3D Optimization for AI Inference Scaling: Balancing Accuracy, Cost, and Latency

Minseok Jung[1]*, Abhas Ricky[1], Rameez Chatni[1]

[1] Cloudera
5470 Great America Parkway
Santa Clara, CA 95054 USA

Abstract

AI inference scaling is often tuned through 1D heuristics (a fixed reasoning pass) or 2D bivariate trade-offs (e.g., accuracy vs. compute), which fail to consider cost and latency constraints. We introduce a 3D optimization framework that jointly calibrates accuracy, cost, and latency within a unified decision space, enabling constraints-aware inference scaling. Using Monte Carlo simulations across three representative scenarios and nine simulated large language models, we evaluate four optimization methods to address the 3D multi-objective optimization (MOO) problem. Framing inference scaling in MOO shapes a feasible space that 1D and 2D optimizations fail to capture, enabling environment-adaptive selection of the inference scaling k. Results show that knee-point optimization based on Pareto frontiers achieves the best balance, while accuracy-maximization remains favorable when accuracy is prioritized. Our results further show that smaller models, when combined with optimal inference scaling, can match or exceed the performance of larger models at a fraction of the cost. The framework establishes a theoretical foundation for deployment-aware inference scaling across diverse operational conditions. Code and simulation available at https://github.com/masonjung/inference-scaling-moo.

Introduction

Beyond scaling AI training, which enhances performance by providing more data, *scaling inference*—which executes multiple reasoning paths before generating a final output—has been highlighted as an efficient method to improve performance (Snell et al. 2024) and reduce erroneous generations (Park et al. 2025). By sampling multiple responses, instead of relying on a single generation, inference scaling increases the likelihood of obtaining higher-quality outputs at the inference stage. This enables even smaller models to surpass the larger models (Wu et al. 2024; Wang et al. 2022). This approach is particularly efficient when parallel computing is available, where advances in AI infrastructure allow multiple inference instances to run concurrently (Bian, Yan, and Venkataraman 2025; NVIDIA 2025).

Prior research on AI inference scaling has primarily focused on 2D relationships between performance and inference scale, demonstrating accuracy gains with increased computational allocation (Wang et al. 2022; Jin, Wei, and Brooks 2025). This approach addresses the limitation of static inference scaling by allowing adaptive compute allocation. For example, in medical AI applications, users may willingly invest additional computational resources to obtain more reliable results. Executing multiple inference passes substantially increases the likelihood of generating higher-quality outputs compared to the single-shot inference (Snell et al. 2024). Such bivariate optimization effectively identifies the inference scale that maximizes accuracy under a limited compute budget. However, the 2D optimization fails to account for constraints on time and cost, factors that critically considered in the real-world deployment settings (NVIDIA 2025). Neglecting these dimensions renders existing scaling analyses incomplete for practical, deployment-aware optimization of AI inference.

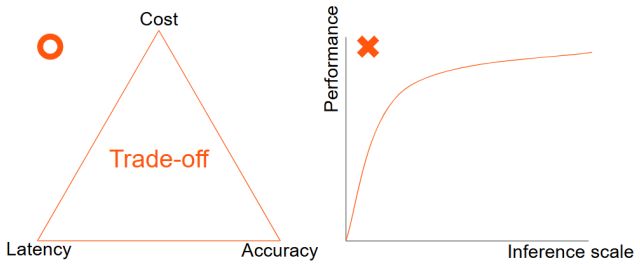


Figure 1: AI inference scaling requires optimizing the multidimensional tradeoff among cost, latency, and accuracy rather than relying on a bivariate performance-compute tradeoff that neglects the cost and latency constraints inherent in real-world deployment.

We argue that inference scaling should be formulated as a **multidimensional optimization problem** (MOO) that jointly considers accuracy, cost, and latency, as illustrated in Fig. 1. Conventional 2D overlooks the associated costs and latency constraints inherent in real-world deployments. For example, in clinical decision support systems, where inference must occur within strict latency and cost budgets, the model can perform only a limited number of reasoning passes, even if additional computation could marginally improve accuracy. To capture these effects more realistically, it is better to model the tradeoff in 3D space to be accountable to core features: *performance, cost, and latency.*

*Contact: minseok.jung@cloudera.com.

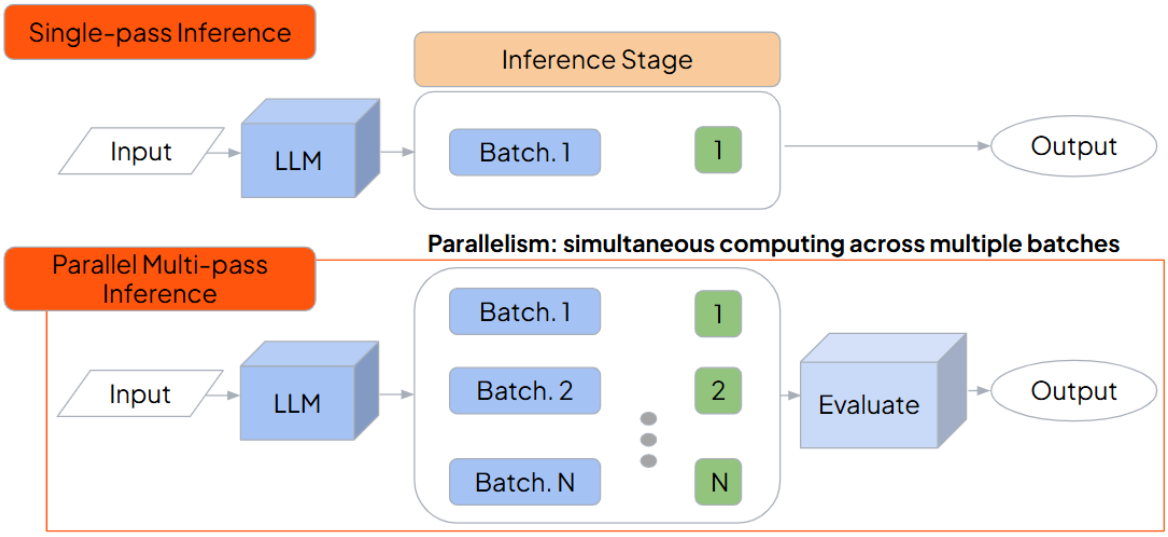


Figure 2: Unlike conventional inference, which performs a single stochastic forward pass (k=1) to generate one output, *parallel inference scaling* executes multiple inference batches concurrently ($k \geq 2$) and aggregates their outputs (e.g., best-of-k) to improve accuracy with minimal increase in latency. Parallelism is particularly effective when sufficient computational resources are available to support concurrent execution.

To address the AI inference scaling problem from a multidimensional perspective, we develop a framework that integrates MOO, simulation for LLM generations, and Monte Carlo (MC) estimation. Building on the 2D bivariate tradeoff between performance-compute, we extend the notion of inference scaling into a tri-objective regime that jointly optimizes accuracy, cost, and latency. In the simulation, we model inference for a stochastic generation with parallelism, and formulate the search for the optimal scale $k^\star$ as a constrained MOO. We further run four optimization methods–accuracy maximization, cube-volume balance, and Pareto-based utopia closet and knee point selection–to characterize efficiency trade-offs under diverse deployment constraints.

The proposed framework provides a unified lens for understanding deployment-aware inference scaling. The main contributions of this research are as follows:

- **MOO framework for inference scaling.** We formalize AI inference scaling as a multi-objective optimization (MOO) problem that jointly considers accuracy, cost, and latency. This framework explicitly incorporates cost and latency–factors overlooked in prior 1D and 2D optimizations–and establishes a theoretical foundation for deployment-aware inference optimization.
- **MC simulation under realistic constraints.** We develop a stochastic simulation that models LLM inference under realistic variability in token length, cost, and latency. Using MC estimation across nine representative model configurations and three constraint scenarios, we map optimality in the 3D objective space.
- **Comparative analysis of optimization strategies.** We evaluate four optimization methods—ACC maximization, maximal-cube, utopia-closest, and knee-point—simulating how each performs under different deployment priorities. The results show that knee-point selection achieves the best relative efficiency, while ACC-maximization is the best when accuracy is prioritized.

This MOO perspective to AI inference scaling not only overcomes the limitation of the bivariate tradeoff optimization but also bridges theoretical scaling-law and real-world deployment needs.

Related Work

Research on AI inference scaling and optimization has progressed along three major directions: (1) scaling inference through parallelism, (2) addressing operational constraints in deployment environments, and (3) formalizing inference selection as a MOO problem. These research threads are deeply interrelated: scaling techniques enable performance gains, system constraints delineate the feasible optimization space, and MOO provides principled methods for balancing the competing objectives.

Inference Scaling through Parallelism. Inference scaling methods enhance model performance by generating multiple candidate outputs and selecting the best among them. Techniques such as best-of-k and self-consistency demonstrate that additional sampling at inference time can enhance reasoning capacity without retraining (Sadhukhan et al. 2025; Wang et al. 2022). Parallelism makes this attainable by executing multiple inference passes simultaneously with minimal latency overhead, leveraging distributed compute resources such as GPUs (NVIDIA 2025). Recent studies also suggest that smaller models can match or even exceed the accuracy of larger ones when parallel inference is effectively utilized (Snell et al. 2024). While these techniques highlight the potential of scaling inference through

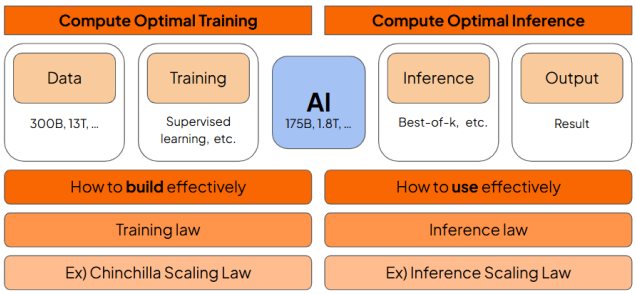


Figure 3: Compute-optimal AI system requires scaling efficiency in both training and inference. Training scaling laws (e.g., Chinchilla law) guide how models are built, while inference scaling laws govern how they are used efficiently under real-world cost and latency constraints.

parallelism, most studies focus on increasing performance via more computing, rather than configuring the optimality with both cost and latency.

Operational Constraints and System Efficiency Another research line emphasized the need to incorporate cost and latency factors for real-world AI systems (NVIDIA 2025; Aubrey 2025). These studies highlight that inference performance is bounded by operational factors such as per-query cost ceilings and service-level latency requirements. Under such constraints, balancing performance, latency, and cost is crucial to achieve effective AI infrastructure operation (Harris 2025). Most of the research remains grounded in bivariate performance-compute tradeoff curve (Brown et al. 2024; Snell et al. 2024; Wang et al. 2022; Jin, Wei, and Brooks 2025; Park et al. 2025). To the best of our knowledge, no prior research has formulated inference optimization that simultaneously accounts for cost, latency, and performance, despite increasing recognition of its necessity.

MOO for Tri-Objective Inference Scaling. From an optimization standpoint, MOO provides an essential framework for balancing multiple competing factors in AI inference. Classical MOO approaches—including distance-to-utopia selection, hypervolume maximization, and knee-point detection—identify Pareto-optimal configurations that maintain balanced efficiency under resource constraints. Also, accuracy-maximization serves as a useful upper bound when performance dominates other considerations, whereas Pareto-based criteria better capture realistic deployment trade-offs. Building on these developments, our work explicitly formulates accuracy, cost, and latency as interdependent objectives within a unified MOO framework, enabling principled and deployment-aware determination of the optimal inference scale k.

Problem Formulation and Modeling

Overview

The objective is to find the optimal inference scale k that jointly balance *accuracy*, *cost*, and *latency* under resource constraint scenarios. We formalize this as an MOO problem with multiple constraints. The key parameters are summarized in Table 1.

Symbol	Definition
k	Number of inference passes
$L_{\text{in}}, L_{\text{out}}$	Input/output token lengths
$\mu_{L_{\text{in/out}}}, \sigma_{L_{\text{in/out}}}$	Mean and std. of in/out token lengths
A_i	ACC of the i-th inference
$A(k)$	Aggregate ACC from k (e.g. best-of-k)
μ_A, σ_A	Mean and std. of single-inference ACC
C_i, T_i	Cost and time of the i-th inference
$c_{\text{in/out}}$	Cost coefficients for in/out tokens
$t_{\text{in/out}}$	Latency coefficients for in/out tokens
P	Parallel factor (concurrent inferences)
$C_{\max}, T_{\max}$	Maximum cost/time constraints
$A_{\min}$	Minimum acceptable ACC
$\hat{\mu}_C(k)$	Expected cost (MC)
$\hat{\mu}_T(k)$	Expected time (MC)
$\hat{\mu}_A(k)$	Expected ACC (MC)
$\mathcal{F}$	Feasible set of inference counts

Table 1: Notation summary for the simulation. Symbols represent model-dependent quantities for ACC, cost, and latency. MC estimates are denoted by the hat symbol ($\hat{h}$).

Stochastic Model of Inference Scaling

Each inference takes input tokens L_{in} and generates output tokens L_{out}, drawn from Gaussian distributions:

$$(L_{\text{in}}, L_{\text{out}}) \sim \left(\mathcal{N}(\mu_{L_{\text{in}}}, \sigma^2_{L_{\text{in}}}), \mathcal{N}(\mu_{L_{\text{out}}}, \sigma^2_{L_{\text{out}}})\right) \quad (1)$$

with a clipping to ensure positivity. The accuracy of a single inference is also modeled as a Gaussian random variable:

$$A_i \sim \mathcal{N}(\mu_A, \sigma_A^2), \quad A_i \in [0, 1]. \quad (2)$$

This assumption of parametric distribution is justified by the Central Limit Theorem (CLT) because both token lengths and single-inference ACC result from the numerous independent stochastic factors. The model is therefore appropriate for applying the CLT assumption, as these aggregated effects yield approximately normal variability around their means. When k independent inferences are executed, aggregate performance is defined by an evaluation metrics. We use the "best-of-k" rule:

$$A(k) = \max\{A_1, A_2, \ldots, A_k\}. \quad (3)$$

The "best-of-k" is the most common AI inference strategy (Snell et al. 2024).

Cost and Latency per Inference

The costs and latency of an inference are modeled as linear functions. For the i-th inference, the cost and latency are:

$$C_i = c_{\text{in}} L_{\text{in},i} + c_{\text{out}} L_{\text{out},i}; \quad T_i = t_{\text{in}} L_{\text{in},i} + t_{\text{out}} L_{\text{out},i} \quad (4)$$

In parallel computing environment, the k inferences are executed concurrently (e.g., eight reasoning batches simultaneously) as shown in Fig. 2. Thus the latency can be divided by the parallel factor P, while each inference cost is same as a sequential execution:

$$C(k) = \sum_{i=1}^{k} C_i, \tag{5}$$

$$T(k) = \frac{1}{P}\sum_{i=1}^{k} T_i = \left(\frac{k}{P}\right)\bar{T}, \quad \bar{T} = \frac{1}{k}\sum_{i=1}^{k} T_i. \tag{6}$$

In this equation, $\bar{T}$ denotes the mean per-inference latency, and P represents the degree of parallelism, reflecting the assumption that P independent inference processes can run concurrently on separate compute units.

Monte Carlo Approximation

Because $C(k), T(k), A(k)$ are stochastic, closed-form expectations are generally intractable. That is to say, their values fluctuate across inference runs due to variability in usage. We therefore employ MC estimation with M trials to approximate the mean and standard deviation.

$$\hat{\mu}_q(k) = \frac{1}{M}\sum_{j=1}^{M} q^{(j)}(k), \quad q \in \{C, T, A\}. \tag{7}$$

Confidence intervals (CI) (e.g., 95%) could be also estimated from percentiles across trials.

Feasible Region and Objective

Deployment settings typically impose an upper bound on computational cost ($C_{\max}$) and latency ($T_{\max}$), as well as a required minimum accuracy ($A_{\min}$). These constraints jointly define the feasible set of inference counts:

$$\mathcal{F} = \{k \mid \hat{\mu}_C(k) \le C_{\max},\ \hat{\mu}_T(k) \le T_{\max},\ \hat{\mu}_A(k) \ge A_{\min}\} \tag{8}$$

Within this feasible region, the inference scaling problem seeks the optimal scale of inferences $k^\star$:

$$\begin{aligned} &\text{Given } \big(\hat{\mu}_A(k), \hat{\mu}_C(k), \hat{\mu}_T(k)\big), \\ &\text{Find } k^\star \in \mathcal{F} \end{aligned} \tag{9}$$

such that $k^\star$ maximizes the joint objective $f(\hat{\mu}_A(k), \hat{\mu}_C(k), \hat{\mu}_T(k))$, where $f(\cdot)$ is defined by an optimization method introduced in Section .

Scenario under Different Budgets

To simulate the MOO under the realistic 3D constraints, we describe three representative budget scenarios in Table 2.

Scenario 1: Low cost and latency (Essay feedback). In educational feedback applications, users prefer quick, low-cost responses over higher-priced and slower outputs. Since minor phrasing inconsistencies are acceptable, this scenario tightens the feasible region $\mathcal{F}$ by imposing stricter cost and time constraints.

Scenario	Cost	Time	Accuracy
1: Essay feedback	Low	Low	Moderate
2: Medical AI	High	High	Very high
3: Proposal writing	Low	High	High

Table 2: Deployment scenarios showing how cost, time, and accuracy priorities differ across domains.

Scenario 2: High cost and time budget (Medical AI). In high-stakes domains such as medical decision support, users are open to allocate greater budgets for both cost and time while very high accuracy is demanded. Here, accuracy maximization naturally pushes toward the highest feasible set k.

Scenario 3: Low cost and high time budget (Proposal writing). In proposal drafting, latency constraints are relaxed, allowing longer inference durations for improved accuracy or style, while cost remains limited due to the exploratory nature of the task. Thus, the feasible region $\mathcal{F}$ prioritizes low-cost efficiency and tolerates higher latency, making accuracy maximization more viable.

These scenarios illustrate that the choice of optimization method depends on the deployment context, as differing priorities for cost, speed, and performance necessitate distinct definitions of optimal inference scaling for each case.

Example Models for Simulations

We simulate nine model configurations derived from three sample LLM families—GPT-5, Nemotron, and Qwen3—each instantiated in large, medium, and small sizes. These models capture a broad spectrum of contemporary inference settings with varying performance, latency, and cost. This setup provides a unified basis for analyzing how different budget regimes (Table 2) influence the optimal inference scale $k^\star$ under MOO.

Methodology

Overview

Given the estimated metrics $\big(\hat{\mu}_A(k), \hat{\mu}_C(k), \hat{\mu}_T(k)\big)$, we identify the optimal inference count $k^\star$ within the feasible set $\mathcal{F}$. We evaluate four optimization strategies: (1) *Accuracy Maximization*, (2) *Maximum Cube Volume*, (3) *Utopia-Closest Selection*, and (4) *Knee-Point Selection*. Each embodies a distinct trade-off measure in the 3D space.

Accuracy Maximization

The most direct approach maximizes expected accuracy:

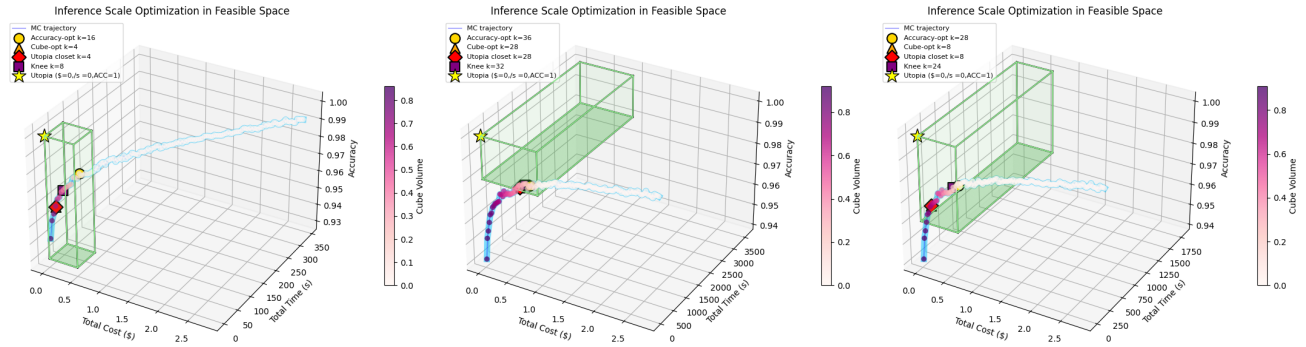
$$k^\star = \arg\max_{k \in \mathcal{F}} \hat{\mu}_A(k). \tag{10}$$

This represents an upper bound that maximizes the ACC while giving a negligible weight on cost and time.

Maximum Cube Volume

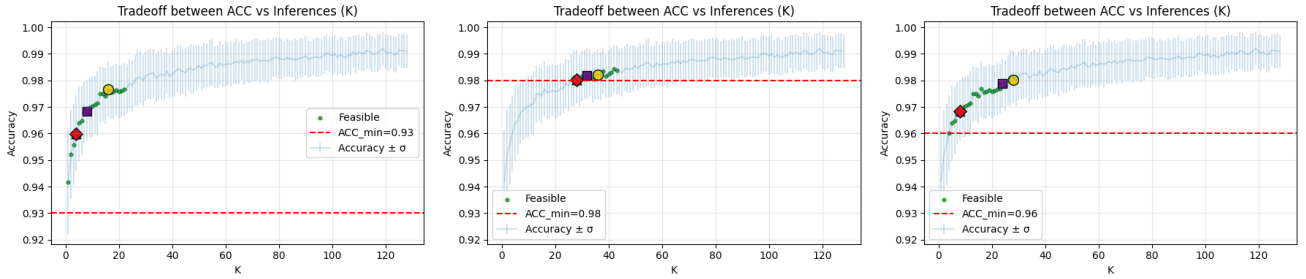
To balance all three objectives in the maximal cube volume in the 3D space, we define normalized goodness scores:

$$g_A(k) = \hat{\mu}_A(k), \quad g_C(k) = 1 - \frac{\hat{\mu}_C(k)}{C_{\max}}, \quad g_T(k) = 1 - \frac{\hat{\mu}_T(k)}{T_{\max}}. \tag{11}$$



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 4: Inference-scaling optimization results for GPT-5 across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes of maximum cost, maximum latency, and minimal ACC ($C_{\max}$, $T_{\max}$, and $A_{\min}$). Markers denote each optimal point (● Maximum-ACC, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); purple colors indicate cube volume (larger is better), and the yellow star marks the utopia point.



(a) 2D ACC–k with optimality for Case 1. (b) 2D ACC–k with optimality for Case 2. (c) 2D ACC–k with optimality for Case 3.

Figure 5: Accuracy-inference scale (k) trade-offs across three scenarios for GPT-5. These show MC means with CI = 95%; the red dashed line indicates minimal ACC ($A_{\min}$). Markers correspond to the optimal configurations identified in Fig. 4. Distinct operational priorities yield different optimal inference scales $k^\star$.

The cube-volume objective maximizes their product:

$$k^\star = \arg\max_{k \in \mathcal{F}} g_A(k)\, g_C(k)\, g_T(k), \tag{12}$$

favoring balanced solutions that make the largest cube under the space rather than prioritizing one or two features.

Pareto Frontier Optimizations

We adopt a Pareto-frontier approach that identifies configurations where no objective can improve without degrading another, representing the best achievable balance among accuracy, cost, and latency. Formally, Pareto frontiers $p_2 = (C_2, T_2, A_2)$ dominates others $p_1 = (C_1, T_1, A_1)$ if

$$C_2 \le C_1, \quad T_2 \le T_1, \quad A_2 \ge A_1, \tag{13}$$

with at least one strict inequality. The Pareto frontier is then defined as

$$\mathcal{P} = \{\, p \in \mathcal{F} \mid \nexists\, q \in \mathcal{F} : q \succ p \,\}. \tag{14}$$

$\mathcal{P}$ represents the boundary of optimal trade-offs where improving one metric inevitably worsens another, forming the efficient surface in the cost–time–accuracy.

Utopia-Closest Selection The utopia point represents an unattainable ideal corresponding to perfect accuracy, zero cost and latency. We can get an efficient solution by picking up a point from the Pareto-frontiers that is the most close to the utopia point that is defined as

$$p^\star = (time = 0, cost = 0, accuracy = 1). \tag{15}$$

It is ideal but unreachable. But this point serves as a geometric reference for evaluating Pareto-optimal points. Each Pareto point is normalized to a comparable scale as

$$\tilde{p} = \left(\frac{C}{C_{\max}}, \frac{T}{T_{\max}}, 1 - A\right), \tag{16}$$

and the solution closest to this ideal is selected as

$$p_{\text{utopia}} = \arg\min_{p \in \mathcal{P}} \|\tilde{p} - p^\star\|. \tag{17}$$

This distance-based criterion selects the configuration that lies nearest to the utopia point on the Pareto surface.

Knee-Point Selection The knee point represents the region of maximum curvature on the Pareto surface. For example, the knee point marks the moment where increasing

accuracy begins to yield diminishing returns relative to the rising cost and latency, representing the relative efficiency. Let $\mathbf{p}(k) = [\frac{\hat{\mu}_C(k)}{C_{\max}}, \frac{\hat{\mu}_T(k)}{T_{\max}}, 1 - \hat{\mu}_A(k)]$ denote the normalized trajectory of inference scaling. The curvature is then:

$$\kappa(k) = \frac{\|\mathbf{p}'(k) \times \mathbf{p}''(k)\|}{\|\mathbf{p}'(k)\|^3}. \tag{18}$$

Intuitively, $\mathbf{p}'(k)$ measures how the trade-off surface changes as k increases, and $\mathbf{p}''(k)$ captures how rapidly that change bends. The denominator normalizes for scale, ensuring curvature reflects the shape of the trade-off rather than its magnitude. The knee point corresponds to:

$$k_{\text{knee}} = \arg\max_k \kappa(k), \quad p_{\text{knee}} = \mathbf{p}(k_{\text{knee}}). \tag{19}$$

where k_{knee} denotes the inference count at which curvature is maximal, and p_{knee} represents the corresponding cost–time–accuracy point on the Pareto surface. The optimal knee point is found by locating the inference count k_{knee} where curvature is maximal and then mapping it to its corresponding trade-off coordinates p_{knee} on the Pareto surface.

Rationales for Optimization Method Selection

These four criteria capture complementary perspectives: **Accuracy Maximization** targets peak performance, **Cube Volume** enforces balanced efficiency, **Utopia-Closest** emphasizes geometric closeness to an ideal, and **Knee-Point** locates the inflection where marginal accuracy gains demand disproportionate cost or time. Because deployment priorities vary (e.g., latency-sensitive, cost-limited, and flawlessness), no single criterion universally dominates. These optimization methods provide practical solutions.

Simulation Overview and Hyperparameters

All simulations adopt consistent parameterization for comparability. Input lengths follow $\mathcal{N}(1024, 64^2)$ and output lengths $\mathcal{N}(2048, 128^2)$; parallelism factor $P = 4$; Monte Carlo trials $M = 300$; random seed 42; and maximum inference count $k_{\max} = 128$. Each scenario applies distinct budget constraints on maximum cost ($C_{\max}$), maximum latency ($T_{\max}$), and minimum acceptable accuracy ($A_{\min}$). Full model configurations are listed in Appendix 3.

Results

Simulation Overview

We systematically evaluated four optimization criteria across three deployment constraint regimes and nine representative simulated LLMs drawn from the GPT-5, NVIDIA Nemotron, and Qwen3 families noted in Appendix 4.

Results Across Deployment Scenarios

Across scenarios, Knee-point optimization produced the most compute-efficient solutions under constrained settings (Scenarios 1 and 3), while accuracy-maximization is the most suite for Scenario 2.

Figures 4 and 5 visualize the corresponding Pareto surfaces in the accuracy–cost–latency space of GPT-5 and conventional 2D tradeoff analysis from Scenario 1. The outcomes are summarized below and in Appendix 2.

Scenario 1: Low Cost and Latency Budget (Essay Feedback). With $C_{\max}$=0.50, $T_{\max}$=60 s, and $A_{\min}$=0.93, the accuracy-optimal configuration occurred at k=16 (97.7 $\pm$ 1.1%, \$0.349, 43.1 s). Cube-optimal and utopia-closest converged at k=4 (96.0 $\pm$ 1.4%, \$0.087, 10.8 s), whereas the knee-point was identified at k=8 (96.8 $\pm$ 1.2%, \$0.174, 21.5 s).

Scenario 2: High Cost and Time Budget (Medical AI). For $C_{\max}$=0.95, $T_{\max}$=3600 s, and $A_{\min}$=0.98, accuracy-maximization yielded the best absolute accuracy at k=36 (98.2 $\pm$ 0.9%, \$0.783, 96.7 s). Knee-point reached k=32 (98.2$\pm$1.0%, \$0.696, 86.0 s) and cube-max and utopia-closet suggested k=28 (98.0$\pm$1.0%, \$0.609, 75.2 s).

Scenario 3: Low Cost, Flexible Latency (Proposal Writing). With $C_{\max}$=0.65, $T_{\max}$=1800 s, and $A_{\min}$=0.96, the accuracy-optimal configuration occurred at k=28 (98.0$\pm$ 1.0%, \$0.609, 75.2 s). The knee-point emerged at k=24 (97.9%$\pm$1.0%, \$0.522, 64.5 s). Cube- and utopia-closest selections (k=8) showing ACC of 0.968$\pm$ 0.012% with total cost of 0.174 in 21.5 s.

Results Across Models

All model families exhibit similar Pareto curvature: larger architectures achieve higher baseline accuracy but incur steeper cost and latency growth as inference multiplicity increases. Smaller models attain comparable accuracy with lower compute budget, reflecting greater efficiency rather than higher absolute performance.

GPT-5 Family. For the GPT series, efficient compute allocation compensates for model size differences. GPT-5 Nano achieved 99.0% accuracy at k=56 from the knee method exceeds GPT-5 Large's 98% at k=16 from Max-ACC while operating at approximately 1/7 of the total cost (\$0.05 vs. \$0.35) under with lower latency (30.1/s vs. 43.1/s) in case 1.

NVIDIA Nemotron Family. Nemotron models exhibit a clear efficiency trend in Scenario 2, demonstrating that smaller architectures can match the accuracy of much larger ones when inference parallelism is leveraged. Nemotron Nano 9B reached perfect accuracy at k=52 (from Max-ACC) with a total cost of \$0.117 and latency of 33.5 s earlier than the Ultra 253B model achieved the same accuracy at k=72 (from Max-ACC) with a cost of \$0.479 and latency of 386.9 s. This four-fold cost difference and ten-fold latency reduction underscore the efficiency of allocating compute to multiple inference paths in smaller models.

Qwen3 Family. Qwen3 models display stable and controllable scaling behavior. In Scenario 3, Qwen3-Max achieved perfect accuracy at k=108 (\$0.63, 464.6 s) with Max-ACC and 0.99 at k=56 (\$0.33, 240.8 s), which is from Knee optimization, demonstrating efficiency of knee method in accuracy–cost-latency trade-offs, by halving cost and latency with a sacrifice of 1% of ACC in scenario 3.

Aggregate Trends- Knee

Across all nine model configurations and three constraint regimes, knee-point optimization consistently pro-

duced the most cost-efficient trade-offs, reducing mean latency by 63% and total cost by 58% relative to accuracy-maximization, while maintaining accuracy within 1–2%. Cube-volume and utopia-closest methods offered additional savings in cost and latency with a tradeoff in ACC. These trends establish a consistent pattern of computing efficiency that generalizes across model scales and deployment contexts. But Max-ACC optimization is the best when high ACC is demanded.

One Null Case for Knee Detection

In one configuration in case 1, we failed to identify a knee point because the feasible Pareto set under the imposed constraints collapsed to at most two non-dominated points, rendering curvature-based knee detection ill-defined. In such cases, utopia-closet or cube-volume can serve an alternative approach to balance efficiency.

Discussion

Toward 3D Optimization from 1D and 2D

Conventional approaches to inference scaling typically relied on 1D heuristics, such as executing a fixed number of reasoning passes (e.g., 16 passes), or on 2D tradeoffs between performance and compute scale. While these methods improved accuracy, they failed to capture the constraints of latency and cost that are essential for AI inference. Because real-world inference should operate within given cost and latency budgets, ignoring these constraints leads to impractical inference. The proposed 3D MOO overcomes the limitation of past approaches by jointly optimizing accuracy, cost, and latency. This method reveals new feasible regions that 1D and 2D optimization fail to capture. The 3D objective space provides an opportunity for a deployment-aware inference optimization.

Efficiency through Scaling with Parallelism

The results demonstrate that additional compute allocation can compensate the performance for the smaller size. When parallelism is available, smaller models can achieve comparable or even superior accuracy to larger architectures with a lower cost and latency. For example, GPT-5 Nano with 56 inference paths showed ACC of 0.99 ($0.05, 30.1s), which is greater than GPT-5 with 24 inference paths that achieved ACC of 0.98 ($0.52, 64.5s) under the knee-point selection criterion in Scenario 3. This result suggests that inference scaling offers an efficient path for small models, especially when parallel inference is available.

Knee-Point Optimization for the Best Balance

Among MOOs, knee-point selection consistently achieved the most balanced efficiency in the feasible space. The Knee MOO identifies the point of maximum curvature on the Pareto frontier, showing the relative efficiency in tri-objectives. While it could be fail to be identified for the linearity of the frontiers that does not show the knee point, it can serve as the most practical solution except for the case that needs the maximal accuracy. Regarding the inference for the minimal cost and the best latency, the solution is always inference with a single path.

Limitations and Future Work

This study relies on stochastic simulation rather than LLM deployments. Thus, it abstracts away system-level factors such as GPU utilization, memory contention, and network queuing delays. Real-world AI infrastructures often exhibit non-Gaussian variability due to heterogeneous hardware, fluctuating energy conditions, and network overhead, all of which complicate inference MOO for further dimensions and higher dimensions. Incorporating empirical traces from AI infrastructure environments would further validate the proposed framework.

Also, the proposed AI inference MOO framework is critical not only for single-agent models but also for multi-agent systems, where multiple agents and tools compound the cost and latency overhead, further amplifying the need for efficient inference scaling. Moreover, as dimensions such as energy consumption, safety constraints, and thermal limits become increasingly relevant, the framework can be extended beyond 3D to support higher-dimensional MOO. Such a generalized formulation would enable jointly optimizing energy, safety, and heat dissipation on top of the proposed 3D space will support deployment-aware reasoning for real-world AI systems.

Future work could extend our framework to infrastructure-aware optimization. This includes integrating hardware metrics from GPUs and Kubernetes-based orchestration to align inference scaling with real-time resource allocation. The optimization layer could operate as an adaptive router that dynamically selects the model and inference scale based on complexity and system load. Such integration would enable real-world MOO across heterogeneous pools, improving inference efficiency on a large scale.

Conclusion

We reframe AI inference scaling as a multidimensional optimization problem that considers accuracy, cost, and latency in one optimization to overcome the limitations of 1D and 2D optimization. By integrating MC simulations with MOO, the proposed framework enables constraint-aware inference scaling. Across diverse model families and scenarios, the result revealed that knee-optimality from the Pareto frontiers makes the scale that shows a relative efficiency among others. Also, we found that smaller models could show a better performance than larger models with a lower cost and latency. Beyond these findings, this framework establishes a foundation for advanced inference systems that respond to operational conditions. Embedding MOO into orchestration layers will enable efficient inference across heterogeneous AI models and diverse environments. As foundation models and agentic systems continue to expand, principled inference scaling will become essential to deploy AI efficiently.

Acknowledgments

This project is a part of Cloudera's Accelerators for ML Projects (AMPs). The AMP is accessible via https://github.com/cloudera/CAI_AMP_Inference_Scaling_Optimization. Thanks to Nashua Springberry and Michael Schuler for constructive comments on the design and programming for the simulation.

References

Aubrey, K. 2025. How the Economics of Inference Can Maximize AI Value. *NVIDIA Technical Blog*. Available at https://blogs.nvidia.com/blog/ai-inference-economics/.

Bian, S.; Yan, M.; and Venkataraman, S. 2025. Scaling Inference-Efficient Language Models. *arXiv preprint arXiv:2501.18107*.

Brown, B.; Juravsky, J.; Ehrlich, R.; Clark, R.; Le, Q. V.; Ré, C.; and Mirhoseini, A. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Harris, D. 2025. Think SMART: How to Optimize AI Factory Inference Performance. *NVIDIA Technical Blog*. Available at https://blogs.nvidia.com/blog/think-smart-optimize-ai-factory-inference-performance/.

Jin, Y.; Wei, G.-Y.; and Brooks, D. 2025. The Energy Cost of Reasoning: Analyzing Energy Usage in LLMs with Test-time Compute. *arXiv preprint arXiv:2505.14733*.

NVIDIA. 2025. The Art of Balancing AI Inference Cost and Performance. *https://nvdam.widen.net/s/hrprjhtmm9/the-it-leaders-guide-to-ai-inference-and-performance*.

Park, Y.-J.; Greenewald, K.; Alim, K.; Wang, H.; and Azizan, N. 2025. Know What You Don't Know: Uncertainty Calibration of Process Reward Models. *arXiv preprint arXiv:2506.09338*.

Sadhukhan, R.; Chen, Z.; Zheng, H.; Zhou, Y.; Strubell, E.; and Chen, B. 2025. Kinetics: Rethinking Test-Time Scaling Laws. *arXiv preprint arXiv:2506.05333*.

Snell, C.; Lee, J.; Xu, K.; and Kumar, A. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wu, Y.; Sun, Z.; Li, S.; Welleck, S.; and Yang, Y. 2024. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*.

Model Configuration and Hyperparameters

To evaluate the proposed inference scaling framework, we define a set of representative model configurations that approximate contemporary large language models (LLMs) across different families and scales. These configurations are not tied to proprietary systems but serve as controllable abstractions for stochastic simulation. Each model specifies per-token costs, latencies, and accuracy distributions used in the Monte Carlo analysis. Costs are expressed in U.S. dollars per million tokens, and latency values represent seconds per input (t_{in}) or output (t_{out}) token. Unless otherwise stated, the parallelism factor is fixed at $P = 4$. Table 3 summarizes all hyperparameters.

Model	c_{in}	c_{out}	t_{in}	t_{out}	$\mu_{L_{\text{in}}}$	$\mu_{L_{\text{out}}}$	Accuracy Mean $\pm$ Std	P_{default}
GPT-5	1.25e-6	10.00e-6	0.0005	0.0050	1024	2048	0.94 ± 0.02	4
GPT-5 Mini	0.25e-6	2.00e-6	0.00025	0.0020	1024	2048	0.92 ± 0.03	4
GPT-5 Nano	0.05e-6	0.40e-6	0.00010	0.0010	1024	2048	0.91 ± 0.04	4
Nvidia Nemotron Ultra 253B	0.90e-6	2.80e-6	0.0010	0.0100	1024	2048	0.93 ± 0.05	4
Nvidia Nemotron H 47B	0.40e-6	1.50e-6	0.0004	0.0040	1024	2048	0.92 ± 0.06	4
Nvidia Nemotron Nano 9B v2	0.20e-6	1.00e-6	0.00012	0.0012	1024	2048	0.91 ± 0.07	4
Qwen3-Max	0.90e-6	2.40e-6	0.0008	0.0080	1024	2048	0.90 ± 0.04	4
Qwen3-Next-80B-A3B	0.50e-6	1.25e-6	0.0004	0.0040	1024	2048	0.89 ± 0.05	4
Qwen3-30B-A3B	0.35e-6	0.90e-6	0.00025	0.0020	1024	2048	0.88 ± 0.06	4

Table 3: Representative model configurations used for stochastic simulation. Costs are given in USD per million tokens. Latency values denote seconds per token for input (t_{in}) and output (t_{out}). All models assume a default parallelism of $P = 4$.

These parameters ensure that the optimal inference scale $k^\star$ under the default budget constraints ($C_{\max} = 0.50$, $T_{\max} = 60$ s) typically falls within a practical range ($k \approx 10$–20), allowing interpretable comparisons across model families while maintaining computational realism.

Simulations

Scenario 1: Low Cost and Time Budget (Essay Feedback)

This scenario represents lightweight applications such as automated essay evaluation or short-form feedback generation, where users prioritize quick turnaround and low cost while maintaining reasonable accuracy. We set the budget constraints to a maximum cost of \$0.50 and a latency limit of 60 seconds, with a minimum acceptable accuracy of $A_{\min} = 0.93$. Under these deployment limits, four optimization strategies—accuracy-maximization, cube-optimal (multiplicative balance), utopia-closest, and knee-point—are evaluated across nine representative models.

Table 4 summarizes the simulation results. The knee-point method consistently achieves the best efficiency across most models, delivering near-maximal accuracy with notably reduced cost and latency. Cube- and utopia-based selections favor minimal resource use but sacrifice accuracy, while accuracy-maximization frequently over-computes within the same budget. Overall, knee-point optimization provides the most balanced solution for latency-sensitive or cost-limited settings such as real-time educational feedback.

Model	**Accuracy-Optimal**	**Cube-Optimal**	**Utopia-Closest**	**Knee-Point**
G-5	k=16 / 0.98 / 0.35 / 43.1	k=4 / 0.96 / 0.09 / 10.8	k=4 / 0.96 / 0.09 / 10.8	k=8 / 0.97 / 0.17 / 21.5
G-5M	k=52 / 0.99 / 0.23 / 56.6	k=4 / 0.95 / 0.02 / 4.4	k=4 / 0.95 / 0.02 / 4.4	k=40 / 0.98 / 0.17 / 43.5
G-5N	k=108 / 1.00 / 0.09 / 58.1	k=4 / 0.95 / 0.00 / 2.2	k=4 / 0.95 / 0.00 / 2.2	k=56 / 0.99 / 0.05 / 30.1
N-253B	k=8 / 0.99 / 0.05 / 42.9	k=4 / 0.97 / 0.03 / 21.5	k=4 / 0.97 / 0.03 / 21.5	–
N-47B	k=24 / 1.00 / 0.08 / 51.6	k=4 / 0.97 / 0.01 / 8.6	k=4 / 0.97 / 0.01 / 8.6	k=8 / 0.99 / 0.03 / 17.2
N-9B	k=52 / 1.00 / 0.12 / 33.5	k=4 / 0.97 / 0.01 / 2.6	k=4 / 0.97 / 0.01 / 2.6	k=48 / 1.00 / 0.11 / 31.0
Q3-M	k=12 / 0.96 / 0.07 / 51.6	k=4 / 0.94 / 0.02 / 17.2	k=4 / 0.94 / 0.02 / 17.2	k=8 / 0.96 / 0.05 / 34.4
Q3-80B	k=24 / 0.98 / 0.07 / 51.6	k=4 / 0.94 / 0.01 / 8.6	k=4 / 0.94 / 0.01 / 8.6	k=8 / 0.96 / 0.03 / 17.2
Q3-30B	k=52 / 1.00 / 0.11 / 56.6	k=4 / 0.94 / 0.01 / 4.4	k=4 / 0.94 / 0.01 / 4.4	k=48 / 0.99 / 0.11 / 52.2

Table 4: Scenario 1 (Essay Feedback). Optimal configurations across nine models under $C_{\max} = 0.50$, $T_{\max} = 60$ s, and $A_{\min} = 0.93$. Each cell shows k / accuracy / total cost (\$) / total latency (s). Model abbreviations: G–5 (GPT–5), G–5M (GPT–5 Mini), G–5N (GPT–5 Nano), N–253B (Nemotron Ultra 253B), N–47B (Nemotron H 47B), N–9B (Nemotron Nano 9B v2), Q3–M (Qwen3-Max), Q3–80B (Qwen3-Next-80B-A3B), and Q3–30B (Qwen3-30B-A3B). Standard deviations (omitted for brevity) were all within approximately ± 0.01.

Scenario 2: High Cost and Time Budget (Medical AI)

This scenario reflects mission-critical domains such as medical diagnostics or clinical report generation, where users prioritize maximum accuracy and reliability over latency or cost considerations. Here, the system operates under relaxed constraints with a maximum cost of \$0.95 and a latency budget of 3600 seconds, while requiring a minimum target accuracy of $A_{\min} = 0.98$. Under these generous budgets, we again evaluate four optimization strategies—accuracy-maximization, cube-optimal, utopia-closest, and knee-point—across nine representative model configurations.

Table 5 summarizes the results. Accuracy-maximization consistently delivers the highest absolute accuracy, making it preferable in precision-critical settings such as medical AI. However, the knee-point approach achieves nearly identical accuracy at substantially lower computational cost and latency, demonstrating superior efficiency when resource constraints or throughput considerations are present. Cube- and utopia-based methods remain the most cost-efficient but trade off small amounts of accuracy. Overall, these findings suggest that while accuracy-maximization remains ideal for clinical-grade precision, knee-point optimization provides a more balanced and deployable alternative.

Model	Accuracy-Optimal	Cube-Optimal	Utopia-Closest	Knee-Point
G–5	k=36 / 0.98 / 0.78 / 96.7	k=28 / 0.98 / 0.61 / 75.2	k=28 / 0.98 / 0.61 / 75.2	k=32 / 0.98 / 0.70 / 86.0
G–5M	k=128 / 0.99 / 0.56 / 139.3	k=32 / 0.98 / 0.14 / 34.8	k=32 / 0.98 / 0.14 / 34.8	k=64 / 0.99 / 0.28 / 69.7
G–5N	k=128 / 1.00 / 0.11 / 68.8	k=16 / 0.98 / 0.01 / 8.6	k=16 / 0.98 / 0.01 / 8.6	k=56 / 0.99 / 0.05 / 30.1
N–253B	k=72 / 1.00 / 0.48 / 386.9	k=8 / 0.99 / 0.05 / 42.9	k=8 / 0.99 / 0.05 / 42.9	k=48 / 1.00 / 0.32 / 258.0
N–47B	k=52 / 1.00 / 0.18 / 111.8	k=8 / 0.99 / 0.03 / 17.2	k=8 / 0.99 / 0.03 / 17.2	k=48 / 1.00 / 0.17 / 103.2
N–9B	k=52 / 1.00 / 0.12 / 33.5	k=8 / 0.99 / 0.02 / 5.2	k=8 / 0.99 / 0.02 / 5.2	k=48 / 1.00 / 0.11 / 31.0
Q3–M	k=128 / 1.00 / 0.75 / 550.5	k=32 / 0.98 / 0.19 / 137.6	k=32 / 0.98 / 0.19 / 137.6	k=56 / 0.99 / 0.33 / 240.8
Q3–80B	k=128 / 1.00 / 0.39 / 275.3	k=24 / 0.98 / 0.07 / 51.6	k=24 / 0.98 / 0.07 / 51.6	k=68 / 1.00 / 0.21 / 146.3
Q3–30B	k=128 / 1.00 / 0.28 / 139.3	k=16 / 0.98 / 0.04 / 17.4	k=16 / 0.98 / 0.04 / 17.4	k=68 / 1.00 / 0.15 / 74.0

Table 5: Scenario 2 (Medical AI). Optimal configurations across nine models under $C_{\max} = 0.95$, $T_{\max} = 3600$ s, and $A_{\min} = 0.98$. Each cell shows k / accuracy / total cost (\$) / total latency (s). Model abbreviations follow Table 4. Standard deviations (omitted for brevity) were all within approximately ± 0.01.

Scenario 3: Moderate Cost and Strict Latency (Proposal Writing)

This scenario models creative or document-generation tasks such as research proposal drafting or long-form text refinement, where users can tolerate longer inference durations but still operate under limited cost budgets. We set a moderate cost constraint of \$0.65 and a latency limit of 1800 seconds, with a minimum acceptable accuracy of $A_{\min} = 0.96$. Under these deployment conditions, four optimization strategies—accuracy-maximization, cube-optimal, utopia-closest, and knee-point—are compared across nine representative language models.

Table 6 presents the results. The knee-point method again demonstrates superior efficiency, consistently achieving accuracy close to the accuracy-optimal solution while reducing total cost and latency by up to 40–70%. Accuracy-maximization delivers the best possible accuracy but often over-computes under the same cost limit, while cube- and utopia-based methods minimize resource use but underperform in accuracy-sensitive settings. Overall, knee-point optimization remains the most balanced approach for moderate-cost, latency-constrained tasks such as real-time writing or interactive drafting systems.

Model	Accuracy-Optimal	Cube-Optimal	Utopia-Closest	Knee-Point
G–5	k=28 / 0.98 / 0.61 / 75.2	k=8 / 0.97 / 0.17 / 21.5	k=8 / 0.97 / 0.17 / 21.5	k=24 / 0.98 / 0.52 / 64.5
G–5M	k=128 / 0.99 / 0.56 / 139.3	k=8 / 0.96 / 0.04 / 8.7	k=8 / 0.96 / 0.04 / 8.7	k=68 / 0.99 / 0.30 / 74.0
G–5N	k=128 / 1.00 / 0.11 / 68.8	k=16 / 0.98 / 0.01 / 8.6	k=16 / 0.98 / 0.01 / 8.6	k=56 / 0.99 / 0.05 / 30.1
N–253B	k=72 / 1.00 / 0.48 / 386.9	k=4 / 0.97 / 0.03 / 21.5	k=4 / 0.97 / 0.03 / 21.5	k=48 / 1.00 / 0.32 / 258.0
N–47B	k=52 / 1.00 / 0.18 / 111.8	k=4 / 0.97 / 0.01 / 8.6	k=4 / 0.97 / 0.01 / 8.6	k=48 / 1.00 / 0.17 / 103.2
N–9B	k=52 / 1.00 / 0.12 / 33.5	k=8 / 0.99 / 0.02 / 5.2	k=8 / 0.99 / 0.02 / 5.2	k=48 / 1.00 / 0.11 / 31.0
Q3–M	k=108 / 1.00 / 0.63 / 464.6	k=12 / 0.96 / 0.07 / 51.6	k=12 / 0.96 / 0.07 / 51.6	k=56 / 0.99 / 0.33 / 240.8
Q3–80B	k=128 / 1.00 / 0.39 / 275.3	k=12 / 0.97 / 0.04 / 25.8	k=12 / 0.97 / 0.04 / 25.8	k=68 / 1.00 / 0.21 / 146.3
Q3–30B	k=128 / 1.00 / 0.28 / 139.3	k=8 / 0.96 / 0.02 / 8.7	k=8 / 0.96 / 0.02 / 8.7	k=68 / 1.00 / 0.15 / 74.0

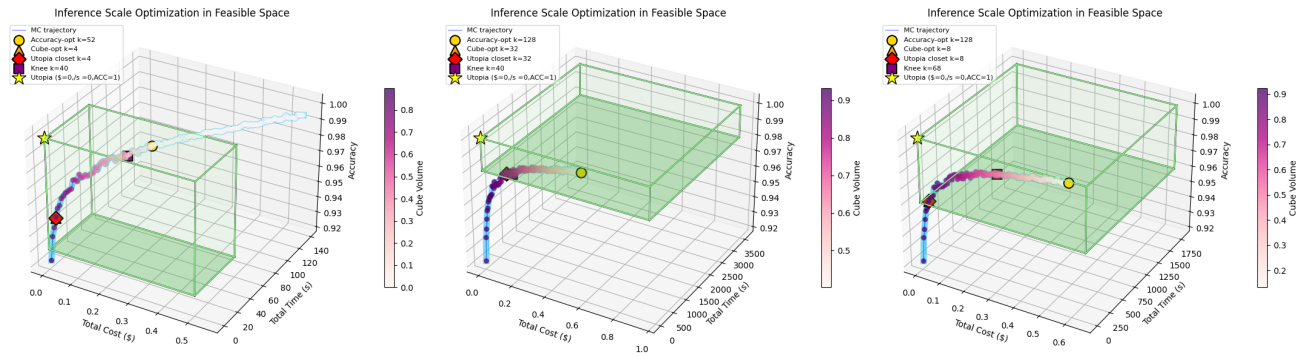
Table 6: Scenario 3 (Proposal Writing). Optimal configurations across nine models under $C_{\max} = 0.65$, $T_{\max} = 1800$ s, and $A_{\min} = 0.96$. Each cell shows k / accuracy / total cost (\$) / total latency (s). Model abbreviations follow Table 4. Standard deviations (omitted for brevity) were all within approximately ± 0.01.

Optimization Results for Inference Scaling across Models and Scenarios

This section provides 3D and 2D visualizations of the optimization process of the simulation. It includes the simulation for three scenarios across three models with large, medium, and small model size. The result for the sample GPT-5 simulation has been noted in the main body of the paper.

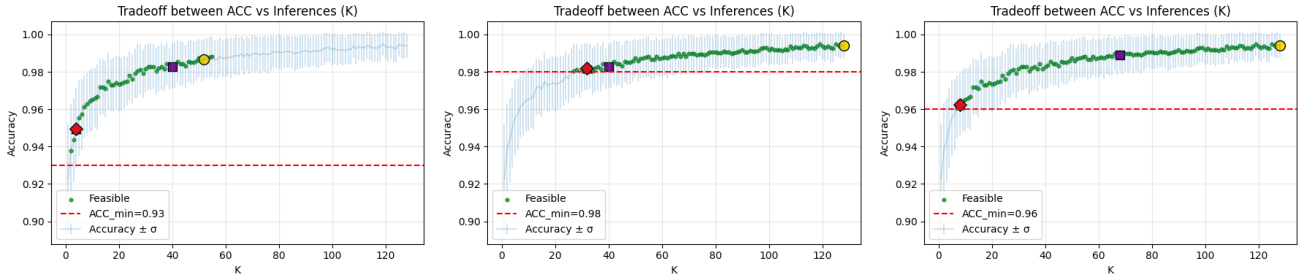
GPT5-mini

This shows the MOO from the GPT-mini case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 6: Inference-scaling optimization results for GPT5-mini across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

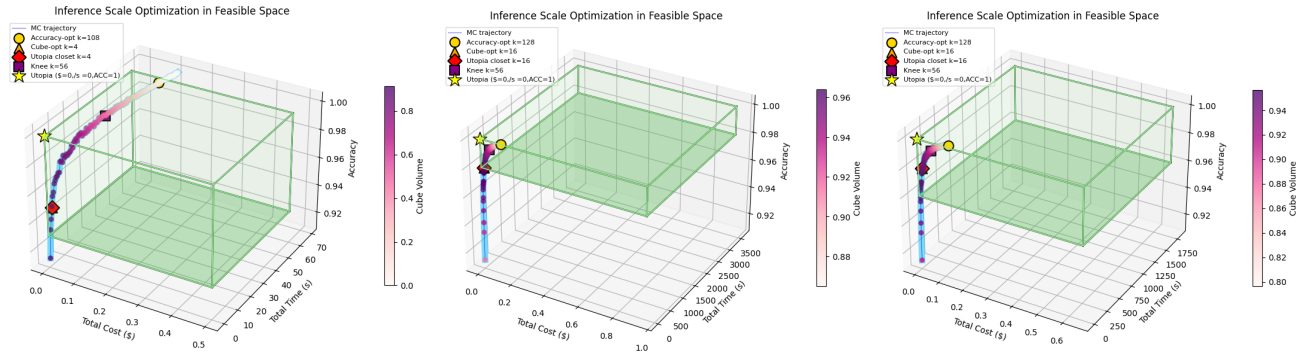


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 7: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

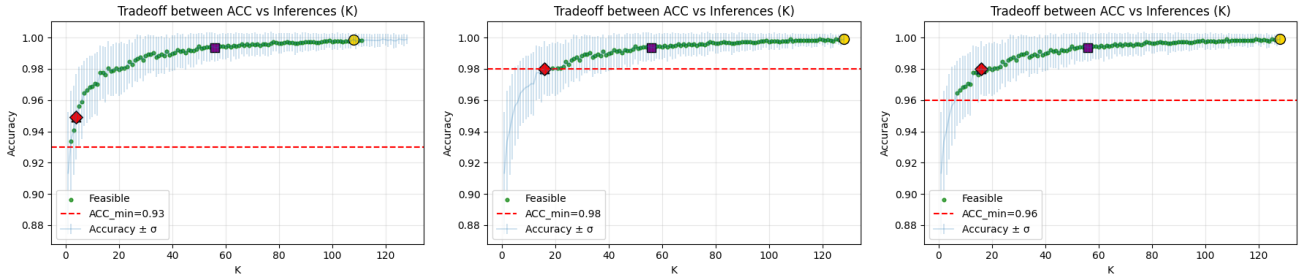
GPT5-nano

This shows the MOO from the GPT-nano case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 8: Inference-scaling optimization results for GPT5-nano across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

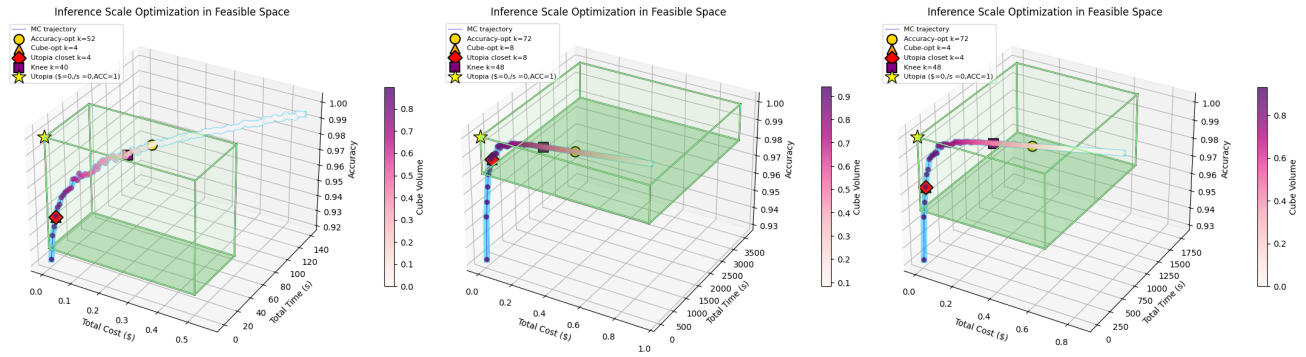


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 9: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

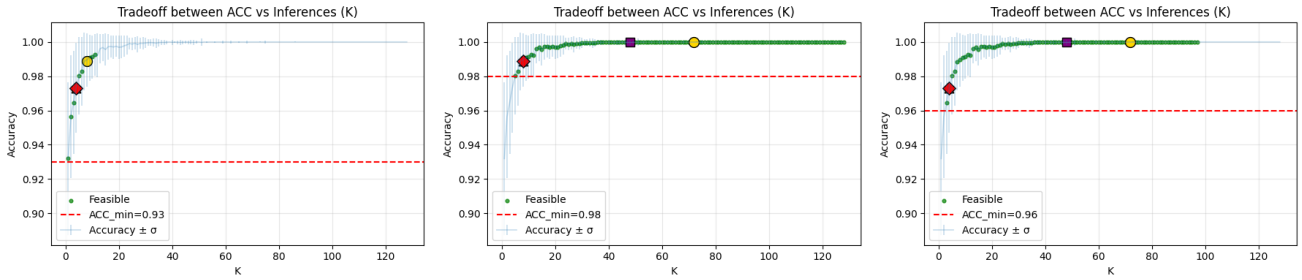
Nvidia Nemotron Ultra 253B

This shows the MOO from the Nvidia Nemotron Ultra 253B case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 10: Inference-scaling optimization results for Nvidia Nemotron Ultra 253B across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

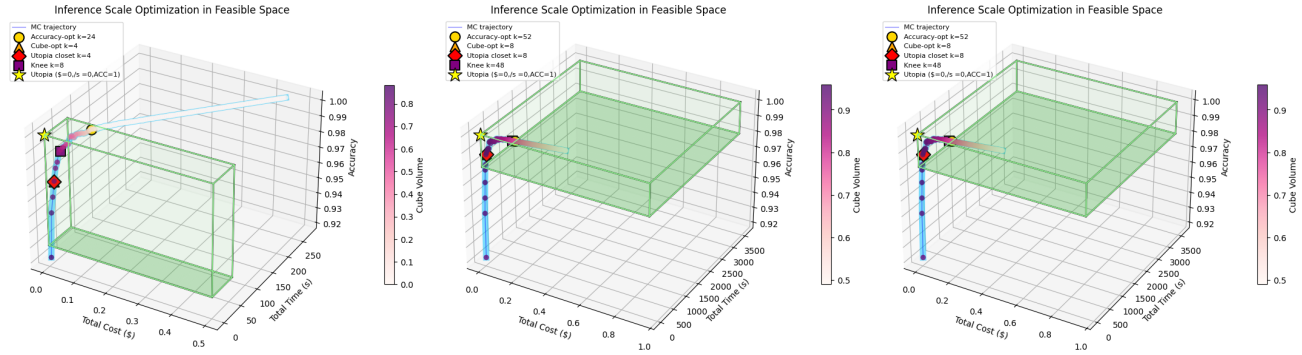


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 11: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

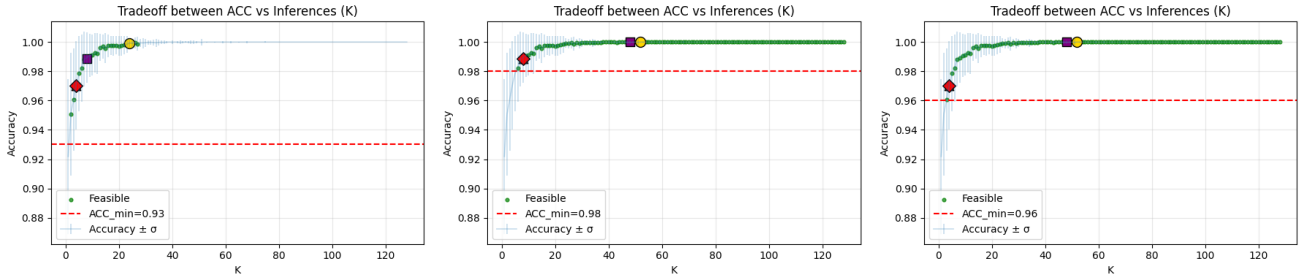
Nvidia Nemotron H 47B

This shows the MOO from the Nvidia Nemotron H 47B case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 12: Inference-scaling optimization results for Nvidia Nemotron H 47B across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

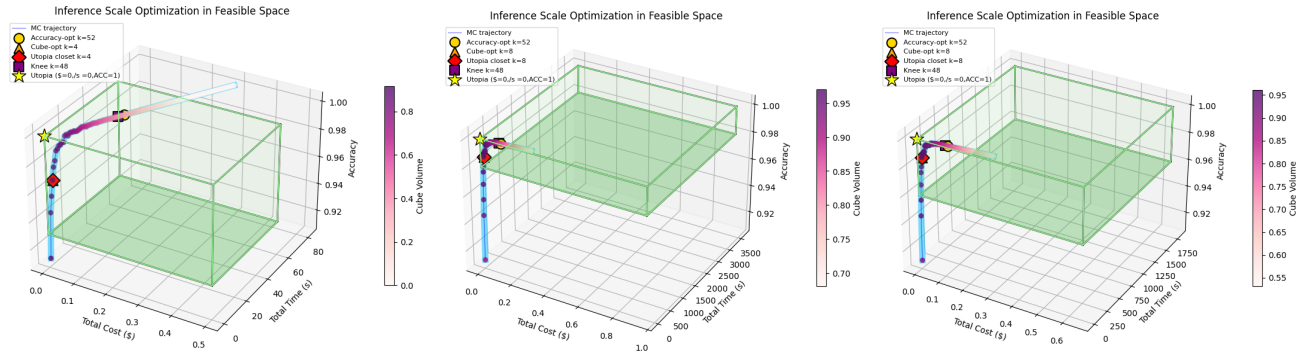


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 13: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

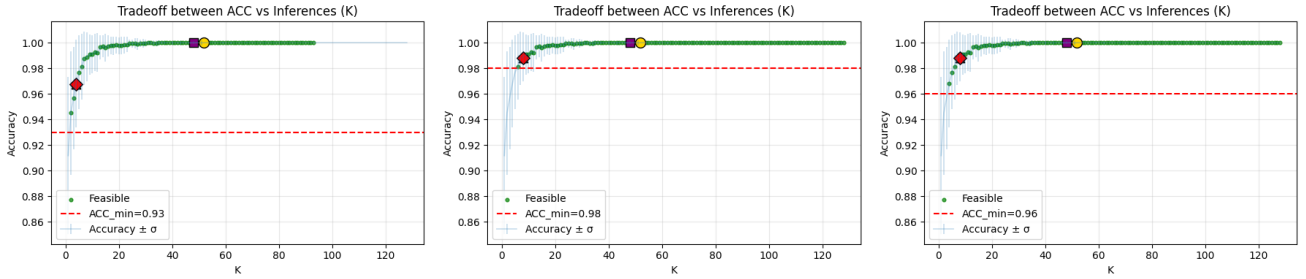
Nvidia Nemotron Nano 9B V2

This shows the MOO from the Nvidia Nemotron Nano 9B V2 case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 14: Inference-scaling optimization results for Nvidia Nemotron Nano 9B V2 across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

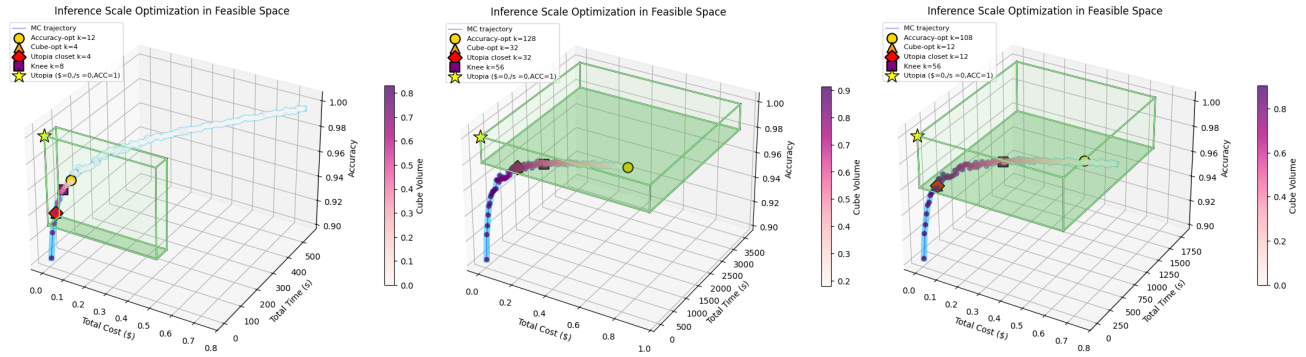


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 15: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

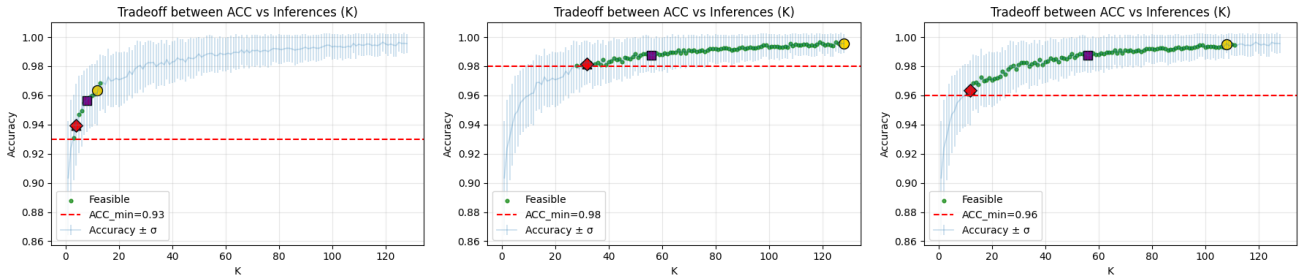
Qwen3-max

This shows the MOO from the Qwen3-max case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 16: Inference-scaling optimization results for Qwen3-max across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

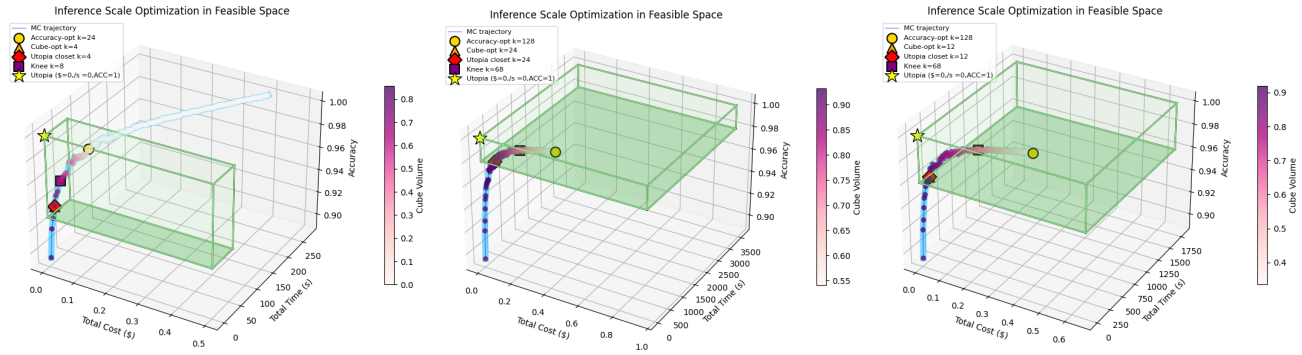


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 17: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

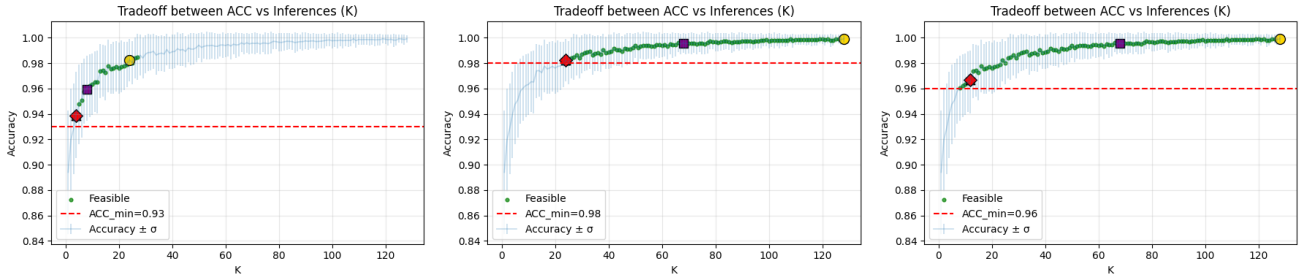
Qwen3-next 30B A3B

This shows the MOO from the Qwen3-next 80B A3B case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 18: Inference-scaling optimization results for Qwen3-next 80B A3B across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.

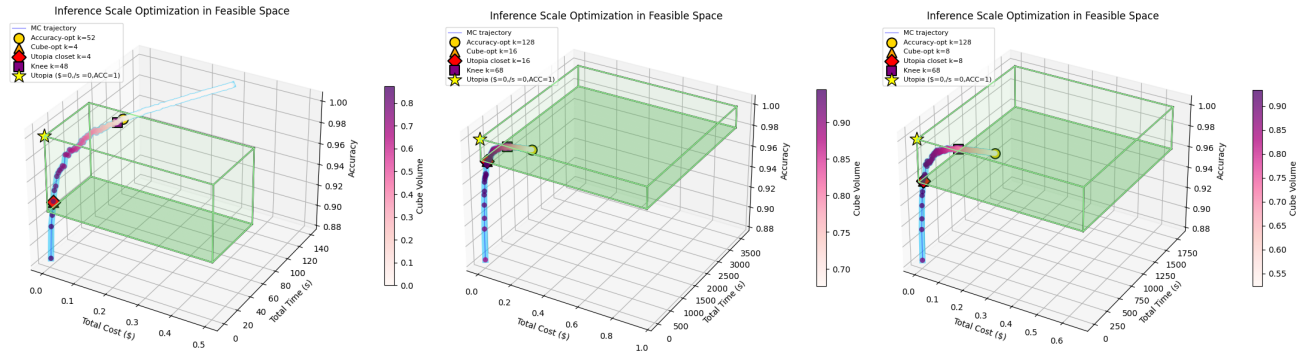


(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 19: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.

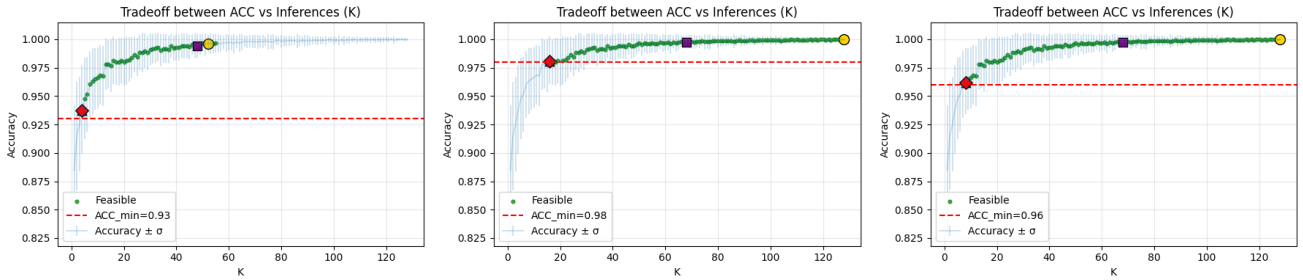
Qwen3 30B A3B

This shows the MOO from the Qwen3 30B A3B case across three scenarios.



(a) 3D MOO results for Case 1. (b) 3D MOO results for Case 2. (c) 3D MOO results for Case 3.

Figure 20: Inference-scaling optimization results for Qwen3 30B A3B across three simulated scenarios. (a–c) 3D feasible cubes in cost–time–accuracy space with constraint planes at $C_{\max}$, $T_{\max}$, and $A_{\min}$. Markers denote optimization criteria (● Accuracy-Optimal, ▲ Cube-Optimal, ♦ Utopia-Closest, ■ Knee-Point); colors indicate cube volume (larger is better), and the yellow star marks the utopia point.



(a) 2D ACC–K with optimality for Case 1. (b) 2D ACC–K with optimality for Case 2. (c) 2D ACC–K with optimality for Case 3.

Figure 21: Accuracy-inference compute (K) trade-offs across three scenarios. These show Monte Carlo means with CI = 95%; the red dashed line indicates $A_{\min}$. Distinct operational priorities yield different optimal inference scales $k^{\star}$.