Prompt-Adaptive Quantization: Adaptive Per-Prompt Routing for Efficient LLM Inference

Gabriel Jimenez[*], Vivann Khanna, Rishi Sastri, Raine Ma, Soham Chatterjee[†], Kevin Zhu[†], Sunishchal Dev[†]

Algoverse AI Research

gabrieln_jimenez@icloud.com

Abstract

Large Language Models (LLMs) produce strong results but are costly to serve. Static post-training quantization reduces memory and compute, yet uses a single bit width for all prompts, wasting resources on easy inputs and degrading accuracy on harder ones. We introduce **Prompt-Adaptive Quantization (PAQ)**, a per-prompt precision framework that requires no retraining of the underlying model. PAQ trains a lightweight BERT router with perplexity-guided supervision to select the smallest adequate quantization level (2, 4, 8, or 16 bits) per input. At inference, prompts are automatically routed to the appropriate pre-quantized LLM variant. Overall, PAQ serves as a novel framework for adaptive per-prompt quantization, reducing latency while maintaining strong accuracy across tasks.

Introduction

Large language models (LLMs) have achieved state-of-the-art performance across a wide range of tasks, from open-domain question answering to multi-step reasoning (Minaee et al. 2024). Yet their deployment is often constrained by compute and memory requirements (Zhou et al. 2024). Quantization reduces these costs by compressing parameters from full-precision to lower-bit representations. Approaches such as GPTQ (Frantar et al. 2022b), AWQ (Lin et al. 2023b), and SmoothQuant (Xiao et al. 2023) show that *static* quantization can yield substantial efficiency improvements while maintaining competitive accuracy. However, static quantization assigns a fixed bit-width across all prompts, wasting resources on simple inputs and degrading performance on complex or long-context queries (Lu et al. 2024).

Adaptive inference suggests that tailoring compute to input difficulty can preserve model quality while reducing cost (Leviathan, Kalman, and Matias 2023; Schuster et al. 2023; Shazeer et al. 2017b). Early-exit methods (Leviathan, Kalman, and Matias 2023; Schuster et al. 2023) and mixture-of-experts architectures (Shazeer et al. 2017b) dynamically allocate computation across layers or subnetworks. Yet this adaptivity has not been extended to quantization.

We address this gap with **Prompt-Adaptive Quantization (PAQ)**, a framework that dynamically selects quantization levels on a per-prompt basis. PAQ uses a lightweight BERT router trained with perplexity-guided supervision: each prompt is labeled with the smallest quantized LLaMA-3.1-8B variant (2, 4, 8, or 16 bits) achieving perplexity below a threshold ($\tau = 1.13$). At inference, the router directs prompts to the appropriate model, reducing compute and memory while preserving accuracy.

Our contributions are as follows:

1. We introduce the first end-to-end system for per-prompt adaptive quantization that requires no retraining of the base model.
2. We develop a perplexity-driven labeling strategy for training a prompt router.
3. We perform a comprehensive evaluation across QA and reasoning benchmarks, showing that PAQ delivers near-baseline accuracy while substantially reducing inference cost.

Related Work

Static quantization methods such as GPTQ (Frantar et al. 2022a) and AWQ (Lin et al. 2023a) apply a fixed bit-width across all inputs, while adaptive approaches seek to adjust precision based on input or model characteristics. For example, OWQ (Lee et al. 2024) uses outlier-aware Hessian metrics to achieve 3.1-bit models with minimal degredation, QAQ (Cheng et al. 2024) applies distinct strategies for key/-value caches, and Delta-CoMe (Ping et al. 2024) enables training-free mixed precision for fine-tuned LLMs. In contrast, PAQ routes entire prompts to pre-quantized models based on predicted complexity using perplexity thresholds. Unlike confidence-based early-exit networks (Schuster et al. 2021) or MoE gating (Shazeer et al. 2017a), this approach provides an interpretable, model-agnostic signal for selecting the minimal sufficient precision without retraining or architectural changes.

Methodology

When using quantized versions of an LLM, there is typically a trade-off between efficiency and accuracy. Lower-bit representations reduce memory usage and inference latency, but also limit the numerical precision of weights and activations.

[*]Lead author

[†]Senior author

Accepted at the AAAI 2026 Workshop AIR-FM, Assessing and Improving Reliability of Foundation Models in the Real World.

As a result, performance typically degrades on complex or ambiguous prompts, where higher precision is often necessary to maintain reliable predictions.

Prompt-Adaptive Quantization (PAQ) addresses this challenge by dynamically selecting the smallest pre-quantized model sufficient for each prompt using a custom BERT-based router. In practice, we use freely available pre-quantized LLaMA-3.1-8B models at 2-, 4-, 8-, and 16-bit precision from Hugging Face (Bartowski 2024; Grattafiori et al. 2024) as well as quantized versions of LLaMA-3.2-3B (Grattafiori et al. 2024) and Qwen3-8B (Yang et al. 2025) at the same bit-widths, ensuring a wide range of quantization levels to balance efficiency and accuracy. Our contribution lies in building a system that automatically routes prompts to the appropriate model, achieving significant computational and memory savings without compromising task performance.

Pre-Quantized Models

First, we define a set of pre-quantized models, each operating at different bit lengths:

$$\{f^{(2)}, f^{(4)}, f^{(8)}, f^{(16)}\},$$

where the superscript denotes the bit-width of the model. From probability logging, we then compute the *perplexity* of the model's response.

Perplexity provides a simple measure of how confident a model is in its token predictions for a prompt, measuring the average uncertainty across all predicted tokens (Jurafsky et al. 2023). Intuitively, lower perplexity can be seen as the model assigning higher confidence to its predictions, which often corresponds to less variability in responses and a tendency toward greater accuracy (Choi et al. 2020). Conversely, higher perplexity reflects greater uncertainty, which may indicate that a higher-precision model would be more appropriate.

Labeling Prompts with the Smallest Sufficient Model

We explore three threshold strategies to balance efficiency and accuracy: a conservative threshold ($T = 1.07$) that prioritizes accuracy by routing more prompts to higher-bit models, a medium threshold ($T = 1.13$) that balances efficiency and accuracy, and an aggressive ($T = 1.21$) threshold that maximizes efficiency by routing more prompts to lower-bit models. We selected three thresholds that maintain high accuracy while utilizing all quantization levels, based on average accuracy of the base LLMs on train datasets. For example, the conservative threshold chosen ($T = 1.07$) had 77.2% accuracy, however, demonstrated a high usage of the 16-bit model. The aggressive threshold ($T = 1.21$) had a total precision of 76. 08% while demonstrating a preference for the 2-bit model, and the medium threshold ($T = 1.13$) was more balanced with an accuracy of 76.55% while maintaining a much closer distribution between bit widths. We use perplexity thresholds rather than direct accuracy because perplexity can provide a more reliable measure of the model's understanding of the prompt (Gonen et al. 2024).

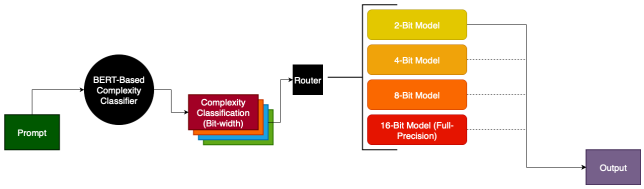

Figure 1: Architecture of Prompt-Adaptive Quantization (PAQ). A ModernBERT classifier routes prompts to appropriate quantized models (2-16 bit) based on predicted complexity.

By capturing the model's level of uncertainty and confusion, perplexity helps ensure that correct responses are not the result of chance or spurious patterns, allowing us to more confidently assess whether the model genuinely comprehends the prompts. For each threshold T, we define:

$$k^* = \min k \in 2, 4, 8, 16 \mid \text{PPL}(x; f^{(k)}) \leq T,$$

where $\text{PPL}(x; f^{(k)})$ is the perplexity of model $f^{(k)}$ on x. We train separate BERT routers for each threshold strategy to evaluate their trade-offs.

For training, we constructed a balanced dataset of 110,000 prompts, with a maximum of 50,000 drawn from each benchmark (Cobbe et al. 2021b; Rajpurkar et al. 2016a; Dua et al. 2019a; Peter et al. 2018). Each router is trained on prompts labeled according to its respective threshold. To avoid biasing the routers toward a single precision level, we downsampled so that prompts were approximately evenly distributed across the four quantization levels for each threshold setting. We randomly split each dataset into train and validation sets, using validation to tune hyperparameters and using a separate test set for final performance reporting. In order to measure the performance of all 3 threshold-level routers, we evaluate their ability to match expected routing results based on pure threshold prompt sorting.

Router Performance

After evaluating all 3 threshold-level routers in performance, we determined that the medium threshold ($T = 1.13$) was the most aligned with our goals in maintaining accuracy while increasing efficiency. On our early router evaluation dataset (3000 samples), the medium threshold performed consistently well across all precision classes, striking a balance between aggressive quantization and stability. While the conservative router ($T = 1.07$) preserved slightly higher accuracy in routing, it failed to yield meaningful computational savings. Conversely, the extreme router ($T = 1.21$) achieved the highest compression and throughput gains but at a notable cost to accuracy, showing extreme preference towards 2-bit model usage (53.3%). Under the medium configuration, the overall accuracy reached 70.61%. This suggests that the router successfully allocated more complex inputs to higher-precision paths, thereby preserving semantic fidelity without overusing costly precision tiers.

Router Architecture and Training

The router is a lightweight classifier designed to predict the minimal sufficient quantization level for a given prompt, implemented using ModernBERT (Warner et al. 2024). Each input is processed by the classifier to produce scores for the four quantization levels (2, 4, 8, or 16 bits), which are converted into probabilities to select the most likely bit-width.

The router is trained as a standard classifier, with the correct quantization level as the target. The loss encourages higher probability for the correct class, teaching the router which prompts can safely use lower-bit models and which require higher precision. We initially trained with 5,000 samples per dataset, but found that more data was needed for the model to generalize. To address this, we added 45,000 samples per dataset, and for datasets that did not reach 50,000 samples, we used the full training split. PAQ is trained on the train splits of five frequently used datasets: SQuAD v2 (Rajpurkar et al. 2016b), DROP (Dua et al. 2019b), GSM8K (Cobbe et al. 2021a), ARC-Easy, and ARC-Challenge (Peter et al. 2018).

We train with a batch size of 16 for five epochs using AdamW (Loshchilov et al. 2019), monitoring validation accuracy to ensure generalization and avoid overfitting. Early observations show the router quickly learns to distinguish simple from complex prompts, assigning low-bit models to easy queries and reserving high-bit models for more challenging cases.

We experimented with different labeling strategies and data splits, finding that including a diverse mix of prompts from multiple datasets improved generalization across QA and reasoning tasks. This training setup allows the router to serve as a reliable method for adaptive quantization, balancing computational efficiency and accuracy without retraining the underlying LLM.

Inference Procedure

During inference, the system operates in three steps:

1. The router predicts the probability distribution $\hat{p}^{(k)}$ over the four quantization levels for a new prompt x.
2. The predicted quantization level is selected as:

$$\hat{k} = \arg\max_k \hat{p}^{(k)}.$$

3. The prompt is forwarded to the pre-quantized model $f^{(\hat{k})}$ to produce the output.

This adaptive routing allows low-bit models to handle easy prompts, reducing compute and memory, while higher-precision models process harder prompts to maintain performance (Xia et al. 2008; Roy et al. 2021).

Experiments

Experimental Setup

We evaluate PAQ on seven widely used benchmarks: SQuAD v2 (Rajpurkar et al. 2016b), DROP (Dua et al. 2019b), GSM8K (Cobbe et al. 2021c), MMLU (Hendrycks et al. 2021), HumanEval (Chen, Tworek et al. 2021), ARC-Easy, and ARC-Challenge (Peter et al. 2018). We report task-specific metrics (F1/EM, accuracy, or code correctness) and latency per prompt. Models are served using `transformers` on an NVIDIA RTX 6000 Ada GPU. Baselines include each individual quantized variant (2-, 4-, 8-, 16-bit) and the 16-bit model for all prompts, as well as a randomized router to verify that results are due to the actual performance of the router. We use Llama 8B (Grattafiori et al. 2024), Llama 3B (Grattafiori et al. 2024), and Qwen 8B (Yang et al. 2025) to evaluate performance. Unless otherwise noted, results are averaged across datasets to highlight trends in scaling, routing efficiency, and precision–accuracy tradeoffs.

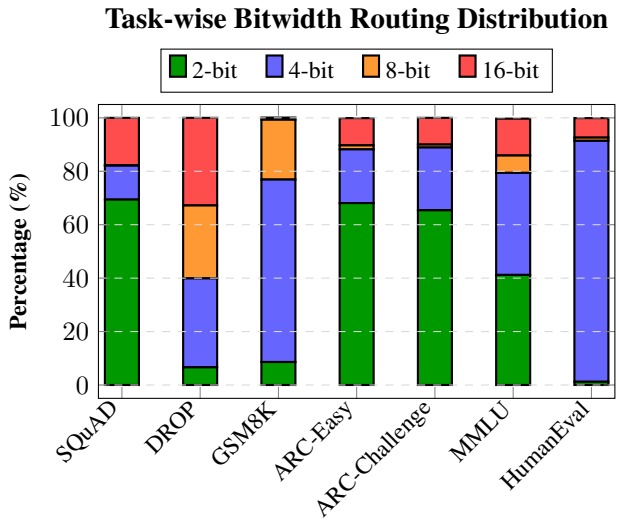

Figure 2: Adaptive bitwidth routing distribution across benchmark tasks for LLaMA-8B using a BERT-based router. Each stacked bar shows the percentage of samples routed to each bitwidth precision (2-bit, 4-bit, 8-bit, and 16-bit) for different evaluation tasks.

Accuracy Results

Table 1 reports per-dataset accuracy and average inference time. Overall, the BERT router achieves 0.707 average accuracy across LLaMA 3B, LLaMA 8B, and Qwen 8B, slightly surpassing the 16-bit baseline (0.705) as well as exceeding all 2-bit, 4-bit, and 8-bit models. This demonstrates that adaptive routing preserves model quality even under aggressive quantization.

QA and Multiple Choice Datasets: On SQuAD, the router achieves 0.970 average accuracy across all models, showing a slight decrease from the 16-bit average accuracy of 0.972. ARC-Easy and ARC-Challenge exhibit minor reductions or equal performance in routed accuracy compared to their highest-performing static configurations (ARC-Easy: 0.412 vs. 0.412; ARC-Challenge: 0.405 vs. 0.416). This indicates that while low-bit routing accelerates inference, higher-bit models are selectively used for more difficult prompts, yet careful tuning may still further improve router accuracy.

Dataset	LLaMA 3B						LLaMA 8B						Qwen 8B					
	2b	4b	8b	16b	BERT	Rand	2b	4b	8b	16b	BERT	Rand	2b	4b	8b	16b	BERT	Rand
SQuAD	0.941	0.946	0.952	0.957	0.950	0.949	0.967	0.974	0.979	0.975	0.973	0.978	0.980	0.985	0.983	0.985	0.986	0.986
DROP	0.543	0.647	0.680	0.674	0.662	0.632	0.677	0.817	0.822	0.821	0.809	0.783	0.762	0.814	0.803	0.793	0.815	0.793
GSM8K	0.243	0.546	0.596	0.605	0.578	0.487	0.402	0.605	0.626	0.672	0.623	0.574	0.766	0.816	0.822	0.813	0.823	0.810
ARC-Easy	0.374	0.409	0.405	0.412	0.412	0.411	0.454	0.489	0.474	0.492	0.484	0.483	0.456	0.460	0.432	0.426	0.457	0.442
ARC-Chall.	0.341	0.390	0.403	0.416	0.405	0.372	0.438	0.529	0.515	0.512	0.529	0.517	0.508	0.510	0.468	0.484	0.505	0.503
MMLU	0.388	0.510	0.514	0.517	0.510	0.485	0.538	0.572	0.576	0.562	0.580	0.558	0.561	0.665	0.671	0.667	0.667	0.644
HumanEval	0.055	0.116	0.207	0.201	0.165	0.116	0.091	0.207	0.250	0.250	0.232	0.171	0.104	0.091	0.067	0.079	0.098	0.073
Overall Acc.	0.548	0.633	0.648	0.651	0.643	0.619	0.629	0.717	0.718	0.723	0.724	0.696	0.710	0.754	0.746	0.743	0.755	0.741
Avg Time (s)	3.9	4.3	7.9	23.6	8.0	9.8	4.1	4.5	8.6	24.6	8.4	10.3	4.2	4.4	9.8	25.2	8.6	10.8

Table 1: Condensed comparison of accuracy and average inference time across fixed quantization levels (2–16 bit) and adaptive routing strategies (ModernBERT Router vs Random Router) for **LLaMA 3B**, **LLaMA 8B**, and **Qwen 8B**. ModernBERT Router routes based on predicted prompt complexity, while Random Router uses uniform random selection.

Reasoning Datasets: GSM8K, MMLU, DROP and HumanEval require higher precision for challenging prompts. DROP, in particular, is demanding due to longer context and reasoning-intensive questions. The router routes a significant portion of these prompts to 8-bit and 16-bit models, improving accuracy over random routing (e.g., DROP: 0.762 vs. 0.736; MMLU: 0.586 vs. 0.572), demonstrating that the router captures task- and prompt-specific needs.

Efficiency Analysis

Routing achieves a mean latency of 8.3 s per prompt, representing a 65.9% reduction relative to the 16-bit baseline (24.5 s). Lower-bit models alone have an average latency of 4.3 s (2-bit: 4.1 s, 4-bit: 4.4 s), meaning PAQ increases latency by 4 s on average compared to low bit-width models but compensates with improved accuracy. Speedups relative to 16-bit vary by dataset, ranging from 1.5× (DROP) to 6.4× (SQuAD), and overhead by router is negligible (< 0.02 s).

Perplexity and Routing Behavior

Perplexity analysis reveals a strong link between prompt complexity and bitwidth selection. Simple QA and Multiple Choice tasks (SQuAD, ARC-Easy) show lower perplexity and are routed mainly to 2- or 4-bit models, while reasoning-intensive prompts (GSM8K, DROP) have higher perplexity and are more frequently routed to 8- or 16-bit models.

Routing distributions (Figure 2) further illustrate this trend: factual or surface-level tasks use low-bit models 70–90% of the time, whereas complex reasoning tasks trigger high-bit routing 20-60% of the time. Overall, across 6,164 test prompts, 41.7% are routed to 2-bit, 30.0% to 4-bit, 10.2% to 8-bit, and 18.0% to 16-bit inference. This adaptive allocation reduces average latency to 8.3 s (from 24.5 s for 16-bit) while preserving baseline accuracy, demonstrating that the router effectively internalizes task-level complexity to optimize precision.

Discussion

PAQ demonstrates that per-prompt adaptive quantization improves efficiency while maintaining high accuracy across diverse tasks:

- **Efficiency:** Routing cuts latency to 8.3 s per prompt, 66% faster than 16-bit inference with minimal overhead.
- **Routing behavior:** Most prompts use low-bit models (41.7% 2-bit, 30.0% 4-bit), while high-bit models handle only complex inputs (10.2% 8-bit, 18.0% 16-bit), showing effective precision allocation.
- **Accuracy trade-offs:** Performance remains strong (0.643 LLaMA 3B, 0.724 LLaMA 8B, 0.755 Qwen 8B). QA tasks stay near baseline, and reasoning-intensive tasks show minor variation, reflecting a balanced efficiency–accuracy trade-off.

By using task- and prompt-specific complexity cues, PAQ achieves major speedups with minimal accuracy loss, showing the router's ability to adaptively allocate computation. Minor drops on reasoning-heavy prompts indicate room for further tuning, but overall PAQ maintains strong efficiency and performance. In large-scale systems handling high volumes of prompts, PAQ's adaptive per-prompt quantization could reduce computational costs and latency while maintaining accuracy, enabling more efficient deployment of large language models in real-time services or multi-user applications.

Conclusion

We introduced **PAQ**, a per-prompt adaptive quantization framework for LLMs. Leveraging a lightweight ModernBERT-based router to select among pre-quantized LLaMA-3.1 and Qwen 8B models, PAQ reduces average latency by 66% while maintaining near-baseline accuracy. It efficiently routes low-complexity prompts, such as SQuAD and ARC, to low-bit models and allocates higher precision for reasoning-intensive prompts like GSM8K and DROP, demonstrating the router's ability to match model precision to prompt complexity. Overall, PAQ provides a practical, interpretable approach to input-aware, efficient LLM inference, opening avenues for adaptive large model deployment.

References

Bartowski. 2024. Llama-3.1-8B-Instruct-GGUF. https://huggingface.co/bartowski/Llama-3.1-8B-Instruct-GGUF.

Chen, M.; Tworek, J.; et al. 2021. Evaluating Large Language Models Trained on Code. arXiv:2107.03374.

Cheng, W.; Zhang, X.; Wu, S.; Zhang, Z.; Li, Y.; Liang, B.; Hu, Y.; and Zhao, X. 2024. QAQ: Quality Adaptive Quantization for LLM KV Cache. *arXiv preprint arXiv:2403.04643*.

Choi; Youngduck; Lee; Youngnam; Cho; Junghyun; Baek; Jineon; Park; Byungsoo; Lee; Seonghoon; Heo; Gunhee; Shin; Yooju; Kang; and Seewoo. 2020. Perplexity-based Data Pruning & Deep Knowledge Tracing. *arXiv preprint arXiv:2002.00925*.

Cobbe; Karl; Kosaraju; Vineet; Bavarian; Mohammad; Chen; Mark; Jun; Heewoo; Kaiser; Lukasz; Plappert; Matthias; Tworek; Jerry; Hilton; Jacob; Nakano; Reiichiro; et al. 2021a. Training Verifiers to Solve Math Word Problems. In *Advances in Neural Information Processing Systems*.

Cobbe; Karl; Kosaraju; Vineet; Bavarian; Mohammad; Chen; Mark; Jun; Heewoo; Kaplan; Jared; Aschenbrenner; Prafulla; Wainwright; John; Zhang; Rewon; Sankaran; Kirthevasan; et al. 2021b. Training Verifiers to Solve Math Word Problems. In *Advances in Neural Information Processing Systems*, 7965–7977.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; et al. 2021c. Training Verifiers to Solve Math Word Problems. In *NeurIPS*.

Dua; Dheeru; Wang; Yizhong; Dasigi; Pradeep; Stanovsky; Gabriel; Singh; Sameer; Gardner; and Matt. 2019a. DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning Over Paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2368–2378.

Dua, D.; Wang, Y.; Dasigi, P.; Stanovsky, G.; Singh, S.; and Gardner, M. 2019b. DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning over Paragraphs. In *NAACL*.

Frantar; Elias; Ashkboos; Saleh; Hoefler; Torsten; Alistarh; and Dan. 2022a. GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers. *arXiv preprint arXiv:2210.17323*.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022b. GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers. arXiv:2210.17323.

Gonen, H.; Iyer, S.; Blevins, T.; Smith, N. A.; and Zettlemoyer, L. 2024. Demystifying Prompts in Language Models via Perplexity Estimation. arXiv:2212.04037.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; Yang, A.; Fan, A.; Goyal, A.; Hartshorn, A.; Yang, A.; Mitra, A.; Sravankumar, A.; Korenev, A.; Hinsvark, A.; Rao, A.; Zhang, A.; Rodriguez, A.; Gregerson, A.; Spataru, A.; Roziere, B.; Biron, B.; Tang, B.; Chern, B.; Caucheteux, C.; Nayak, C.; Bi, C.; Marra, C.; McConnell, C.; Keller, C.; Touret, C.; Wu, C.; Wong, C.; Ferrer, C. C.; Nikolaidis, C.; Allonsius, D.; Song, D.; Pintz, D.; Livshits, D.; Wyatt, D.; Esiobu, D.; Choudhary, D.; Mahajan, D.; Garcia-Olano, D.; Perino, D.; Hupkes, D.; Lakomkin, E.; AlBadawy, E.; Lobanova, E.; Dinan, E.; Smith, E. M.; Radenovic, F.; Guzmán, F.; Zhang, F.; Synnaeve, G.; Lee, G.; Anderson, G. L.; Thattai, G.; Nail, G.; Mialon, G.; Pang, G.; Cucurell, G.; Nguyen, H.; Korevaar, H.; Xu, H.; Touvron, H.; Zarov, I.; Ibarra, I. A.; Kloumann, I.; Misra, I.; Evtimov, I.; Zhang, J.; Copet, J.; Lee, J.; Geffert, J.; Vranes, J.; Park, J.; Mahadeokar, J.; Shah, J.; van der Linde, J.; Billock, J.; Hong, J.; Lee, J.; Fu, J.; Chi, J.; Huang, J.; Liu, J.; Wang, J.; Yu, J.; Bitton, J.; Spisak, J.; Park, J.; Rocca, J.; Johnstun, J.; Saxe, J.; Jia, J.; Alwala, K. V.; Prasad, K.; Upasani, K.; Plawiak, K.; Li, K.; Heafield, K.; Stone, K.; El-Arini, K.; Iyer, K.; Malik, K.; Chiu, K.; Bhalla, K.; Lakhotia, K.; Rantala-Yeary, L.; van der Maaten, L.; Chen, L.; Tan, L.; Jenkins, L.; Martin, L.; Madaan, L.; Malo, L.; Blecher, L.; Landzaat, L.; de Oliveira, L.; Muzzi, M.; Pasupuleti, M.; Singh, M.; Paluri, M.; Kardas, M.; Tsimpoukelli, M.; Oldham, M.; Rita, M.; Pavlova, M.; Kambadur, M.; Lewis, M.; Si, M.; Singh, M. K.; Hassan, M.; Goyal, N.; Torabi, N.; Bashlykov, N.; Bogoychev, N.; Chatterji, N.; Zhang, N.; Duchenne, O.; Çelebi, O.; Alrassy, P.; Zhang, P.; Li, P.; Vasic, P.; Weng, P.; Bhargava, P.; Dubal, P.; Krishnan, P.; Koura, P. S.; Xu, P.; He, Q.; Dong, Q.; Srinivasan, R.; Ganapathy, R.; Calderer, R.; Cabral, R. S.; Stojnic, R.; Raileanu, R.; Maheswari, R.; Girdhar, R.; Patel, R.; Sauvestre, R.; Polidoro, R.; Sumbaly, R.; Taylor, R.; Silva, R.; Hou, R.; Wang, R.; Hosseini, S.; Chennabasappa, S.; Singh, S.; Bell, S.; Kim, S. S.; Edunov, S.; Nie, S.; Narang, S.; Raparthy, S.; Shen, S.; Wan, S.; Bhosale, S.; Zhang, S.; Vandenhende, S.; Batra, S.; Whitman, S.; Sootla, S.; Collot, S.; Gururangan, S.; Borodinsky, S.; Herman, T.; Fowler, T.; Sheasha, T.; Georgiou, T.; Scialom, T.; Speckbacher, T.; Mihaylov, T.; Xiao, T.; Karn, U.; Goswami, V.; Gupta, V.; Ramanathan, V.; Kerkez, V.; Gonguet, V.; Do, V.; Vogeti, V.; Albiero, V.; Petrovic, V.; Chu, W.; Xiong, W.; Fu, W.; Meers, W.; Martinet, X.; Wang, X.; Wang, X.; Tan, X. E.; Xia, X.; Xie, X.; Jia, X.; Wang, X.; Goldschlag, Y.; Gaur, Y.; Babaei, Y.; Wen, Y.; Song, Y.; Zhang, Y.; Li, Y.; Mao, Y.; Coudert, Z. D.; Yan, Z.; Chen, Z.; Papakipos, Z.; Singh, A.; Srivastava, A.; Jain, A.; Kelsey, A.; Shajnfeld, A.; Gangidi, A.; Victoria, A.; Goldstand, A.; Menon, A.; Sharma, A.; Boesenberg, A.; Baevski, A.; Feinstein, A.; Kallet, A.; Sangani, A.; Teo, A.; Yunus, A.; Lupu, A.; Alvarado, A.; Caples, A.; Gu, A.; Ho, A.; Poulton, A.; Ryan, A.; Ramchandani, A.; Dong, A.; Franco, A.; Goyal, A.; Saraf, A.; Chowdhury, A.; Gabriel, A.; Bharambe, A.; Eisenman, A.; Yazdan, A.; James, B.; Maurer, B.; Leonhardi, B.; Huang, B.; Loyd, B.; Paola, B. D.; Paranjape, B.; Liu, B.; Wu, B.; Ni, B.; Hancock, B.; Wasti, B.; Spence, B.; Stojkovic, B.; Gamido, B.; Montalvo, B.; Parker, C.; Burton, C.; Mejia, C.; Liu, C.; Wang, C.; Kim, C.; Zhou, C.; Hu, C.; Chu, C.-H.; Cai, C.; Tindal, C.; Feichtenhofer, C.; Gao, C.; Civin, D.; Beaty, D.; Kreymer, D.; Li, D.; Adkins, D.; Xu, D.; Testuggine, D.; David, D.; Parikh, D.; Liskovich, D.; Foss, D.; Wang, D.; Le, D.; Holland, D.; Dowling, E.; Jamil, E.; Montgomery,

E.; Presani, E.; Hahn, E.; Wood, E.; Le, E.-T.; Brinkman, E.; Arcaute, E.; Dunbar, E.; Smothers, E.; Sun, F.; Kreuk, F.; Tian, F.; Kokkinos, F.; Ozgenel, F.; Caggioni, F.; Kanayet, F.; Seide, F.; Florez, G. M.; Schwarz, G.; Badeer, G.; Swee, G.; Halpern, G.; Herman, G.; Sizov, G.; (Jack)Zhang, G.; Lakshminarayanan, G.; Inan, H.; Shojanazeri, H.; Zou, H.; Wang, H.; Zha, H.; Habeeb, H.; Rudolph, H.; Suk, H.; Aspegren, H.; Goldman, H.; Zhan, H.; Damlaj, I.; Molybog, I.; Tufanov, I.; Leontiadis, I.; Veliche, I.-E.; Gat, I.; Weissman, J.; Geboski, J.; Kohli, J.; Lam, J.; Asher, J.; Gaya, J.-B.; Marcus, J.; Tang, J.; Chan, J.; Zhen, J.; Reizenstein, J.; Teboul, J.; Zhong, J.; Jin, J.; Yang, J.; Cummings, J.; Carvill, J.; Shepard, J.; McPhie, J.; Torres, J.; Ginsburg, J.; Wang, J.; Wu, K.; U, K. H.; Saxena, K.; Khandelwal, K.; Zand, K.; Matosich, K.; Veeraraghavan, K.; Michelena, K.; Li, K.; Jagadeesh, K.; Huang, K.; Chawla, K.; Huang, K.; Chen, L.; Garg, L.; A, L.; Silva, L.; Bell, L.; Zhang, L.; Guo, L.; Yu, L.; Moshkovich, L.; Wehrstedt, L.; Khabsa, M.; Avalani, M.; Bhatt, M.; Mankus, M.; Hasson, M.; Lennie, M.; Reso, M.; Groshev, M.; Naumov, M.; Lathi, M.; Keneally, M.; Liu, M.; Seltzer, M. L.; Valko, M.; Restrepo, M.; Patel, M.; Vyatskov, M.; Samvelyan, M.; Clark, M.; Macey, M.; Wang, M.; Hermoso, M. J.; Metanat, M.; Rastegari, M.; Bansal, M.; Santhanam, N.; Parks, N.; White, N.; Bawa, N.; Singhal, N.; Egebo, N.; Usunier, N.; Mehta, N.; Laptev, N. P.; Dong, N.; Cheng, N.; Chernoguz, O.; Hart, O.; Salpekar, O.; Kalinli, O.; Kent, P.; Parekh, P.; Saab, P.; Balaji, P.; Rittner, P.; Bontrager, P.; Roux, P.; Dollar, P.; Zvyagina, P.; Ratanchandani, P.; Yuvraj, P.; Liang, Q.; Alao, R.; Rodriguez, R.; Ayub, R.; Murthy, R.; Nayani, R.; Mitra, R.; Parthasarathy, R.; Li, R.; Hogan, R.; Battey, R.; Wang, R.; Howes, R.; Rinott, R.; Mehta, S.; Siby, S.; Bondu, S. J.; Datta, S.; Chugh, S.; Hunt, S.; Dhillon, S.; Sidorov, S.; Pan, S.; Mahajan, S.; Verma, S.; Yamamoto, S.; Ramaswamy, S.; Lindsay, S.; Lindsay, S.; Feng, S.; Lin, S.; Zha, S. C.; Patil, S.; Shankar, S.; Zhang, S.; Zhang, S.; Wang, S.; Agarwal, S.; Sajuyigbe, S.; Chintala, S.; Max, S.; Chen, S.; Kehoe, S.; Satterfield, S.; Govindaprasad, S.; Gupta, S.; Deng, S.; Cho, S.; Virk, S.; Subramanian, S.; Choudhury, S.; Goldman, S.; Remez, T.; Glaser, T.; Best, T.; Koehler, T.; Robinson, T.; Li, T.; Zhang, T.; Matthews, T.; Chou, T.; Shaked, T.; Vontimitta, V.; Ajayi, V.; Montanez, V.; Mohan, V.; Kumar, V. S.; Mangla, V.; Ionescu, V.; Poenaru, V.; Mihailescu, V. T.; Ivanov, V.; Li, W.; Wang, W.; Jiang, W.; Bouaziz, W.; Constable, W.; Tang, X.; Wu, X.; Wang, X.; Wu, X.; Gao, X.; Kleinman, Y.; Chen, Y.; Hu, Y.; Jia, Y.; Qi, Y.; Li, Y.; Zhang, Y.; Zhang, Y.; Adi, Y.; Nam, Y.; (Sid)Wang, Y.; Zhao, Y.; Hao, Y.; Qian, Y.; Li, Y.; He, Y.; Rait, Z.; DeVito, Z.; Rosnbrick, Z.; Wen, Z.; Yang, Z.; Zhao, Z.; and Ma, Z. 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*.

Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *ICLR*.

Jurafsky; Daniel; Martin; and H, J. 2023. *Speech and Language Processing*. Pearson, 3rd edition. Draft available online at https://web.stanford.edu/~jurafsky/slp3/.

Lee, C.; Jin, J.; Kim, T.; Kim, H.; and Park, E. 2024. OWQ: Outlier-Aware Weight Quantization for Efficient Fine-Tuning and Inference of Large Language Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Leviathan, Y.; Kalman, M.; and Matias, Y. 2023. Fast Inference from Transformers via Speculative Decoding. In *Proceedings of ICML*.

Lin; Ji; Tang; Jiaming; Tang; Haotian; Yang; Shang; Dang; Xingyu; Han; and Song. 2023a. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. *arXiv preprint arXiv:2306.00978*.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Dang, X.; and Han, S. 2023b. Activation-Aware Weight Quantization for LLM Compression and Acceleration. arXiv:2306.00978.

Loshchilov; Ilya; Hutter; and Frank. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Lu, Y.; Elhoushi, M.; Xiong, W.; Wahab, A.; Cao, S.; Cheng, H.; Heafield, K.; and Liu, Z. 2024. Systematic Characterization of LLM Quantization: A Performance, Energy, and Quality Perspective. arXiv:2508.16712.

Minaee, S.; Mikolov, T.; Nikzad, N.; Chenaghlu, M.; Socher, R.; Amatriain, X.; and Gao, J. 2024. Large Language Models: A Survey. arXiv:2402.06196.

Peter, C.; Isaac, C.; Oren, E.; Tushar, K.; Ashish, S.; Carissa, S.; and Oyvind, T. 2018. Think you have solved question answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint arXiv:1803.05457*.

Ping, B.; Wang, S.; Wang, H.; Han, X.; Xu, Y.; Yan, Y.; Chen, Y.; Chang, B.; Liu, Z.; and Sun, M. 2024. DeltaCoMe: Training-Free Delta-Compression with Mixed-Precision for Large Language Models. In *NeurIPS*.

Rajpurkar; Pranav; Zhang; Jian; Lopyrev; Konstantin; Liang; and Percy. 2016a. SQuAD: 100 and000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016b. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*.

Roy; Aurko; Saffar; Mohammad; Vaswani; Ashish; Grangier; and David. 2021. Routing Transformers. In *Advances in Neural Information Processing Systems*, 9287–9297.

Schuster, T.; Fisch, A.; Jaakkola, T.; and Barzilay, R. 2021. Consistent Accelerated Inference via Confident Adaptive Transformers. In *EMNLP*.

Schuster, T.; Fisch, A.; Jaakkola, T.; and Barzilay, R. 2023. Skip & Seek: Simple Acceleration Methods for Transformer Inference. In *Proceedings of ACL*.

Shazeer; Noam; Mirhoseini; Azalia; Maziarz; Krzysztof; Davis; Andy; Le; V, Q.; Hinton; Geoffrey; Dean; and Jeff. 2017a. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations*.

Shazeer, N.; et al. 2017b. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *ICLR*.

Warner, B.; Chaffin, A.; Clavié, B.; Weller, O.; Hallström, O.; Taghadouini, S.; Gallagher, A.; Biswas, R.; Ladhak, F.; Aarsen, T.; Cooper, N.; Adams, G.; Howard, J.; and Poli, I. 2024. Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference. arXiv:2412.13663.

Xia; Feng; Liu; Tie-Yan; Wang; Jue; Zhang; Wensheng; Li; and Hang. 2008. Cascade Ranking for Efficient Learning to Rank. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 75–82.

Xiao, G.; Lin, J.; Seznec, M.; Wu, H.; Demouth, J.; and Han, S. 2023. SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models. In *International Conference on Machine Learning*.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; Zheng, C.; Liu, D.; Zhou, F.; Huang, F.; Hu, F.; Ge, H.; Wei, H.; Lin, H.; Tang, J.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Zhou, J.; Lin, J.; Dang, K.; Bao, K.; Yang, K.; Yu, L.; Deng, L.; Li, M.; Xue, M.; Li, M.; Zhang, P.; Wang, P.; Zhu, Q.; Men, R.; Gao, R.; Liu, S.; Luo, S.; Li, T.; Tang, T.; Yin, W.; Ren, X.; Wang, X.; Zhang, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Wang, Z.; Cui, Z.; Zhang, Z.; Zhou, Z.; and Qiu, Z. 2025. Qwen3 Technical Report. arXiv:2505.09388.

Zhou, Z.; Ning, X.; Hong, K.; Fu, T.; Xu, J.; Zhou, S.; Tu, Z.; Chen, G.; Yang, H.; Wang, Y.; et al. 2024. Efficient Inference for Large Language Models: A Survey. arXiv:2404.14294.