A Survey on Small Language Models

Chien Van Nguyen[1*], **Xuan Shen**[2*], **Ryan Aponte**[3*], **Yu Xia**[4], **Samyadeep Basu**[5],
Zhengmian Hu[5], **Jian Chen**[6], **Mihir Parmar**[7], **Sasidhar Kunapuli**, **Joe Barrow**[8],
Junda Wu[4], **Ashish Singh**[9], **Yu Wang**[1], **Jiuxiang Gu**[8], **Nesreen K. Ahmed**[10],
Nedim Lipka[8], **Ruiyi Zhang**[8], **Xiang Chen**[8], **Tong Yu**[8], **Sungchul Kim**[8], **Hanieh Deilamsalehy**[8],
Namyong Park[11], **Mike Rimer**, **Zhehao Zhang**[12], **Huanrui Yang**[13],
Puneet Mathur[8], **Gang Wu**[8], **Franck Dernoncourt**[8], **Ryan A. Rossi**[8], **Thien Huu Nguyen**[1]

[1]University of Oregon, [2]Northeastern University, [3]Carnegie Mellon University
[4]University of California, San Diego, [5]University of Maryland, College Park
[6]State University of New York at Buffalo, [7]Arizona State University
[8]Adobe Research, [9]University of Massachusetts Amherst, [10]Intel AI Research
[11]Meta AI, [12]Dartmouth College, [13]University of Arizona

`{chienn,thienn}@uoregon.edu`, `{dernonco,ryrossi@}@adobe.com`

Abstract

Small Language Models (SLMs) have become increasingly important due to their efficiency and performance to perform various language tasks with minimal computational resources, making them ideal for various settings including on-device, mobile, edge devices, among many others. In this article, we present a comprehensive survey on SLMs, focusing on their architectures, training techniques, and model compression techniques. We propose a novel taxonomy for categorizing the methods used to optimize SLMs, including model compression, pruning, and quantization techniques. We summarize the benchmark datasets that are useful for benchmarking SLMs along with the evaluation metrics commonly used. Additionally, we highlight key open challenges that remain to be addressed. Our survey aims to serve as a valuable resource for researchers and practitioners interested in developing and deploying small yet efficient language models.

1 Introduction

Although large language models (LLMs) have demonstrated impressive performance on a wide array of benchmarks and real-world situations, their success comes at significant cost. LLMs are resource-intensive to train and run, requiring significant compute *and* data. This often means that they are run on centralized and specialized hardware for both training and inference.

As a response to these challenges, there has been a growing interest in small language models (SLMs). Small language models aim to retain the accuracy and/or adaptability of large language models, while being subject to some constraint(s), such as training or inference hardware, data availability, bandwidth, or generation time. Improving model performance relative to these constraints can then improve downstream goals such as privacy, cost, or the ability to run on consumer devices.

The inherent difficulty of a survey of small language models is that the definitions of "small" and "large" are a function of both context and time. GPT-2, a "large language model" in 2019 at 1.5B parameters, is smaller than many "small" language models covered in this survey. However, although the scale changes, the goals of training small language models remain relatively stable.

In this survey, we explore the architectures, training, and model compression techniques that enable the building and inferencing of SLMs. In addition, we summarize the benchmark datasets and evaluation metrics commonly used in evaluating SLM performance. For this, we propose a novel taxonomy for organizing the methods along two axes:

- The **techniques** used in pre-processing (model architecture), training, and post-processing (model compression) SLMs; and
- The **constraints** the technique is attempting to optimize for, such as inference compute, training time, speed, etc.

An overview of these axes can be found in Table 1 (techniques) and Table 2 (constraints).

It is important to note that progress on any one of these goals does not necessarily imply progress on the others. In fact, there are often trade-offs between them. For instance, memory-efficient

---

[*] These authors contributed equally to this work.

Proceedings of Recent Advances in Natural Language Processing,pages 807–821
Varna, Sep 8–10, 2025
https://doi.org/10.26615/978-954-452-098-4-093

training methods like quantization-aware training (Dettmers et al., 2022a, 2024) are often slower than their full-precision counterparts. However, by using mixed precision to represent the weights and gradients, they allow training or finetuning using less memory. Finally, although there have been several recent surveys on LLMs and their learning methods (Rogers et al., 2020; Min et al., 2021; Zhu et al., 2023; Shen et al., 2023), to the best of our knowledge, this is the first survey focused on SLMs.

Organization of the Survey. This survey is structured into three main sections, each covering a key aspect of optimizing SLMs. **Section 2** focuses on model architectures, including lightweight designs, efficient self-attention approximations, and neural architecture search to efficiently build smaller models. **Section 3** covers efficient pre-training and fine-tuning techniques to enhance performance for SLMs while managing resource constraints. **Section 4** explores model compression techniques, such as pruning, quantization, and knowledge distillation, which reduce model size and latency without sacrificing significant accuracy. **Section 5** introduces an overview of benchmark datasets and evaluation metrics, providing a comprehensive framework for assessing the effectiveness of these methods. **Section 6** discusses the applications that are enabled by SLMs, organized by constraints.

Summary of Main Contributions. The key contributions of this work are as follows:

- A comprehensive survey of existing work on small language models for practitioners. We also survey the problem settings, evaluation metrics, and datasets used in the literature.
- We introduce a few intuitive taxonomies for SLMs and survey existing work using these taxonomies.
- We identify important applications, open problems, and challenges of SLMs for future work to address.

2 Model Architectures

This section discusses the architectural designs for developing SLMs. Specifically, we cover lightweight architectures (Section 2.1), efficient self-attention approximations (Section 2.2), and neural architecture search (Section 2.3).

2.1 Lightweight Architectures

Lightweight language model architectures are designed to achieve efficient performance with fewer parameters and reduced computational overhead, which is ideal for deployment on resource-constrained devices such as mobile phones, edge devices, and embedded systems. Representative lightweight models often follow the encoder-only and decoder-only architectures.

Lightweight encoder-only architectures are mostly optimized versions of BERT (Devlin et al., 2019). For example, MobileBERT (Sun et al., 2020) introduces an inverted-bottleneck structure to maintain a balance between self-attention and feed-forward networks, achieving a 4.3x size reduction and a 5.5x speedup compared to the base version of BERT. DistilBERT (Sanh, 2019) and TinyBERT (Jiao et al., 2019) achieve more than 96% of BERT's performance while being less than 45% smaller and 60% faster by leveraging language modeling, distillation, and cosine-distance losses.

Lightweight decoder-only architectures are designed to scale down autoregressive language models, such as GPT (Radford et al., 2018, 2019) and the LLaMA series (Touvron et al., 2023), into compact and efficient models. These models emphasize knowledge distillation, memory overhead optimization, parameter sharing, embedding sharing to enhance efficiency and scalability. BabyLLaMA (Timiryasov and Tastet, 2023a) and BabyLLaMA-2 (Tastet and Timiryasov, 2024) distill knowledge from multiple teachers into a 58M-parameter model and a 345M-parameter model respectively, demonstrating that distillation can exceed teacher models' performance particularly under data-constrained conditions. TinyLLaMA (Zhang et al., 2024), with only 1.1B parameters, achieves high efficiency by optimizing memory overhead, e.g., via FlashAttention (Dao et al., 2022), while maintaining competitive performance for various downstream tasks. MobilLLaMA (Thawakar et al., 2024) applies a parameter-sharing scheme that reduces both pre-training and deployment costs, introducing a 0.5B-parameter model for resource-constrained devices. MobileLLM (Liu et al., 2024b) investigates the impact of model depth (i.e., number of layers) and width (i.e., number of heads) on performance, effectively conducting a targeted architecture search within a smaller parameter range for language models with millions of parameters.

Technique	General Mechanism	Training Compute	Dataset Size	Inference Runtime	Memory	Storage Space	Latency
3***Model Architectures** (Sec. 2)	Lightweight Models (Sec. 2.1)	✓		✓	✓		✓
	Efficient Self-Attention (Sec. 2.2)	✓		✓	✓		✓
	Neural Arch. Search (Sec. 2.3)			✓	✓	✓	
3***Training Techniques** (Sec. 3)	Pre-training (Sec. 3.1)	✓	✓	✓	✓	✓	
	Finetuning (Sec. 3.2)	✓	✓				
4***Model Compression** (Sec. 4)	Pruning (Sec. 4.1)			✓	✓	✓	✓
	Quantization (Sec. 4.2)			✓	✓	✓	✓
	Knowledge Distillation (Sec. 4.3)		✓				

Table 1: General techniques used for optimizing small language models, categorized by type of model optimization and most central constraints they address.

2.2 Efficient Self-Attention Approximations

Deploying large language models can be challenging due to the substantial number of parameters in the self-attention layers, as well as the computational cost associated with self-attention. In this section, we discuss strategies towards decreasing this computational cost which can ultimately be useful in creating small language models.

Reformer (Kitaev et al., 2020) improves the complexity of the self-attention from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ by replacing the dot product attention with one which uses locality-sensitivity hashing. Roy et al. (2021) use a sparse routing module based on an online k-means clustering, which reduces the complexity of the attention computation.

To reduce the computational quadratic complexity of the self-attention layer from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$, several works, including (Wang et al., 2020a; Katharopoulos et al., 2020; Xiong et al., 2021; Beltagy et al., 2020), propose linear attention mechanisms. In particular, (Katharopoulos et al., 2020) express self-attention as a linear dot-product of kernel feature maps, thus reducing the quadratic complexity. The authors further show that transformers with this linear attention mechanism can be viewed as a recurrent neural network which enables faster inference. Building on these foundations, recent advancements have led to more advanced architectures. Notable examples include Mamba (Gu and Dao, 2023; Dao and Gu, 2024), which introduces a selective state space model with input-dependent transitions, and RWKV (Peng et al., 2023a), which combines elements of transformers and RNNs with a linear attention mechanism. These models not only achieve linear time and space complexity but also demonstrate competitive performance across various tasks. This ongoing trend towards efficient sequence modeling architectures aims to maintain the expressiveness of attention-based models while significantly reducing computational complexity.

Hybrid models that combine the efficiency of SSMs with the recall capabilities of attention mechanisms have also gained attention. MambaFormer (Park et al., 2024) interleaves Mamba-based SSM layers with attention modules, improving in-context learning capabilities. Similarly, Jamba (Lieber et al., 2024) employ sequentially stacked Mamba-Attention layers to enhance performance on long-sequence tasks. Samba (Ren et al., 2024) extends this idea by introducing a block structure that alternates between Mamba, MLP, and SWA layers, achieving constant throughput as sequence lengths increase. Hymba (Dong et al., 2024) further innovates with a hybrid-head architecture combining attention for recall and SSMs for efficient summarization, achieving state-of-the-art efficiency and accuracy for small LMs. These hybrid designs illustrate the effectiveness of combining complementary mechanisms to address the limitations of standalone architectures.

2.3 Neural Architecture Search Techniques

This section discusses automated methods to discover the most efficient model architectures for specific tasks and hardware constraints. Previous research has primarily concentrated on Neural Architecture Search (NAS) for vision tasks (Tan and Le, 2019; Zoph and Le, 2016; Wu et al., 2019; Guo et al., 2020) and BERT models (Xu et al., 2021; Jawahar et al., 2023; Ganesan et al., 2021), as these models have comparatively fewer parameters, which reduces the cost of the search process for efficient architectures. However, models with over a billion parameters pose a significant challenge in searching for smaller, more efficient models.

3 Training Techniques

This section explores training techniques specifically optimized for Small Language Models (SLMs), with a primary focus on how these methods enable efficient training within limited resource environments. A key consideration is the interplay between model size and bit-precision, as a model with a large parameter count at a very low bit-precision may have a similar memory footprint to a model with fewer parameters at a higher bit-precision.

3.1 Low-Resource Pre-training

Low-Precision Training SLMs are designed to operate under strict memory constraints. Therefore, training with extremely low precision allows these models to fit within limited resources. This approach enables significant memory savings, allowing for larger batch sizes or more complex models within the same memory footprint. Automatic Mixed Precision (AMP) with FP16 (Micikevicius et al., 2018) has been widely adopted for its efficiency, but its limited dynamic range can lead to numerical instability. BFLOAT16 (Burgess et al., 2019), with its broader dynamic range, offers greater stability and is particularly effective for smaller batch sizes. Further efficiency gains can be achieved with FP8 formats, supported by hardware like NVIDIA's Hopper architecture. These formats reduce memory usage and accelerate computation but require advanced techniques, such as dynamic scaling, stochastic rounding, and hybrid formats, to maintain numerical stability. Innovations like FP8-LM (Peng et al., 2023b) and methods for scaling FP8 training to trillion-token LLMs (Fishman et al., 2024) demonstrate the effectiveness of these approaches. For even greater savings, integer-based training with INT8 and INT4 formats offers compelling benefits. Techniques like Jetfire (Xi et al., 2024) and INT4 training (Xi et al., 2023) rely on precise quantization to minimize accuracy loss. Emerging methods such as BitNet (Wang et al., 2023) and BitNet-1.58 (Ma et al., 2024), which use 1-bit weights and low-bit activations, achieve extreme memory reductions. It is important to note that the choice of precision—ranging from FP16 to INT4 or 1-bit should be guided by the trade-offs between hardware compatibility, training speed, and model accuracy.

Parallelism Training: SLMs are typically pretrained across multiple machine nodes to leverage distributed computing resources efficiently. Several system-level optimization techniques have been developed to this end. Zero Redundancy Data Parallelism (ZeRO) (Rajbhandari et al., 2020) offers three progressive stages of optimization, each partitioning more training states across devices: ZeRO-1 partitions optimizer states, ZeRO-2 adds gradient partitioning, and ZeRO-3 further partitions model parameters. PyTorch's Fully Sharded Data Parallel (FSDP) (Zhao et al., 2023) implements similar concepts. These parallelism techniques enable training with larger batch sizes, significantly improving efficiency and scalability for SLMs.

3.2 Fine-tuning Techniques

Fine-tuning on smaller, task-specific datasets allows models to leverage the knowledge gained during pre-training, enabling them to excel in specialized tasks or domains. Fine-tuning techniques are designed to address challenges like limited computing resources, data quality, availability, and robustness, ensuring efficient adaptation to new tasks without extensive retraining.

3.2.1 Parameter-Efficient Fine-Tuning

Parameter-Efficient Fine-Tuning (PEFT) updates a small subset of parameters or adds lightweight modules, keeping most of the pre-trained model's parameters fixed. This approach reduces computational costs during SLM fine-tuning, preserves the model's pre-trained knowledge, minimizes overfitting, and improves flexibility.

LoRA uses low-rank decomposition (Hu et al., 2021), Prompt Tuning (Lester et al., 2021) inserts learnable prompts into inputs, and Llama-Adapter (Zhang et al., 2023b; Gao et al., 2023) adds prompts to LLaMA's attention blocks. Dynamic Adapters

(Kong et al., 2024; Feng et al., 2024; Gou et al., 2023; Liu et al., 2023b; Luo et al., 2024) automatically combine multiple adapters as a mixture-of-experts model to enable multi-tasking and prevent forgetting (Han et al., 2024; Yang et al., 2024).

To further optimize PEFT, some tools combine these techniques with fused kernels for improved performance and resource efficiency. For example, Unsloth (Daniel Han and team, 2023) is a cutting-edge tool that enables fine-tuning of large-scale models up to 5x faster, while reducing memory usage by as much as 80%. By leveraging innovations such as dynamic 4-bit quantization and gradient checkpointing, Unsloth accelerates training without sacrificing accuracy.

3.2.2 Data Augmentation

Data augmentation increases the complexity, diversity and quality of training data, leading to improved generalization and performance on downstream tasks. AugGPT (Dai et al., 2023) rephrases training samples using ChatGPT. Evol-Instruct (Xu et al., 2023) uses multistep revisions to generate diverse, open-domain instructions with increased complexity. Reflection-tuning (Li et al., 2023a, 2024) enhances data quality and instruction-response consistency for instruction tuning by refining both instructions and responses using GPT-4 based on predefined criteria. FANNO (Zhu et al., 2024) augments instructions and generates responses by incorporating external knowledge sources through retrieval-augmented generation. LLM2LLM (Lee et al., 2024b) generates more hard samples based on model prediction on training data during training.

Data augmentation is also effective for synthesizing new data when training data is limited, such as for low-resource languages (Whitehouse et al., 2023), medical and clinical applications (Chintagunta et al., 2021), and privacy-sensitive data (Song et al., 2024), enabling models to generalize better and perform more robustly in constrained settings.

4 Model Compression Techniques

Model compression techniques focus on reducing the size and complexity of large pre-trained language models while maintaining their performance. As a result, these methods are a key approach to deriving SLMs from LLMs. In this section, we propose a taxonomy for model compression that categorizes such techniques by whether they perform pruning (Section 4.1), quantization (Section 4.2), or knowledge distillation (Section 4.3).

4.1 Pruning Techniques

Weight pruning is a model optimization technique that reduces the number of parameters to enhance computational efficiency and lower memory usage, all while maintaining performance levels. We differentiate between two major approaches for pruning: unstructured pruning and structured pruning.

Unstructured pruning removes less significant individual weights, offering fine-grained control and flexibility in reducing model size. For example, to perform irregular pruning on large language models, SparseGPT (Frantar and Alistarh, 2023) reformulates the pruning task as a sparse regression problem, optimizing both the remaining and pruned weights using a layer-wise approximate regression solver. SparseGPT can efficiently handle large-scale models like OPT-175B and BLOOM-176B. Additionally, (Boža, 2024) integrates the ADMM (Boyd et al., 2011) algorithm for weight updates to further mitigate pruning errors. Wanda (Sun et al., 2023) incorporates both weights and activations into consideration during pruning process, and eliminates the need of weight updates. In addition, the n:m pruning strategy (Zhou et al., 2021) brings unstructured pruning to model acceleration by pruning exactly n weights out of every m, balancing pruning flexibility and computational efficiency for significant speedups. NVIDIA's TensorRT leverages such sparse patterns to optimize memory access and reduce computational loads, accelerating inference on GPUs, particularly hardware like the A100. Additionally, the n:m sparse pattern can also be applied in edge AI applications on NVIDIA Jetson Nano to enhance power efficiency and optimize model size. Finally, unstructured pruning often results in sparse matrices requiring specialized hardware or algorithms to maximize computational benefits (Frantar and Alistarh, 2023).

Structured pruning (Wang et al., 2020b; Santacroce et al., 2023; Ma et al., 2023; Tao et al., 2023; Xia et al., 2024; Kurtić et al., 2024) aims to compress LLMs while maintaining performance by removing groups of parameters in a structured manner, which enables more efficient hardware implementation. A major direction in this approach concerns the sparsity of neurons in the model. For instance, Li et al. (2023b) observes prevalent sparsity in feed-forward networks. Liu et al. (2023e) proposes using small neural networks for dynamic pruning based on input, termed "contextual sparsity". Mirzadeh et al. (2024) change the activation functions in pre-trained models to ReLU and fine-tune to improve activation sparsity.

Recent work has also addressed the redundancy in the Transformer architecture to achieve reduction of GPU memory usage and speed enhancement (Michel et al., 2019; Voita et al., 2019; Ge et al., 2024). For example, Sajjad et al. (2023); Xia et al. (2022) investigates the layer redundancy for effective structured pruning. We also highlight input-dependent pruning methods, such as contextual sparsity (Liu et al., 2023e) and FastGen (Ge et al., 2024), which should be considered along with the challenges of efficient implementation for optimizing computation and memory.

4.2 Quantization

Quantization is widely adopted to compress LLMs with vast parameter counts. The GPTQ (Frantar et al., 2022) focuses on layer-wise weight-only quantization, using inverse Hessian matrices to minimize the reconstruction error. To fully leverage the benefits of fast integer matrix multiplication, more quantization methods (Liu et al., 2023a; Dettmers et al., 2022b; Kim et al., 2023; Xiao et al., 2023; Yao et al., 2022; Lin et al., 2024; Liu et al., 2023d, 2024a, 2023c; Shao et al., 2023) that quantize both weights and activations are increasingly being adopted for LLMs. AWQ (Lin et al., 2024) and ZeroQuant (Yao et al., 2022) take activation into account to assess the importance of weights, enabling more effective optimization for weight quantization. In addition, for K/V Cache Quantization (Hooper et al., 2024; Liu et al., 2024c; Yue et al., 2024), Key-Value cache is specifically quantized for enabling efficient long-sequence length inference.

Another challenge of activation quantization lies in the outliers that fall outside the typical activation distribution. SmoothQuant (Xiao et al., 2023) smoothes activation outliers by migrating quantization difficulty from activations to weights. SpinQuant (Liu et al., 2024a) introduces rotation matrices to transform outliers into a new space. Recently, quantization-aware training (QAT) methods, such as LLM-QAT (Liu et al., 2023d) and EdgeQAT (Shen et al., 2024b), have gained attention due to the strong performance. Both methods adopt distillation with float16 models to recover the quantizationi error. We also note recent work (Shen et al., 2024a,b; Zeng et al., 2024) that implements the quantized LLMs on mobile devices and FPGAs to demonstrate the effectiveness and efficiency of the weight and activation quantization for LLMs.

4.3 Knowledge Distillation Techniques

In its classical form, knowledge distillation (Hinton et al., 2015) involves training an efficient model, known as the "student," to replicate the behavior of a larger, more complex model, referred to as the "teacher." In this section, we particularly focus on distillation strategies from one or multiple white-box teacher language model to a target student language model.

Babyllama (Timiryasov and Tastet, 2023b) is among the first to develop a compact 58M parameter language model using a Llama model as the teacher. A key finding of this work is that distillation from a robust teacher can outperform traditional pre-training on the same dataset. In a similar vein, (Gu et al., 2024) introduce modifications in the distillation loss, which enables the student models to generate better quality responses with improved calibration and lower exposure bias. Sequence-level distillation loss can also be improved by using a generalized version of f-divergences as shown in (Wen et al., 2023). Liang et al. (2023) extend layer-wise distillation strategies for language models by using task-aware filters which distill only the task specific knowledge from the teacher. Recent works (Wan et al., 2024a,b) show that multiple language models can be fused as a teacher towards distilling knowledge into small language models by strategically merging their output probability distributions.

One of the issues in knowledge distillation for language models is that the distillation strategies are primarily effective when (1) the teacher and the student language model share the same tokenizer and (2) the teacher's pre-training data is available. Boizard et al. (2024) addresses this issue by introducing an universal logit distillation loss inspired from the optimal transport literature. Often distillation is also combined with pruning techniques towards creating smaller language models. For example, (Sreenivas et al., 2024; Muralidharan et al., 2024) show that an iterative step of pruning a large language model followed by retraining with distillation losses, can enable strong smaller models.

Setting	Constraints	Datasets	Metrics
Efficient Inference	Latency	SuperGLUE (Sarlin et al., 2020), SQuAD (Rajpurkar et al., 2016), TriviaQA (Joshi et al., 2017), CoQA (Reddy et al., 2019), Natural Questions (NQ) (Kwiatkowski et al., 2019)	Inference Time (Narayanan et al., 2023), Throughput (Arora et al., 2024)
On-device/Mobile	Memory	TinyBERT (Jiao et al., 2020) and OpenOrca (Lian et al., 2023)	Peak Memory Usage (Lee et al., 2024a), Memory Footprint, Compression Ratio (Cao et al., 2024)
Privacy-Preserving	Privacy	PrivacyGLUE (Shankar et al., 2023), MIMIC (Johnson et al., 2020)	Privacy Budget (Yu et al., 2024), Noise Level (Havrilla et al., 2024)
Energy-Efficient AI	Energy Optimization	-	Energy Efficiency Ratio (Stojkovic et al., 2024), Thermal Efficiency, Idle Power Consumption (Patel et al., 2024)

Table 2: Overview of Settings, Constraints, and Metrics.

Recent advancements have explored methods beyond traditional label distillation by incorporating additional supervision during the distillation process to create smaller language models. Hsieh et al. (2023) find that using “rationales” as an additional source of supervision during distillation makes it more sample-efficient. Moreover, the authors find that the distilled model outperforms large-language models on commonly used NLI, Commonsense QA and arithmetic reasoning benchmarks. In a similar vein, (Dai et al., 2024; Magister et al., 2023; Ho et al., 2023; Fu et al., 2023) distill the reasoning chain from a larger language model to a smaller language model along with the label information. Such distilled models have been shown to possess improved arithmetic, multi-step math, symbolic and commonsense reasoning abilities.

5 Evaluation

Table 2 presents different evaluation settings along with their corresponding datasets and metrics for SLMs. In this section, we focus on the evaluation metrics for SLMs. These settings and metrics are organized according to the constraints they address for SLMs.

Latency Two key metrics to evaluate latency are inference time (Narayanan et al., 2023) and throughput (Arora et al., 2024). Inference time measures how quickly a model can process input and generate an output, which is crucial for user-facing applications that require immediate feedback. Throughput, on the other hand, evaluates the number of tokens or samples a model can process in a given period, making it especially relevant for large-scale tasks or time-sensitive applications.

Memory When deploying models in memory-constrained environments, memory efficiency becomes a primary consideration. Metrics such as peak memory usage (Lee et al., 2024a) capture the highest amount of memory the model consumes during inference. Similarly, memory footprint and compression ratio (Cao et al., 2024) are used to measure how compact a model is and the efficiency of the compression techniques applied, enabling models to operate within memory constraints without sacrificing performance.

Privacy Privacy budget (Yu et al., 2024), a measure rooted in differential privacy, quantifies the model’s ability to protect sensitive information during both training and inference. Alongside this, noise level (Havrilla et al., 2024) measures the trade-off between privacy and accuracy by assessing how much noise is added to ensure privacy while maintaining the model’s performance.

Energy Optimization The energy efficiency ratio (Stojkovic et al., 2024) evaluates the energy used relative to the model’s overall performance, providing insights into how energy-intensive an SLM is in practice. Other metrics, such as thermal efficiency and idle power consumption (Patel et al., 2024), measure the energy consumed when the model is either actively processing tasks or idle, which is crucial for long-term deployment in energy-constrained environments like embedded systems or mobile devices.

6 Applications

In this section, we consider applications of SLMs, that is, specific use-cases like translation and autocompletion.

6.1 Real-Time Interaction

GPT-4o, released in May 2024, processes text, vision, and audio input end-to-end and is faster than GPT-4 Turbo (OpenAI, 2024). The demonstration involved responses in the style of human conver-

Category	Application	Description	Need for SLM Application	Inference Runtime	Memory	Storage Space	Latency	Comm. Overhead
4*Real-Time Interaction	**Chatbots**	Handle frequent queries and basic troubleshooting.	Real-time response needed, lightweight	✓	✓		✓	✓
	Voice Interfaces	Used in voice assistants and dictation tools.	Low latency required for real-time	✓	✓		✓	
	Translation	Basic translation between languages.	Real-time translation with low-resources	✓	✓		✓	✓
5*Content Generation	**Text Summarization**	Summarize articles and reports.	Faster inference, minimal resource use	✓	✓	✓	✓	
5*& Processing	**Sentiment Analysis**	Assess customer sentiment across platforms.	Efficient analysis in low-resource envir.	✓	✓	✓	✓	
	Text Classification	Filter emails, classify content.	Low latency, on-the-fly processing	✓	✓	✓	✓	
	NLP for Search	Improves search engine functionality.	Low latency for real-time search	✓	✓		✓	
	Autocompletion	Suggest completions in IDEs or text editors.	Fast prediction with low memory	✓	✓	✓	✓	

Table 3: Taxonomy of Applications of Small Language Models.

sation. LLaMA-Omni combine a speech encoder, adaptor, LLM, and streaming decoder to enable real-time interaction with speech input based on LLaMA-3-8B-Instruct (Fang et al., 2024). Emotionally Omni-present Voice Assistant, or EMOVA, apply LLaMA-3.1-8B as an end-to-end speech model that can generate poems and describe images at the user's request. Google Deepmind's Project Astra uses Gemini to process audio and video information from a smartphone or glasses and respond to respond to queries like mathematics problems and memorize object sequences (Deepmind, 2024).

6.2 Content Generation and Processing

LLMR uses LLMs in mixed reality to generate and modify 3D scenes. It combines language models used in several roles - a Scene Analyzer GPT to summarize objects and give further details like color, Skill Library GPT to determine what is required to fufill a user's request, Builder GPT to generate code for the request, and Inspector GPT to evaluate its code (Torre et al., 2024). DreamCodeVR assists users in editing an application in the Unity engine through code generation (Giunchi et al., 2024; Juliani et al., 2020). This permits users to edit VR applications without requiring extensive programming knowledge.

6.3 Edge Inference and Privacy

On-device LLMs maintain usability even when MobileLLM improve on various chat benchmarks and performs comparably with LLaMA-2-7B in API calling (Liu et al., 2024b). Apple Intelligence applies an 3B parameter model to perform on-device inference for a broad range of tasks, such as text and notification summarization, image and emoji generation, and code completion for XCode (Gunter et al., 2024; Research, 2024). On-device inference reduces latency as measured by the time to first generated token (Hu et al., 2024; Gerganov). HuatuoGPT is a domain-adapted LLM for medical dialogue and BioMistral is an LLM tailored for biomedical work (Zhang et al., 2023a; Labrak et al., 2024). Applications related to medicine may need to adhere to stringent privacy regulations and represent a promising area for future work. TalkBack with GeminiNano assists blind and low vision people by describing and captioning images and runs on Android devices (Team, 2024). On-device inference makes this technology usable without an internet connection.

Mixture-of-Experts can reduce inference cost by using a gating network to use only a subset of layers during inference time (Shazeer et al., 2017). Google's GLaM uses mixture-of-experts (Du et al., 2022) but is a 1.2T parameter model. EdgeMoE extend misture-of-experts to edge computing using an Nvidia Jetson TX2 and Raspberry Pi 4B, with the latter device being CPU-only (Sarkar et al., 2023). Based on experimental findings that most weights contribute little to the final computation, the authors compress weights and predict the relevant experts in advance.

7 Conclusion

This paper has provided an extensive survey of Small Language Models (SLMs), covering a wide range of topics including model architectures, training methodologies, and model compression techniques that are crucial for optimizing SLMs. We hope that this survey will serve as a valuable resource for both researchers and practitioners working on SLMs.

References

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Sinan Kaplan, Megan Leszczynski, Isys Johnson, Vishal Subbiah, Azalia Mirhoseini, James Zou, and Christopher Ré. 2024. Simple linear attention language models balance the recall-throughput tradeoff.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Nicolas Boizard, Kevin El Haddad, Céline Hudelot, and Pierre Colombo. 2024. Towards cross-tokenizer distillation: the universal logit distillation loss for llms.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. [link].

Vladimír Boža. 2024. Fast and optimal weight update for pruned large language models. *arXiv preprint arXiv:2401.02938*.

Neil Burgess, Jelena Milanovic, Nigel Stephens, Konstantinos Monachopoulos, and David Mansell. 2019. Bfloat16 processing for neural networks. In *2019 IEEE 26th Symposium on Computer Arithmetic (ARITH)*, pages 88–91. IEEE.

Zhiwei Cao, Qian Cao, Yu Lu, Ningxin Peng, Luyang Huang, Shanbo Cheng, and Jinsong Su. 2024. Retaining key information under high compression ratios: Query-guided compressor for llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12685–12695, Bangkok, Thailand. Association for Computational Linguistics.

Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2021. Medically aware gpt-3 as a data generator for medical dialogue summarization. In *Machine Learning for Healthcare Conference*, pages 354–372. PMLR.

Chengwei Dai, Kun Li, Wei Zhou, and Songlin Hu. 2024. Beyond imitation: Learning key reasoning steps from dual chain-of-thoughts in reasoning distillation.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, et al. 2023. Auggpt: Leveraging chatgpt for text data augmentation. *arXiv preprint arXiv:2302.13007*.

Michael Han Daniel Han and Unsloth team. 2023. Unsloth.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10041–10071. PMLR.

Google Deepmind. 2024. Project astra a universal ai agent that is helpful in everyday life.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022a. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022b. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

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

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, et al. 2024. Hymba: A hybrid-head architecture for small language models. *arXiv preprint arXiv:2411.13676*.

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. 2022. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR.

Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. 2024. Llama-omni: Seamless speech interaction with large language models.

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. 2024. Mixture-of-loras: An efficient multitask tuning for large language models. *arXiv preprint arXiv:2403.03432*.

Maxim Fishman, Brian Chmiel, Ron Banner, and Daniel Soudry. 2024. Scaling fp8 training to trillion-token llms. *arXiv preprint arXiv:2409.12517*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning.

Vinod Ganesan, Gowtham Ramesh, and Pratyush Kumar. 2021. Supershaper: Task-agnostic super pre-training of bert models with variable hidden dimensions. *arXiv preprint arXiv:2110.04711*.

Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, Hongsheng Li, and Yu Qiao. 2023. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. 2024. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *The Twelfth International Conference on Learning Representations*.

Georgi Gerganov. llama.cpp.

Daniele Giunchi, Nels Numan, Elia Gatti, and Anthony Steed. 2024. DreamCodeVR: Towards Democratizing Behavior Design in Virtual Reality with Speech-Driven Programming. In *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, Orlando, USA. IEEE.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T Kwok, and Yu Zhang. 2023. Mixture of cluster-conditional lora experts for vision-language instruction tuning. *arXiv preprint arXiv:2312.12379*.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models.

Tom Gunter, Zirui Wang, Chong Wang, Ruoming Pang, Andy Narayanan, Aonan Zhang, et al. 2024. Apple intelligence foundation language models.

Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. 2020. Single path one-shot neural architecture search with uniform sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 544–560. Springer.

Jiayi Han, Liang Du, Hongwei Du, Xiangguo Zhou, Yiwen Wu, Weibo Zheng, and Donghong Han. 2024. Slim: Let llm learn more and forget less with soft lora and identity mixture. *arXiv preprint arXiv:2410.07739*.

Alex Havrilla, Yilun Du, Chuanyang Zheng, Phillip Isola, and Joshua B. Tenenbaum. 2024. Understanding the effect of noise in llm training data with algorithmic chains of thought.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers.

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. *arXiv preprint arXiv:2401.18079*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander 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.

Cunchen Hu, Heyang Huang, Liangliang Xu, Xusheng Chen, Jiang Xu, Shuang Chen, Hao Feng, Chenxi Wang, Sa Wang, Yungang Bao, Ninghui Sun, and Yizhou Shan. 2024. Inference without interference: Disaggregate llm inference for mixed downstream workloads.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Ganesh Jawahar, Haichuan Yang, Yunyang Xiong, Zechun Liu, Dilin Wang, Fei Sun, Meng Li, Aasish Pappu, Barlas Oguz, Muhammad Abdul-Mageed, et al. 2023. Mixture-of-supernets: Improving weight-sharing supernet training with architecture-routed mixture-of-experts. *arXiv preprint arXiv:2306.04845*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. 2020. Mimic-iv. *PhysioNet. Available online at: https://physionet. org/content/mimiciv/1.0/(accessed August 23, 2021)*, pages 49–55.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Arthur Juliani, Vincent-Pierre Berges, Ervin Teng, Andrew Cohen, Jonathan Harper, Chris Elion, Chris Goy, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. 2020. Unity: A general platform for intelligent agents.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR.

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

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Rui Kong, Qiyang Li, Xinyu Fang, Qingtian Feng, Qingfeng He, Yazhu Dong, Weijun Wang, Yuanchun Li, Linghe Kong, and Yunxin Liu. 2024. Lora-switch: Boosting the efficiency of dynamic llm adapters via system-algorithm co-design. *arXiv preprint arXiv:2405.17741*.

Eldar Kurtić, Elias Frantar, and Dan Alistarh. 2024. Ziplm: Inference-aware structured pruning of language models. *Advances in Neural Information Processing Systems*, 36.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Yanis Labrak, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. 2024. Biomistral: A collection of open-source pretrained large language models for medical domains.

Jaewook Lee, Yoel Park, and Seulki Lee. 2024a. Designing extremely memory-efficient cnns for on-device vision tasks.

Nicholas Lee, Thanakul Wattanawong, Sehoon Kim, Karttikeya Mangalam, Sheng Shen, Gopala Anumanchipali, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. 2024b. Llm2llm: Boosting llms with novel iterative data enhancement. *arXiv preprint arXiv:2403.15042*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. 2024. Selective reflection-tuning: Student-selected data recycling for llm instruction-tuning. *arXiv preprint arXiv:2402.10110*.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Heng Huang, Jiuxiang Gu, and Tianyi Zhou. 2023a. Reflection-tuning: Data recycling improves llm instruction-tuning. *arXiv preprint arXiv:2310.11716*.

Zonglin Li, Chong You, Srinadh Bhojanapalli, Daliang Li, Ankit Singh Rawat, Sashank J. Reddi, Ke Ye, Felix Chern, Felix Yu, Ruiqi Guo, and Sanjiv Kumar. 2023b. The lazy neuron phenomenon: On emergence of activation sparsity in transformers. In *The Eleventh International Conference on Learning Representations*.

Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. Openorca: An open dataset of gpt augmented flan reasoning traces. `https://https://huggingface.co/Open-Orca/OpenOrca`.

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.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. 2024. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. 2023a. Qllm: Accurate and efficient low-bitwidth quantization for large language models. *arXiv preprint arXiv:2310.08041*.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. 2023b. Moelora: An moe-based parameter efficient fine-tuning method for multi-task medical applications. *arXiv preprint arXiv:2310.18339*.

Shih-yang Liu, Zechun Liu, Xijie Huang, Pingcheng Dong, and Kwang-Ting Cheng. 2023c. Llm-fp4: 4-bit floating-point quantized transformers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 592–605.

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

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. 2024a. Spinquant–llm quantization with learned rotations. *arXiv preprint arXiv:2405.16406*.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. 2024b. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. *arXiv:2402.14905*.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. 2023e. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. 2024c. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*.

Tongxu Luo, Jiahe Lei, Fangyu Lei, Weihao Liu, Shizhu He, Jun Zhao, and Kang Liu. 2024. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *arXiv preprint arXiv:2402.12851*.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit llms: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2023. Teaching small language models to reason.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *International Conference on Learning Representations*.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. 2021. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56:1 – 40.

Seyed Iman Mirzadeh, Keivan Alizadeh-Vahid, Sachin Mehta, Carlo C del Mundo, Oncel Tuzel, Golnoosh Samei, Mohammad Rastegari, and Mehrdad Farajtabar. 2024. ReLU strikes back: Exploiting activation sparsity in large language models. In *The Twelfth International Conference on Learning Representations*.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Compact language models via pruning and knowledge distillation.

Deepak Narayanan, Keshav Santhanam, Peter Henderson, Rishi Bommasani, Tony Lee, and Percy Liang. 2023. Cheaply evaluating inference efficiency metrics for autoregressive transformer apis.

OpenAI. 2024. Hello gpt-4o.

Jongho Park, Jaeseung Park, Zheyang Xiong, Nayoung Lee, Jaewoong Cho, Samet Oymak, Kangwook Lee, and Dimitris Papailiopoulos. 2024. Can mamba learn how to learn? a comparative study on in-context learning tasks. *arXiv preprint arXiv:2402.04248*.

Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Brijesh Warrier, Nithish Mahalingam, and Ricardo Bianchini. 2024. Characterizing power management opportunities for llms in the cloud. In *ASPLOS '24: Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, New York, NY, USA. Association for Computing Machinery.

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. 2023a. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077, Singapore. Association for Computational Linguistics.

Houwen Peng, Kan Wu, Yixuan Wei, Guoshuai Zhao, Yuxiang Yang, Ze Liu, Yifan Xiong, Ziyue Yang, Bolin Ni, Jingcheng Hu, et al. 2023b. Fp8-lm: Training fp8 large language models. *arXiv preprint arXiv:2310.18313*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Pranav Rajpurkar, Jian Zhang, Konstantin Liu, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Liliang Ren, Yang Liu, Yadong Lu, Yelong Shen, Chen Liang, and Weizhu Chen. 2024. Samba: Simple hybrid state space models for efficient unlimited context language modeling. *arXiv preprint arXiv:2406.07522*.

Apple Machine Learning Research. 2024. Introducing apple's on-device and server foundation models.

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.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2023. On the effect of dropping layers of pre-trained transformer models. *Computer Speech & Language*, 77:101429.

V Sanh. 2019. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Michael Santacroce, Zixin Wen, Yelong Shen, and Yuanzhi Li. 2023. What matters in the structured pruning of generative language models? *arXiv preprint arXiv:2302.03773*.

Rishov Sarkar, Hanxue Liang, Zhiwen Fan, Zhangyang Wang, and Cong Hao. 2023. Edge-moe: Memory-efficient multi-task vision transformer architecture with task-level sparsity via mixture-of-experts.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. 2020. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947.

Atreya Shankar, Andreas Waldis, Christof Bless, Maria Andueza Rodriguez, and Luca Mazzola. 2023. Privacyglue: A benchmark dataset for general language understanding in privacy policies. *Applied Sciences*, 13(6):3701.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. 2023. Omniquant: Omnidirectionally calibrated quantization for large language models. *arXiv preprint arXiv:2308.13137*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large language model alignment: A survey. *ArXiv*, abs/2309.15025.

Xuan Shen, Peiyan Dong, Lei Lu, Zhenglun Kong, Zhengang Li, Ming Lin, Chao Wu, and Yanzhi Wang. 2024a. Agile-quant: Activation-guided quantization for faster inference of llms on the edge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 18944–18951.

Xuan Shen, Zhenglun Kong, Changdi Yang, Zhaoyang Han, Lei Lu, Peiyan Dong, Cheng Lyu, Chih-hsiang Li, Xuehang Guo, Zhihao Shu, et al. 2024b. Edgeqat: Entropy and distribution guided quantization-aware training for the acceleration of lightweight llms on the edge. *arXiv preprint arXiv:2402.10787*.

Yiping Song, Juhua Zhang, Zhiliang Tian, Yuxin Yang, Minlie Huang, and Dongsheng Li. 2024. Llm-based privacy data augmentation guided by knowledge distillation with a distribution tutor for medical text classification. *arXiv preprint arXiv:2402.16515*.

Sharath Turuvekere Sreenivas, Saurav Muralidharan, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Llm pruning and distillation in practice: The minitron approach.

Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. 2024. Dynamollm: Designing llm inference clusters for performance and energy efficiency.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR.

Chaofan Tao, Lu Hou, Haoli Bai, Jiansheng Wei, Xin Jiang, Qun Liu, Ping Luo, and Ngai Wong. 2023. Structured pruning for efficient generative pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10880–10895.

Jean-Loup Tastet and Inar Timiryasov. 2024. Babyllama-2: Ensemble-distilled models consistently outperform teachers with limited data. *arXiv preprint arXiv:2409.17312*.

Gemini Team. 2024. Gemini: A family of highly capable multimodal models.

Omkar Thawakar, Ashmal Vayani, Salman Khan, Hisham Cholakal, Rao M Anwer, Michael Felsberg, Tim Baldwin, Eric P Xing, and Fahad Shahbaz Khan. 2024. Mobillama: Towards accurate and lightweight fully transparent gpt. *arXiv preprint arXiv:2402.16840*.

Inar Timiryasov and Jean-Loup Tastet. 2023a. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *arXiv preprint arXiv:2308.02019*.

Inar Timiryasov and Jean-Loup Tastet. 2023b. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty.

Fernanda De La Torre, Cathy Mengying Fang, Han Huang, Andrzej Banburski-Fahey, Judith Amores Fernandez, and Jaron Lanier. 2024. Llmr: Real-time prompting of interactive worlds using large language models.

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. 2023. Llama: Open and efficient foundation language models.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024a. Knowledge fusion of large language models.

Fanqi Wan, Longguang Zhong, Ziyi Yang, Ruijun Chen, and Xiaojun Quan. 2024b. Fusechat: Knowledge fusion of chat models.

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

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020a. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2020b. Structured pruning of large language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162, Online. Association for Computational Linguistics.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. f-divergence minimization for sequence-level knowledge distillation.

Chenxi Whitehouse, Monojit Choudhury, and Alham Fikri Aji. 2023. Llm-powered data augmentation for enhanced cross-lingual performance. *arXiv preprint arXiv:2305.14288*.

Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. 2019. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10734–10742.

Haocheng Xi, Yuxiang Chen, Kang Zhao, Kai Jun Teh, Jianfei Chen, and Jun Zhu. 2024. Jetfire: Efficient and accurate transformer pretraining with int8 data flow and per-block quantization. *arXiv preprint arXiv:2403.12422*.

Haocheng Xi, Changhao Li, Jianfei Chen, and Jun Zhu. 2023. Training transformers with 4-bit integers. *Advances in Neural Information Processing Systems*, 36:49146–49168.

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*.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. *arXiv preprint arXiv:2204.00408*.

Guangxuan Xiao, Ji Lin, Mickael 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*, pages 38087–38099. PMLR.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. 2021. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14138–14148.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Jin Xu, Xu Tan, Renqian Luo, Kaitao Song, Jian Li, Tao Qin, and Tie-Yan Liu. 2021. Nas-bert: task-agnostic and adaptive-size bert compression with neural architecture search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1933–1943.

Shu Yang, Muhammad Asif Ali, Cheng-Long Wang, Lijie Hu, and Di Wang. 2024. Moral: Moe augmented lora for llms' lifelong learning. *arXiv preprint arXiv:2402.11260*.

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. *Advances in Neural Information Processing Systems*, 35:27168–27183.

Da Yu, Peter Kairouz, Sewoong Oh, and Zheng Xu. 2024. Privacy-preserving instructions for aligning large language models.

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. *arXiv preprint arXiv:2402.12065*.

Shulin Zeng, Jun Liu, Guohao Dai, Xinhao Yang, Tianyu Fu, Hongyi Wang, Wenheng Ma, Hanbo Sun, Shiyao Li, Zixiao Huang, et al. 2024. Flightllm: Efficient large language model inference with a complete mapping flow on fpgas. In *Proceedings of the 2024 ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pages 223–234.

Hongbo Zhang, Junying Chen, Feng Jiang, Fei Yu, Zhihong Chen, Jianquan Li, Guiming Chen, Xiangbo Wu, Zhiyi Zhang, Qingying Xiao, Xiang Wan, Benyou Wang, and Haizhou Li. 2023a. Huatuogpt, towards taming language models to be a doctor. *arXiv preprint arXiv:2305.15075*.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. 2023b. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. 2023. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*.

Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. 2021. Learning n: m fine-grained structured sparse neural networks from scratch. *arXiv preprint arXiv:2102.04010*.

He Zhu, Junyou Su, Tianle Lun, Yicheng Tao, Wenjia Zhang, Zipei Fan, and Guanhua Chen. 2024. Fanno: Augmenting high-quality instruction data with open-sourced llms only. *arXiv preprint arXiv:2408.01323*.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2023. A survey on model compression for large language models. *ArXiv*, abs/2308.07633.

Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*.