Towards Understanding Challenges Surrounding Agentic System Design

Michael Shen* **Xiaoyue Zhou*** **Ishan Patwardhan*** **Yueying Li** **Udit Gupta**
ECE/CS Department, Cornell Tech, New York City, United States
{mts247, xz957, ip259, yl3469, ugupta}@cornell.edu

1. INTRODUCTION AND RELATED WORK

Agentic systems are changing the modern inference paradigm by introducing dynamic, heterogeneous workloads that are less predictable than traditional inference pipelines [1]. Because these agents evolve at runtime with different control flows, tool calls, and multi-step reasoning, they create execution paths where resource demands and data patterns are constantly changing [2, 8, 9]. This unpredictability necessitates a shift in how we build and evaluate infrastructure. Our work aims to call attention to the need for commensurate accuracy/energy benchmarking to address current challenges of practical, efficient agentic system design. We show how cost-aware system design can yield 2.3×–3.4× lower latency and energy, or 12.3× fewer tokens at near iso-accuracy.

2. METHOD

We design and evaluate three agentic systems [7, 11] in multi-turn settings to study interaction patterns between LLM-tool and LLM-LLM configurations. For LLM–tool systems: a coding agent that iterates with a compiler, and a RAG [6] pipeline. For LLM-LLM systems: a reflection-based system consisting of a coding agent paired with a reflector agent [10].

Agent → Compiler ⟳
Vector Search → Agent ⟳
Tool-LLM Loops

Coding Agent → Reflector Agent
LLM-LLM Loop

We use Qwen3 [12] models on NVIDIA A6000 Ada GPUs, evaluated on the HumanEval [3] / TriviaQA [4] benchmarks.

3. EXPERIMENTS & RESULTS

The breadth of the agentic system design space makes applying traditional design space exploration methods like gradient descent or Bayesian optimization extremely difficult. This complexity also hampers online serving and design space exploration, as the immense number of parameters renders model routing and reinforcement learning impractical due to sparse reward and the curse of dimensionality [13].

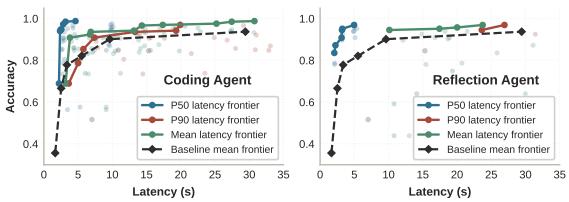

Figure 1: Accuracy–latency pareto curves for Qwen3 model combinations. Coding includes two agent–compiler loops, while reflection involves only model-to-model interaction.

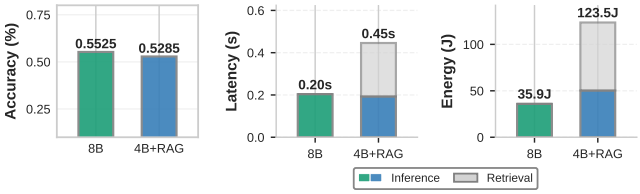

Figure 2: A standalone LLM outperforms a smaller LLM augmented with a retrieval-augmented generation context.

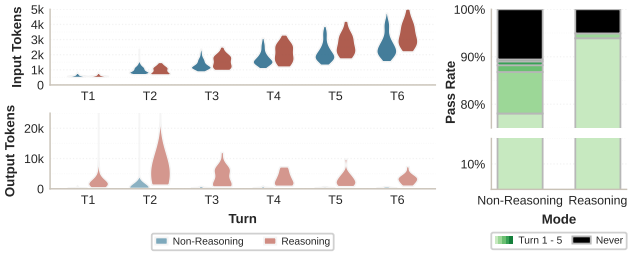

Figure 3: Token counts increase significantly with reasoning and iterative model calls, with only marginal accuracy gains.

Takeaway 1: Increasing model complexity and adding tools often yields inconsistent accuracy gains while significantly increasing latency and compute cost. Switching models across iterations introduces varied performance–accuracy tradeoffs (Figure 1), with optimal P50 configurations often differing from mean or P90 frontiers (see below table). Similarly, tools like RAG can 2.3× latency and 3.4× energy at iso-accuracy (Figure 2).

Config	P50	P90	Mean	Acc.	Config	P50	P90	Mean	Acc.
0.6b→1.7b	2.21	3.63	3.17	0.689	1.7b→4b	–	5.78	–	0.854
4b→8b	–	7.37	3.81	0.909	8b→14b	–	–	6.75	0.924

Takeaway 2: Reasoning cycles and iterative agent calls frequently yield diminishing returns, where the cost outweighs downstream accuracy improvements. Figure 3 shows how agentic iterations increase prefill context as tokens cascade across turns. Additionally, while reasoning reduces the number of turns required, they significantly inflate generation volume, increasing P50 tokens by up to 9.7× and P90 by 53.6×.

4. CONCLUSION

Our work aims to highlight the urgent need to rethink inference infrastructure for dynamic agentic systems, and the need to propose new approaches to benchmarking and design that better balance accuracy, efficiency, and adaptability.

ACKNOWLEDGMENTS

This material is based upon work supported by a National Science Foundation Graduate Research Fellowship under Grant No. DGE – 2139899 and NSF Grant CFF-2326608. Computational resources were generously provided by Google, Amazon, and Chameleon Cloud [5].

References

[1] M. Cemri et al. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

[2] G. I. Chaudhry et al. Murakkab: Resource-efficient agentic workflow orchestration in cloud platforms. *arXiv preprint arXiv:2508.18298*, 2025.

[3] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. 2021.

[4] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, 2017.

[5] K. Keahey, J. Anderson, Z. Zhen, P. Riteau, P. Ruth, D. Stanzione, M. Cevik, J. Colleran, H. S. Gunawi, C. Hammock, J. Mambretti, A. Barnes, F. Halbach, A. Rocha, and J. Stubbs. Lessons learned from the chameleon testbed. In *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*. USENIX Association, July 2020.

[6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[7] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[8] M. Luo et al. Autellix: An efficient serving engine for llm agents as general programs. *arXiv preprint arXiv:2502.13965*, 2025.

[9] K. Santhanam et al. Alto: An efficient network orchestrator for compound ai systems. In *Proceedings of the 4th Workshop on Machine Learning and Systems*, pages 117–125, 2024.

[10] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

[11] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, A. H. Awadallah, R. W. White, D. Burger, and C. Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation, 2023.

[12] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[13] H. Zhang, T. Feng, and J. You. Router-r1: Teaching llms multi-round routing and aggregation via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.