Sparse Prompt Architecture Design for Scalable Fine-Tuning of Billion-Parameter Language Models

Andrahas White
Department of Computer Science, Binghamton University, Binghamton, NY, USA.
awhite@binghamton.edu

Abstract

The rapid scaling of language models into the hundreds of billions of parameters has created a pressing need for fine-tuning methods that do not demand full model replication or exorbitant computational budgets. Sparse prompt architectures have emerged as a promising paradigm, enabling selective insertion of learnable continuous vectors while keeping the pretrained backbone frozen. This paper presents a systems-oriented analysis of sparse prompt design for scalable fine-tuning, moving beyond accuracy metrics to examine structural trade-offs across the entire machine learning lifecycle. We investigate how sparse selection mechanisms can be architected to reduce serving latency, memory pressure, and communication overhead in distributed deployments. The discussion integrates infrastructure considerations such as model partitioning, dynamic batching, and edge-cloud orchestration, arguing that sparsity patterns must be co-designed with the target hardware topology. We further address cross-cutting concerns of robustness, fairness, and sustainability, showing that unconstrained prompt sparsification can inadvertently amplify dataset biases, introduce instability under distribution shift, and create uneven energy profiles across geo-distributed inference nodes. Governance frameworks for auditing prompt-based adapters, model cards that capture sparsity-induced behavioral boundaries, and policy guidelines for multi-tenant prompt serving are proposed as integral components of responsible deployment. By treating sparse prompts not as isolated algorithmic artifacts but as deeply embedded components within socio-technical infrastructures, the paper provides a holistic design methodology. The analysis culminates in a call for open, standardized evaluation suites that account for the full stack of hardware, software, and organizational constraints, ensuring that scalable fine-tuning architectures remain inclusive, verifiable, and environmentally conscious.

Keywords

sparse prompts, fine-tuning, large language models, scalable systems, parameter-efficient tuning, infrastructure, governance, fairness, sustainability.

1. Introduction

The contemporary landscape of natural language processing is dominated by language models whose parameter counts routinely exceed tens or hundreds of billions. While these models demonstrate remarkable few-shot and zero-shot capabilities, adapting them to specialized domains, regulated industries, or low-resource languages almost invariably demands some form of fine-tuning. Full parameter fine-tuning, wherein every weight is updated, is increasingly impractical due to the prohibitive costs of storing optimizer states, gradients, and multiple model replicas across GPU clusters. The systems community has therefore turned its attention to parameter-efficient fine-tuning methods that introduce a small number of additional trainable parameters while preserving the integrity of the frozen base model. Early approaches such as prompt tuning appended continuous task-specific vectors to the input

sequence, demonstrating that a tiny fraction of added capacity could rival full fine-tuning when the base model reached sufficient scale [1]. Prefix tuning extended this idea by prepending learned activations to all transformer layers, thereby providing a stronger inductive bias for generative tasks [2]. In parallel, low-rank adaptation methods injected trainable rank-decomposition matrices into attention blocks, achieving compelling accuracy while maintaining an extremely compact parameter footprint [3]. The adapter paradigm inserted small bottleneck modules between transformer layers, offering modularity and composability across tasks [4]. Even simpler techniques, such as updating only bias terms, proved surprisingly competitive in many settings [5]. Taken together, these methods constitute a vibrant design space where the primary goal has been to maximize downstream performance per trainable parameter.

However, the overwhelming focus on parameter counts and accuracy metrics obscures a broader set of system-level imperatives that become acute when fine-tuning is performed at scale across globally distributed infrastructures. The central contention of this paper is that sparse prompt architectures must be formalized not merely as algorithmic recipes but as full-stack systems that interface with hardware accelerators, network fabrics, serving pipelines, and organizational governance structures. This perspective is motivated by the convergence of several trends. First, the proliferation of mixture-of-experts architectures has normalized the idea that large models can activate only selective sub-networks per input token, and this sparsity can be exploited for both training and inference acceleration [6]. Second, quantization and distillation techniques have shown that aggressive compression can be applied without catastrophic degradation, highlighting the latent redundancy in dense models [7, 8]. Third, the operational reality of large-scale serving involves complex trade-offs among latency, throughput, and energy consumption that are rarely considered in method-centric evaluations. Sparse prompts, in which only a subset of inserted vectors are activated based on input context, present a natural junction where these concerns intersect. By designing the selection logic at the architectural level, it becomes possible to control not just the number of trained parameters but also the dynamic computational graph during inference, directly influencing tail latency and memory bandwidth utilization.

The contribution of this work is a systematic exploration of the structural, infrastructural, and socio-technical dimensions of sparse prompt architecture design. Rather than proposing a novel algorithm, we map the design space across multiple layers of abstraction, emphasize trade-offs that are often invisible in benchmark-driven research, and prescribe a set of architectural principles grounded in systems thinking. The remainder of the paper is organized as follows. Section 2 establishes the key challenges that scalable fine-tuning must address, framing them through the lenses of computational heterogeneity, data governance, and life-cycle management. Section 3 dissects the design space of sparse prompt architectures, analyzing sparsification granularity, selection controllers, and the interplay with model parallelism. Section 4 examines infrastructure and deployment considerations, from edge-cloud splitting to fault tolerance. Section 5 delves into governance, fairness, and sustainability, arguing for integrated auditing and energy-aware scheduling. Section 6 addresses robustness, structural resilience, and the perils of over-sparsification. Section 7 reflects on policy implications and future institutional frameworks, and Section 8 concludes with a synthesis and recommendations.

2. Scalable Fine-Tuning: Challenges Beyond Parameter Budgets

The conventional narrative around scalable fine-tuning has centered on the tension between parameter efficiency and task performance. While this framing has yielded valuable innovations, it systematically overlooks four categories of challenges that dominate production-grade deployments: computational heterogeneity, data locality and provenance, life-cycle dynamics, and environmental externalities. Understanding these challenges is a prerequisite for any architectural discussion because the effectiveness of a sparse prompt design is ultimately evaluated not on a single GPU but within a heterogeneous fabric of accelerator types, memory hierarchies, and geo-distributed edge nodes.

Computational heterogeneity refers to the reality that fine-tuning and inference jobs span diverse hardware, from data center-grade H100 clusters to edge accelerators such as Apple Neural Engine or Qualcomm Hexagon. A sparse prompt module trained on high-bandwidth memory might exhibit drastically different throughput on a memory-constrained edge device that can only afford to materialize a fraction of the prompt embeddings at a time. The design of the sparsification controller must therefore be aware of the target deployment tier, trading off expressiveness against memory and latency budgets in a hardware-conscious manner. Traditional parameter-efficient tuning methods such as adapters and low-rank adaptation do not natively expose this controllability, as they apply fixed transformations regardless of input complexity or available compute. Sparse prompts, by contrast, can be designed with gating mechanisms that dynamically select the number of active prompt tokens based on both linguistic features and runtime resource signals, creating a direct lever for hardware-aware adaptation.

Data locality and provenance introduce a second layer of complexity. In many regulated sectors, such as healthcare, finance, and public administration, fine-tuning data cannot be freely moved across jurisdictional boundaries. Federated fine-tuning frameworks aim to address this by keeping data on local devices or within national clouds, but they exacerbate the difficulty of maintaining consistent sparsity patterns across participants. If different nodes independently learn different sparse activation masks, aggregating them into a global sparse prompt while preserving the inductive biases of each mask becomes a non-trivial systems problem. The architecture must therefore incorporate a protocol for mask reconciliation, versioning, and provenance tracking, ensuring that the lineage of each prompt segment is auditable. This requirement ties directly to broader data governance frameworks that demand traceable model updates, especially when prompts encode task-specific knowledge that might inadvertently capture sensitive distributional signatures.

Life-cycle dynamics refer to the continuous evolution of the base model, the prompt library, and the deployment environment. Billion-parameter models are frequently updated through patching, continued pretraining, or reinforcement learning from human feedback. A sparse prompt that was optimized for one checkpoint may suffer from internal distribution shift when applied to a successor model, as the activation subspaces of the backbone shift. Without life-cycle-aware design, the operational cost of re-validating and potentially re-training sparse prompts after every base model release can overwhelm the savings gained from parameter efficiency. Architectural support for prompt stability metrics, gradual prompt decay, and compatibility layers that map old prompt embeddings to new representation spaces becomes essential for sustainable operations at scale.

Environmental externalities, particularly carbon emissions from GPU clusters, have attracted increasing scrutiny. Parameter-efficient methods are often portrayed as inherently green because they reduce the number of floating-point operations during training. However, this

narrative can be misleading when fine-tuning is performed millions of times across myriad tasks and tenants in a shared cloud infrastructure. The cumulative energy footprint of serving lightweight prompts can still be enormous if the sparsity controller is not optimized for minimal activation. Sparse prompts offer an under-explored opportunity to incorporate carbon-aware scheduling: the same input could be processed with a sparser prompt configuration during peak grid carbon intensity and a denser one when renewable supply is abundant, effectively turning the fine-tuning architecture into a demand-response participant in the energy system. This perspective elevates sparse prompt design from a purely algorithmic exercise to a component of organizational sustainability policy.

3. Design Space of Sparse Prompt Architectures

Sparse prompt architectures rest on the intuition that not all prompt tokens are equally necessary for every input. By inserting a learnable selection mechanism, the system can dynamically activate only a subset of prompt embeddings, reducing both computational cost and potential interference from irrelevant prompt parameters. The design space can be decomposed along three orthogonal axes: sparsification granularity, the architecture of the selection controller, and the integration with the underlying model's parallelism strategy. Each axis entails structural trade-offs that ripple through training dynamics, inference latency, and system maintainability.

Sparsification granularity determines the atomic unit on which sparsity is enforced. The finest granularity operates at the level of individual continuous prompt tokens, where a binary or soft mask is computed per position, and only those positions with mask values above a threshold are injected into the transformer's key-value cache. Coarser granularities group prompt tokens into semantic blocks, such as topic-related segments or layers in a prefix-prompt hybrid, and sparsify at the block level. Coarse-grained sparsification simplifies the mask routing logic and is more amenable to compiler-level optimizations that pre-allocate memory for contiguous blocks, but it may retain tokens that are only partially useful, wasting capacity. Fine-grained sparsification offers higher precision but requires unstructured gather operations that can be detrimental on hardware with cache line constraints. An intermediate design uses a two-stage hierarchy: a lightweight token-level saliency scorer followed by a block-level aggregator that enforces structural constraints, combining the adaptability of fine granularity with the hardware friendliness of coarse blocking.

The selection controller is the neural component that decides which prompt portions to activate. Its design must balance expressiveness against the overhead it introduces. A simple yet effective approach is to employ a small linear projection on top of the input's sentence embedding or averaged hidden states, outputting a logit vector over prompt token slots from which the top-k are selected. More expressive controllers incorporate lightweight attention over a set of learned prototypes, enabling context-dependent routing that mimics mixture-of-experts routing but at the prompt level. A critical system consideration is whether the controller is end-to-end differentiable or uses discrete sampling with gradient estimation. Differentiable soft selection enables stable training with standard optimizers but leads to dense computations during training, as all prompt tokens receive gradient signal through the softmax. Hard selection with reinforcement learning or straight-through estimators trains a genuinely sparse forward pass but can suffer from high variance and convergence difficulties. Recent work has introduced mechanisms for selectively inserting prompts to achieve competitive performance with reduced prompt token overhead [18]. Such selective insertion operates by learning per-layer decisions about whether to attach prompt vectors at all,

effectively turning sparsification into a gating problem across depth. This design illustrates the potential of unifying token-level and layer-level sparsity in a single trainable controller.

Integrating sparse prompt architectures with model parallelism is a non-trivial systems challenge. In tensor-parallel deployments, the prompt embeddings and their masks must be partitioned across devices so that the sparsification decision does not introduce cross-device synchronization bottlenecks. If the selection controller resides on a single device, it becomes a serialization point that stalls the entire pipeline. Distributing the controller itself, perhaps by sharding the scoring function along with the prompt table, aligns with the dataflow patterns of the backbone and can reduce communication volume, as each device only needs to fetch the mask for its local shard. However, inconsistent sparsity patterns across devices can lead to imbalanced computation, especially when prompt tokens disproportionately fall onto one pipeline stage. Co-designing the sparsification mask with load-balancing constraints, drawing inspiration from mixture-of-experts load-balancing losses, yields systems that maintain stable throughput under varying input distributions. Moreover, in inference scenarios where model quantization is employed, the precision of the prompt embeddings and the mask storage format must be jointly considered, because mismatched numerical representations can lead to subtle degradation that only manifests in end-to-end latency.

The choice of sparsity scheduling also affects the operational life cycle. Static sparsity patterns, fixed after training, offer predictable performance and are easier to compile into optimized inference graphs. Dynamic sparsity, where the prompt configuration adapts per input, yields superior accuracy-latency Pareto frontiers but requires runtime decision engines that consume power and silicon area. A pragmatic system can adopt hybrid scheduling: a default sparse configuration is compiled ahead of time for the most frequent request classes, while a dynamic controller is invoked only for out-of-distribution inputs or when latency budgets permit. This tiered model aligns with the classic systems principle of separating the fast path from the slow path and provides a graceful degradation mechanism under overload.

4. Infrastructure and Deployment Considerations

Deploying sparse prompt-tuned models at scale requires rethinking the software stack from the cluster manager down to the kernel launch configuration. The insertion of sparse, context-dependent prompt tokens transforms the computational graph from a static, pre-determined shape into a dynamic one, which poses challenges for graph optimization, memory pre-allocation, and pipelining. Infrastructure providers must decide whether to represent prompt placement as a series of gather-scatter operations on the key-value cache or to incorporate prompts as virtual layers with conditional execution semantics. The former approach aligns with key-value cache-centric serving systems that treat the cache as a mutable tensor, while the latter is more natural for graph-based compilers that perform ahead-of-time layout planning.

Serving systems designed for generative large language models, such as Orca, leverage iteration-level scheduling to handle the variable sequence lengths and attention patterns typical of autoregressive decoding [19]. The introduction of sparse prompts amplifies this variability because the prompt mask may change not only across requests but also across decoding steps within a single generation. A request that begins with a sparsified prompt might, in later steps, require the insertion of additional prompt tokens as the context grows and triggers the selection of new semantic slots. Supporting intra-generation prompt adaptation calls for a streaming control interface between the prompt controller and the inference engine, where mask updates are injected asynchronously without stalling the

attention kernel. Designing such an interface demands careful attention to consistency; the attention mask for future tokens must reflect the updated prompt set, necessitating just-in-time updates to the causal masking logic.

Geo-distributed deployments introduce further infrastructure complexity. When users in different continents interact with a fine-tuned model, serving latency can be minimized by placing prompt adapters in edge data centers close to the user, while the large frozen backbone remains in a central compute region. Sparse prompts are particularly well-suited for this edge-cloud split because the prompt selection controller and the prompt embeddings themselves constitute a minuscule fraction of the total model size. An edge node can host a local cache of prompt embeddings and a lightweight controller, performing sparsification and embedding lookups before sending a compact representation of the selected prompts along with the user input to the central backbone. This architecture reduces ingress bandwidth requirements and allows the backbone to treat incoming requests as a standard batch of token sequences with pre-computed key-value cache prefixes. The primary engineering challenge lies in maintaining consistency between the prompt inventory at the edge and the version of the backbone in the cloud, a problem that can be managed through distributed key-value stores with version vectors and lazy cache invalidation policies inherited from content delivery networks.

Fault tolerance and resilience demand specific design provisions. In a distributed serving cluster, the failure of a node hosting a portion of the prompt embeddings could render a subset of tasks degraded. Sparse architectures can be engineered to be fault-tolerant by introducing redundancy in prompt storage through erasure coding or replication across pod replicas. Moreover, because the sparsification controller can be tuned to rely on multiple partially overlapping prompt sets, a single missing prompt segment may be compensated for by activating alternative semantically similar slots. This graceful degradation property transforms the prompt set from a rigid table into a flexible associative memory that can tolerate partial infrastructure outages without complete service disruption. Such design patterns echo the resilience strategies long employed in distributed storage systems and offer a path toward mission-critical deployments of prompt-tuned language models.

5. Governance, Fairness, and Sustainability

The proliferation of prompt-based adapters across organizations raises important questions about algorithmic governance. Each sparse prompt configuration encodes a specific task adaptation and, by extension, encapsulates a set of normative and cultural assumptions derived from its fine-tuning corpus. When multiple tenants share a common backbone and operate independent prompt storefronts, the potential for conflicting or biased adaptations to coexist within the same serving infrastructure necessitates robust governance mechanisms. Sparse architectures introduce an additional layer of complexity because the sparsification controller itself can amplify or suppress certain voices based on which prompt slots it deems salient for a given demographic or linguistic group. Fairness auditing must therefore penetrate the selection logic, examining whether the sparsity mask systematically excludes prompt segments that encode minority language conventions or culturally specific knowledge.

A practical governance instrument is the expansion of model card reporting to encompass sparse prompt behaviors. Standard model cards document intended use, evaluation results, and limitations [14]. For sparse prompt systems, model cards should additionally specify the sparsification distribution across different population slices, the provenance of the data used to train the selection controller, and the failure modes observed when the sparsity budget is

severely constrained. Tools that visualize prompt activation patterns across demographic groups can provide transparency, enabling external auditors to detect disparities in which knowledge sources are activated for different user cohorts. Such auditing aligns with the calls for algorithmic impact assessments that are gaining traction in legislative proposals worldwide.

The intersection of sparse prompts and fairness extends to resource allocation within multi-tenant clouds. A sparse prompt that aggressively reduces latency for high-priority tenants while leaving standard tenants with denser, slower configurations could introduce a structural tiering of service quality that mirrors existing social stratifications. Cloud providers and platform operators must establish service-level objectives that bound the disparity in sparsity ratio across tenants, ensuring that efficiency gains do not translate into discriminatory quality-of-service degradation. This requires monitoring frameworks that track the distribution of sparsity masks per tenant category and flag statistical deviations that warrant human review.

Sustainability concerns are accentuated by the dynamic nature of sparse prompts. The ability to modulate sparsity in response to grid signals, as discussed earlier, can transform a model serving fleet into a flexible load resource. To realize this vision, infrastructure orchestration layers must expose carbon intensity metrics to the prompt controller, allowing it to trade off a slight increase in perplexity for a substantial drop in energy consumption during peak fossil-fuel periods. The modeling of such trade-offs benefits from system-level optimization that jointly considers the energy cost of moving prompt embeddings across the memory hierarchy and the carbon cost of computing additional attention operations. Lifecycle assessments that account for the embodied carbon of manufacturing the accelerators that ultimately house and execute sparse prompts provide a holistic view, ensuring that fine-tuning innovations do not simply shift environmental burdens from one segment of the stack to another.

6. Robustness and Structural Resilience

The robustness of sparse prompt architectures under distribution shift is of paramount concern for high-stakes applications. Because the sparsification controller partitions the prompt space based on features observed during training, encountering inputs that lie far from the training manifold can lead to degenerate mask patterns that either collapse to near-zero prompts, effectively stripping the model of task-specific guidance, or inadvertently activate a dissonant set of prompt slots that inject contradictory signals. Such fragility must be understood not as a failure of a specific implementation but as a structural property of learned sparse gating. Mitigations include training the controller with adversarial perturbations that corrupt the input representation used for sparsification, equipping the controller with an uncertainty estimation head that can fall back to a dense prompt ensemble when confidence is low, and incorporating a minimum sparsity floor that guarantees a baseline of adaptation regardless of input novelty.

Another dimension of robustness concerns adversarial attacks that deliberately manipulate the input to cause the selection of harmful or copyrighted prompt segments. An adversary could craft inputs that trigger retrieval of prompt embeddings associated with toxic completions or confidential enterprise knowledge, effectively jailbreaking the adapter. The defense in depth required to counter such threats involves hardening the controller against gradient-based attacks, rate-limiting requests that exhibit erratic sparsity patterns, and cryptographically signing prompt tables to prevent tampering during edge caching. Intrusion detection systems that profile the distribution of sparsity masks across the request stream can raise alerts when sudden shifts indicative of an ongoing attack occur, much like anomaly detection in network security.

The structural resilience of the deployment fleet itself is enhanced by the decoupling of prompt controllers from the backbone. In the event that a newly deployed sparse prompt configuration exhibits catastrophic regression on a subset of inputs, rollback can be performed by redirecting traffic to a previous prompt version stored in a shadow table without restarting the backbone container. This hot-swapping capability reduces mean time to recovery and encourages a culture of bold experimentation with sparsification schemes. To make rollback safe, prompt controllers should be designed with monotonic version identifiers and backward-compatible mask formats, ensuring that a newer controller can still consume prompt embeddings stored in an older layout. Version skew between controllers and embedding tables is a classic distributed systems problem that, if ignored, can lead to silent data corruption; sparse prompt ecosystems benefit from adopting schema evolution practices well-established in database management.

7. Policy Implications and Future Institutional Frameworks

The ascendancy of sparse prompt architectures within the billion-parameter model ecosystem necessitates a re-examination of current regulatory instruments. Existing AI regulations often focus on the model as a monolithic artifact, mandating conformity assessments, risk classifications, and transparency obligations at the model level. Sparse prompt adapters, being lightweight and easily distributable, challenge this monolithic view because they can be created, shared, and modified by a wide range of actors beyond the original model provider. A regulatory regime that treats the prompt adapter as an independent component requiring its own compliance verification would close accountability gaps without stifling innovation. In practice, this could mean that sparse prompt registries similar to app stores emerge, where adapters are submitted with declared intended use, bias test reports, and compatibility matrices against certified backbone versions.

The institutional economics of sparse prompts also merit attention. Because sparse adapters can encode highly valuable domain expertise at a fraction of the cost of full model training, they are likely to become a medium of exchange in data markets. Organizations might license sparse prompt configurations for radiology report generation or legal contract summarization, embedding their proprietary knowledge in a compact, tradeable format. This scenario raises intellectual property questions: does a sparse prompt constitute a derivative work of the backbone, and what rights do prompt authors hold over embeddings that were learned using a combination of private data and public checkpoints? Open-source licensing models that explicitly address the status of adapter weights and sparsity masks would provide much-needed clarity and reduce litigation risk. The academic community, open-source foundations, and standards bodies should collaborate to draft reference license agreements that balance commercial incentives with broad access.

Looking forward, the trajectory of sparse prompt architectures will be shaped by advances in hardware-software co-design. Emerging accelerators with native support for unstructured sparsity can execute fine-grained prompt selection with negligible overhead, encouraging the development of even more expressive controllers. Conversely, the shift toward disaggregated memory architectures, where compute and memory pools are physically separated, will put a premium on prompt sparsification that minimizes inter-pool data movement. Research that co-optimizes the sparsification policy, the memory tiering strategy, and the network topology will unlock new regimes of cost efficiency. At the same time, standardization efforts must ensure that sparse prompt representations remain interoperable across frameworks such as PyTorch, JAX, and ONNX, preventing vendor lock-in and fostering a competitive ecosystem.

The sustainability and fairness dimensions discussed earlier should be embedded into the fabric of such standards. A global consortium that defines eco-rating labels for prompt adapters, akin to ENERGY STAR, could empower consumers and enterprises to make environmentally conscious choices when selecting fine-tuning configurations. Auditing protocols that routinely probe sparse prompt controllers with counterfactual inputs representing diverse demographic groups would become a baseline requirement for deployment in the public sector. These institutional mechanisms, though non-technical in nature, are critical complements to algorithmic advances, ensuring that scalability is achieved not at the expense of societal values but in concert with them.

8. Conclusion

Sparse prompt architectures represent a convergence of parameter efficiency, dynamic computation, and modular adaptation that is ideally suited to the next generation of billion-parameter language models. This paper has argued that realizing their full potential requires a systems perspective that spans hardware-aware controller design, distributed infrastructure orchestration, governance frameworks, and policy innovation. By analyzing the design space along the axes of granularity, selection logic, and parallelism integration, we have highlighted the structural trade-offs that determine real-world viability, from tail latency in geo-distributed serving to the carbon intensity of prompt-based inference. The integration of governance instruments, including expanded model cards and fairness audits for sparsification patterns, ensures that these architectures remain accountable. The challenge ahead is not merely to improve benchmark scores on curated datasets but to build end-to-end socio-technical systems that are efficient, resilient, fair, and sustainable. Achieving this vision demands interdisciplinary collaboration among machine learning practitioners, systems engineers, regulators, and civil society, all working toward fine-tuning infrastructures that respect both computational and human constraints.

References

1. Lester, B., Al-Rfou, R., & Constant, N. (2021). The power of scale for parameter-efficient prompt tuning. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (pp. 3045–3059). Association for Computational Linguistics.
2. Li, X. L., & Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (pp. 4582–4597). Association for Computational Linguistics.
3. Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2022). LoRA: Low-rank adaptation of large language models. In International Conference on Learning Representations.
4. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., ... & Gelly, S. (2019). Parameter-efficient transfer learning for NLP. In International Conference on Machine Learning (pp. 2790–2799). PMLR.
5. Zaken, E. B., Ravfogel, S., & Goldberg, Y. (2022). BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (pp. 1–9). Association for Computational Linguistics.

6. Fedus, W., Zoph, B., & Shazeer, N. (2022). Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research, 23(120), 1–39.

7. Dettmers, T., Lewis, M., Belkada, Y., & Zettlemoyer, L. (2022). LLM.int8(): 8-bit matrix multiplication for transformers at scale. In Advances in Neural Information Processing Systems.

8. Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In Advances in Neural Information Processing Systems Workshop.

9. Wang, Z., Zhang, Y., & Zhai, C. (2022). Continual prompt tuning for dialog state tracking. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (pp. 1124–1136). Association for Computational Linguistics.

10. Narayanan, D., Shoeybi, M., Casper, J., LeGresley, P., Patwary, M., Korthikanti, V., ... & Catanzaro, B. (2021). Efficient large-scale language model training on GPU clusters using Megatron-LM. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 1–14). ACM.

11. Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (pp. 610–623). ACM.

12. Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 3645–3650). Association for Computational Linguistics.

13. Holstein, K., Wortman Vaughan, J., Daumé III, H., Dudik, M., & Wallach, H. (2019). Improving fairness in machine learning systems: What do industry practitioners need? In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (pp. 1–16). ACM.

14. Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., ... & Gebru, T. (2019). Model cards for model reporting. In Proceedings of the Conference on Fairness, Accountability, and Transparency (pp. 220–229). ACM.

15. Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., ... & Liang, P. (2021). On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258.

16. Tay, Y., Dehghani, M., Bahri, D., & Metzler, D. (2020). Efficient transformers: A survey. arXiv preprint arXiv:2009.06732.

17. Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

18. Zhu, W., & Tan, M. (2023, December). SPT: Learning to selectively insert prompts for better prompt tuning. In Proceedings of the 2023 conference on empirical methods in natural language processing (pp. 11862-11878).

19. Yu, G., Jeong, J. S., Kim, G. W., Kim, S., & Chun, B. G. (2022). Orca: A distributed serving system for transformer-based generative models. In 16th USENIX Symposium

on Operating Systems Design and Implementation (OSDI 22) (pp. 521–538). USENIX Association.

20. Frantar, E., Ashkboos, S., Hoefler, T., & Alistarh, D. (2023). GPTQ: Accurate post-training quantization for generative pre-trained transformers. In International Conference on Learning Representations.

21. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., ... & Sifre, L. (2022). Training compute-optimal large language models. In Advances in Neural Information Processing Systems.

22. Dao, T., Fu, D. Y., Ermon, S., Rudra, A., & Ré, C. (2022). FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In Advances in Neural Information Processing Systems.

23. Gehman, S., Adewumi, T., Lal, N., & Koyejo, S. (2020). RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In Findings of the Association for Computational Linguistics: EMNLP 2020 (pp. 3356–3369). Association for Computational Linguistics.

24. Weidinger, L., Mellor, J., Rauh, M., Griffin, C., Uesato, J., Huang, P. S., ... & Gabriel, I. (2021). Ethical and social risks of harm from language models. arXiv preprint arXiv:2112.04359.

25. Jobin, A., Ienca, M., & Vayena, E. (2019). The global landscape of AI ethics guidelines. Nature Machine Intelligence, 1(9), 389–399.