

Small Language Models: Architectures, Techniques, Evaluation, Problems and Future Adaptation

Tanjil Hasan Sakib², Md. Tanzib Hosain², Md. Kishor Morol¹

¹Cornell University

²American International University-Bangladesh

{20-43633-2, 20-42737-1}@student.aiub.edu, mmorol@cornell.edu

Abstract

Small Language Models (SLMs) have gained substantial attention due to their ability to execute diverse language tasks successfully while using fewer computer resources. These models are particularly ideal for deployment in limited environments, such as mobile devices, on-device processing, and edge systems. In this study, we present a complete assessment of SLMs, focussing on their design frameworks, training approaches, and techniques for lowering model size and complexity. We offer a novel classification system to organize the optimization approaches applied for SLMs, encompassing strategies like pruning, quantization, and model compression. Furthermore, we assemble SLM's studies of evaluation suite with some existing datasets, establishing a rigorous platform for measuring SLM capabilities. Alongside this, we discuss the important difficulties that remain unresolved in this sector, including trade-offs between efficiency and performance, and we suggest directions for future study. We anticipate this study to serve as a beneficial guide for researchers and practitioners who aim to construct compact, efficient, and high-performing language models.

1 Introduction

Large Language Models (LLMs) have exhibited remarkable efficacy across a wide spectrum of benchmarks and real-world applications. However, their accomplishment comes at a significant cost. Training and operating these models demand immense computational resources, including specialized tools and large datasets. As a result, LLMs are often deployed on centralized and specialized infrastructure for both training and inference. These aspects make them undesirable for certain environments that require lower resource use or quicker deployment timeframes.

To surmount these issues, SLMs have garnered considerable interest. These models endeavor to preserve the efficacy and adaptability of LLMs while operating under stricter constraints, such as less computational power, decreased memory availability, bandwidth restrictions, and quicker inference times. By enhancing model efficacy relative to these limits,

SLMs can yield benefits such as lower operational costs, enhanced privacy, and the potential to operate on consumer-grade hardware. Table 1 presents a comparative overview of prominent SLMs, detailing their respective research labs, parameter sizes (in billions), training token counts (in billions), token-to-parameter ratios¹, ALScore² values, architectural types, and announcement years³.

A fundamental issue in surveying small language models resides in determining what comprises “small” versus “large.” Model classification depends on situation and time period. For example, GPT-2, which was designated a “large” model in 2019 with 1.5 billion parameters, is presently smaller than many of the models categorized as “small” in the present landscape. Despite these modifications in scale, the essential objectives of training smaller models remain reasonably identical.

It is crucial to observe that development in one element of these axes does not always indicate progress in others, as there are often trade-offs. For example, strategies such as quantization-aware training attempt to make training more memory-efficient, but they may result in delayed processing compared to full-precision alternatives. Nonetheless, employing mixed precision approaches to characterize weights and gradients can allow for more efficient use of memory during training or fine-tuning without a significant reduction in accuracy.

While several recent surveys have studied large language models and their associated training approaches [Rogers *et al.*, 2020], SLMs’ practical inference improvement factor’s comprehensive study is still missing. That is why, through this study, we focus on concentrating faster and lighter techniques with their problems and adaptation across fields on SLMs.

The key contributions of this study are as follows:

- A complete survey of extant research on small language models, encompassing important issue contexts like training and model compression strategies with their evaluation and adaptation frequently exploited in this domain.
- The introduction of intuitive strategies for categorizing

¹Chinchilla scaling $\geq 20:1$.

²“ALScore” is a quick and dirty rating of the model’s power. The formula is: $\text{Sqr Root of (Parameters x Tokens)} \div 300$. Any ALScore ≥ 1.0 is a powerful model in mid-2023.

³<https://lifearchitect.ai/>

- SLM-related approaches and a summary of existing work based on these strategies.
- Identification of unresolved barriers and future research potential for small language models.

2 Small Language Models Architectures

This section covers numerous architectural innovations that enable the building of efficient small language models. It focuses on some key approaches.

2.1 Lightweight Models

Lightweight designs are especially intended to deliver great performance while minimizing computational and memory overhead. These models are well-suited for deployment in resource-constrained situations, such as mobile devices, embedded systems, and edge computing platforms. They are commonly categorized into encoder-only and decoder-only configurations.

Encoder-only Architectures: This category largely comprises of optimized variations of BERT, which have been fine-tuned to increase efficiency without compromising performance. Key instances include:

- **MobileBERT** [Sun *et al.*, 2020]: Introduced an inverted bottleneck structure to balance self-attention and feed-forward layers, resulting in a $4.3\times$ reduction in model size and a $5.5\times$ increase in processing speed compared to the original BERT.
- **DistilBERT** [Sanh, 2019] and **TinyBERT** [Jiao *et al.*, 2019]: Both models dramatically reduce model size (over 60%) while keeping more than 96% of the original BERT's performance, making them perfect for realistic deployment circumstances.

Decoder-only Architectures: This category focuses on autoregressive models such as GPT [Radford *et al.*, 2019] and LLaMA [Touvron *et al.*, 2023]. Prominent instances include:

- **BabyLLaMA** [Timiryasov and Tastet, 2023]: Employs knowledge distillation from various teacher models to produce a 58M-parameter model, excelling in low-data settings.
- **TinyLLaMA** [Zhang *et al.*, 2024]: Utilizes FlashAttention [Dao *et al.*, 2022] to optimize memory overhead, achieving great efficiency with just 1.1B parameters.
- **MobileLLM** [Li *et al.*, 2024a]: Implements block-wise weight sharing and grouped-query attention techniques, decreasing latency and boosting scalability for real-world applications.

2.2 Streamlined Self-attention Mechanisms

Self-attention processes are crucial in language models but are frequently resource-intensive, causing scaling issues. This section outlines strategies that decrease the computational overhead of self-attention layers while retaining model accuracy:

- **Reformer** [Kitaev *et al.*, 2020]: Reduces the quadratic complexity of self-attention ($O(N^2)$) to logarithmic complexity ($O(N \log N)$) via locality-sensitive hashing.

- **Linear Attention** [Katharopoulos *et al.*, 2020]: Reformulates self-attention into a linear dot-product of kernel feature maps, enabling quicker inference and memory-efficient processing.
- **Mamba** [Liu *et al.*, 2024]: Introduces input-dependent state-space transitions to achieve linear complexity while keeping robust model expressiveness.
- **RWKV** [Peng *et al.*, 2023]: Combines transformer-based and RNN-based architectures with a linear attention mechanism, delivering efficient yet competitive task performance.

For handling longer sequences, hybrid models such as **Longformer** [Dao and Gu, 2024] and **Nystrom-based methods** provide scalability by lowering computational complexity to linear levels. These models maintain good empirical performance, making them particularly useful for jobs involving vast text inputs.

2.3 Neural Architecture Search (NAS)

NAS techniques employ automated algorithms to discover optimal architectures tailored to specific duties and hardware constraints. Initially applied to vision tasks and smaller BERT models, NAS has recently been extended to large-scale language models, addressing challenges such as high parameter counts and computational costs. Recent advancements include:

- **MobileLLM**: Conducted a systematic exploration of architectural depth (number of layers) and width (number of attention heads) to design efficient models within constrained parameter ranges. It also introduced initialization techniques to reduce the search space, thereby accelerating the convergence of Neural Architecture Search (NAS) processes, making them practical for large-scale language models.

2.4 Small Multi-modal Integration

Recent advancements in large multi-modal models (LMMs) have demonstrated comparable or even superior performance relative to their predecessors, all while substantially reducing the number of parameters. Notable examples include the LLaVA-Next series, Idefics2 [Laurençon and Picard, 2024], and InternVL2 [Chen *et al.*, 2023]. This progress has been fueled in part by the development of more efficient and compact language models, such as Gemma and phi-3-mini, which underscore the significance of curated datasets in optimizing performance.

A significant focus in this discipline has been on minimizing the size of vision encoders during multi-modal fusion. For example, InternVL2 leverages intermediate outputs of large visual encoders while omitting later blocks, thus enhancing efficiency. Smaller variants such as PaliGemma [Beyer and Jones, 2024] and Mini-Gemini [Li and Zhao, 2024] employ lightweight vision encoders to further streamline multi-modal processing.

Monolithic multi-modal models extend this efficiency even further by wholly eradicating the need for a dedicated visual encoder. Instead, these models utilize lightweight architectures

Model	Laboratory	Params (B)	Tokens (B)	Tokens:Params	ALScore	Architecture	Announced
OPT	Meta	175	300	1:71	0.76	Dense	2023
Pythia	EleutherAI	12	300	25:1	0.2	Dense	2023
Cerebras-GPT	Cerebras	13	260	20:1	0.2	Dense	2023
Phi-1	Microsoft	1.3	150	116:1	0.0	Dense	2023
Phi-2	Microsoft	2.7	1400	519:1	0.2	Dense	2023
TinyLlama	SUTD/Independent	1.1	3000	2,728:1	0.2	Dense	2024
Gemma	Google DeepMind	7	6000	858:1	0.7	Dense	2024
DCLM	International	7	2600	372:1	0.4	Dense	2024
Bloom	BigScience	176	366	3:1	0.8	Dense	2022
Galactica	Meta AI	120	450	4:1	0.8	Dense	2022
Qwen	Alibaba	72	18000	250:1	3.8	Dense	2024
EvaByte	SambaNova	6.5	1500	231:1	0.3	Dense	2025
MiniCPM-2.4B	Tsinghua	2.4	1100	459:1	0.2	Dense	2024
OpenELM	Apple	3.04	1500	494:1	0.2	Dense	2024
Phi-3-mini	Microsoft	3.8	3300	869:1	0.4	Dense	2024
Zamba 7B	Zyphra	7	1050	150:1	0.3	Dense	2024
Gemma-2	Google DeepMind	27	13000	482:1	2.0	Dense	2024
SmollM	Hugging Face	1.7	1000	589:1	0.1	Dense	2024
H2O-Danube3-4B	H2O.ai	4	6000	1,500:1	0.5	Dense	2024
Fox-1	TensorOpera	1.6	3005	1,879:1	0.2	Dense	2024
phi-3.5-MoE	Microsoft	60	4900	82:1	1.8	MoE	2024
Qwen2.5	Alibaba	72	18000	250:1	3.8	Dense	2024

Table 1: Various SLMs based on Parameters, Training Tokens, and Architectural Characteristics.

to generate visual tokens. For instance, Chameleon employs a VQ-VAE model to encode and decode images into discrete tokens, while Mono-InternVL [Luo and He, 2024] relies on an MLP-based architecture to produce visual tokens for image patches. Mono-InternVL integrates a modality-specific feed-forward network, referred to as a multi-modal Mixture-of-Experts, to effectively differentiate between modalities.

These innovations underscore the ongoing trend toward more compact and efficient multi-modal models that maintain competitive performance across a wide spectrum of tasks.

3 Training Strategies

This section examines essential training strategies utilized in the development of small language models (SLMs), concentrating on pre-training and fine-tuning. While many of these methods are similar to those employed for large language models (LLMs), the techniques discussed here emphasize efficiency and scalability, particularly in resource-constrained scenarios. Table 2 shows a taxonomical overview of SLMs’ training techniques innovations.

3.1 Foundational Pre-training Strategies

Pre-training Strategies is a foundational step in constructing SLMs, where mixed-precision training has emerged as a critical method for enhancing efficiency. This approach employs low-precision representations during forward and backward propagation while retaining high-precision weights for revisions. For instance, Automatic Mixed Precision (AMP) [Mickevicius *et al.*, 2018], maintains a master copy of weights in 32-bit floating-point (FP32) precision while performing computations in 16-bit floating-point (FP16) precision. Despite its utility, accuracy issues due to FP16’s limited numerical range have been noted [Rae *et al.*, 2021]. To address

this, Brain Floating Point (BFLOAT16) offers an enhanced dynamic range with additional exponent bits, leading to superior training performance and numerical stability compared to FP16.

Modern GPU architectures, such as NVIDIA’s Hopper, further enhance these capabilities by supporting 8-bit floating-point (FP8) precision, facilitating greater computational efficiency without compromising accuracy. Specialized tensor cores in these GPUs enable quicker training for both SLMs and LLMs.

To complement mixed-precision methods, optimization techniques such as Adam, AdamW, and memory-efficient variants like Adafactor and Sophia [Liu and Zhao, 2024] are widely employed. Gradient clipping prevents exploding gradients, while effective initialization strategies ensure a stable starting point for training. These combined approaches enhance efficiency, stabilize the training process, and improve the robustness of resultant models.

Distributed training has become indispensable for pre-training SLMs, leveraging multiple machine nodes to manage large datasets and models efficiently. Techniques such as Zero Redundancy Optimizer (ZeRO) and PyTorch’s Fully Sharded Data Parallel (FSDP) [Zhao *et al.*, 2023] distribute training states across devices. ZeRO, for example, partitions optimizer states, gradients, and model parameters across nodes, while FSDP implements similar concepts to facilitate larger batch sizes and enhanced scalability.

3.2 Task-specific Adaptation

Task-Specific applies pre-trained models to task-specific datasets, allowing SLMs to specialize in domains or tasks while resolving constraints like limited data, computational resources, and robustness challenges.

Strategy	Methods	Features	Performance	Advantages
Pre-training	AMP, BFLOAT16, FP8	Low precision computations while maintaining accuracy.	50% faster training and 40% memory savings compared to FP32.	Reduced memory usage, faster training on large datasets.
Fine-tuning	LoRA, Prompt Tuning	Updates specific parameters or introduces lightweight modules.	Improves task-specific accuracy with minimal resource overhead.	Efficient adaptation for domain-specific tasks.
Distributed	ZeRO, FSDP	Multi-node task distribution.	Enables training of models 2x larger than standard setups.	Scalable for large-scale training.
Data Augmentation	AugGPT, Evol-Instruct	Generates diverse training samples.	Enhances robustness with 20% improved generalization.	Improved downstream task performance.

Table 2: Training Techniques for Small Language Models.

Resource-efficient Fine-tuning

Resource-Efficient Fine-Tuning minimizes computational overhead by updating only a subset of model parameters or integrating lightweight modules, leaving most pre-trained weights unaltered. Approaches like LoRA , which employs low-rank decomposition, and Prompt Tuning [Lester *et al.*, 2021], which introduces learnable prompts, exemplify PEFT methods. Llama-Adapter [Zhang *et al.*, 2023] extends this by adding prompts to LLaMA’s attention blocks, enabling task-specific alterations with minimal resources.

Dynamic Adapters dynamically combine multiple adapters into a mixture-of-experts model, effectively enabling multi-tasking and mitigating catastrophic forgetting. These techniques are highly adaptable, making PEFT an attractive choice for fine-tuning in resource-limited environments.

Diversity-enhanced Data Augmentation

Data augmentation is a potent technique to enhance the diversity, complexity, and quality of training data. By augmenting datasets, models generalize better and perform robustly in subsequent tasks. For instance, AugGPT generates rephrased training samples using ChatGPT, while Evol-Instruct [Xu *et al.*, 2023] employs multistep revisions to create diverse open-domain instructions. Reflection-tuning refines both instructions and responses using GPT-4 to enhance instruction-response consistency.

Methods like FANNO [Zhu *et al.*, 2024] incorporate external knowledge sources for retrieval-augmented generation, and LLM2LLM generates hard samples based on model predictions to enhance learning dynamics. In scenarios where data is limited, such as low-resource languages , medical applications or privacy-sensitive data , data augmentation facilitates robust performance with limited datasets.

4 Model Compression Strategies

Model compression approaches are critical for lowering the size and complexity of big pre-trained language models while retaining their performance. These techniques are crucial in reducing big language models (LLMs) into short language models (SLMs) appropriate for deployment in resource-constrained situations. This section categorizes model compression approaches into some basic strategies. Table 3 provides a taxonomical overview of SLMs’ model compression techniques innovations

4.1 Streamlined Pruning Strategies

Pruning strategies strive to minimize the number of parameters in a model, boosting computational efficiency and decreasing memory use while preserving acceptable levels of performance. Two basic ways to pruning are unstructured and organized pruning.

Fine-grained Pruning: This strategy eliminates individual weights of lower relevance, enabling fine-grained control over model size. For instance, SparseGPT reformulates the pruning process as a sparse regression problem, enabling efficient pruning for big models like OPT-175B and BLOOM-176B. Wanda [Sun *et al.*, 2023] analyzes both weights and activations, minimizing the requirement for weight updates during pruning. The n:m pruning approach prunes exactly n weights out of every m, balancing flexibility and efficiency, and is exploited by NVIDIA’s TensorRT to speed inference on GPUs.

Group-based Pruning: Unlike Fine-grained Pruning, Group-based Pruning removes entire groupings of parameters, such as neurons or channels, which allows implementation on standard hardware. Methods like contextual sparsity dynamically prune neurons based on input, maximizing computational efficiency. Additionally, approaches like as FastGen [Ge *et al.*, 2023] focus on eliminating duplication in Transformer layers, enabling more effective use of GPU memory and faster processing performance.

Recent work has also examined input-dependent pruning algorithms, such as dynamic sparsity, which adjust pruning tactics to individual inputs for increased performance. Structured pruning techniques remain vital for enabling hardware-accelerated compression without compromising model efficacy.

4.2 Precision-optimized Quantization

Quantization is a frequently utilized approach for compressing LLMs with huge parameter counts. This strategy decreases model size and computing needs by encoding weights and activations with lower precision, such as 8-bit integers.

Weight Compression Approaches: GPTQ [Frantar *et al.*, 2022] accomplishes layer-wise weight-only quantization, reducing reconstruction error using inverse Hessian matrices. AWQ and ZeroQuant expand this method by considering activations alongside weights, optimizing the trade-off between precision and computational efficiency.

Strategy	Methods	Features	Performance	Advantages
Pruning	SparseGPT, Wanda	Removes irrelevant weights for efficiency.	Reduces model size by 60% with minimal accuracy loss (< 2%).	Efficient size control, suitable for deployment.
Quantization	GPTQ, SmoothQuant	Reduces weight and activation precision.	Lowers computational cost by 50%, inference time improved by 30%.	Reduced size and faster inference.
Knowledge Transfer	BabyLlama, Distillation	Transfers task-specific knowledge from larger models.	Retains 90% of teacher model performance with a 70% smaller size.	High efficiency for smaller models.

Table 3: Model Compression Techniques for Small Language Models.

Efficient Activation Quantization: Activation quantization creates issues owing to outliers in activation distributions. Techniques like SmoothQuant [Xiao *et al.*, 2023] transfer quantization difficulties from activations to weights, whereas SpinQuant [Han *et al.*, 2015] turns outliers into a new space using rotation matrices. Recent improvements in quantization-aware training (QAT), such as LLM-QAT, employ distillation from higher-precision models to recover mistakes made during quantization.

Additionally, developing approaches for Key-Value cache [Hooper and Zhang, 2024] quantization enable efficient inference across extended sequences by compressing memory-intensive components. These developments have broadened the utility of quantization for implementing LLMs on resource-constrained hardware, including mobile devices and FPGAs.

4.3 Knowledge Transfer Techniques

Knowledge Transfer Techniques moves knowledge from a bigger, pre-trained instructor model to a smaller, more efficient student model. This procedure minimizes model size while retaining performance, making it an essential technique for developing SLMs.

Traditional Knowledge Transfer: In its form, Knowledge Transfer entails teaching a student model to imitate the behavior of a teacher model [Gu *et al.*, 2024]. BabyLlama, for example, employs Llama as the teacher to generate a 58M-parameter student model, proving that distillation can outperform pre-training on the same dataset.

Enhanced Knowledge Transfer: Recent improvements include sequence-level distillation, which employs generalized f-divergences to increase response quality , and task-aware distillation, which selectively transmits task-specific knowledge. Multi-teacher distillation techniques combine outputs from numerous teacher models to boost the performance of student models.

Supervised Knowledge Transfer: Incorporating additional monitoring during distillation has demonstrated to help the learning process. For example, rationale-based supervision promotes sample efficiency and raises performance in tasks like commonsense reasoning and arithmetic problem-solving. Furthermore, strategies like reasoning chain distillation convey multi-step reasoning capabilities from instructor models, arming student models with sophisticated problem-solving skills.

5 Evaluation

Evaluation plays a vital role in measuring the effectiveness of small language models (SLMs) across diverse application contexts. Table 4 gives a detailed breakdown of the various settings, limitations, datasets, and metrics utilized for assessing SLMs. This section goes into the various datasets and evaluation criteria, categorized according to the restrictions they are designed to address.

5.1 Datasets

Datasets are crucial to the evaluation process, as they enable models to generalize effectively across varied contexts. The typically used datasets for pre-training and evaluating SLMs are given in Table 4. These datasets are particularly constructed to satisfy distinct limitations, ensuring robust evaluation across multiple use cases.

Efficient Inference: For applications where latency and throughput are crucial, models need to create outputs with little delay while retaining high processing rates. This requirement is particularly critical for real-time applications such as question answering, text classification, and natural language understanding. Key datasets in this category include SuperGLUE, SQuAD, TriviaQA, CoQA, and Natural Questions (NQ), which are extensively used for measuring performance in these tasks.

Privacy-Preserving: With increased concerns about data privacy, privacy-preserving datasets play a critical role in facilitating the development of SLMs that can handle sensitive information securely. PrivacyGLUE [Havrilla, 2024] applies differential privacy approaches to common tasks such as sentiment analysis, giving a standard for privacy-conscious models. Similarly, anonymized datasets like MIMIC and n2c2⁴ offer de-identified clinical notes for medical applications. Federated datasets like LEAF⁵ further enhance privacy by allowing data to remain spread across devices, providing privacy by design using federated learning frameworks.

TinyML and On-device: Models implemented in resource-constrained situations must run effectively without losing accuracy. Frameworks such as TinyBERT and OpenOrca provide specific datasets and tools for training and evaluation in such circumstances. TinyBERT, a distilled version of BERT, is optimized for both size and speed, making it appropriate for

⁴<https://portal.dbmi.hms.harvard.edu/projects/n2c2-nlp/>

⁵<https://github.com/TalwalkarLab/leaf>

applications with minimal latency needs. Similarly, OpenOrca datasets balance performance with resource limits, permitting the construction of compact and efficient models appropriate for deployment on low-power devices.

Energy-Efficient AI: In applications where energy consumption is a significant concern, datasets focus on enabling models to optimize energy usage while preserving performance. While specific statistics are not necessarily explicitly connected to this environment, the evaluation generally uses larger datasets to estimate energy efficiency under typical workloads.

5.2 Metrics

Evaluation measures quantify the performance of SLMs under various limitations, delivering insights into their applicability for particular applications. Table 4 organizes these metrics based on their importance to latency, memory, privacy, and energy efficiency.

Latency: Latency is a critical measure for applications requiring quick feedback, such as chatbots or live translations. Metrics such as inference time assess how quickly a model processes input and creates output, while throughput evaluates the amount of tokens or samples a model can handle within a certain time period. These metrics are particularly significant for large-scale tasks or time-sensitive applications.

Memory: Memory efficiency is crucial when deploying models in situations with limited memory resources, such as mobile or embedded devices. Metrics such as peak memory usage capture the highest memory consumption during inference, while memory footprint and compression ratio [Cao *et al.*, 2024] evaluate the compactness of a model and the effectiveness of compression approaches. These criteria ensure that models may run efficiently within limited situations without compromising performance.

Privacy: Protecting sensitive information is critical for models implemented in fields such as healthcare and finance. The privacy budget [Yu *et al.*, 2024] quantifies a model’s ability to maintain differential privacy, while noise level [Havrilla and Iyer, 2024] analyzes the trade-off between privacy and accuracy by assessing the amount of noise added to secure data.

Energy Optimization: As energy efficiency becomes increasingly critical, especially for edge devices and embedded systems, measures like the energy efficiency ratio [Stojkovic *et al.*, 2024b] provide insights into the energy usage relative to model performance. Additional measures, such as thermal efficiency and idle power consumption [Patel *et al.*, 2024], analyze energy use during active processing and idle phases, delivering a full perspective of a model’s energy footprint.

6 Problems and Future Works

Despite their growing adoption and versatility, small language models (SLMs) face several major hurdles that impede their full potential. Addressing these problems is crucial for advancing the field and ensuring the reliability and fairness of SLMs across diverse uses. This part discusses some of the most pressing challenges, including hallucination, biases, inference-time energy consumption, and data privacy issues. Each of

these open problems offers a chance for future study and development.

6.1 Problems of Hallucination

One of the most common problems in both small and large language models is hallucination, which refers to the generation of information that is either truly wrong or irrelevant to the input context. This problem not only undermines user trust but can also have serious effects in high-stakes domains such as healthcare, law, and finance.

Hallucination can be generally grouped into two categories:

- **Factuality Hallucination:** The generated material defies verifiable facts or established knowledge bases.
- **Faithfulness Hallucination:** The result is unrelated to the user’s question or the task’s requirements, failing to stay contextually relevant.

Studies like HallusionBench [Guan *et al.*, 2024] show that bigger models usually exhibit fewer hallucinations, suggesting that increased parameter counts can enhance factual consistency. However, results from the AMBER benchmark suggest that hallucination types and severity change greatly with model size and design. Interestingly, while bigger models may lower the general rate of hallucinations, they are also more likely to produce complicated and nuanced errors, which may be harder to identify and mitigate.

Future study should focus on building benchmarks and assessment measures that account for the variety of hallucination types. Additionally, knowing how model size, training data quality, and design affect hallucination can pave the way for more powerful mitigation methods.

6.2 Addressing Bias in Models

Bias in language models is a longstanding problem that remains even in smaller, more targeted models. Bias often comes from the training data, which may contain social stereotypes, prejudices, or imbalances. This can result in models repeating or even increasing these biases, leading to outcomes that may be unfair or discriminatory.

Measuring Bias: Effective bias reduction starts with solid measurement methods. Several benchmarks have been created for this reason, including:

- **Bias Benchmark for Question Answering (BBQ):** Focuses on finding flaws in question-answering tasks [Parikh *et al.*, 2021].
- **RealToxicityPrompts:** Measures the tendency of models to create toxic outputs when asked [Gehman *et al.*, 2020].
- **CrowS-Pairs:** A dataset intended to test for stereotypical links in generated text.

Impact of Model Size: Research has shown that larger models tend to display higher levels of bias, as tested across different standards. For instance, the LLaMA series showed greater bias scores on RealToxicityPrompts and StereoSet as model size increased. However, improvements in younger generations, such as LLaMA-2, show that targeted interventions during training can successfully decrease bias levels.

Addressing bias requires a multi-faceted approach, including better management of training data, introduction of

Setting	Constraints	Datasets	Metrics
Efficient Inference	Latency	SuperGLUE, SQuAD, TriviaQA, CoQA, NQ	Inference Time, Throughput
On-device/Mobile	Memory	TinyBERT, OpenOrca	Peak Memory Usage, Memory Footprint, Compression Ratio
Privacy-Preserving	Privacy	PrivacyGLUE, MIMIC, n2c2, LEAF	Privacy Budget, Noise Level
Energy-Efficient AI	Energy Optimization	-	Energy Efficiency Ratio, Thermal Efficiency, Idle Power Consumption

Table 4: Overview of settings, constraints, and metrics for evaluating small language models.

fairness-aware loss functions, and model reviews that value equity and inclusion.

6.3 Energy-efficient Inference

Energy economy is a critical issue for applying SLMs in resource-constrained settings, such as mobile devices or embedded systems. These models must balance computing speed with minimal energy consumption to stay effective for real-world uses.

Research using the MELODI standard [Husom *et al.*, 2024] shows several factors affecting energy efficiency:

- GPU-based inference is usually more energy-efficient than CPU-based inference, especially for large-scale jobs.
- Devices like computers require significantly more energy during inference compared to specialized hardware.
- Response token length is one of the greatest predictors of energy usage, stressing the value of concise outputs.

[Stojkovic *et al.*, 2024a] showed that optimization techniques could lower energy usage by up to 20%, highlighting the possibility for further advancements in this area. Future work should study dynamic adaptation techniques that improve energy use based on application context, device capabilities, and workload characteristics.

6.4 Data Security and Privacy

Privacy issues are important when deploying SLMs, especially in applications that handle private user information. The risks connected with data leaks and illegal access must be carefully mitigated to ensure user trust and compliance with privacy regulations.

Privacy problems can be divided into three main areas:

- **Training Data Leakage:** Research by [Liu *et al.*, 2023] showed that data from later steps of pretraining is more sensitive to extraction. The writers suggest that attention mechanisms could be a key factor.
- **System Prompt Leaking:** [Li *et al.*, 2024b] found risks associated with illegal access to system prompts, which could lead to unintended behavior or misuse of the model.
- **Inference-Time Data Exposure:** Models applied in digital assistants or other apps often access sensitive user data, such as location or health records. Ensuring this data stays safe during inference is important.

Mitigation tactics include the use of differential privacy techniques, federated learning frameworks, and strong encryption protocols. However, more study is needed to handle the unique challenges offered by SLMs, especially as they become more integrated into privacy-sensitive applications.

7 Conclusion

The growing importance of Small Language Models (SLMs) lies in their ability to provide efficient and effective solutions across diverse applications, especially in resource-constrained settings. This paper offers a thorough review of SLMs, covering their designs, training methods, and model compression techniques, all of which contribute to their optimization and scalability. By giving an intuitive taxonomy of evaluation measures and summarizing diverse datasets, we have given a method for measuring SLMs’ success in different settings.

While the breakthroughs in SLM study have been impressive, several basic challenges remain unresolved. Issues such as hallucination, biases, inference-time energy economy, and data protection represent important areas that require further study.

We hope that this poll serves as a foundational resource for both academics and practitioners. By finding key areas for innovation and cooperation, we aim to inspire further improvements in the design, training, and implementation of SLMs. As these models continue to develop, they have the potential to revolutionize AI applications by making intelligence systems more accessible, efficient, and trustworthy. Our goal is to see SLMs driving impactful solutions across industries, bridging the gap between technical skills and social needs.

References

- [Beyer and Jones, 2024] Luc Beyer and Megan Jones. Paligemma: Towards compact vision encoders for multi-modal models. *Transactions on Image Processing*, 2024.
- [Cao *et al.*, 2024] Zhiwei Cao, Qian Cao, Yu Lu, Ningxin Peng, Luyang Huang, Shanbo Cheng, and Jinsong Su. Retaining key information under high compression ratios: Query-guided compressor for llms. *arXiv preprint arXiv:2406.02376*, 2024.
- [Chen *et al.*, 2023] Jie Chen, Hao Wang, and Ming Zhang. Internl12: Scalable multi-modal models with reduced vision

- encoder complexity. In *Advances in Neural Information Processing Systems*, 2023.
- [Dao and Gu, 2024] Tri Dao and Albert Gu. 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, 2024.
- [Dao *et al.*, 2022] Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *arXiv preprint arXiv:2205.14135*, 2022.
- [Frantar *et al.*, 2022] Elias Frantar, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.
- [Ge *et al.*, 2023] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023.
- [Gehman *et al.*, 2020] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020.
- [Gu *et al.*, 2024] Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024.
- [Guan *et al.*, 2024] Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, et al. Hallusionbench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14375–14385, 2024.
- [Han *et al.*, 2015] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1135–1143, 2015.
- [Havrilla and Iyer, 2024] Alex Havrilla and Maia Iyer. Understanding the effect of noise in llm training data with algorithmic chains of thought. *arXiv preprint arXiv:2402.04004*, 2024.
- [Havrilla, 2024] M. et al. Havrilla. Balancing noise and accuracy in privacy-preserving models. *Proceedings of the ACM*, 2024.
- [Hooper and Zhang, 2024] J. Hooper and T. Zhang. Key-value cache quantization for efficient long-sequence inference. *Journal of Machine Learning Research*, 2024.
- [Husom *et al.*, 2024] Erik Johannes Husom, Arda Goknil, Lwin Khin Shar, and Sagar Sen. The price of prompting: Profiling energy use in large language models inference. *arXiv preprint arXiv:2407.16893*, 2024.
- [Jiao *et al.*, 2019] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.
- [Katharopoulos *et al.*, 2020] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.
- [Kitaev *et al.*, 2020] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.
- [Laurençon and Picard, 2024] Jean Laurençon and Thomas Picard. Idefcis2: Efficient multi-modal fusion with lightweight visual encoders. *Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [Lester *et al.*, 2021] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.
- [Li and Zhao, 2024] Peng Li and Hua Zhao. Mini-gemini: Efficient multi-modal models with lightweight vision encoders. *Proceedings of the International Conference on Machine Learning*, 2024.
- [Li *et al.*, 2024a] Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*, 2024.
- [Li *et al.*, 2024b] Qinbin Li, Junyuan Hong, Chulin Xie, Jeffrey Tan, Rachel Xin, Junyi Hou, Xavier Yin, Zhun Wang, Dan Hendrycks, Zhangyang Wang, et al. Llm-pbe: Assessing data privacy in large language models. *arXiv preprint arXiv:2408.12787*, 2024.
- [Liu and Zhao, 2024] Fang Liu and Ting Zhao. Sophia: A memory-efficient optimizer for large-scale model training. *Transactions on Machine Learning Research*, 2024.
- [Liu *et al.*, 2023] Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.
- [Liu *et al.*, 2024] Chengkai Liu, Jianghao Lin, Jianling Wang, Hanzhou Liu, and James Caverlee. Mamba4rec: Towards efficient sequential recommendation with selective state space models. *arXiv preprint arXiv:2403.03900*, 2024.
- [Luo and He, 2024] Qiang Luo and Yan He. Mono-internvl: Mlp-based architectures for efficient multi-modal fusion. *Transactions on Neural Networks and Learning Systems*, 2024.
- [Micikevicius *et al.*, 2018] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *International Conference on Learning Representations*, 2018.

- [Parrish *et al.*, 2021] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel R Bowman. Bbq: A hand-built bias benchmark for question answering. *arXiv preprint arXiv:2110.08193*, 2021.
- [Patel *et al.*, 2024] Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Brijesh Warrier, Nithish Mahalingam, and Ricardo Bianchini. Characterizing power management opportunities for llms in the cloud. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, pages 207–222, 2024.
- [Peng *et al.*, 2023] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.
- [Radford *et al.*, 2019] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI Blog, 2019.
- [Rae *et al.*, 2021] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susanah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.
- [Rogers *et al.*, 2020] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.
- [Sanh, 2019] Victor Sanh. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.
- [Stojkovic *et al.*, 2024a] Jovan Stojkovic, Esha Choukse, Chaojie Zhang, Inigo Goiri, and Josep Torrellas. Towards greener llms: Bringing energy-efficiency to the forefront of llm inference. *arXiv preprint arXiv:2403.20306*, 2024.
- [Stojkovic *et al.*, 2024b] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. Dynamollm: Designing llm inference clusters for performance and energy efficiency. *arXiv preprint arXiv:2408.00741*, 2024.
- [Sun *et al.*, 2020] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 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, 2020. Association for Computational Linguistics.
- [Sun *et al.*, 2023] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.
- [Timiryasov and Tastet, 2023] Inar Timiryasov and Jean-Loup Tastet. Baby llama: Knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *arXiv preprint arXiv:2308.02019*, 2023.
- [Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2308.02019*, 2023.
- [Xiao *et al.*, 2023] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.
- [Xu *et al.*, 2023] Can Xu, Qingfeng Sun, Kai Zheng, Xubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.
- [Yu *et al.*, 2024] Da Yu, Peter Kairouz, Sewoong Oh, and Zheng Xu. Privacy-preserving instructions for aligning large language models. *arXiv preprint arXiv:2402.13659*, 2024.
- [Zhang *et al.*, 2023] Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.
- [Zhang *et al.*, 2024] Peiyuan Zhang, Guangtao Zeng, Tian-duo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.
- [Zhao *et al.*, 2023] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.
- [Zhu *et al.*, 2024] He Zhu, Junyou Su, Tianle Lun, Yicheng Tao, Wenjia Zhang, Zipei Fan, and Guanhua Chen. Fanno: Augmenting high-quality instruction data with open-sourced llms only. *arXiv preprint arXiv:2408.01323*, 2024.