveness in saving storage and memory.

### 2.2 Benchmarks and Datasets

The main goal of these benchmarks and datasets is to measure the efficiency and performance of compressed LLMs in comparison to their uncompressed counterparts. These benchmarks and datasets typically consist of diverse tasks and datasets that cover a range of natural language processing challenges.

#### 2.2.1 Common Benchmarks and Datasets

The majority of research evaluates compressed LLMs on well-established NLP benchmarks and datasets. For instance, WikiText-2 (Merity et al., 2017), C4 (Raffel et al., 2020), and PTB (Marcus et al., 1993) are designed for evaluating the perplexity performance of language models. LAMBADA (Paperno et al., 2016), PIQA (Tata and Patel, 2003), and OpenBookQA (Mihaylov et al., 2018) are designed to evaluate the zero-shot ability of language models. GSM8K (Cobbe et al., 2021), CommonsenseQA (Talmor et al., 2019) and StrategyQA (Geva et al., 2021) are designed to evaluate the reasoning ability of language models.

#### 2.2.2 BIG-Bench

BIG-Bench (BBH) (Srivastava et al., 2023) is a benchmark suite designed for LLMs, covering over 200 NLP tasks, e.g., Text Comprehension Tasks, Inference Tasks, and Mathematical Reasoning Tasks. The aim of BBH is to evaluate the performance of LLMs across these various complex tasks. The compressed LLMs use BBH to measure their capability across a multidimensional spectrum of tasks.

#### 2.2.3 Unseen Instructions Datasets

Unseen instructions datasets are used to evaluate the performance of LLMs on unseen tasks. For instance, the Vicuna-Instructions (Zheng et al., 2023) dataset created by GPT-4 includes 80 complex questions across nine different categories like generic, knowledge-based, and writing tasks. Another dataset, User-Oriented-Instructions (Wang et al., 2023d), consists of 252 carefully selected instructions inspired by various user-focused applications such as Grammarly, StackOverflow, and Overleaf. These datasets evaluate how well compact LLMs can handle and carry out new tasks by presenting them with unfamiliar instructions.

#### 2.2.4 EleutherAI LM Harness

The EleutherAI LM Harness (Gao et al., 2023) is an advanced framework for evaluating LLMs, providing a unified testing platform that supports over 60 standard academic benchmarks along with hundreds of subtasks and variants. The standardized evaluation tasks provided by the harness ensure the reproducibility and comparability of evaluation, which is essential for implementing fair and reproducible evaluations for the compressed LLMs.

## 3 Quantization

Quantization (Gray and Neuhoff, 1998) refers to the process of reducing the number of bits (i.e., precision) in the parameters of the model with minimal loss in inference performance. Quantization can be categorized into two main approaches: **Quantization-Aware Training (QAT)**, and **Post-Training Quantization (PTQ)**. The primary distinction between the two approaches lies in

| Category† | Methods | LLM | Bit Width | | | Perplexity Difference‡ | | Speedup |
|---|---|---|---|---|---|---|---|---|
| | | | Weights | Activations | KV Cache | Wikitext-2 | C4 | |
| QAT | LLM-QAT | LLaMA-30B | 4 | 8 | 16 | 0.5 | 0.9 | – |
| | BitDistiller | LLaMA2-13B | 2 | 16 | 16 | 1.9 | – | – |
| | OneBit | LLaMA-13B | 1 | 16 | 16 | 4.09 | 3.64 | – |
| Weight-Only Quantization | LUT-GEMM | LLaMA-65B | 3 | 16 | 16 | 0.14 | – | 2.04× |
| | SqueezeLLM | LLaMA-13B | 3 | 16 | 16 | 0.51 | 0.67 | 2.4× |
| | GPTQ | OPT-175B | 3 | 16 | 16 | 0.34 | 0.23 | 3.24× |
| | AWQ | LLaMA2-70B | 3 | 16 | 16 | 0.42 | – | 3.2× |
| | OWQ | LLaMA-65B | 3.01 | 16 | 16 | 0.72 | – | – |
| | SpQR | LLaMA-30B | 3.89 | 16 | 16 | 0.15 | 0.1 | 2.0× |
| | QuIP | LLaMA2-70B | 2 | 16 | 16 | 3.007 | 3.228 | – |
| Weight-Activation Quantization | ZeroQuant | GPT-J-6B | 8 | 8 | 16 | 0.16 | – | 3.67× |
| | LLM.int8() | OPT-13B | 8 | 8 | 16 | – | 0.00 | 1.22× |
| | SmoothQuant | OPT-175B | 8 | 8 | 16 | 0.18 | – | 1.56× |
| | RPTQ | OPT-175b | 4 | 4 | 16 | 2.26 | 2.15 | – |
| | Olive | BLOOM-7B | 4 | 4 | 16 | 2.11 | 2.24 | 4.5× |
| | OS+ | LLaMA-65B | 4 | 4 | 16 | 5.77 | – | – |
| | QT | OPT-1.3B | 8 | 8 | 16 | 17.74 | – | – |
| | ZeroQuant-FP | LLaMA-30B | 4 | 8 | 16 | 0.18 | 0.13 | – |
| | OmniQuant | LLaMA-7B | 4 | 6 | 16 | 0.41 | 0.55 | – |
| KV Cache Quantization | KVQuant | LLaMA-65B | 16 | 16 | 2 | 0.19 | 0.11 | 1.4× |
| | WKVQuant | LLaMA-13B | 4 | 16 | 4 | 0.12 | 0.14 | – |

† : The results presented in the table are solely derived from the original papers.

‡ : (The perplexity of the quantized LLM) - (The perplexity of the origin LLM).

Table 1: The performance of various representative LLM quantization methods.

whether retraining is needed during quantization. PTQ enables direct use of quantized models in inference, while QAT requires retraining to rectify errors introduced by quantization. Table 1 shows the performance of many representative LLM quantization methods.

### 3.1 Quantization-Aware Training

QAT involves retraining a quantized model to counteract performance degradation caused by quantization. For instance, LLM-QAT (Liu et al., 2023b) implements the standard QAT framework directly onto LLMs. LLM-QAT distills knowledge by generating data from the LLM itself, and train the quantized LLM to align with the output distribution of the original LLM based on the generated data. BitDistiller (Du et al., 2024) merges QAT with self-distillation, enhancing LLM performance at sub-4-bit precisions. It employs tailored asymmetric quantization, clipping, and a Confidence-Aware Kullback-Leibler Divergence objective for faster convergence and superior results. OneBit (Xu et al., 2024) introduces a novel 1-bit parameter representation method and an effective parameter initialization method to implement 1-bit quantization for LLM weight matrices, paving the way for the extremely low bit-width deployment of LLMs.

**Remark 1.** *While QAT can mitigate quantization's accuracy degradation, retraining demands a lot of effort due to tens or hundreds of billions of parameters in LLMs. A practical solution is to incorporate Parameter-Efficient Fine-Tuning (PEFT) into the retraining process of QAT. Currently, methods like QLORA (Dettmers et al., 2023), PEQA (Kim et al., 2023a), and LoftQ (Li et al., 2023a) combine quantization with PEFT for model fine-tuning efficiency. However, these methods are typically task-dependent. L4Q (Jeon et al., 2024) makes a preliminary attempt to enhance generality by leveraging LoRA-wise learned quantization step size for LLMs. We think that introducing PEFT to enhance QAT efficiency is not only feasible but also holds significant promise, warranting thorough exploration.*

### 3.2 Post-Training Quantization

PTQ efficiently converts a full-precision LLM to low-precision without retraining, saving memory and computational costs. We categorize PTQ for LLMs into three groups: **Weight-Only Quantization**, **Weight-Activation Quantization**, and **KV Cache Quantization**. The disparity between these groups lies in their quantization objectives. Weight-only quantization focuses solely on quantizing weights, whereas weight-activation quantization extends its objective to both weights and activations. Prior research (Yao et al., 2023) indicates that activation quantization is typically more sensitive to weight quantization, allowing weight-only quantization to achieve lower bit-width. However, since quantized weights necessitate dequantization before multiplication with activations, weight-only quantization inevitably introduces additional computational overhead during inference and cannot enjoy the accelerated low-bit operation supported by specific hardware. Furthermore, KV cache quantization targets the KV cache, which stores keys and values of attention layers. The KV cache often consumes lots of memory, acting as a bottleneck for input streams containing lengthy tokens. By implementing KV cache quantization, it is possible to increase throughput and accommodate inputs with longer tokens more efficiently.

#### 3.2.1 Weight-Only Quantization

Weight-only quantization is the most conventional and widespread method. For example, LUT-GEMM (Park et al., 2024) uses binary-coding quantization (BCQ) (Rastegari et al., 2016) format, which factorizes the parameters of LLMs into binary parameters and a set of scaling factors, to accelerate quantized matrix multiplications in weight-only quantization. GPTQ (Frantar et al., 2023) proposes a layer-wise quantization method based on Optimal Brain Quantization (OBQ) (Frantar and Alistarh, 2022), which updates weights with inverse Hessian information, and quantizes LLMs into 3/4-bit. QuIP (Chee et al., 2023) optimally adjusts weights by utilizing the LDL decomposition of the Hessian matrix derived from vectors drawn uniformly at random from a calibration set, and multiplies weight and Hessian matrices with a Kronecker product of random orthogonal matrices to ensure incoherence between weight and Hessian matrices. Combining these two steps, QuIP successfully quantizes LLMs into 2-bits with minimal performance loss.

To further minimize quantization errors in the weight-only quantization of LLMs, many studies identify sensitive weights, which have an important effect on LLMs' quantization performance, and store these sensitive weights in high precision. For example, AWQ (Lin et al., 2023) stores the top 1% of weights that have the most significant impact on LLM performance in high-precision, and integrates a per-channel scaling method to identify optimal scaling factors. Here, ''channel'' denotes individual dimensions or feature maps within the model. Similar with AWQ, OWQ (Lee et al., 2024) store weights sensitive to activation outliers in high-precision, and quantizes other non-sensitive weights. Different from OWQ, SpQR (Dettmers et al., 2024) employs the L2 error between the original and quantized predictions as a weight sensitivity metric. Furthermore, SqueezeLLM (Kim et al., 2023b) introduces a weights clusters algorithm based on sensitivity, using k-means centroids as quantized weight values, to identify sensitive weights. The sensitivity is approximated by the Hessian matrix of weights. Then, SqueezeLLM stores sensitive weights in an efficient sparse format, and quantize other weights. SqueezeLLM quantizes LLMs in 3-bit, and achieves a more than 2× speedup compared to the FP16 baseline.

#### 3.2.2 Weight-Activation Quantization

Alongside studies centered on weight-only quantization in LLMs, there is a plethora of research focusing primarily on weight-activation quantization in LLMs. For example, ZeroQuant (Yao et al., 2022) is the first work to implement weight-activation quantization for LLMs, which uses group-wise quantization for weight and token-wise quantization for activations, and reduces the precision for weights and activations of LLMs to INT8.

LLMs have outliers in activations, and the performance of LLMs declines considerably, if these activations with outliers are directly quantized. Recent studies try to treat these outliers specially to reduce quantization errors in weight-activation quantization. For example, LLM.int8() (Dettmers et al., 2022) stores these outlier feature dimensions into high-precision, and uses vector-wise

quantization, which assigns separate normalization constants to each inner product within matrix multiplication, to quantize other features. LLM.int8() quantizes weights and activations of LLMs into 8-bit without any performance degradation. SmoothQuant (Xiao et al., 2023) designs a per-channel scaling transformation to smooths the activation outliers based on the discovery that different tokens have similar variations across channels of activations. RPTQ (Yuan et al., 2023a) finds that the range of values varies greatly between different channels, and integrates a channel reordering method, which clusters and reorders the channels in the activation and uses the same quantization parameters to quantize the values in each cluster, into layer normalization and linear layer weights to efficiently reduce the effect of numerical range differences between channels. OliVe (Guo et al., 2023) thinks that outliers are more important than the normal values, and uses an outlier-victim pair (OVP) quantization to handle outlier values locally with low hardware overheads and significant performance benefits. OS+ (Wei et al., 2023) further finds that outliers are concentrated in specific and asymmetric channels. Based on the findings, OS+ incorporates channel-wise shifting to eliminate the impact of asymmetry and channel-wise scaling to balance the distribution of outliers. LLM-FP4 (Liu et al., 2023a) uses floating-point formats (specifically FP8 and FP4) to address the limitations of traditional integer quantization (such as INT8 and INT4) to deal with outliers. Furthermore, LLM-FP4 (Liu et al., 2023a) points out that exponent bits and clipping range are important factors that effect the performance of FP quantization, and introduces a search-based framework for determining the optimal exponent bias and maximal quantization value. OmniQuant (Shao et al., 2024b) handles the activation outliers by equivalently shifting the challenge of quantization from activations to weights, and optimizes the clipping threshold to adjust the extreme values of the weights.

#### 3.2.3 KV Cache Quantization

With the increasing number of input tokens supported by LLMs, the memory usage of the KV cache also increases. Recent efforts begin to focus on KV cache quantization to reduce the memory footprint of LLMs and accelerate their inference. For example, KVQuant (Hooper et al., 2024) proposes several KV Cache Quantization methods, such as Per-Channel Key Quantization, PreRoPE Key Quantization, and Non-Uniform KV cache quantization, to implement 10 million context length LLM inference. Through an in-depth analysis of the element distribution within the KV cache, KIVI (Liu et al., 2024) finds that key caches should be quantized per-channel, while value caches should be quantized per-token. Finally, KIVI succeeds in quantizing the KV cache to 2 bits without fine-tuning. WKVQuant (Yue et al., 2024) presents an innovative approach for quantizing LLMs by integrating past-only quantization to refine attention computations, employing a two-dimensional quantization strategy to manage the distribution of key/value (KV) caches effectively, and utilizing cross-block reconstruction regularization for optimizing parameters. This method enables the quantization of both weights and KV caches, resulting in memory savings that rival those of weight-activation quantization, while nearly matching the performance levels of weight-only quantization.

## 4 Pruning

Pruning (LeCun et al., 1989) is a powerful technique to reduce the size or complexity of a model by removing redundant components. Pruning can be divided into **Unstructured Pruning**, **Semi-Structured Pruning**, and **Structured Pruning**. Structured pruning removes entire components like neurons, attention heads, or layers based on specific rules while preserving the overall network structure. On the other hand, unstructured pruning prunes individual parameters, resulting in an irregular sparse structure. Semi-structured pruning is a method that lies between structured pruning and unstructured pruning, capable of achieving fine-grained pruning and structural regularization simultaneously. It prunes partial parameters based on specific patterns rather than entire channels, filters, or neurons, making it a fine-grained form of structured pruning. Table 2 shows the performance of many representative LLM pruning methods.

### 4.1 Unstructured Pruning

Unstructured pruning preserves the pruned model's performance, hence, works related to unstructured pruning of LLMs often dispense with

| Category† | Methods | LLM | Perplexity Difference (WikiText-2)‡ | Compression Rate | Speed up |
|---|---|---|---|---|---|
| Unstructured Pruning | SparseGPT | OPT-175B | −0.14 | 50% | – |
| | Wanda | LLaMA-65B | 1.01 | 50% | – |
| | SAMSP | LLaMA2-13B | 0.63 | 50% | – |
| | DSnoT | LLaMA-65B | 2.08e4 | 90% | – |
| Structured Pruning | LLM-Pruner | LLaMA-13B | 3.6 | 20% | – |
| | Shortened LLaMA | LLaMA-7B | 10.5 | 35% | – |
| | FLAP | LLaMA-65B | 7.09 | 50% | – |
| | SliceGPT | LLaMA2-70B | 1.73 | 30% | 1.87× |
| Semi-Structured Pruning | E-Sparse | LLaMA-65B | 2.13 | 2:4 | 1.53× |
| | SparseGPT | OPT-175B | 0.39 | 2:4 | 2× |
| | Wanda | LLaMA-65B | 2.69 | 2:4 | 1.24× |

† : The results presented in the table are solely derived from the original papers.

‡ : (The perplexity of the pruned LLM) - (The perplexity of the origin LLM).

Table 2: The performance of various representative LLM pruning methods.

retraining to restore performance. Nevertheless, unstructured pruning renders the pruned model irregular, necessitating specialized handling or software optimizations for inference acceleration. An innovative approach in this domain is SparseGPT (Frantar and Alistarh, 2023), which introduces a one-shot pruning strategy without retraining. SparseGPT frames pruning as an extensive sparse regression problem and solves it using an approximate sparse regression solver. SparseGPT achieves significant unstructured sparsity, even up to over 50% on the largest GPT models like OPT-175B and BLOOM-176B, with minimal increase in perplexity. To reduce the cost about the weight update process required by SparseGPT, Wanda (Sun et al., 2024) achieves model sparsity by pruning weights with the smallest magnitudes multiplied by the norm of the corresponding input activations, without the need for retraining or weight updates. To further minimize pruning-induced errors while upholding the desired overall sparsity level, SAMSP (Shao et al., 2024a) utilizes the Hessian matrix as a metric for weight matrix sensitivity evaluation, and dynamically adjusts sparsity allocation based on sensitivity. Furthermore, DSnoT (Zhang et al., 2024) minimizes the reconstruction error between dense and sparse models through iterative weight pruning-and-growing on top of sparse LLMs to enhance LLM performance across various sparsity rates, especially at high sparsity levels. To provide hardware support for handling unstructured pruning on the GPU Tensor Core hardware, Flash-LLM (Xia et al., 2023) introduces an unstructured sparse matrix multiplication method, which loads weight matrices in a sparse format from global memory and reconstructs them in a dense format within high-speed on-chip buffers for computation using tensor cores.

## 4.2 Structured Pruning

Compared to unstructured pruning, structured pruning offers the advantage of being hardware-agnostic, allowing for accelerated inference on traditional hardware post-pruning. However, the removal of larger and potentially more critical components in structured pruning may result in performance degradation, typically requiring efficient parameter fine-tuning for recovery. We divide LLMs structured pruning works into several groups based on pruning metrics: **Loss-based Pruning**, **Magnitude-based Pruning**, and **Regularization-based Pruning**.

**Loss-based Pruning** (Molchanov et al., 2019) assesses the significance of a pruning unit by measuring its impact on loss or gradient information (e.g., first-order or second-order derivatives of loss). For example, LLM-Pruner (Ma et al., 2023) introduces a one-shot structured pruning on LLMs based on gradient information. Specifically, LLM-Pruner identifies dependent structures via a dependency detection algorithm and selects optimal pruning groups using gradient information, rather than solely relying on loss changes, in a task-agnostic manner. Different from LLM-Pruner, which focuses on narrowing LLMs' width, Shortened LLaMA (Kim et al., 2024) introduces a one-shot depth pruning on LLMs. Shortened LLaMA chooses the Transformer block as the prunable unit, and prunes these unimportant Transformer blocks, where the importance of Transformer blocks is evaluated by loss and

its second-order derivative. After pruning, both LLM-Pruner and Shortened LLaMA utilize LoRA to rapidly recover the performance of the pruned model.

**Magnitude-based Pruning** (Han et al., 2015) involves devising a heuristic metric based on the magnitudes of pruning units, and use the metric to assess the importance of pruning units, subsequently pruning those units whose scores fall below a predefined threshold. For example, FLAP (An et al., 2024) utilizes a structured fluctuation metric to assess and identify columns in the weight matrix suitable for pruning, measuring the variation of each input feature relative to a baseline value to estimate the impact of removing a column of weights. Additionally, FLAP uses an adaptive structure search to optimize global model compression, and restores the model's performance post-pruning through a baseline bias compensation mechanism, avoiding the need for fine-tuning. To further maintain the pruned model's performance, SliceGPT (Ashkboos et al., 2024) leverages the computational invariance of transformer networks and optimizes the pruning process through Principal Component Analysis (PCA). Specifically, SliceGPT employs PCA as the pruning metric, applying it at each layer of the transformer network to project the signal matrix onto its principal components and eliminate insignificant columns or rows from the transformed weight matrices, ultimately aiming to compress the model effectively.

**Regularization-based Pruning** (Wen et al., 2016) typically adds a regularization term (e.g., $L_0$, $L_1$, and $L_2$ regularization) into the loss function to induce sparsity for LLMs. For example, Sheared LLaMA (Xia et al., 2024) uses a pair of Lagrange multipliers based on pruning masks to impose constraints on the pruned model shape directly, thereby formulating pruning as a constrained optimization problem. Through solving this optimization problem, Sheared LLaMA derives optimal pruning masks. Additionally, Sheared LLaMA introduces dynamic batch loading, a strategy that adapts training data loading based on each domain's loss reduction rate, enhancing the efficiency of data utilization during training.

**Remark 2.** *Structured pruning typically reduces model size by removing redundant parameters, but it may degrade model performance. A novel approach is to combine knowledge distillation (Hinton et al., 2015) with structured pruning. Knowledge distillation allows knowledge extracted from a LLM to be transferred to a smaller model, helping the smaller model maintain its performance while reducing its size.*

### 4.3 Semi-Structured Pruning

Apart from unstructured pruning and structured pruning, there are many studies which use semi-structured pruning to prune partial weights of LLMs based on specific patterns. N:M sparsity, where every M contiguous elements leave N non-zero elements, is an example of semi-structured pruning. For example, E-Sparse (Li et al., 2023b) implements N:M sparsity by introducing information entropy as a metric for evaluating parameter importance to enhances the significance of parameter weights and input feature norms. E-Sparse incorporates global naive shuffle and local block shuffle to efficiently optimize information distribution and mitigate the impact of N:M sparsity on LLM accuracy. Furthermore, many pruning studies can also be generalized to semi-structured patterns. For example, SparseGPT (Frantar and Alistarh, 2023) and Wanda (Sun et al., 2024) also explore N:M sparsity of LLMs. SparseGPT (Frantar and Alistarh, 2023) employs block-wise weight partitioning, with each block containing M weights. It identifies and prunes N weights with the lowest reconstruction error (based on Hessian information), ensuring a sparsity ratio of N:M. This process iteratively prunes and updates model weights, addressing one block at a time until the desired sparsity level is achieved across the entire model. Wanda (Sun et al., 2024) achieves structured N:M pruning by dividing the weight matrix into groups of M consecutive weights and computing an importance score for each weight. The score is determined by the product of the weight's magnitude and the norm of the corresponding input activations. Within each weight group, the N weights with the highest scores are retained, while the rest are set to zero, thereby implementing structured N:M pruning. Furthermore, choosing the optimal pruning strategy is crucial for compatibility with the target hardware. For instance, Choquette et al. (2021) introduce the Ampere Tensor Core GPU architecture (e.g., A100 GPUs) and propose 2:4 fine-grained semi-structured sparsity to accelerate Sparse Neural Networks on this

hardware. However, the current implementation of the Ampere architecture supports only the 2:4 ratio, leaving other ratios without acceleration.

**Remark 3.** *LLMs often perform well on multiple tasks, which means they contain a multitude of parameters for various tasks. Dynamic pruning (Xia et al., 2020) methods can dynamically prune different parts of the model based on the current task's requirements to provide better performance on specific tasks. This helps strike a balance between performance and efficiency.*

**Remark 4.** *For PTQ and pruning, preparing a high-quality calibration dataset to assist in improving the performance of compressed LLMs is crucial. Specifically, Williams and Aletras (2023) make a extensive empirical study on the effect of calibration data upon model compression methods, and find that the performance of downstream tasks can vary significantly depending on the calibration data selected. High-quality calibration data can improve the performance and accuracy of the compressed model, so careful selection and preparation of calibration data are necessary.*

## 5 Knowledge Distillation

Knowledge Distillation (KD) (Hinton et al., 2015) is a technique aimed at transferring knowledge from a large and complex model (i.e., teacher model) to a smaller and simpler model (i.e., student model). We classify these methods into two clear categories (Gu et al., 2024): **Black-box KD**, where only the teacher's outputs are accessible, typically from closed-source LLMs, and **White-box KD**, where the teacher's parameters or output distribution are available, usually from open-source LLMs.

### 5.1 Black-box KD

Black-box KD usually prompts the teacher LLM to generate a distillation dataset for fine-tune the student LM, thereby transfering capabilities from teacher LLM to the student LM. In Black-box KD, teacher LLMs such as ChatGPT (gpt-3.5-turbo) and GPT4 (OpenAI, 2024) are typically employed, while smaller LMs (SLMs), such as GPT-2 (Radford et al., 2019), T5 (Raffel et al., 2020), FlanT5 (Chung et al., 2024), and CodeT5 (Wang et al., 2021), are commonly utilized as student LMs. On the other hand, researchers find that LLMs have emergent abilities, which refers to a significant improvement in performance when the model reaches a certain scale, showcasing surprising capabilities. Lots of Black-box KD methods try to distill emergent abilities from LLMs to student LMs, and we introduce three commonly used emergent ability distillation methods: Chain-of-Thought (CoT) Distillation, In-Context Learning (ICL) Distillation, and Instruction Following (IF) Distillation.

#### 5.1.1 Chain-of-Thought Distillation

CoT (Wei et al., 2022; Wang et al., 2023b) prompts LLMs to generate intermediate reasoning steps, enabling them to tackle complex reasoning tasks step by step. Li et al. (2024b) and Hsieh et al. (2023) employ LLMs to prompt the generation of explanations and leverage a multi-task learning framework to bolster the reasoning capabilities of smaller models while enhancing their capacity for generating explanations. Magister et al. (2023) show that LLMs' reasoning capability can be transferred to SLMs via knowledge distillation, but there's a trade-off between model and dataset size in reasoning ability. Ho et al. (2023) use zero-shot CoT techniques to prompt LLMs to generate diverse rationales to enrich the distillation dataset for the student models. Shridhar et al. (2023) distill two student models: a problem decomposer and a subproblem solver, which the problem decomposer decomposes complex problems into a sequence of subproblems, and the subproblem solver solves these subproblems step by step. Wang et al. (2023a) incorporate contrastive decoding during rationale generation for teacher models and address shortcut issues by introducing a counterfactual reasoning objective during student model training. Fu et al. (2023) demonstrate that increasing task-specific capabilities through distillation may inadvertently lead to reduced performance in solving generalized problems, and focus on improving mathematical capability of student LMs via distillation. PaD (Zhu et al., 2024) prompts LLMs to generate Program-of-Thought (PoT) rationales instead of CoT rationales to construct distillation dataset, and fine-tunes SLMs with the distillation dataset. Wang et al. (2023e) establishes a multi-round interactive learning paradigm that enables student LMs to provide feedback to teacher LLMs during the distillation process, thereby obtaining tailored training data. Additionally, DRA introduces a self-reflection

learning mechanism, allowing the student LMs to learn from their mistakes and enhance their reasoning abilities. Li et al. (2024c) find that negative data generated from teacher LMs also has reasoning knowledge, and guides student LMs to learn knowledge from both negative samples besides positive ones.

#### 5.1.2 In-Context Learning Distillation

ICL (Dong et al., 2023; Wang et al., 2023c) employs structured prompts with task descriptions and examples for LLMs to learn new tasks without gradient updates. Huang et al. (2022) introduce a method called in-context learning distillation, which transfers in-context learning ability from LLMs to smaller models by combining in-context learning objectives with language modeling objectives. Specifically, it trains the student model to improve its generalization across various tasks by imitating the soft label predictions of the teacher model and the hard label ground truth values. Additionally, the method incorporates two few-shot learning paradigms: Meta In-context Tuning (Meta-ICT) and Multitask In-context Tuning (Multitask-ICT). In Meta-ICT, the student model adapts to new tasks with in-context learning and guidance from the teacher. Conversely, Multitask-ICT treats all target tasks as training tasks, directly using examples from them in distillation. The outcomes show that Multitask-ICT is more effective, despite its increased computational requirements. AICD (Liu, 2024) leverages the autoregressive nature of LLMs to perform meta-teacher forcing on CoTs within the context, jointly optimizing the likelihood of all in-context CoTs, thereby distilling the capabilities of in-context learning and reasoning into smaller models.

#### 5.1.3 Instruction Following Distillation

IF (Ouyang et al., 2022; Brooks et al., 2023) aims to bolster the zero-shot ability of LLMs through fine-tuning using a collection of instruction-like prompt-response pairs. For instance, Lion (Jiang et al., 2023) prompts the LLM to identify and generate the ''hard'' instructions, which are then utilized to enhance the student model's capabilities. LaMini-LM (Wu et al., 2024) develops an extensive collection of 2.58 million instructions, comprising both existing and newly generated instructions, and fine-tunes a diverse array of models by using these instructions. SELF-INSTRUCT (Wang et al., 2023d) uses student LMs themselves as teachers to generate instruction following dataset, and fine-tunes students themselves with the dataset. Selective Reflection-Tuning (Li et al., 2024a) leverages the teacher LLMs to reflect on and improve existing data, while the student LMs assess and selectively incorporate these improvements, thereby increasing data quality and compatibility with the student LMs.

**Remark 5.** *Black-Box Distillation uses the teacher model's outputs as supervision, but the teacher model's outputs may not cover all possible input scenarios. Thus, understanding how to handle a student model's generalization on unknown data and how to increase data diversity is an area that requires further investigation.*

### 5.2 White-box KD

White-box KD enables the student LM to gain a deeper understanding of the teacher LLM's internal structure and knowledge representations, often resulting in higher-level performance improvements. A representative example is MINILLM (Gu et al., 2024), which is the first work to study distillation from the Open-source generative LLMs. MINILLM uses a reverse Kullback-Leibler divergence objective, which is more suitable for KD on generative language models, to prevent the student model from overestimating the low-probability regions of the teacher distribution, and derives an effective optimization approach to learn the objective. Further, GKD (Agarwal et al., 2024) explores distillation from auto-regressive models, where generative language models are a subset. GKD trains the student using self-generated outputs, incorporating teacher feedback, and allows flexibility in using different loss functions when the student cannot fully replicate the teacher's distribution. Different from the above studies, which focus on learning the teacher distribution, TED (Liang et al., 2023) proposes a task-aware layer-wise distillation method, which designs task-aware filters, which align the hidden representations of the teacher and student models at each intermediate layer, to reduce the knowledge gap between the student and teacher models.

**Remark 6.** *Although white-box distillation allows student LMs to learn the knowledge of teacher LLMs more deeply compared to black-box distillation, currently, open-source LLMs perform worse*

*than closed-source ones, limiting the improvement of student LMs performance in white-box distillation. This is one of the barren factors hindering the development of white-box distillation. A feasible solution is to distill knowledge from closed-source LLMs to open-source LLMs through black-box distillation, and then use white-box distillation to transfer knowledge from open-source LLMs to student LLMs.*

**Remark 7.** *White-box distillation often involves understanding and utilizing the internal structure of LLMs, such as layer connections and parameter settings. A more in-depth exploration of different network structures and interactions between layers can improve the effectiveness of white-box distillation.*

## 6 Low-Rank Factorization

Low-Rank Factorization (Srebro and Jaakkola, 2003) reduces a large matrix into smaller ones to save space and computational effort. For example, it decomposes a large matrix $W$ into two small matrices $U$ and $V$ (i.e., $W \approx UV$), where $U$ is $m \times k$ and $V$ is $k \times n$, with $k$ much smaller than $m$ and $n$. Recent studies try to employ low-rank factorization to compress LLMs and achieve significant success in this regard. For example, LPLR (Saha et al., 2023) compresses weight matrices of LLMs through randomized low-rank and low-precision factorization. Specifically, LPLR approximates the column space of the matrix using random sketching techniques, quantizes these columns, and then projects the original columns onto this quantized space to create two low-rank factors stored in low-precision. ASVD (Yuan et al., 2023b) finds that the activation distribution has an effect on the compression performance. To sovle the problem, ASVD proposes to scale the weight matrix with a diagonal matrix that contains scaling factors corresponding to the activation distribution of the input feature channels. Moreover, ASVD assigns the most suitable compression ratio to different layers by analyzing the singular values distribution in each layer's weight matrix, ensuring minimal loss of model performance during the compression process. Furthermore, Sharma et al. (2024) demonstrate that the performance of LLMs can be significantly improved by applying Layer-Selective Rank Reduction (LASER) to specific layers of Transformer models. LASER involves selectively reducing the rank higher-order components of weight matrices, which is shown to improve the model's handling of rare training data and its resistance to question paraphrasing.

## 7 Challenges and Future Directions

### 7.1 More Advanced Methods

The research on model compression techniques for LLMs is still in its early stages. These compressed LLMs, as demonstrated in prior studies (Frantar and Alistarh, 2023; Liu et al., 2023b; Ho et al., 2023), continue to exhibit a significant performance gap when compared to their uncompressed counterparts. By delving into more advanced model compression methods tailored for LLMs, we have the potential to enhance the performance of these uncompressed LLMs.

### 7.2 Scaling up Model Compression Methods from Other Models

In our paper, we introduce several representative model compression methods for LLMs. However, many classic model compression methods remain prevalent in traditional small models. For example, lottery tickets (Frankle and Carbin, 2019) and parameter sharing (Savarese and Maire, 2019) are widely used model compression methods in small models. These methods still hold significant potential in the era of LLMs. Future work should focus on exploring how to extend these compression methods to LLMs to achieve further compression.

### 7.3 LLM Inference and Deployment

The efficiency of compressed LLMs during deployment is also a significant area for exploration. This involves multiple evaluation metrics, including arithmetic intensity, memory size, and throughput. Furthermore, we can use an analytical tool, the Roofline Model (Williams et al., 2009), to assess the resource efficiency of compressed LLMs on specific hardware. Evaluating the deployment efficiency of compressed LLMs on specific hardware can guide researchers in selecting and analyzing the advantages and disadvantages of various model compression methods and further optimizing these methods.

### 7.4 The Effect of Scaling Law

The scaling law (Kaplan et al., 2020) underscores the significant impact of model size, dataset size, and compute resources on the performance of

LLMs. However, the scaling law presents a fundamental challenge for LLM compression, i.e., there is a trade-off between model size and performance in compressed LLMs. Delving into the mechanisms and theories underpinning the scaling law is crucial for elucidating and potentially overcoming this limitation.

### 7.5 AutoML for LLM Compression

Existing compression techniques have made remarkable progress, but they still heavily depend on manual design. For instance, designing appropriate student architectures for knowledge distillation requires a significant amount of human effort. To reduce this reliance on manual design, a feasible solution is to combine Automated Machine Learning (AutoML) techniques such as Meta-Learning (Finn et al., 2017) and Neural Architecture Search (NAS) (Zoph and Le, 2017) with model compression. By combining with AutoML techniques, model compression can automatically select appropriate hyperparameters and tailor architectures and scales of compressed models, thus minimizing human involvement and lowering the associated costs. Furthermore, AutoML can identify optimal model compression strategies tailored to specific task requirements, thereby further enhancing compression rates without compromising model performance.

### 7.6 Explainability of LLM Compression

Earlier research (Stanton et al., 2021; Xu et al., 2021) has raised significant concerns regarding the explainability of model compression techniques applied to Pre-trained Language Models (PLMs). Notably, these same challenges extend to LLM compression methods as well. For example, CoT-distillation can enhance SLMs' reasoning performance, yet the mechanism through which it imparts CoT ability remains unclear. This challenge underscores the importance of integrating explainability with model compression approaches for the advancement of LLM compression applications. Explainability not only clarifies the changes and trade-offs in the compression process but also enhances efficiency and accuracy. Additionally, interpretability aids in evaluating the compressed model's performance to ensure it aligns with practical requirements.

## 8 Conclusion

In this survey, we have explored model compression techniques for LLMs. Our coverage spanned compression methods, metrics, and benchmark datasets. By diving into LLM compression, we've highlighted its challenges and opportunities. This survey aims to be a valuable reference, providing insights into the current landscape and promoting ongoing exploration of this pivotal topic.

## Acknowledgments

We would like to thank the anonymous reviewers and the action editor for their valuable feedback and discussions. The work of Jian Li is supported partially by National Natural Science Foundation of China (No. 62106257). The work of Yong Liu is supported partially by National Natural Science Foundation of China (No. 62076234), Beijing Outstanding Young Scientist Program (No. BJJWZYJH012019100020098), the Unicom Innovation Ecological Cooperation Plan, and the CCF-Huawei Populus Grove Fund.

## References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. Generalized knowledge distillation for auto-regressive language models. In *The Twelfth International Conference on Learning Representations*.

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. 2024. Fluctuation-based adaptive structured pruning for large language models. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20–27, 2024, Vancouver, Canada*, pages 10865–10873. AAAI Press. https://doi.org/10.1609/aaai.v38i10.28960

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. 2024. SliceGPT: Compress large language models by deleting rows and columns. In *The Twelfth International Conference on Learning Representations*.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17–24, 2023*, pages 18392–18402. IEEE. https://doi.org/10.1109/CVPR52729.2023.01764

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutsk, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual*.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. 2023. QuIP: 2-bit quantization of large language models with guarantees. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Jack Choquette, Wishwesh Gandhi, Olivier Giroux, Nick Stam, and Ronny Krashinsky. 2021. NVIDIA A100 tensor core GPU: Performance and innovation. *IEEE Micro*, 41(2):29–35. https://doi.org/10.1109/MM.2021.3061394

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. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

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

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 – December 9, 2022*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*.

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

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A survey for in-context learning. *CoRR*, abs/2301.00234.

Dayou Du, Yijia Zhang, Shijie Cao, Jiaqi Guo, Ting Cao, Xiaowen Chu, and Ningyi Xu. 2024. Bitdistiller: Unleashing the potential of sub-4-bit llms via self-distillation. *CoRR*, abs/2402.10631. https://doi.org/10.18653/v1/2024.acl-long.7

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse,

trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019*. OpenReview.net.

Elias Frantar and Dan Alistarh. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *Advances in Neural Information Processing Systems*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10421–10430. PMLR.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361. https://doi.org/10.1162/tacl_a_00370

R. M. Gray and D. L. Neuhoff. 1998. Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2383. https://doi.org/10.1109/18.720541

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Cong Guo, Jiaming Tang, Weiming Hu, Jingwen Leng, Chen Zhang, Fan Yang, Yunxin Liu, Minyi Guo, and Yuhao Zhu. 2023. Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization. In *Proceedings of the 50th Annual International Symposium on Computer Architecture, ISCA 2023, Orlando, FL, USA, June 17–21, 2023*, pages 3:1–3:15. ACM. https://doi.org/10.1145/3579371.3589038

Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9–14, 2023*, pages 14852–14882. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.830

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. 2024. Kvquant: Towards 10 million context length LLM inference with KV cache quantization. *CoRR*, abs/2401.18079.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas

Pfister. 2023. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9–14, 2023*, pages 8003–8017. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-acl.507

Yukun Huang, Yanda Chen, Zhou Yu, and Kathleen R. McKeown. 2022. In-context learning distillation: Transferring few-shot learning ability of pre-trained language models. *CoRR*, abs/2212.10670.

Hyesung Jeon, Yulhwa Kim, and Jae-Joon Kim. 2024. L4Q: Parameter efficient quantization-aware training on large language models via lora-wise LSQ. *CoRR*, abs/2402.04902.

Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. 2023. Lion: Adversarial distillation of proprietary large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3134–3154, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.189

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. 2024. Shortened llama: A simple depth pruning for large language models. *ICLR Workshop on Mathematical and Empirical Understanding of Foundation Models (ME-FoMo)*.

Jeonghoon Kim, Jung Hyun Lee, Sungdong Kim, Joonsuk Park, Kang Min Yoo, Se Jung Kwon, and Dongsoo Lee. 2023a. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. 2023b. Squeezellm: Dense-and-sparse quantization. *CoRR*, abs/2306.07629.

Yann LeCun, John S. Denker, and Sara A. Solla. 1989. Optimal brain damage. In *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27–30, 1989]*, pages 598–605. Morgan Kaufmann.

Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. 2024. OWQ: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20–27, 2024, Vancouver, Canada*, pages 13355–13364. AAAI Press. https://doi.org/10.1609/aaai.v38i12.29237

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. 2024a. Selective reflection-tuning: Student-selected data recycling for LLM instruction-tuning. *CoRR*, abs/2402.10110.

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. 2024b. Explanations from large language models make small reasoners better. In *2nd Workshop on Sustainable AI*.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Bin Sun, Xinglin Wang, Heda Wang, and Kan Li. 2024c. Turning dust into gold: Distilling complex reasoning capabilities from llms by leveraging negative data. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20–27, 2024, Vancouver, Canada*, pages 18591–18599. AAAI Press. https://doi.org/10.1609/aaai.v38i17.29821

Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. 2023a. Loftq: Lora-fine-tuning-aware quantization for large language models. *CoRR*, abs/2310.08659.

Yun Li, Lin Niu, Xipeng Zhang, Kai Liu, Jianchen Zhu, and Zhanhui Kang. 2023b. E-sparse: Boosting the large language model inference through entropy-based N: M sparsity. *CoRR*, abs/2310.15929.

Zhuo Li, Hengyi Li, and Lin Meng. 2023c. Model compression for deep neural networks: A survey. *Computers*, 12(3):60. https://doi.org/10.3390/computers12030060

Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2023. Less is more: Task-aware layer-wise distillation for language model compression. In *International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 20852–20867. PMLR.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. 2023. AWQ: Activation-aware weight quantization for LLM compression and acceleration. *CoRR*, abs/2306.00978.

Shih-yang Liu, Zechun Liu, Xijie Huang, Pingcheng Dong, and Kwang-Ting Cheng. 2023a. LLM-FP4: 4-bit floating-point quantized transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 592–605, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.39

Yuxuan Liu. 2024. Learning to reason with autoregressive in-context distillation. In *The Second Tiny Papers Track at ICLR 2024*.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023b. LLM-QAT: Data-free quantization aware training for large language models. *CoRR*, abs/2305.17888.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. 2024. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. *CoRR*, abs/2402.02750.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adámek, Eric Malmi, and Aliaksei Severyn. 2023. Teaching small language models to reason. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2023, Toronto, Canada, July 9–14, 2023*, pages 1773–1781. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-short.151

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330. https://doi.org/10.21236/ADA273556

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*. OpenReview.net.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 – November 4, 2018*, pages 2381–2391. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1260

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. 2019. Importance estimation for neural network pruning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16–20, 2019*, pages 11264–11272. Computer Vision Foundation / IEEE. https://doi.org/10.1109/CVPR.2019.01152

OpenAI. 2024. Gpt-4 technical report.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller,

Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7–12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics. https://doi.org/10.18653/v1/P16-1144

Gunho Park, Baeseong park, Minsub Kim, Sungjae Lee, Jeonghoon Kim, Beomseok Kwon, Se Jung Kwon, Byeongwook Kim, Youngjoo Lee, and Dongsoo Lee. 2024. LUT-GEMM: Quantized matrix multiplication based on LUTs for efficient inference in large-scale generative language models. In *The Twelfth International Conference on Learning Representations*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:140:1–140:67.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pages 525–542. Springer. https://doi.org/10.1007/978-3-319-46493-0_32

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866. https://doi.org/10.1162/tacl_a_00349

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

Pedro Savarese and Michael Maire. 2019. Learning implicitly recurrent cnns through parameter sharing. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019*. OpenReview.net.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, 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, et al. 2022. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100.

Hang Shao, Bei Liu, and Yanmin Qian. 2024a. One-shot sensitivity-aware mixed sparsity pruning for large language models. In *ICASSP 2024 – 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11296–11300. https://doi.org/10.1109/ICASSP48485.2024.10445737

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng

Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2024b. Omniquant: Omnidirectionally calibrated quantization for large language models. In *The Twelfth International Conference on Learning Representations*.

Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. 2024. The truth is in there: Improving reasoning with layer-selective rank reduction. In *The Twelfth International Conference on Learning Representations*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9–14, 2023*, pages 7059–7073. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.findings-acl.441

Nathan Srebro and Tommi S. Jaakkola. 2003. Weighted low-rank approximations. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21–24, 2003, Washington, DC, USA*, pages 720–727. AAAI Press.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Johan Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew M. Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, Cesar Ferri, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Christopher Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, C. Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Dylan Schrader, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodolà, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Xinyue Wang, Gonzalo Jaimovitch-Lopez, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Francis Anthony Shevlin, Hinrich Schuetze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kocon, Jana Thompson, Janelle Wingfield, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan

Batchelder, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Guerr, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh Dhole, Kevin Gimpel, Kevin Omondi, Kory Wallace Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros-Colón, Luke Metz, Lütfi Kerem Senel, Maarten Bosma, Maarten Sap, Maartje Ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramirez-Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael Andrew Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michal Swedrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mitch Walker, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T., Nanyun Peng, Nathan Andrew Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nicole Martinez, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter W. Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Milkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramon Risco, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan Le Bras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Russ Salakhutdinov, Ryan Andrew Chi, Seungjae Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel Stern Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima Shammie Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven Piantadosi, Stuart Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsunori Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Venkatesh Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A. Alemi, and Andrew Gordon Wilson. 2021. Does knowledge distillation really work? In *Advances in Neural*

*Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6–14, 2021, virtual*, pages 6906–6919.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1421

Sandeep Tata and Jignesh M. Patel. 2003. Piqa: An algebra for querying protein data sets. In *Proceedings of the 15th International Conference on Scientific and Statistical Database Management (SSDBM 2003), 9–11 July 2003, Cambridge, MA, USA*, pages 141–150. IEEE Computer Society. https://doi.org/10.1109/SSDM.2003.1214975

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

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

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax

Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. 2023a. SCOTT: Self-consistent chain-of-thought distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9–14, 2023*, pages 5546–5558. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.304

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2023b. 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*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023c. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1–5, 2023*. OpenReview.net.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023d. 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), ACL 2023, Toronto, Canada, July 9–14, 2023*, pages 13484–13508. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.754

Yue Wang, Weishi Wang, Shafiq R. Joty, and Steven C. H. Hoi. 2021. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7–11 November, 2021*, pages 8696–8708. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.685

Zhaoyang Wang, Shaohan Huang, Yuxuan Liu, Jiahai Wang, Minghui Song, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. 2023e. Democratizing reasoning ability: Tailored learning from large language model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6–10, 2023*, pages 1948–1966. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.120

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. 2023. 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*, pages 1648–1665, Singapore. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.emnlp-main.102

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Miles Williams and Nikolaos Aletras. 2023. How does calibration data affect the post-training pruning and quantization of large language models? *CoRR*, abs/2311.09755.

Samuel Williams, Andrew Waterman, and David A. Patterson. 2009. Roofline: An insightful visual performance model for multicore architectures. *Communications of the ACM*, 52(4):65–76. https://doi.org/10.1145/1498765.1498785

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Aji. 2024. LaMini-LM: A diverse herd of distilled models from large-scale instructions. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 944–964, St. Julian's, Malta. Association for Computational Linguistics.

Haojun Xia, Zhen Zheng, Yuchao Li, Donglin Zhuang, Zhongzhu Zhou, Xiafei Qiu, Yong Li, Wei Lin, and Shuaiwen Leon Song. 2023. Flash-llm: Enabling cost-effective and highly-efficient large generative model inference with unstructured sparsity. *Proceedings of the VLDB Endowment*, 17(2):211–224. https://doi.org/10.14778/3626292.3626303

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations*.

Patrick Xia, Shijie Wu, and Benjamin Van Durme. 2020. Which *bert? A survey organizing contextualized encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16–20, 2020*, pages 7516–7533. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.608

Guangxuan Xiao, Ji Lin, Mickaël Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning, ICML 2023, 23–29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Ke Xu, Julian J. McAuley, and Furu Wei. 2021. Beyond preserved accuracy: Evaluating loyalty and robustness of BERT compression. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7–11 November, 2021*, pages 10653–10659. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.832

Yuzhuang Xu, Xu Han, Zonghan Yang, Shuo Wang, Qingfu Zhu, Zhiyuan Liu, Weidong Liu, and Wanxiang Che. 2024. Onebit: Towards extremely low-bit large language models. *CoRR*, abs/2402.11295.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In *NeurIPS*.

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

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

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. 2023b. ASVD: Activation-aware singular value decomposition for compressing large language models. *CoRR*, abs/2312.05821.

Yuxuan Yue, Zhihang Yuan, Haojie Duanmu, Sifan Zhou, Jianlong Wu, and Liqiang Nie. 2024. Wkvquant: Quantizing weight and key/value cache for large language models gains more. *CoRR*, abs/2402.12065.

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

Yuxin Zhang, Lirui Zhao, Mingbao Lin, Sun Yunyun, Yiwu Yao, Xingjia Han, Jared Tanner, Shiwei Liu, and Rongrong Ji. 2024. Dynamic sparse no training: Training-free fine-tuning for sparse LLMs. In *The Twelfth International Conference on Learning Representations*.

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. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 – 16, 2023*.

Xuekai Zhu, Biqing Qi, Kaiyan Zhang, Xinwei Long, Zhouhan Lin, and Bowen Zhou. 2024. PaD: Program-aided distillation can teach small models reasoning better than chain-of-thought fine-tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2571–2597, Mexico City, Mexico. Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.142

Barret Zoph and Quoc V. Le. 2017. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*. OpenReview.net.