Towards Automatically Optimizing Retrieval Augmented AI Systems

Melissa Z. Pan[1] **Negar Arabzadeh**[1] **Mathew Jacob**[2] **Fiodar Kazhamiaka**[3]
Esha Choukse[3] **Matei Zaharia**[1]
[1] UC Berkeley, [2] University of Washington, [3] Microsoft Azure Research - Systems

Abstract

Large Language Models (LLMs) are increasingly deployed in real-world systems, with Retrieval-Augmented Generation (RAG) as a dominant production workload. Yet LLM deployments are energy-intensive, as inference accounts for over 90% of the model lifecycle in cloud workloads. We show that RAG workflows with near-identical accuracy can differ drastically in energy consumption—a property we call **workflow fungibility**. For example, pairing Llama3-8B with stronger retrievers matches the accuracy of Llama3-70B while using over $5\times$ less energy. To study this effect, we profile retrieval and generation configurations across FinanceBench and FRAMES, mapping the joint accuracy–energy landscape. Our results reveal configurations within $\leq 3\%$ accuracy that differ by up to $20.2\times$ in energy, exposing large hidden opportunities for efficiency. We further demonstrate that lightweight regressors can predict accuracy from a small set of configuration knobs, enabling prediction-guided pruning of the design space. These findings establish workflow fungibility as a key lever for sustainable RAG, and point toward systematic, energy-aware configuration as a critical direction for retrieval-based LLM systems.

1 Introduction

The shift to using Large Language Models (LLMs) as a modular component in real-world systems marks a new deployment paradigm [1–5]. Retrieval Augmented AI Systems, which we broadly refer to as RAG, is a representative and widely-deployed LLM system that powers enterprise search, customer support, and knowledge-intensive applications across industry [6–10]. Yet, this rapidly growing deployment faces a grand challenge: unsustainable energy consumption. With LLM inference constituting over 90% of the model lifecycle in major cloud workloads [11], test-time scaling with LLMs creates immense energy demands that threaten both economic viability and environmental sustainability.

To reduce AI energy consumption, we exploit RAG's modular architecture. RAG systems do not prescribe a specific pipeline, but are rather a method for solving knowledge-intensive tasks with many algorithmic choices for components such as retrievers, rerankers, generation model, query reformulation, and so on. As we show in Section 2.2, different configurations of these components can deliver nearly identical accuracy while consuming vastly different amounts of energy. For instance, the right choice of retriever can enable a system with Llama3-8B to match the accuracy of Llama3-70B while using 5.42× less average energy per query (energy breakdown shown in Appendix C). This insight, which we term **workflow fungibility**, demonstrates that careful RAG configuration can unlock massive energy savings without degrading task quality.

As a first step toward exploiting *workflow fungibility* in RAG, we study the atomic units of the pipeline—retrieval and generation—and empirically map their joint accuracy–energy landscape across representative configurations. Varying the retriever, retrieval depth k, and the generator LLM reveals many configurations with near-identical accuracy but multi-$\times$ differences in per-query energy. This reframes the objective from finding a single "best" pipeline to *navigating a set of near-optimal*

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: .

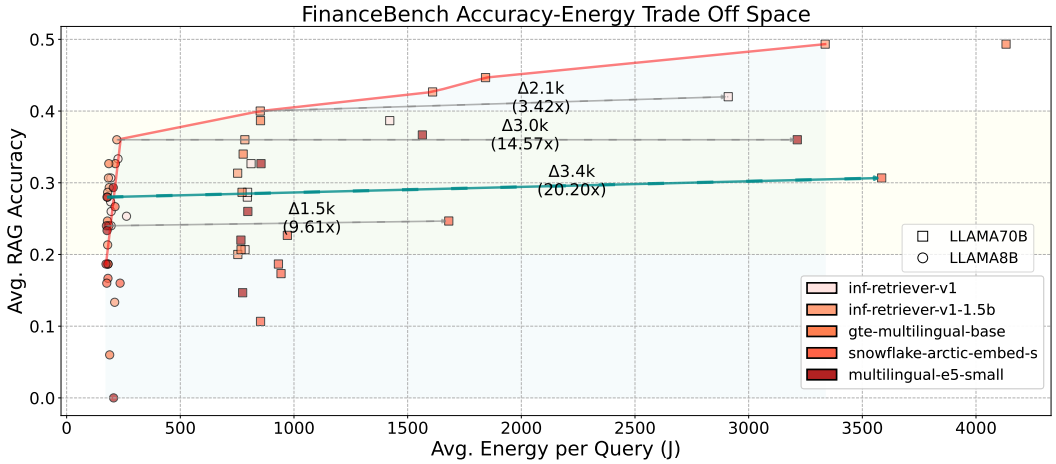

Figure 1: Accuracy–energy tradeoffs for FinanceBench. Each point represents an unique RAG configuration (retriever, retrieval depth, generator), with circles denoting Llama3-8B and squares denoting Llama3-70B as generators. Color encodes embedding model choice for retrieval. Same color and shape represent varying k (depth). The x-axis reports per-query energy (J), while the y-axis reports average RAG accuracy reported by LLM-as-a-Judge with Llama3-70b. The red curve traces the Pareto frontier. We highlight four configuration pairs that differ by less than 3% accuracy but by 3.4-20.2x difference in per-query energy, showing the optimization opportunities hidden in RAG.

pipelines under energy constraints. On FinanceBench and FRAMES, we find configurations within $\leq 3\%$ accuracy that differ by up to $20.2\times$ in energy.

However, systematically exploiting workflow fungibility for accuracy–energy optimization is challenging due to an exponentially large search space. To make such navigation practical, we train lightweight models that *predict downstream answer accuracy* from a few configuration knobs (retriever, k, LLM). Coupled with measured energy, these predictions allow practitioners to shortlist energy-efficient candidates near the empirical Pareto frontier without exhaustive profiling. As proof of concept that we can use ML solution towards the challenge of wide configuration space, we train a random forest model that effectively predict optimal configurations with RMSE under 0.05. This result demonstrates an opportunity to systematically optimize RAG for accuracy-energy trade-off.

Recent systems improve RAG efficiency but do not navigate the full configuration space or treat energy as a first-class objective. Hermes [12] accelerates retrieval but leaves end-to-end configuration open. METIS [13] selects from a small set of options (e.g., synthesis method, number of chunks, intermediate length) via a simple rule-based mapping from an LLM-generated query profile. Execution frameworks like Murakkab [14] and HedraRAG [15] schedule predefined workflows and assume extensive profiling is available. In short, existing approaches rely on profiling narrow configuration families and do not scale to the combinatorial space. We instead extend the search with ML-based accuracy prediction and pair it with component-level energy profiles, enabling prediction-guided pruning toward energy-aware configuration.

Our contributions include:

1. **Workflow fungibility:** RAG configurations can achieve similar accuracy with up to $20.2\times$ differences in energy.
2. **Accuracy–energy profiling:** We map the joint landscape of retrievers, depths, and generators on FinanceBench and FRAMES.
3. **Prediction-guided search:** Lightweight regressors accurately predict accuracy from configuration knobs, enabling efficient pruning.
4. **Sustainable RAG:** Electro is the first step toward end-to-end energy-aware optimization in compound LLM systems.

2 Workflow Fungibility: Design Space Exploration

To systematically optimize energy efficiency in RAG, we must first understand how different configurations behave across their vast design space. This section characterizes the RAG pipeline and

establishes our key insight: multiple workflows can achieve near-identical accuracy with drastically different energy consumption. Through systematic profiling across benchmarks, we show that **workflow fungibility**—the interchangeability of components without loss of accuracy—offers orders-of-magnitude opportunities for energy savings.

2.1 The Design Space

In practice, retrieval-augmented AI systems for knowledge-intensive and Q&A tasks (which we refer to collectively as *RAG*) comprise many stages with no fixed scope of components, composed to fit application needs. Common techniques include query reformulation; dense, sparse, hybrid, or web-augmented retrieval; retrieval-depth control (*top-*k, dynamic k); reranking (cross-encoder or LLM-based); context construction (chunking, deduplication); context compression (summarization/filtering); and generation policies (model/routing, decoding, response caching).

As a first step toward this problem, we focus on two components that dominate the accuracy–energy trade space: Retrieval and Generation. Even for a minimal retriever→generator pipeline, combinatorial choices across these knobs yield hundreds of configurations with distinct accuracy and energy footprints, motivating a need for systematic exploration.

2.2 Discovering Workflow Fungibility

Setup. We profile RAG configurations on two knowledge-intensive Q&A benchmarks with increasing level of task difficulty: FinanceBench and FRAMES. Our design space spans five embedding models, retrieval depths ($k \in \{1, 3, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$), and two generators (Llama3-8B, Llama3-70B). We use a FAISS [16] HNSW index and drive the system with a Poisson load generator while monitoring GPU power via `nvidia-smi` to compute per-query energy (J).

Findings. Fig. 1 illustrates the core insight of **workflow fungibility**. Configurations that differ by less than 3% in accuracy often vary by orders of magnitude in energy, revealing large hidden system efficiency opportunities. Generator choice dominates this tradeoff: Llama3-70B configurations consistently incur much higher energy, while Llama3-8B paired with stronger retrievers can match its accuracy at a fraction of the cost. Retrieval depth further shapes the curve: larger k inflates energy without guaranteed accuracy gains, and in some cases even reduces performance. We provide the trade-off space result for FRAMES in Appx. A and component-level energy analysis in Appx. C.

These findings reinforce two insights. First, there is no single "best" RAG pipeline. The efficient choice depends on accuracy targets and energy constraints. Second, current compound AI systems miss massive energy reduction opportunities underscores the need for systematic, accuracy–energy aware exploration of the RAG design space.

3 Towards Prediction-Guided Navigation of the RAG Design Space

To systematically exploit *workflow fungibility* for RAG pipelines, exhaustive profiling is infeasible due to the exponentially growing configuration space. The challenge is then to find an energy-efficient configuration while maintaining the task quality without sweeping through the entire design space.

A promising direction towards this challenge is prediction-guided navigation. Energy consumption of a RAG configuration can be approximated by aggregating the profiled energy of its individual components, so the main difficulty lies in predicting downstream accuracy. Accuracy does not follow simple trends across knobs: as shown in Section 2, changes in retrieval depth or embedding model can have nonlinear effects. Thus, the central task for systematic optimization is to build models that estimate accuracy efficiently, and then cross-reference these predictions with component-level energy profiles to identify Pareto-efficient configurations.

Our key idea is to use lightweight models that predict downstream performance from configuration features. Our results demonstrate that even simple regressors can approximate downstream accuracy with high fidelity, establishing prediction-guided methods as a feasible first step toward scalable, energy-aware configuration of RAG pipelines.

3.1 Performance Predictor

Given the vast configuration space of compound AI systems, exhaustive exploration is prohibitively expensive. Even modest RAG pipelines combine multiple retrievers, retrieval depths, and large language models, resulting in thousands of potential configurations. Rather than exhaustively evaluating each combination directly, a promising approach is to predict system performance from a subset of configuration features, thereby narrowing the search space. As an initial step, we train lightweight regressors to predict downstream accuracy from three knobs: retriever type, number of retrieved documents (k), and choice of LLM. Our design space includes two LLMs, five retrievers, and 12 retrieval depths (full spec in Appendix B). We evaluate predictions using 10-fold cross-validation on FinanceBench and FRAMES.

To contextualize results, we compare against a random baseline that assigns performance values at random. Unsurprisingly, this baseline performs poorly on both FinanceBench and Frames.

Table 1: Accuracy prediction Performance.

Model	FinanceBench		Frames	
	RMSE	MAE	RMSE	MAE
Random	0.4405	0.3535	0.4874	0.4016
DecisionTree	0.0494	0.0323	0.0122	0.0098
KNN	0.0490	0.0318	**0.0098**	**0.0078**
Linear Regression	0.0527	0.0404	0.0116	0.0094
MLP	0.0500	0.0365	0.0115	0.0090
Random Forest	**0.0470**	**0.0303**	0.0101	0.0083
SVR	0.0721	0.0660	0.0507	0.0437

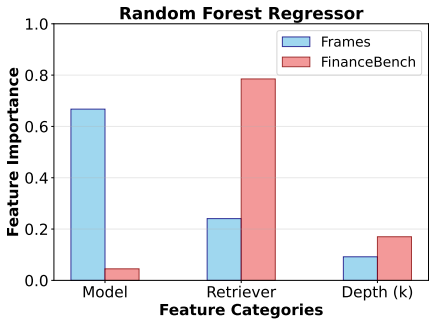

Figure 2: Feature importance comparison.

Across all models, we observe substantial improvements over random guessing shown in Table 1. Decision Trees, KNN, Random Forests, and MLP regressors consistently achieve strong performance, with Random Forests reaching the best predictive accuracy on both datasets. Even Linear Regression reduce error significantly, showing that much of the mapping from configuration to performance can be captured by simple linear trends.

Our modeling results also reveal insights beyond prediction accuracy. Feature importance analysis (Fig. 2) shows that different components matter for different tasks. For FinanceBench, retrieval type dominates, followed by number of retrieved items (k); in Frames, the choice of LLM emerges as the most critical predictor of accuracy, followed by retrieval type features, with k playing a smaller role. This divergence highlights that the most influential factors are task-dependent. Optimizers that treat all knobs equally risk wasting profiling effort on parameters with marginal impact. These findings establish predictive modeling as a practical first step toward scalable configuration optimization, reducing the cost of exploring large design spaces while preserving accuracy.

4 Discussion and Ongoing Work

With workflow fungibility established and prediction-guided navigation shown feasible, we are building toward a novel framework that unifies these two steps: systematic profiling to expose the accuracy–energy landscape, and lightweight prediction to efficiently traverse it. Together, these capabilities form the foundation for automatic configuration of RAG pipelines under energy and quality constraints.

This work represents the first step in a larger, ongoing project on efficient and sustainable Retrieval-Augmented AI systems. Moving forward, we are extending our approach with runtime adaptation that dynamically adjusts configurations under changing workloads, expanding profiling to broader domains and tasks, and developing new search algorithms for navigating massive design spaces. A natural next direction is to move beyond accuracy prediction to directly predict for optimal configuration for a given task. Together, these directions point toward a future where LLM-based retrieval pipelines can automatically self-configure for both efficiency and quality, making large-scale LLM deployment to be both effective and sustainable.

References

[1] Matei Zaharia, Omar Khattab, Lingjiao Chen, Jared Quincy Davis, Heather Miller, Chris Potts, James Zou, Michael Carbin, Jonathan Frankle, Naveen Rao, and Ali Ghodsi. The shift from models to compound ai systems. https://bair.berkeley.edu/blog/2024/02/18/compound-ai-systems/, 2024.

[2] Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. Dspy: Compiling declarative language model calls into self-improving pipelines, 2023. URL https://arxiv.org/abs/2310.03714.

[3] Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'arcy, David Wadden, Matt Latzke, Minyang Tian, Pan Ji, Shengyan Liu, Hao Tong, Bohao Wu, Yanyu Xiong, Luke Zettlemoyer, Graham Neubig, Dan Weld, Doug Downey, Wen tau Yih, Pang Wei Koh, and Hannaneh Hajishirzi. Openscholar: Synthesizing scientific literature with retrieval-augmented lms, 2024. URL https://arxiv.org/abs/2411.14199.

[4] Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R D Costa, José R Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an ai co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

[5] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. Chatdev: Communicative agents for software development, 2024. URL https://arxiv.org/abs/2307.07924.

[6] Rama Akkiraju, Anbang Xu, Deepak Bora, Tan Yu, Lu An, Vishal Seth, Aaditya Shukla, Pritam Gundecha, Hridhay Mehta, Ashwin Jha, Prithvi Raj, Abhinav Balasubramanian, Murali Maram, Guru Muthusamy, Shivakesh Reddy Annepally, Sidney Knowles, Min Du, Nick Burnett, Sean Javiya, Ashok Marannan, Mamta Kumari, Surbhi Jha, Ethan Dereszenski, Anupam Chakraborty, Subhash Ranjan, Amina Terfai, Anoop Surya, Tracey Mercer, Vinodh Kumar Thanigachalam, Tamar Bar, Sanjana Krishnan, Samy Kilaru, Jasmine Jaksic, Nave Algarici, Jacob Liberman, Joey Conway, Sonu Nayyar, and Justin Boitano. Facts about building retrieval augmented generation-based chatbots, 2024. URL https://arxiv.org/abs/2407.07858.

[7] Microsoft. Retrieval Augmented Generation (RAG) in Azure AI Search. https://learn.microsoft.com/en-us/azure/search/retrieval-augmented-generation-overview?tabs=docs, August 2025. Accessed: 2025-08-27.

[8] Akhil Gupta and Oliver Chiu. Production-Quality RAG Applications with Databricks. https://www.databricks.com/blog/production-quality-rag-applications-databricks, May 2024. Accessed: 2025-08-27.

[9] Sarah Packowski, Inge Halilovic, Jenifer Schlotfeldt, and Trish Smith. Optimizing and evaluating enterprise retrieval-augmented generation (rag): A content design perspective, 2024. URL https://arxiv.org/abs/2410.12812.

[10] Krutika Ingale and Saurabh Trivedi. RAG to Riches: Harnessing the power of Retrieval-Augmented Generation. https://medium.com/workday-engineering/rag-to-riches-harnessing-the-power-of-retrieval-augmented-generation-from-scratch-60d58e3bb19b, October 2024. Accessed: 2025-08-27.

[11] 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*, ASPLOS '24, page 207–222, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703867. doi: 10.1145/3620666.3651329. URL https://doi.org/10.1145/3620666.3651329.

[12] Michael Shen, Muhammad Umar, Kiwan Maeng, G. Edward Suh, and Udit Gupta. Hermes: Algorithm-system co-design for efficient retrieval-augmented generation at-scale. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, ISCA '25, page 958–973, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400712616. doi: 10.1145/3695053.3731076. URL https://doi.org/10.1145/3695053.3731076.

[13] Siddhant Ray, Rui Pan, Zhuohan Gu, Kuntai Du, Shaoting Feng, Ganesh Ananthanarayanan, Ravi Netravali, and Junchen Jiang. Metis: Fast quality-aware rag systems with configuration adaptation, 2025. URL https://arxiv.org/abs/2412.10543.

[14] Gohar Irfan Chaudhry, Esha Choukse, Haoran Qiu, Íñigo Goiri, Rodrigo Fonseca, Adam Belay, and Ricardo Bianchini. Murakkab: Resource-efficient agentic workflow orchestration in cloud platforms, 2025. URL https://arxiv.org/abs/2508.18298.

[15] Zhengding Hu, Vibha Murthy, Zaifeng Pan, Wanlu Li, Xiaoyi Fang, Yufei Ding, and Yuke Wang. Hedrarag: Coordinating llm generation and database retrieval in heterogeneous rag serving, 2025. URL https://arxiv.org/abs/2507.09138.

[16] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. 2024.

Appendix

A FRAMES Result

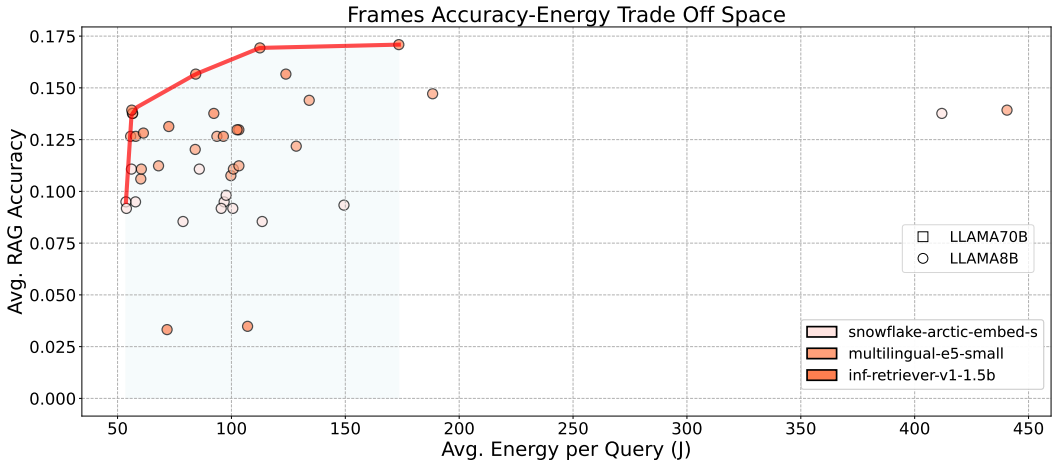

Figure 3: Accuracy–energy tradeoffs for FRAMES. Each point represents an unique RAG configuration (retriever, retrieval depth, generator), with circles denoting Llama3-8B and squares denoting Llama3-70B as generators. Color encodes embedding model choice for retrieval. Same color and shape represent varying k (depth). The x-axis reports per-query energy (J), while the y-axis reports average RAG accuracy reported by LLM-as-a-Judge with Llama3-70b. The red curve traces the Pareto frontier.

B Performance Predictor Setup

Table 2: Configuration design space modeled in Electro for Performance Predictor.

Component	**Options**
Generation Model	Llama-3.1-8B-Instruct, Llama-3.3-70B-Instruct
Retriever (Embedding Model)	bm25, hybrid, wiki-gte-multilingual, wiki-e5-small, wiki-snowflake-arctic-s, wiki-inf-retriever-v1-1.5b
Retrieval Depth (k)	{1, 3, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50}

C RAG Energy Component-Level Analysis

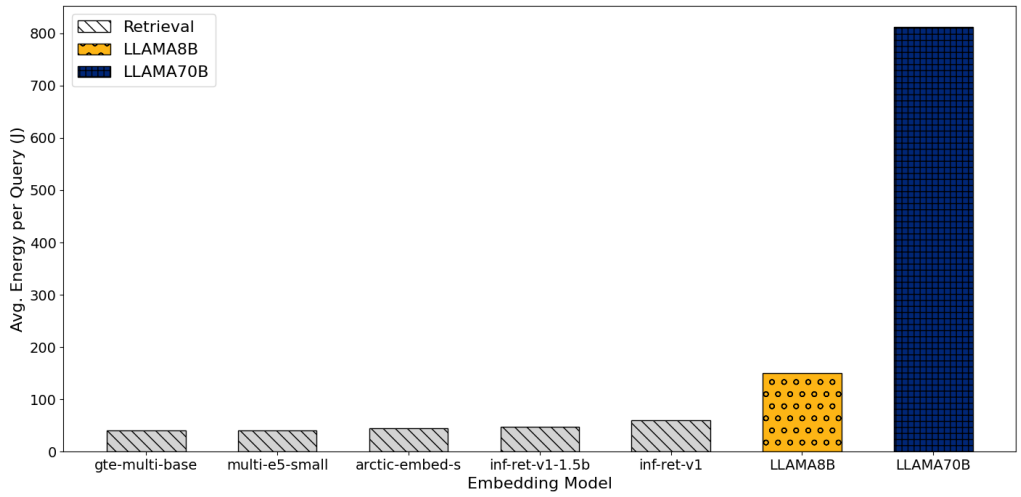

Figure 4: Average energy per query on FinanceBench breakdown by RAG component. The grey bars denote various embedding model for retrieval.

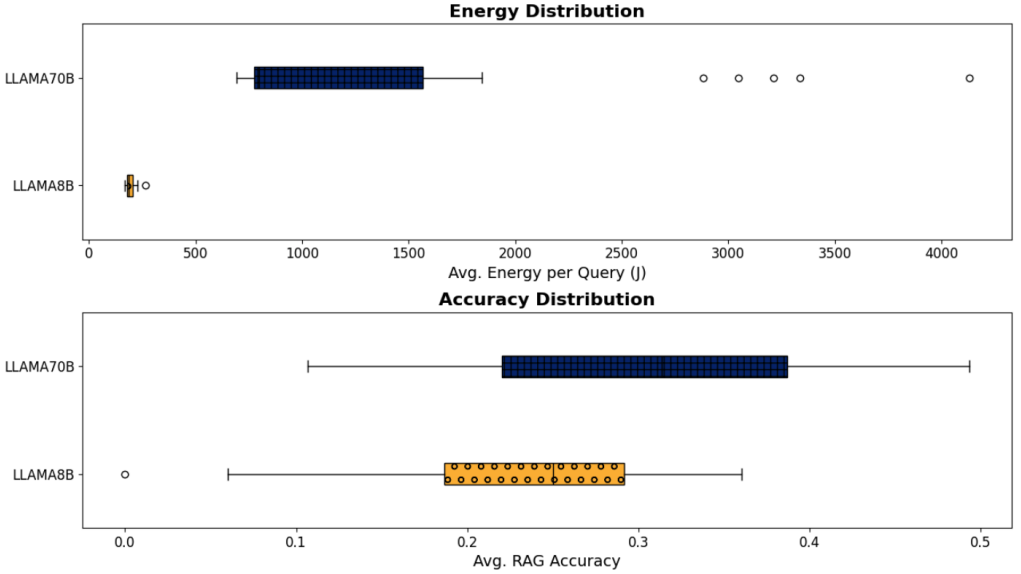

Figure 5: Performance and quality distribution on FinanceBench with breakdown by model. The generation model greatly affects both energy consumption and accuracy: larger models achieve higher accuracy but demand significantly more energy.