`LLM-Box` : An Agentic Framework for Guided Black-Box Optimization in Mapping LLMs onto Specialized Hardware Accelerators

Sujay Pandit[1]*, **Akanksha Jain**[3], **Rami Cohen**[3], **Zhijie Deng**[3], **Sagar Karandikar**[2,3]*,
Sagi Perel[4], **Anand Raghunathan**[1], **Parthasarathy Ranganathan**[3]

[1]Purdue University [2]UC Berkeley [3]Google [4]Google DeepMind
`{pandit8, raghunathan}@purdue.edu`
`{avjain, ramic, zjdeng, skarandikar, sagipe, parthas}@google.com`

Abstract

Identifying efficient execution strategies for Large Language Models (LLMs) on specialized hardware accelerators requires exploring a vast design space where exhaustive search is computationally prohibitive. Traditional black-box optimization (BBO) methods offer a principled alternative, but their efficiency degrades in high-dimensional, sparse spaces with many infeasible points. We propose `LLM-Box`, a framework that integrates an LLM agent to guide multi-objective BBO toward the Pareto frontier while significantly reducing sampling of infeasible points. By leveraging the LLM agent to retrieve and structure prior exploration data through retrieval-augmented generation (RAG), and by warm-starting and filtering BBO suggestions, our approach guides the search towards feasible and promising regions of the design space. As a result, `LLM-Box` identifies Pareto-optimal configurations with a hypervolume difference of less than 3% using $40-150\times$ fewer simulations than an exhaustive search, and compared to a well-known BBO tool, achieves 2% better accuracy with $20\times$ fewer trials. Moreover, the framework demonstrates zero-shot generalization, transferring knowledge from prior models and hardware to unseen targets.

1 Introduction

Mapping rapidly evolving Large Language Models (LLMs) [1, 2, 3, 4] onto specialized hardware accelerators involves navigating a vast combinatorial design space of execution strategies—encompassing parallelism choices, collective communication strategies, KV cache sharding strategies, and reconfigurable interconnect topologies [5, 6, 7, 8, 9]. Each point in the design space must often be validated through detailed simulation, making exhaustive exploration computationally prohibitive.

A common approach in prior works has been to employ black-box optimization (BBO) methods to accelerate exploration [7, 6, 10, 11, 12, 13]. These methods offer a principled way to sample points and improve over random or exhaustive search. However, the effectiveness of vanilla BBO is limited in high-dimensional, constrained design spaces that are typical in hardware/software co-design, where large regions are infeasible and domain-specific constraints dominate performance outcomes [14, 13]. To address this, prior research has often relied on manually encoding domain knowledge or heuristics into the optimizer [6, 11, 12, 13, 15, 16]. While effective for specific contexts, such

---

[*] Work done at Google.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

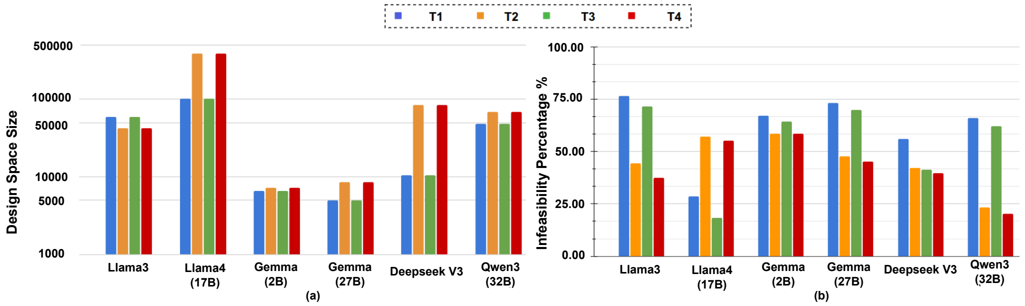

Figure 1: (a) Design space size for execution strategies of LLMs on Google internal TPU pods (T1, T2, T3 & T4) (b) Inherent infeasibility of the design space determined jointly by model-level and hardware-level constraints (e.g., model size, degree of parallelizability, number of chips, memory capacity and bandwidth)

manual interventions do not directly scale across different hardware platforms, workload classes, or evolving design objectives.

This paper explores a new direction: using LLM agents as a source of structured priors to guide this design space exploration (DSE). Unlike fixed/manually added heuristics, LLMs embed broad knowledge of LLM model architecture, accelerator architecture, algorithmic patterns, and can reason about parameter interactions at a higher level of abstraction[17, 18]. We propose that LLMs can assist BBO by (i) initializing the search with informed trials, (ii) filtering out infeasible or low-promise trials, and (iii) transferring insights across prior DSE studies.

We evaluate this idea by integrating the Gemini LLM with the Google Vizier BBO service [19] to optimize the mapping of modern LLM workloads onto Google's TPU pods. A TPU pod is a collection of TPU chips interconnected with reconfigurable high-speed links. To the best of our knowledge, this is the first effort to explore this design space using LLMs.
Our Contributions. We (i) introduce `LLM-Box`, an LLM-guided framework that complements conventional BBO; (ii) show improved sample efficiency—40–150× fewer simulations than exhaustive search and better accuracy than BBO baselines; and (iii) demonstrate robust transfer learning across models and hardware, enabling faster, more generalizable exploration.

2 Background

2.1 Exhaustive Design Space Exploration

Our baseline approach to this DSE utilizes an internal performance modeling simulator to conduct exhaustive simulations across the entire mapping design space, estimating performance metrics, namely, latency and queries per second (QPS). We use it to establish a ground-truth Pareto frontier representing the optimal trade-offs. While this exhaustive sweep provides a valuable baseline for analysis, it is time-consuming and computationally expensive.

2.2 Standard Black-Box Optimization

To reduce the number of required simulations, we create a second baseline by employing a BBO tool, Google Vizier [19], that internally uses a combination of Gaussian process bandits and genetic algorithms to sample the design space. In this setup, as shown in Figure 2(a), Vizier iteratively suggests configurations ("trials") to be evaluated by the performance modeling simulator and receives performance metrics feedback. At each step, Vizier updates a surrogate with the received feedback and optimizes a multi-objective acquisition to propose the next batch of trials.

3 LLM-guided Design Space Exploration

Our proposed framework, shown in Figure 2(b), integrates the LLM agent into the Vizier–Simulator loop. The LLM's role is not to replace the optimizer (Vizier), but to provide it with "context" and filter suggestions. It leverages three sources of knowledge: (1) intrinsic knowledge of hardware and

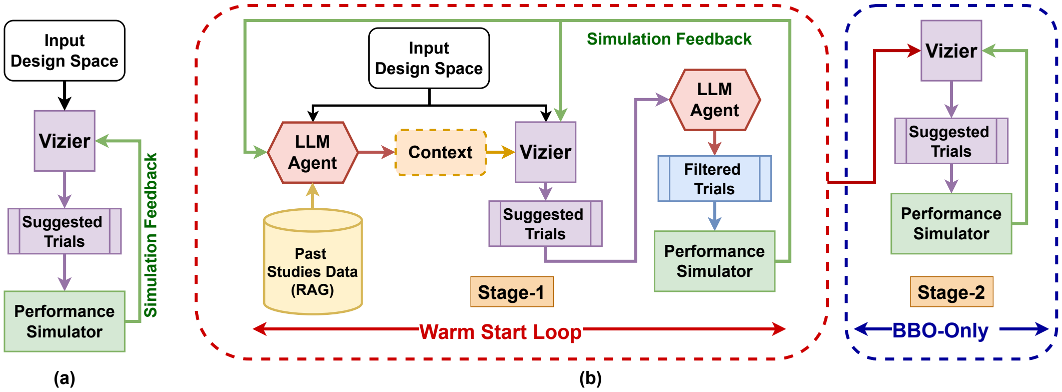

Figure 2: (a) Standard Vizier (BBO-only) framework. (b) The proposed `LLM-Box` framework. An LLM agent, supported by retrieval from past studies (RAG), provides context to warm-start the search and filters Vizier's trial suggestions before simulation. During Stage 1(warm-start), the LLM remains in the decision loop; In Stage 2, Vizier continues optimization alone using the enhanced posterior built from LLM-guided trials.

LLM models from its pre-training, (2) target system and model architecture information provided in prompts, and (3) historical data from past exploration studies using Retrieval-Augmented Generation (RAG). The core idea is to leverage the LLM's ability to utilize this domain knowledge to guide the search away from infeasible regions and towards promising ones. The process operates in two stages.

Stage 1 (Warm-start): In each iteration, the LLM, informed by the target system's specifications and prior studies (via RAG), emits a *context* object that constrains the search space [20]. Vizier proposes a batch of candidates from this constrained space; the LLM then applies a lightweight feasibility/quality filter using feedback from already-evaluated trials and retrieved exemplars. The accepted candidates are evaluated by the simulator, and the resulting metrics are fed back to both Vizier and the LLM to refine constraints and filtering in subsequent iterations.

Stage 2 (BBO-only): After the warm-start trials, the LLM is taken out of the decision loop. Vizier continues optimization from its posterior built on all data collected in Stage 1, using a standard multi-objective acquisition (e.g., hypervolume-improvement based) over the *final* hard constraints learned during warm-start. Trials are suggested directly by Vizier and evaluated by the simulator.

4 Evaluation

We evaluated our approach against the performance modeling simulator (ground truth) and Vizier baselines across a suite of modern LLMs (Qwen3, Llama3, Llama4, DeepseekV3, Gemma-2B and Gemma-27B) on four Google internal TPU pods. Each TPU pod contains multiple chips connected in a dynamically reconfigurable interconnect topology [5]. For any given model and hardware pair, the design space of possible execution strategies encompasses parameters like batch size, KV cache sharding strategies, collective operations, parallelism choices (data, model, pipeline, or expert), and interconnect topologies. We use Pareto-hypervolume (HPV) error relative to the ground truth to evaluate the quality of Pareto-frontiers obtained using Vizier and `LLM-Box` [21]. For the `LLM-Box` framework, we employed Gemini-2.5-Pro[1] as the reasoning model to guide Vizier, and Gemini-Embedding-001[22] to support retrieval-augmented generation (RAG) from prior exploration data. Although the empirical results naturally depend on the specific reasoning model and BBO algorithm chosen, the framework we present is agnostic to these choices, and the broader methodology extends across a wide range of hardware platforms and workload scenarios.

4.1 Transfer Learning

Across Systems We provided the LLM with the Pareto-frontier trials (obtained using ground truth simulation) for all models on one TPU pod (T1). We then tasked it with finding the Pareto curve for the same models on the other three target pods. Figure 3(a) shows that the LLM-guided search achieved significantly lower error with 100 trials than the Vizier-only baseline with 2000 trials. Crucially, as seen in Figure 3(b), the infeasibility rate was reduced by 20% compared to Vizier,

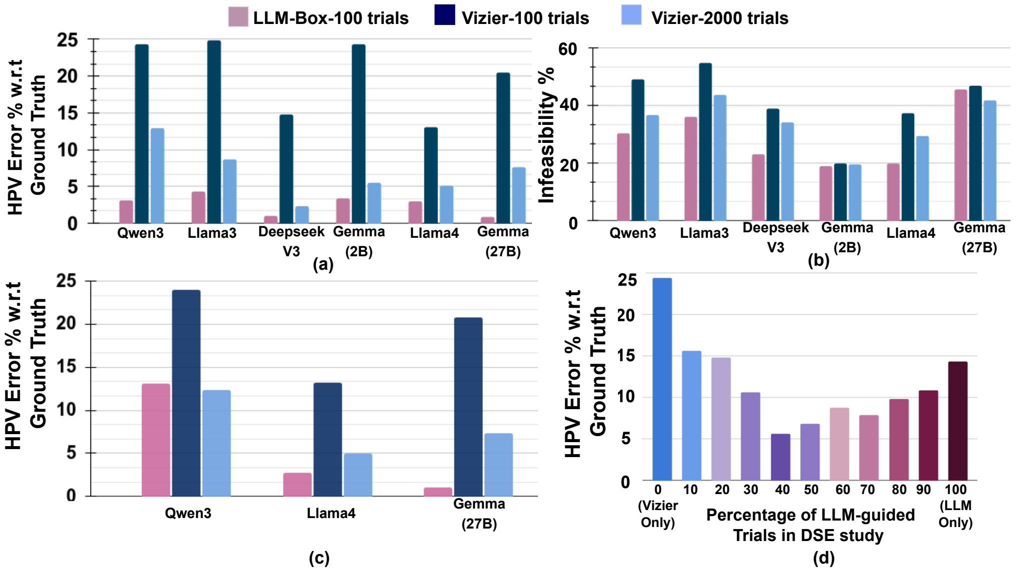

Figure 3: A comparison of our LLM-Box framework with Vizier-only baselines. (a) and (b) evaluate the Hypervolume (HPV) error and infeasibility rate for transfer learning across systems. (c) shows the HPV error for transfer learning across models. (d) presents an ablation study on the impact of the warm-start budget (percentage of LLM-guided trials). **Note:** All results are averaged across TPU pods. "*X trials*" in the labels are results for the corresponding framework with *X* simulated trials.

demonstrating effective knowledge transfer.

Across Models In another experiment, we gave the LLM the Pareto-optimal trials from a set of source models (Llama 3, DeepseekV3, Gemma-2B) and targeted new, unseen models (Llama4, Qwen 3, Gemma-27B). Figure 3(c) shows that the LLM-guided approach led to a much more accurate Pareto-frontier than Vizier could achieve with 20× more trial budget.

4.2 Warm start

Figure 3(d) shows an ablation study on the role of LLM, confirming that a balanced warm-start is crucial. Using an LLM agent to actively guide the initial exploration trials before letting Vizier take over yielded the best results. However, relying solely on LLM hinders exploration, while using only Vizier suffers from the inefficiencies of initial random exploration. We also find that the duration of the optimal involvement of the LLM agent depends on the inherent infeasibility of the design space.

5 Conclusion and Future Work

We presented `LLM-Box` , a framework that augments multi-objective black-box optimization with large language model guidance for efficient hardware/software co-design of ML accelerators. By combining intrinsic domain knowledge from pre-trained LLMs with retrieval from prior explorations, our approach provides warm-start priors and trial filtering that steers the search toward feasible and high-quality design points. Empirically, `LLM-Box` identifies Pareto-optimal execution strategies with 40–150× fewer simulations than exhaustive search and achieves improved sample efficiency compared to a state-of-the-art BBO tool. Moreover, the framework demonstrates robust transfer learning, effectively generalizing across both models and hardware platforms. Our ablation study highlights the critical role of balanced warm-starting—too little LLM involvement limits efficiency, while too much hinders exploration. These findings suggest that LLM-guided BBO offers a promising paradigm for tackling expensive design space exploration problems. Looking forward, we envision extending this methodology to larger design spaces and investigating alternative modes of interaction where LLMs and optimizers collaborate more effectively.

References

[1] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, and Inderjit Dhillon et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1(1):4, 2024.

[4] Meta AI. Introducing LLaMA 4: Advancing Multimodal Intelligence. Meta AI Blog, April 2025. Accessed: 2025-08-19.

[5] Norm Jouppi, George Kurian, Sheng Li, Peter Ma, Rahul Nagarajan, Lifeng Nai, Nishant Patil, Suvinay Subramanian, Andy Swing, Brian Towles, et al. Tpu v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings. In *Proceedings of the 50th annual international symposium on computer architecture*, pages 1–14, 2023.

[6] Dan Zhang, Safeen Huda, Ebrahim Songhori, Kartik Prabhu, Quoc Le, Anna Goldie, and Azalia Mirhoseini. A full-stack search technique for domain optimized deep learning accelerators. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 27–42, 2022.

[7] Srivatsan Krishnan, Amir Yazdanbakhsh, Shvetank Prakash, Jason Jabbour, Ikechukwu Uchendu, Susobhan Ghosh, Behzad Boroujerdian, Daniel Richins, Devashree Tripathy, Aleksandra Faust, and Vijay Janapa Reddi. Archgym: An open-source gymnasium for machine learning assisted architecture design. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*, New York, NY, USA, 2023. Association for Computing Machinery.

[8] Jaehong Cho, Minsu Kim, Hyunmin Choi, and Jongse Park. LLM-sim: A simulation infrastructure for LLM inference serving systems. In *Machine Learning for Computer Architecture and Systems 2024*, 2024.

[9] Cong Guo, Feng Cheng, Zhixu Du, James Kiessling, Jonathan Ku, Shiyu Li, Ziru Li, Mingyuan Ma, Tergel Molom-Ochir, Benjamin Morris, Haoxuan Shan, Jingwei Sun, Yitu Wang, Chiyue Wei, Xueying Wu, Yuhao Wu, Hao Frank Yang, Jingyang Zhang, Junyao Zhang, Qilin Zheng, Guanglei Zhou, Hai Li, and Yiran Chen. A survey: Collaborative hardware and software design in the era of large language models. *IEEE Circuits and Systems Magazine*, 25(1):35–57, 2025.

[10] Sheng-Chun Kao and Tushar Krishna. Gamma: Automating the hw mapping of dnn models on accelerators via genetic algorithm. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–9, 2020.

[11] Sheng-Chun Kao, Michael Pellauer, Angshuman Parashar, and Tushar Krishna. Digamma: domain-aware genetic algorithm for hw-mapping co-optimization for dnn accelerators. In *Proceedings of the 2022 Conference & Exhibition on Design, Automation & Test in Europe*, DATE '22, page 232–237. European Design and Automation Association, 2022.

[12] Chirag Sakhuja, Zhan Shi, and Calvin Lin. Leveraging domain information for the efficient automated design of deep learning accelerators. In *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 287–301, 2023.

[13] Luigi Nardi, David Koeplinger, and Kunle Olukotun. Practical design space exploration. In *2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 347–358. IEEE, 2019.

[14] Aviral Kumar, Amir Yazdanbakhsh, Milad Hashemi, Kevin Swersky, and Sergey Levine. Data-driven offline optimization for architecting hardware accelerators, 2022.

[15] Chen Bai, Qi Sun, Jianwang Zhai, Yuzhe Ma, Bei Yu, and Martin D.F. Wong. Boom-explorer: Risc-v boom microarchitecture design space exploration framework. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–9, 2021.

[16] Amir Yazdanbakhsh, Christof Angermueller, Berkin Akin, Yanqi Zhou, Albin Jones, Milad Hashemi, Kevin Swersky, Satrajit Chatterjee, Ravi Narayanaswami, and James Laudon. Apollo: Transferable architecture exploration. *arXiv preprint arXiv:2102.01723*, 2021.

[17] Kavya Sreedhar, Josh Ogbonda, Pengqi Yin, Narges Shahidi, Kanthi Nagaraj, Zhijie Deng, Rami Cohen, Ton Kalker, Sameer Kumar, Amir Yazdanbakhsh, et al. Leveraging llms to improve hardware-software co-design workflow productivity and accessibility. In *Machine Learning for Computer Architecture and Systems 2025*, 2025.

[18] Hannah Lin, Martin Maas, Maximilian Roquemore, Arman Hasanzadeh, Fred Lewis, Yusuf Simonson, Tzu-Wei Yang, Amir Yazdanbakhsh, Deniz Altinbüken, Florin Papa, Maggie Nolan Edmonds, Aditya Patil, Don Schwarz, Satish Chandra, Chris Kennelly, Milad Hashemi, and Parthasarathy Ranganathan. Eco: An llm-driven efficient code optimizer for warehouse scale computers, 2025.

[19] Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and David Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1487–1495, 2017.

[20] Andreas Krause and Cheng Ong. Contextual gaussian process bandit optimization. *Advances in neural information processing systems*, 24, 2011.

[21] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.

[22] Jinhyuk Lee, Feiyang Chen, Sahil Dua, Daniel Cer, Madhuri Shanbhogue, Iftekhar Naim, Gustavo Hernández Ábrego, Zhe Li, Kaifeng Chen, Henrique Schechter Vera, et al. Gemini embedding: Generalizable embeddings from gemini. *arXiv preprint arXiv:2503.07891*, 2025.