COAST: FROM SUPPORT-LIMITED RETRIEVAL POLICIES TO TIMELY EXECUTION ON SHARED RAG STACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

A budgeted RAG router commits twice: it proposes a retrieval configuration from logged evidence and then decides whether that work is still timely in the live shared stack. Conflating these decisions leaves a blind spot because logging support and average resource charge do not encode the queue and accelerator contention present at execution. COAST (Coverage-constrained Offline Allocation and Serving-time Tail control) turns this boundary into an explicit propose–admit interface. Its doubly robust proposal policy masks conditionally unsupported actions and selects a validation-feasible expected-charge point; its serving layer intersects that proposal with calibrated, monotone quantile-latency and violation-risk tests, while sustained drift changes proposals without bypassing admission. In matched Stack A runs that hold the SUPPORTDR proposal policy and downstream stack fixed, adding admission changes EM/F1 from 51.2/65.3 to 51.3/65.4, while P99 moves from 304 to 252 ms and retrieval-budget violations from 1.4% to 0.42%. Across 22,328 Natural Questions, HotpotQA, and TriviaQA requests and five repetitions, full COAST reaches 51.4/65.6 EM/F1—paired differences of +0.3 [0.0,0.6]/+0.4 [0.1,0.7] against DENSE+RR—while reducing P95/P99 from 278/412 to 156/248 ms, limiting violations to 0.4%, and staying within the predeclared request-attributed service-charge budget. At 2× load, its P95 increases by 11.2% versus 38.4% for DENSE+RR; after replacing both generator and reranker, adding admission to SUPPORTDR moves P95/P99 from 221/338 to 194/304 ms, and full COAST reaches 188/288 ms. These results make live admissibility a distinct design target for RAG: logged support determines what can be proposed responsibly, whereas request-state risk determines what should execute now.

1 INTRODUCTION

A retrieval action can be statistically supported, answer-useful, and already too late to execute. A router may rank a deep dense-and-rerank path while queues are short; before that work reaches the shared accelerator, a burst can make the same path dominate the request tail. The evidence available at these two moments is different. Logged propensities and outcomes delimit what an offline policy can learn, whereas queue depths, recent stage times, utilization, and offered load determine whether the proposed work fits the request that is arriving now.

This distinction is visible without changing the RAG stack. On Stack A, SUPPORTDR proposes from the same 25-action catalog with 51.2/65.3 EM/F1 and 304 ms P99. Attaching the serving-state admission rule leaves the proposal policy, observations, models, hardware, and decoding fixed; the resulting SUPPORTDR+CTRL system records 51.3/65.4 EM/F1, 252 ms P99, and 0.42% rather than 1.4% retrieval-budget violations. Admission deliberately changes the executed action mixture through veto and renormalization. The paper's central estimand is this complete matched system contrast: what changes when a fixed supported proposal rule is followed by a live admissibility rule?

Existing adaptive RAG methods primarily decide whether, when, or how deeply to retrieve from query and retrieval evidence (Asai et al., 2024; Jeong et al., 2024; Jiang et al., 2023; Trivedi et al., 2023; Ray et al., 2025; Raza et al., 2026). Logged-policy learning supplies estimators for answer

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

allocation under partial action coverage (Dudík et al., 2011; Swaminathan & Joachims, 2015; Jiang & Li, 2016), while serving systems optimize scheduling, placement, and batching (Dean & Barroso, 2013; Jiang et al., 2025). These lines leave an operational boundary unmeasured: the action justified by logged support need not be the action admissible under the service state in which it executes. Measuring that boundary requires a proposal policy that can be held fixed, an admission operator that exposes its vetoes, and complete-request outcomes rather than a latency surrogate alone.

COAST implements this propose–admit decomposition. Offline, a doubly robust objective ranks the 25 sparse, dense, cascade, depth, and reranking configurations after a conditional propensity-and-ESS mask; validation selects a checkpoint within a predeclared expected service-charge budget. Online, a 0.95-quantile latency model and calibrated violation-risk model are projected to be monotone in depth and reranking, intersected with the supported proposal, and renormalized. A hysteretic drift monitor can replace the proposal distribution, but every fallback proposal still passes through the same admission operator. This interface makes proposal support, candidate risk, veto behavior, executed violations, and end-to-end P95/P99 jointly observable.

Five independent repetitions each serve all 22,328 Natural Questions, HotpotQA, and TriviaQA requests. Full COAST reaches 51.4/65.6 EM/F1, with paired differences of +0.3 [0.0,0.6]/+0.4 [0.1,0.7] relative to DENSE+RR, while moving P95/P99 from 278/412 to 156/248 ms and violations from 10.6% to 0.4%. Its request-attributed service charge is $1.18/1K versus $1.52/1K. At doubled load, COAST P95 grows by 11.2% rather than 38.4% for DENSE+RR. Replacing both generator and reranker preserves the matched direction: at 2× load, adding admission to SUPPORTDR moves P95/P99 from 221/338 to 194/304 ms, while full COAST reaches 188/288 ms.

Contributions. **(1) Decision object.** A formal executed-policy operator and matched admission contrast distinguish support-limited proposal learning from request-state execution. **(2) Control composition.** COAST combines support-masked doubly robust proposals, monotone quantile-risk admission, and drift-conditioned fallback behind one auditable interface. **(3) System evidence.** Matched interventions expose the admission effect; the complete system occupies a joint quality–tail–charge region across load, time-window, query-exclusion, and second-stack tests.

2 RELATED WORK

Adaptive retrieval chooses work; COAST also admits execution. RAG, REALM, Atlas, RETRO, and Re2G establish retrieval-conditioned generation and retrieve–rerank pipelines (Lewis et al., 2020; Guu et al., 2020; Izacard et al., 2023; Borgeaud et al., 2022; Glass et al., 2022). Self-RAG, Adaptive-RAG, FLARE, interleaved retrieval, and RECOMP adapt retrieval frequency, query-conditioned depth, or context construction for answer utility (Asai et al., 2024; Jeong et al., 2024; Jiang et al., 2023; Trivedi et al., 2023; Xu et al., 2024). METIS jointly selects chunk and synthesis configurations with resource-aware scheduling (Ray et al., 2025); SAGE imitates an oracle depth from first-pass retrieval features for a P95 objective (Raza et al., 2026). These methods make the acquisition choice itself adaptive. COAST exposes the subsequent boundary: a supported 25-way sparse/dense/cascade-and-reranking proposal is intersected with current queue- and accelerator-conditioned admissibility. Holding the proposal rule fixed then makes the system effect of this boundary directly measurable.

Learning from logged actions. Doubly robust estimation and counterfactual risk minimization provide objectives for policies learned from bandit feedback (Dudík et al., 2011; Swaminathan & Joachims, 2015; Jiang & Li, 2016). Their positivity requirement is a property of the logging distribution, not of the service state in which a chosen action later runs. COAST uses propensity and effective-sample-size masks to delimit the offline action set, then measures the separate online effect through calibrated candidate risk, vetoes, executed violations, and complete-request tails. MESS+ addresses a complementary constrained-routing problem: virtual queues optimize cost while guaranteeing long-run satisfaction rates across LLM choices (Woisetschläger et al., 2025). Its action is a model endpoint and its constraint is aggregate answer satisfaction; COAST chooses retrieval configurations within one generator and empirically controls request-state latency risk.

Serving optimization and the new estimand. Dense retrieval and reranking build on DPR, ANCE, ColBERT, cross-encoders, and listwise rerankers (Karpukhin et al., 2020; Xiong et al., 2021; Khattab & Zaharia, 2020; Nogueira & Cho, 2019; Lin et al., 2022; Ma et al., 2023). Serving work reduces

tail delay through early exit, confidence thresholds, redundancy, bounded queues, and adaptive scheduling (Bolukbasi et al., 2017; Xin et al., 2020; Schuster et al., 2022; Dean & Barroso, 2013; Vulimiri et al., 2013; Suresh et al., 2015). RAGO optimizes placement, resource allocation, and batching from a RAG workload abstraction (Jiang et al., 2025); those system-configuration decisions alter the execution substrate that our matched-stack intervention holds fixed. Consequently, METIS and RAGO are adjacent system optimizers, and MESS+ operates over a different action and guarantee. Our direct adaptive comparisons instead preserve the observations, 25 actions, stack, and tuning budget: SUPERVISEDROUTE captures offline imitation, SUPPORTDR adds logged-action correction, and SUPPORTDR+CTRL changes only serving-state admission. This ladder tests the paper's specific claim without attributing scheduler, model-choice, or hardware-placement changes to the retrieval policy. Appendix Table 3 compares adaptive object, live observation, learning signal, and measured target. The SUPPORTDR/SUPPORTDR+CTRL pair then measures the complete-request change induced by inserting admissibility after a fixed support-limited proposal, while full COAST evaluates the resulting quality–tail–charge point.

3 PROBLEM SETTING

A budgeted RAG router crosses two decision boundaries: logged outcomes determine which retrieval work can be proposed, and the live shared-stack state determines which proposal is timely enough to execute. We model their composition and the matched change produced by inserting the second boundary.

3.1 OPERATIONAL ACTION SPACE

For each query q, a retrieval configuration is an action $a \in \mathcal{A}$:

$$a = (r, k, \rho), \tag{1}$$

where the retriever family is indexed by $r \in \{\text{NORET}, \text{SPARSE}, \text{DENSE}, \text{CASCADE}\}$, the retrieved-document count by $k \in \{0, 2, 4, 8, 16\}$, and the decision to apply a cross-encoder reranker by $\rho \in \{0, 1\}$. The valid catalog comprises one NORET action, $(k{=}0, \rho{=}0)$, plus $3 \times 4 \times 2 = 24$ retrieval actions: the other three families each combine four positive depths with reranking on/off. Thus $|\mathcal{A}| = 25$ rather than the unconstrained Cartesian product of 40 tuples.

The serving interface scores all 25 actions. Offline eligibility is support-aware: action a may receive policy mass at state x only when its conditional logging propensity and validation effective sample size meet the thresholds in Section 5.1. Regular traffic concentrates on six deployed pipelines; stratified controller-safe exploration supplies coverage for the remaining catalog while preserving known propensities in every eligible stratum.

3.2 BUDGETS AND METRICS

$T(a, q, s)$ measures end-to-end latency for query q executed with action a in service state s; $C(a, q, s)$ is the corresponding request-attributed service charge, computed using the fixed resource-price schedule in Appendix Table 21. We evaluate end-to-end targets of 200 ms at P95 and 300 ms at P99, an aggregate expected-charge budget, and a 75 ms retrieval-plus-reranking sub-budget. The remaining 125 ms is planning headroom for routing, generation, and other request stages at the P95 operating point. The identity $75 + 125 = 200$ defines the allocation used to tune admission; neither P95 nor P99 follows algebraically from that allocation, so both are measured over complete requests.

With fixed greedy decoding, EM and token-level F1 are macro-averaged over independently trained and served repetitions. We define the high-quality region by paired non-inferiority to DENSE+RR: on the standard split, the lower 95% confidence bound for both quality deltas must exceed -0.5 points. A separately refitted query-hash-disjoint evaluation tests whether that conclusion survives stricter exclusion; Appendix A.2 reports its sensitivity at 0.25 and 0.5 points.

3.3 OFFLINE ALLOCATION AND SERVING-STATE ADMISSION

The offline problem constrains a proposal distribution over the logged context distribution. It seeks high doubly robust answer value inside conditional action support and under aggregate expected

service charge:

$$\max_{\pi \in \Pi_{\text{sup}}} \quad \widehat{V}_{\text{DR}}(\pi) \tag{2}$$

$$\text{s.t.} \quad \widehat{\mathbb{E}}_{\pi}[C] \le B_C, \qquad \pi(a \mid x) = 0 \text{ for } a \notin \mathcal{A}_{\text{sup}}(x). \tag{3}$$

We predeclare $B_C = \$1.20/1\text{K}$ queries for validation-time operating-point selection; the selected nominal point has charge \$1.18/1K. The charge sums request-attributed stage usage, including concurrent requests separately, and is the common policy-selection metric rather than physical device busy time.

Serving-time admission asks whether a proposed action should execute for the current request. Let $g(x, a) \in \{0, 1\}$ be an admissibility rule and π^g the executed distribution:

$$\pi^g(a \mid x) = \frac{\pi(a \mid x) g(x, a)}{\sum_{a'} \pi(a' \mid x) g(x, a')}, \tag{4}$$

with NoRet used when the denominator is zero. For any complete-request metric M, the matched admission contrast is

$$\Delta_g M(\pi) = M(\pi^g) - M(\pi^{\text{pass}}), \tag{5}$$

where $g_{\text{pass}}(x, a) = 1$. Equation 5 holds the proposal rule and execution stack fixed while allowing admission to change the executed action mixture, as intended. We report this contrast through EM/F1, charge, P95/P99, veto rate v_{veto}, shadow false-safe rate $p_{\text{false-safe}}$, and executed violation rate p_{viol}. The gate predicts retrieval-plus-reranking risk; complete-request measurements determine the end-to-end tails.

4 System Design

COAST is a control layer in front of a standard RAG stack: the application issues `Answer(query, budget_ms)`, the router selects a retrieval action, and the downstream stack executes sparse or dense retrieval, optional reranking, and LLM generation.

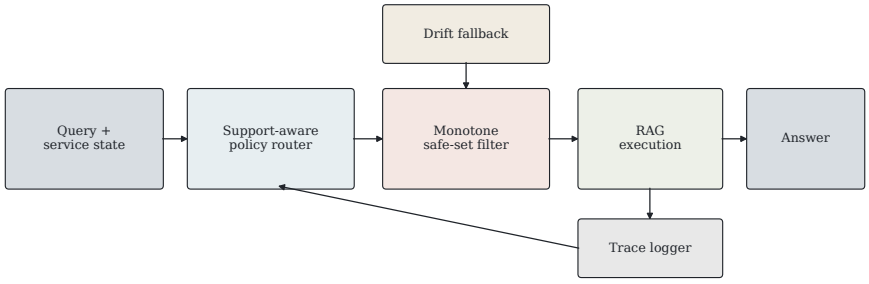

Figure 1: COAST separates supported action allocation from service-state admission. The trace logger updates calibration and drift statistics; sustained drift changes the candidate distribution while retaining the admission test.

Serving stack. Sparse retrieval uses Elasticsearch 8.12 with BM25 scoring. Dense retrieval uses FAISS 1.7.4 with HNSW (M=32, efSearch=128) over 1024-dimensional e5-large-v2 embeddings stored in fp16. Stack A uses bge-reranker-large and Llama-3.1-8B-Instruct served by vLLM 0.4.2. Stack B changes both to Mistral-7B-Instruct-v0.3 and cross-encoder/ms-marco-MiniLM-L-12-v2. Each stack is retrained independently while preserving the corpus, 25-action catalog, SLOs, and hardware class.

Reranking and generation share 2× NVIDIA A100 80GB: vLLM uses tensor parallelism 2, and the reranker runs on a separate CUDA stream on GPU 0. This colocation creates the contention that the controller must observe. Retrieval and routing run on AMD EPYC 9654 with 512 GB RAM. Routing adds 0.19 ms P95 overhead at 160 QPS and 0.31 ms at 320 QPS. Every comparator uses the same stack instance, indexes, model checkpoints, decoding rule, and latency accounting; only the action policy changes.

Features and request accounting. The router consumes 18 pre-action features in five groups: request form (query length and punctuation statistics), semantic difficulty (three lexical-rarity features and four embedding projections), queue state (sparse, dense, reranking, and generation depths), recent service state (exponentially weighted retrieval and generation latency plus GPU utilization), and load multiplier. End-to-end latency begins when the request enters the router queue and ends when the final token is emitted. Retrieval latency includes queueing, search, context fetch, and reranking.

Generation uses greedy decoding, an average input length of 1842 tokens, and a common 192-token cap; capped requests remain in EM/F1, with method-wise counts in Appendix Table 18. Generation is never shortened by the controller. Because retrieval depth changes context and prefill work, end-to-end tails retain action-conditioned generation; the 75 ms predictor governs only retrieval plus reranking.

Logging and support. Each request records query hash, de-identified features, action, known propensity, latency decomposition, service charge, and controller decision; raw query text is excluded. The regular 99.5% stream mixes NoRet, Sparse, Dense, Dense+RR, Cascade, and Cascade+RR with weights 5/20/30/25/15/5. The remaining 0.5% is concentrated in predeclared feature strata, where the gate randomizes over controller-safe alternatives with conditional propensity at least 0.01. Rare open strata keep aggregate exploration low while meeting the eligibility threshold.

Matched tuning. Every adaptive comparator uses the same stack, split, pre-action observation boundary, decoding, seeds, and validation data. Their model classes reflect the comparison: HeuristicGate is a restricted threshold rule, SupervisedRoute is a multiclass LightGBM imitator, and the SupportDR variants and COAST use the same two-layer proposal network. Learned variants receive 40 validation trials. SupportDR uses the clipped doubly robust value, support mask, and charge penalty in Equation 8, omitting the entropy-concentration term; SupportDR+Ctrl attaches the identical monotone safe-set controller without COAST's drift-conditioned proposal distribution. This pair holds the proposal policy fixed and changes only the execution-time admissibility operator. Comparison with full COAST measures the bundled addition of entropy concentration and drift-conditioned proposals.

5 Method

COAST forms a support-limited proposal, removes actions whose current predicted retrieval risk exceeds admission thresholds, and samples from the remaining mass. Drift changes the proposal, not the admission rule, exposing both distributions for Equation 5.

5.1 Offline Ranking Under Support Limits

Each served request contributes the logged record $(x_i, a_i, t_i, c_i, \mu_i)$: its 18-dimensional feature vector is $x_i = \phi(q_i, s_i)$, the action executed for retrieval is a_i, measured latency is t_i, request-attributed service charge is c_i, and the randomized logging policy's exact assignment probability is $\mu_i = \mu(a_i \mid x_i)$. Quality labels occur only in the reference-labeled replay slice, so proposal and quality models use labeled records while latency and charge predictors use all traces. Collection day and request identity define 70%/15%/15% event partitions for fitting, selection, and serving evaluation. A stricter quality refit removes every evaluation query hash from labeled policy and quality tuples in both training and validation; latency, risk, and charge models retain the event-disjoint partition.

Almost all regular traffic mass falls on the six deployed mixture actions; observations of the other actions come from safe exploration at a low rate within predeclared feature strata. Within each action–stratum cell, inverse-propensity weights $w_j = 1/\max(\mu_j, \epsilon)$ define $\mathrm{ESS} = (\sum_j w_j)^2 / \sum_j w_j^2$ and normalized support $\mathrm{ESS}/n_{\mathrm{cell}}$. We include a in $\mathcal{A}_{\mathrm{sup}}(x)$ only when its stratum-conditional propensity satisfies $\mu(a \mid x) \geq \epsilon$ and validation $\mathrm{ESS}/n_{\mathrm{cell}} \geq 0.20$; logits outside this set are masked before normalization. All support claims refer to this declared conditional rule and the executed-policy diagnostics in Appendix Table 15. To score such policies from logged data, we use a doubly robust estimator with a LightGBM quality model (Ke et al., 2017; Dudík et al., 2011; Swaminathan &

Joachims, 2015; Jiang & Li, 2016):

$$\widehat{V}_{\mathrm{DR}}(\pi_\theta) = \frac{1}{n}\sum_{i=1}^{n}\Big[\sum_{a\in\mathcal{A}}\pi_\theta(a \mid x_i)\,\hat{Q}(x_i,a) + \frac{\pi_\theta(a_i \mid x_i)}{\tilde{\mu}_i}\left(y_i - \hat{Q}(x_i,a_i)\right)\Big], \tag{6}$$

where $y_i = (\mathrm{EM}_i + \mathrm{F1}_i)/2$ with both terms scaled to $[0,1]$, and $\tilde{\mu}_i = \max(\mu_i, \epsilon)$ with $\epsilon = 0.01$. Identification assumes consistency, conditional exchangeability of logged actions given the 18 pre-action features, and positivity within $\mathcal{A}_{\mathrm{sup}}(x)$. Inside that boundary, doubly robust consistency requires either the logged propensity or quality model to be correct; clipping and the support mask provide finite-sample stability while confining allocation to covered actions. The candidate policy is a two-layer, 128-unit MLP. Masked logits produce

$$\pi_\theta(a \mid x) = \frac{\mathbb{I}[a \in \mathcal{A}_{\mathrm{sup}}(x)]\exp z_\theta(x,a)}{\sum_{a'}\mathbb{I}[a' \in \mathcal{A}_{\mathrm{sup}}(x)]\exp z_\theta(x,a')}. \tag{7}$$

We optimize the complete offline objective

$$\min_\theta -\widehat{V}_{\mathrm{DR}}(\pi_\theta) + \lambda_C\left[\widehat{\mathbb{E}}_{\pi_\theta}\hat{C}(x,a) - B_C\right]_+ + \lambda_H\,\widehat{\mathbb{E}}[H(\pi_\theta(\cdot \mid x))], \tag{8}$$

where $\hat{C}$ is a LightGBM per-request service-charge predictor and B_C is the predeclared budget in Section 3.3. Equation 8 is the Lagrangian training objective; operating-point selection then discards validation checkpoints whose estimated expected charge exceeds B_C or whose mass escapes the support mask. Among the remaining checkpoints, validation DR value selects λ_C, λ_H, and temperature, while propensity clipping remains fixed at $\epsilon = 0.01$. Adam trains for at most 50 epochs with early stopping. Five seeds independently fit the quality model, charge model, and policy. The logged μ_i values, rather than an estimated propensity model, enter Equation 6. Appendix Table 15 reports executed support mass, global ESS, and maximum weights rather than inferring overlap from aggregate value alone.

5.2 State-Conditioned Monotone Admission

The offline objective regulates aggregate charge over logged states; the controller instead admits each request under its current queue and accelerator state. Full traces train a LightGBM 0.95-quantile regressor $\hat{T}_{0.95}(x,a)$ with pinball loss and a binary violation-risk model $\hat{p}_v(x,a) = P(T_{\mathrm{ret}} > 75 \mid x,a)$ with log loss. We define $\hat{T}_{\mathrm{ret}}(x,a) = \hat{T}_{0.95}(x,a) + m$, select m on validation, and calibrate $\hat{p}_v$ by isotonic regression on a query-hash and day-blocked split.

Because deeper retrieval and reranking cannot reduce required work, we project both predictions onto the action lattice so they are nondecreasing in k and ρ. The validated safety margin is 4.6 ms, and the admissible set is

$$\mathcal{A}_{\mathrm{safe}}(x) = \left\{a \in \mathcal{A}_{\mathrm{sup}}(x) : \hat{T}_{\mathrm{ret}}(x,a) \le 75,\ \hat{p}_v(x,a) \le 0.05\right\}. \tag{9}$$

Intersecting the supported allocation with this predicted-admissible set yields the executed policy. We renormalize π_θ over the intersection and sample using a request-seeded random draw; an empty set executes NoRet. The 75 ms gate models retrieval plus reranking. Retrieval depth can also alter context assembly and generation prefill, so complete-request measurement supplies the empirical end-to-end P95/P99. Algorithm 1 gives the serving procedure.

Matched variants expose complementary system contrasts. Controller-only replaces the complete doubly robust proposal objective with the plug-in score $\hat{Q}(x,a) - \lambda_C\hat{C}(x,a)$ before applying the same calibrated set and fallback. The point-prediction gate retains COAST's proposal ranking but jointly substitutes a LightGBM mean-latency threshold $\hat{T}_{\mathrm{mean}}(x,a) \le 75$ for quantile prediction, calibration, margin, binary-risk admission, and monotone projection; its raw binary score supports the ECE diagnostic. The no-controller ablation retains proposal ranking and the fallback monitor while bypassing state-conditioned filtering. SupportDR uses the same support mask, policy capacity, and charge budget but optimizes Equation 8 without the entropy-concentration term; SupportDR+Ctrl applies the identical admission filter without drift-conditioned proposals. This pair changes only the admissibility operator and supplies the matched system-level contrast. Full COAST additionally changes entropy concentration and fallback proposals together, so comparisons involving Controller-only, the point-prediction gate, or full COAST are reported as bundled contrasts.

5.3 DRIFT FALLBACK

Fallback uses the final non-alert training day as its reference distribution. Each scalar feature is discretized into 32 equiprobable reference bins; a 0.5 pseudocount per bin smooths both histograms before their marginal KL values are summed. A continuously running monitor updates once per minute from a rolling window of at least 10,000 requests and tracks retrieval-budget violations over the same requests. KL above 0.15 or violations above 1.5% activate fallback. Hysteresis resets normal routing only after KL remains below 0.10 and violations below 1.0% for 20 minutes.

Fallback proposes SPARSE with $k = 4$ for queries above the 90th training-percentile lexical-rarity score and NORET otherwise. The proposal passes through the same monotone latency filter, with NORET as the terminal action. Thus drift changes candidate priorities while retaining the same admission calculation.

6 EXPERIMENTS

The evaluation follows the claim chain in Section 1: insert admission after a fixed proposal policy, evaluate the complete quality–charge–tail point, and then perturb load, traffic time, generator, and reranker.

6.1 EXPERIMENTAL SETUP

For LLM serving and reranking, we use 2× NVIDIA A100 80GB (80 GB HBM each); retrieval and policy routing use AMD EPYC 9654, backed by 512 GB DDR5 RAM and NVMe storage comprising 2× 3.84 TB SSDs. Every machine uses Ubuntu 22.04.4 LTS; its software environment includes CUDA 12.4 and the NVIDIA 550.90.07 driver. For generation, vLLM 0.4.2 (Kwon et al., 2023) serves Llama-3.1-8B-Instruct; this configuration represents modern systems for LLM serving, including Orca (Yu et al., 2022). Dense retrieval follows the DPR/ANCE line (Karpukhin et al., 2020; Xiong et al., 2021) and is implemented with FAISS plus HNSW (Johnson et al., 2021; Malkov & Yashunin, 2020), while representative ANN and vector-store alternatives include DiskANN, Filtered-DiskANN, Milvus, and VBase (Subramanya et al., 2019; Gollapudi et al., 2023; Wang et al., 2021; Zhang et al., 2023). We rerank with bge-reranker-large; its closest conceptual connection is to passage reranking, spanning BERT cross-encoders through LLM rerankers (Nogueira & Cho, 2019; Lin et al., 2022; Ma et al., 2023).

The corpus contains 120 M documents from the 2023-12 Wikipedia dump and a 2024-01 Common Crawl snapshot. Evaluation contains 3,610 Natural Questions (Kwiatkowski et al., 2019), 7,405 HotpotQA (Yang et al., 2018), and 11,313 TriviaQA (Joshi et al., 2017) requests, totaling 22,328. EM/F1 are macro-averaged across datasets; each of five independent seeds processes the complete request set. Nominal traffic is open-loop Poisson at 160 QPS over persistent HTTP/2 connections with no client think time and at most 512 in-flight requests. Query/result caches and cross-request KV-prefix reuse are disabled; indexes and OS page caches are warmed for 5 minutes and then retained. The burst slice uses a Markov-modulated arrival process with 20-second bursts at 1.8× rate while preserving the nominal long-run mean. A 2 s timeout is charged to end-to-end latency and enters the retrieval-budget count only when retrieval plus reranking exceeds 75 ms.

Run-level 95% t intervals summarize P95, P99, and service charge across seeds. Paired EM/F1 intervals use 10,000 query-hash bootstrap replicates stratified by dataset. The charge aggregates request-attributed CPU/GPU stage usage and physical storage operations under the fixed schedule in Appendix Table 21. A retrieval-budget violation occurs when the finally executed action's realized retrieval-plus-reranking latency exceeds 75 ms. During evaluation, a capacity-isolated shadow replica replays each pre-filter top candidate from the same queue-state snapshot. A false-safe candidate is predicted safe but violates the budget in shadow replay; a false veto is rejected despite remaining within budget. Shadow work is excluded from serving load and charge. Raw decision counts aggregate five complete repetitions with 111,640 requests as denominator. For the five pre-filter risk bins in Appendix Table 8, pooled ECE is $\sum_b (n_b/N)|\bar{p}_b - \bar{y}_b|$.

The fixed routes are NORET; SPARSE, DENSE, and CASCADE at $k = 8$; their reranked counterparts DENSE+RR and CASCADE+RR; and the best charge-feasible point from the complete 25-action static grid. CASCADE fuses sparse and dense candidates before optional reranking. HEURISTICGATE

is a validation-tuned threshold rule over lexical rarity, reranker queue, and load that chooses among NoRet, Sparse, and Dense+RR. SupervisedRoute is a multiclass LightGBM proxy for the offline-imitation decision, trained on the observed action maximizing labeled quality minus a validation-selected charge penalty without propensity correction; SAGE is compared at the interface level in Section 2. SupportDR uses the two-layer proposal network with propensity correction and support masking, and SupportDR+Ctrl changes only by attaching the identical admission operator. Full COAST and its no-controller ablation complete the comparison. All rows share the serving stack, event partitions, seeds, and decoding; adaptive methods share the pre-action observation boundary, while their model classes and action restrictions are stated above.

6.2 RQ1: What Changes When Admission Follows a Fixed Proposal?

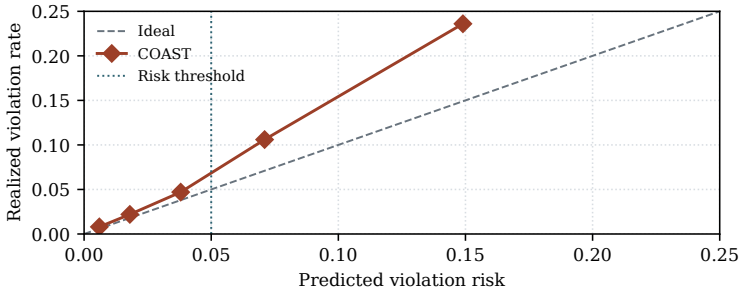

Figure 2: Pre-filter risk calibration at 1× load. Mean predicted risk in each bin is compared with the violation rate of capacity-isolated shadow replays; the vertical line is the 0.05 admission threshold.

Table 1: Admission variants at 1× load. ECE uses pre-filter candidate risk; violations and P99 use executed requests.

Variant	ECE	Viol.	P99
Fixed Sparse	–	0.14%	276
Controller-only	0.074	0.5%	269
Point gate	0.118	1.8%	308
No controller	–	2.7%	287
COAST	0.012	0.4%	248

The SupportDR pair holds its proposal fixed: admission changes EM/F1 by +0.1/+0.1 points while moving P99 from 304 to 252 ms and violations from 1.4% to 0.42%. In the full-policy pair, bypassing admission keeps EM/F1 at 51.4/65.6 but moves P95 from 156 to 194 ms and violations from 0.4% to 2.7%. The point gate records 0.118 ECE, 1.8% violations, and 308 ms P99, versus 0.012, 0.4%, and 248 ms for the complete rule.

Across five repetitions, COAST vetoes 12,615 top proposals with 4.2% false vetoes; 780 requests activate fallback. At 1.5× load, its P99 is 268 ms versus 314/331 ms for point/no-controller variants. Appendix Figure 3 and Tables 8–10 give the full diagnostics.

6.3 RQ2: Can One Point Satisfy Quality, Charge, and Both Tails?

Table 2 aligns every policy's quality, charge, violations, and tails. Relative to Dense+RR, COAST changes EM/F1 by +0.3 [0.0,0.6]/+0.4 [0.1,0.7], exceeding the predeclared -0.5 margins, while reaching 156/248 ms P95/P99 and \$1.18/1K charge. This is a 44%/40% tail and 22% charge reduction; versus SupervisedRoute, \$0.03/1K accompanies 42/64 ms lower tails.

After evaluation hashes are excluded from labeled policy and quality-model training, the retained 2,376,148 tuples have 0 overlap. This allocator refit records 51.2/65.3 EM/F1 versus 51.1/65.2 for fixed Dense+RR; both lower bounds exceed -0.5 ($p = 0.004/0.002$ for EM/F1). With event-

Table 2: Complete standard-split comparison at 160 QPS. EM/F1 and all serving columns in a row come from the same fitted policy; brackets are run-level 95% CIs, charge is request-attributed, and violation counts use all 111,640 requests.

Method	EM/F1	P95 [CI]	P99 [CI]	Viol. count/rate	Charge/1K [CI]
NoRet	42.3/55.8	168 [164,172]	254 [247,262]	0/0.0%	0.92 [0.90,0.94]
Sparse	47.2/61.4	182 [177,188]	276 [269,284]	155/0.14%	0.98 [0.96,1.00]
Dense	49.8/63.6	224 [217,231]	342 [332,353]	6,240/5.6%	1.14 [1.11,1.17]
Dense+RR	51.1/65.2	278 [268,287]	412 [398,426]	11,805/10.6%	1.52 [1.48,1.56]
Cascade	50.4/64.5	238 [230,246]	358 [347,371]	8,285/7.4%	1.21 [1.18,1.24]
Cascade+RR	51.2/65.3	246 [238,254]	374 [362,386]	9,575/8.6%	1.34 [1.31,1.37]
Best Static 25	50.7/64.9	206 [198,214]	318 [306,329]	1,490/1.3%	1.16 [1.13,1.19]
HeuristicGate	50.6/64.7	212 [205,219]	328 [317,339]	890/0.8%	1.18 [1.16,1.20]
SupervisedRoute	50.8/64.9	198 [192,204]	312 [304,321]	1,205/1.1%	1.15 [1.13,1.17]
SupportDR	51.2/65.3	186 [180,192]	304 [296,312]	1,560/1.4%	1.17 [1.15,1.19]
SupportDR+Ctrl	51.3/65.4	158 [152,164]	252 [244,260]	470/0.42%	1.19 [1.17,1.21]
COAST w/o controller	51.4/65.6	194 [187,201]	287 [279,296]	3,015/2.7%	1.18 [1.16,1.20]
COAST	51.4/65.6	156 [149,163]	248 [241,256]	445/0.4%	1.18 [1.16,1.20]

disjoint latency, risk, and charge models, it reaches 157/250 ms and $1.18/1K. Appendix Tables 4–5 report all comparators and margins.

6.4 RQ3: Does the Separation Persist Across Serving Regimes?

At 2× load, COAST reaches 173.5/276 ms with 0.76% violations; its P95 grows by 11.2% versus 38.4% for Dense+RR (Appendix Figure 4). In the later window, COAST records 162/256 ms at 1× and 180/279 ms at 2×; every dataset preserves the tail direction (Appendix Tables 6 and 7).

Second generator–reranker stack. Stack B replaces both generator and reranker and refits every policy and predictor. At base load, COAST records 49.9/64.2 EM/F1 versus 49.6/63.8 for Dense+RR, with paired non-inferiority (Appendix Table 13). In the separate 2× stress, SupportDR+Ctrl moves SupportDR from 221/338 to 194/304 ms and violations from 2.1% to 0.83%; full COAST reaches 188/288 ms and 0.68%, the only displayed Stack B point below both tail targets.

Interpretation. The matched contrast includes the intended veto and renormalization, supporting attribution to admission as a whole. Stack B quality and 2× tails are separate measurements; hash exclusion targets labeled proposal/quality tuples, and support follows the declared propensity/ESS rule. Across 1–2× load, a later window, and two stacks, full COAST stays below both tail targets and fixed-proposal pairs preserve the admission-associated separation.

7 Conclusion

Holding the SupportDR proposal and stack fixed, admission moves P99 from 304 to 252 ms and violations from 1.4% to 0.42%, with a +0.1/+0.1-point EM/F1 change. Full COAST reaches 51.4/65.6 EM/F1, 156/248 ms tails, 0.4% violations, and $1.18/1K charge; its hash-excluded refit retains 51.2/65.3 EM/F1. The direction persists under doubled load and after replacing both generator and reranker. Budgeted RAG should learn what its logs justify, then admit only what its current stack can execute on time.

Ethics Statement. This study uses public QA benchmarks and web corpora without human-subject interaction. Evaluation jointly reports answer quality, vetoes, calibration, support, and budget violations. Decision records retain query hashes and de-identified features rather than raw query text.

AI Use Statement. Language-model tools assisted with manuscript preparation, including formatting, citation, and language revisions.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 527–536. PMLR, 2017. URL https://proceedings.mlr.press/v70/bolukbasi17a.html.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2206–2240. PMLR, 2022. URL https://proceedings.mlr.press/v162/borgeaud22a.html.

Jeffrey Dean and Luiz André Barroso. The tail at scale. *Communications of the ACM*, 56(2):74–80, 2013. doi: 10.1145/2408776.2408794. URL https://doi.org/10.1145/2408776.2408794.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 1097–1104. Omnipress, 2011. URL https://icml.cc/2011/papers/554_icmlpaper.pdf.

Michael Glass, Gaetano Rossiello, Md Faisal Mahbub Chowdhury, Ankita Rajaram Naik, Pengshan Cai, and Alfio Gliozzo. Re2G: Retrieve, rerank, generate. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2701–2715. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.naacl-main.194. URL https://aclanthology.org/2022.naacl-main.194/.

Siddharth Gollapudi, Neel Karia, Varun Sivashankar, Ravishankar Krishnaswamy, Nikit Begwani, Swapnil Raz, Yiyong Lin, Yin Zhang, Neelam Mahapatro, Premkumar Srinivasan, Amit Singh, and Harsha Vardhan Simhadri. Filtered-diskann: Graph algorithms for approximate nearest neighbor search with filters. In *Proceedings of the ACM Web Conference (WWW)*, pp. 3406–3416. ACM, 2023. doi: 10.1145/3543507.3583552. URL https://doi.org/10.1145/3543507.3583552.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3929–3938. PMLR, 2020. URL https://proceedings.mlr.press/v119/guu20a.html.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023. URL https://www.jmlr.org/papers/v24/23-0037.html.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7036–7050. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.389. URL https://aclanthology.org/2024.naacl-long.389/.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings*

of Machine Learning Research, pp. 652–661. PMLR, 2016. URL https://proceedings.mlr.press/v48/jiang16.html.

Wenqi Jiang, Suvinay Subramanian, Cat Graves, Gustavo Alonso, Amir Yazdanbakhsh, and Vidushi Dadu. RAGO: Systematic performance optimization for retrieval-augmented generation serving. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, pp. 974–989. ACM, 2025. doi: 10.1145/3695053.3731093. URL https://doi.org/10.1145/3695053.3731093.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7969–7992. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.495. URL https://aclanthology.org/2023.emnlp-main.495/.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021. doi: 10.1109/TBDATA.2019.2921572. URL https://doi.org/10.1109/TBDATA.2019.2921572.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke 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)*, pp. 1601–1611. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pp. 3146–3154. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html.

Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 39–48. ACM, 2020. doi: 10.1145/3397271.3401075. URL https://doi.org/10.1145/3397271.3401075.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. *Pretrained Transformers for Text Ranking: BERT and Beyond*. Synthesis Lectures on Human Language Technologies. Springer International Publishing, 2022. doi: 10.1007/978-3-031-02181-7. URL https://doi.org/10.1007/978-3-031-02181-7.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*, 2023. doi: 10.48550/arXiv.2305.02156. URL https://arxiv.org/abs/2305.02156.

Yury A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020. doi: 10.1109/TPAMI.2018.2889473. URL https://doi.org/10.1109/TPAMI.2018.2889473.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019. doi: 10.48550/arXiv.1901.04085. URL https://arxiv.org/abs/1901.04085.

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. In *Proceedings of the ACM SIGOPS 31st Symposium on Operating Systems Principles*, pp. 606–622. ACM, 2025. doi: 10.1145/3731569.3764855. URL https://doi.org/10.1145/3731569.3764855.

Muhammad Faizan Raza, Shuo Luna Yang, and Satish Mahadevan Srinivasan. SAGE: SLO-aware adaptive retrieval for production RAG systems. In *2026 12th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 169–175. IEEE, 2026. doi: 10.1109/CoDIT70676.2026.11631166. URL https://doi.org/10.1109/CoDIT70676.2026.11631166.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 17456–17472, 2022. doi: 10.52202/068431-1269. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6fac9e316a4ae75ea244ddcef1982c71-Abstract-Conference.html.

Suhas Jayaram Subramanya, Fnu Devvrit, Harsha Vardhan Simhadri, Ravishankar Krishnawamy, and Rohan Kadekodi. DiskANN: Fast accurate billion-point nearest neighbor search on a single node. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pp. 13748–13758. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/09853c7fb1d3f8ee67a61b6bf4a7f8e6-Abstract.html.

Lalith Suresh, Marco Canini, Stefan Schmid, and Anja Feldmann. C3: Cutting tail latency in cloud data stores via adaptive replica selection. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pp. 513–527. USENIX Association, 2015. URL https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/suresh.

Adith Swaminathan and Thorsten Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research*, 16:1731–1755, 2015. URL https://jmlr.org/papers/v16/swaminathan15a.html.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.557. URL https://aclanthology.org/2023.acl-long.557/.

Ashish Vulimiri, Philip Brighten Godfrey, Radhika Mittal, Justine Sherry, Sylvia Ratnasamy, and Scott Shenker. Low latency via redundancy. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, pp. 283–294. ACM, 2013. doi: 10.1145/2535372.2535392. URL https://doi.org/10.1145/2535372.2535392.

Jianguo Wang, Xiaomeng Yi, Rentong Guo, Hai Jin, Peng Xu, Shengjun Li, Xiangyu Wang, Xiangzhou Guo, Chengming Li, Xiaohai Xu, Kun Yu, Yuxing Yuan, Yinghao Zou, Jiquan Long, Yudong Cai, Zhenxiang Li, Zhifeng Zhang, Yihua Mo, Jun Gu, Ruiyi Jiang, Yi Wei, and Charles Xie. Milvus: A purpose-built vector data management system. In *Proceedings of the 2021 International Conference on Management of Data*, pp. 2614–2627. ACM, 2021. doi: 10.1145/3448016.3457550. URL https://doi.org/10.1145/3448016.3457550.

Herbert Woisetschläger, Ryan Zhang, Shiqiang Wang, and Hans-Arno Jacobsen. MESS+: Dynamically learned inference-time LLM routing in model zoos with service level guarantees. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 60178–60211. Neural Information Processing Systems Foundation, 2025. doi: 10.52202/085713-1804. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4dd1d9b841712bd37b833559f041530c-Abstract-Conference.html.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2246–2251. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.204. URL https://aclanthology.org/2020.acl-main.204/.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=zeFrfgyZln.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=m1JLVigNHp.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for transformer-based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pp. 521–538. USENIX Association, 2022. URL https://www.usenix.org/conference/osdi22/presentation/yu.

Qianxi Zhang, Shuotao Xu, Qi Chen, Guoxin Sui, Jiadong Xie, Zhizhen Cai, Yaoqi Chen, Yinxuan He, Yuqing Yang, Fan Yang, Mao Yang, and Lidong Zhou. VBase: Unifying online vector similarity search and relational queries via relaxed monotonicity. In *Proceedings of the 17th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 377–395. USENIX Association, 2023. URL https://www.usenix.org/conference/osdi23/presentation/zhang-qianxi.

A Additional Evidence and Implementation Details

A.1 Decision Matrix for Closest Systems

The comparison locates COAST at the proposal–execution boundary. Its direct empirical object is the complete-request change produced by execution-time admission while the downstream stack and proposal rule remain fixed.

A.2 Query-Disjoint Quality and Margin Sensitivity

Policy and quality models are retrained after removing every evaluation query hash from their labeled training and validation tuples. Latency, risk, and charge predictors retain the event-disjoint training

Table 3: Decision-level comparison with the closest adaptive retrieval and serving systems. "Measured target" names the quantity each design directly optimizes or evaluates.

System	Adaptive object	Live observation	Learning signal	Measured target
METIS	Chunk/synthesis configuration and scheduling	Query and resource state	Configuration performance	Quality-aware serving latency
SAGE	Retrieval depth	First-pass retrieval features	Oracle-depth imitation	P95-aware depth selection
MESS+	LLM endpoint	Virtual-queue state	Endpoint reward and cost	Long-run satisfaction under cost
RAGO	Placement, resources, and batching	Workload and hardware profile	System profiling	RAG throughput and latency
COAST	Supported retrieval proposal plus execution admission	Query, queues, recent stage time, utilization, and load	Known propensities, answer labels, charge, and latency traces	Matched admission contrast and joint quality–tail–charge point

partition, and their hyperparameters are selected on validation; no final-partition outcome enters model fitting or selection. Table 4 shows the complete paired quality comparison; Table 5 verifies that the high-quality operating region is stable at both 0.25- and 0.5-point margins.

Table 4: Complete query-hash-disjoint quality comparison at 160 QPS. Intervals are paired, dataset-stratified 95% bootstrap CIs; the 2,376,148 retained training tuples have 0 hash overlap with evaluation.

Method	EM	ΔEM vs. Dense+RR [95% CI]	F1	ΔF1 vs. Dense+RR [95% CI]
DENSE+RR	51.1	–	65.2	–
NORET	42.3	-8.8 [-9.1, -8.5]	55.8	-9.4 [-9.7, -9.1]
SPARSE	47.2	-3.9 [-4.2, -3.6]	61.4	-3.8 [-4.1, -3.5]
DENSE	49.8	-1.3 [-1.6, -1.0]	63.6	-1.6 [-1.9, -1.3]
CASCADE	50.4	-0.7 [-1.0, -0.4]	64.5	-0.7 [-1.0, -0.4]
CASCADE+RR	51.2	+0.1 [-0.2, +0.3]	65.3	+0.1 [-0.1, +0.3]
BEST STATIC 25	50.7	-0.4 [-0.7, -0.1]	64.9	-0.3 [-0.6, 0.0]
HEURISTICGATE	50.6	-0.5 [-0.8, -0.1]	64.8	-0.4 [-0.7, -0.1]
SUPERVISEDROUTE	50.9	-0.2 [-0.5, +0.1]	65.0	-0.2 [-0.4, +0.1]
SUPPORTDR	51.3	+0.2 [-0.1, +0.5]	65.4	+0.2 [-0.1, +0.4]
SUPPORTDR+CTRL	51.3	+0.2 [-0.1, +0.5]	65.5	+0.3 [0.0, +0.5]
COAST w/o controller	51.3	+0.2 [-0.1, +0.5]	65.5	+0.3 [0.0, +0.6]
COAST	51.2	+0.1 [-0.2, +0.4]	65.3	+0.1 [-0.2, +0.3]

A.3 PER-DATASET AND TEMPORAL STABILITY

The aggregate result is consistent across all three benchmarks: quality remains in the same local region, and COAST reduces both tails for NQ, HotpotQA, and TriviaQA.

The later shadow window preserves the same interface and budgets while shifting request mix and queue state. Policy ordering remains strong at both load levels, and COAST retains the lowest tail among high-quality routes.

Later-window P99 values at 1×/2× are 420/549 for DENSE+RR, 384/472 for CASCADE+RR, 336/386 for HEURISTICGATE, 319/351 for SUPERVISEDROUTE, and 256/279 for COAST.

A.4 CONTROLLER CALIBRATION AND VETO CONSEQUENCES

Table 8 compares calibrated prediction with shadow-replay violations for the offline top-ranked candidate on every request. Bins below the risk threshold characterize false-safe behavior before

Table 5: Non-inferiority sensitivity from the lower paired confidence bound. "Pass" requires both EM and F1 lower bounds to exceed the stated margin.

Method	0.25-point margin	0.5-point margin
NoRet	fail	fail
Sparse	fail	fail
Dense	fail	fail
Cascade	fail	fail
Cascade+RR	pass	pass
Best Static 25	fail	fail
HeuristicGate	fail	fail
SupervisedRoute	fail	inconclusive
SupportDR	pass	pass
SupportDR+Ctrl	pass	pass
COAST w/o controller	pass	pass
COAST	pass	pass

Table 6: Per-dataset comparison between Dense+RR and COAST on Stack A.

Dataset	Method	EM	F1	P95	P99
NQ	Dense+RR	48.6	62.4	264	398
	COAST	48.9	62.8	148	238
HotpotQA	Dense+RR	53.2	67.8	292	436
	COAST	53.5	68.1	168	264
TriviaQA	Dense+RR	51.4	65.5	278	402
	COAST	51.8	65.9	152	242

Table 7: Later-window shadow evaluation on Stack A. Offline-predicted versus realized policy ranking has Spearman $\rho = 0.89$ at $1\times$ and 0.83 at $2\times$ load.

Method	EM	F1	P95 @ $1\times$	P95 @ $2\times$	Charge/1K @ $1\times$	Charge/1K @ $2\times$
Dense+RR	50.8	64.8	284	391	1.54	1.55
Cascade+RR	50.9	64.9	254	329	1.35	1.36
HeuristicGate	50.4	64.5	216	248	1.19	1.20
SupervisedRoute	50.6	64.7	204	229	1.16	1.17
COAST	50.9	64.9	162	180	1.20	1.21

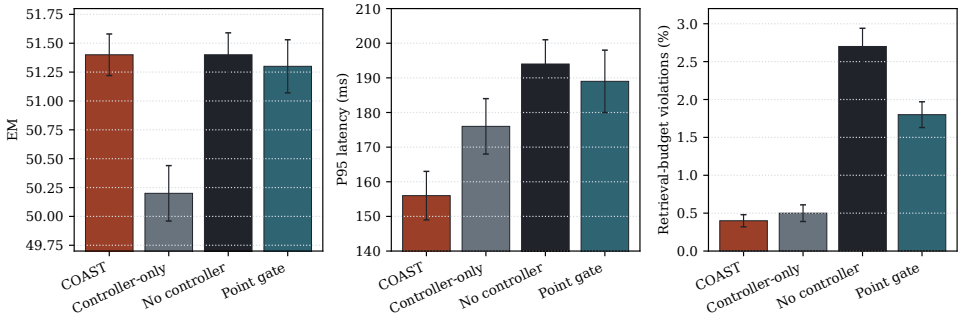

Figure 3: Bundled system contrasts relative to full COAST. Controller-only replaces the complete doubly robust proposal objective with a plug-in score; No controller bypasses admission; and Point gate jointly removes quantile prediction, calibration, margin, binary-risk admission, and monotone projection. Error bars are paired-bootstrap intervals for EM, run-level intervals for P95, and query-bootstrap intervals for violations.

filtering. Table 9 then reports final executed-request violations, conservative vetoes, veto volume, fallback activation, and P99 by traffic slice.

Table 8: Controller risk calibration at 1× nominal load. Counts aggregate one offline top-ranked candidate per request over five repetitions before safe-set filtering.

Predicted-risk bin	Mean prediction	Realized violations [95% CI]	Count
$[0, 0.01)$	0.006	0.008 [0.007,0.009]	42,355
$[0.01, 0.025)$	0.018	0.022 [0.020,0.024]	29,810
$[0.025, 0.05)$	0.038	0.047 [0.044,0.050]	20,690
$[0.05, 0.10)$	0.071	0.106 [0.101,0.112]	12,255
$[0.10, 1]$	0.149	0.236 [0.226,0.246]	6,530

Table 9: Safe-set behavior across load and traffic shift. Counts and request-level rates aggregate five complete repetitions (111,640 requests); false-veto is conditional on vetoed requests.

Slice	ECE	Final viol.	False-veto count/rate	Veto count/rate	Fallback count/rate	P99
Nominal, 1×	0.012	0.4%	530/4.2%	12,615/11.3%	780/0.7%	248
Bursty, 1×	0.057	1.4%	974/5.1%	19,100/17.1%	2,345/2.1%	271
Later, 1×	0.049	1.2%	739/4.8%	15,405/13.8%	1,565/1.4%	256
Nominal, 1.5×	0.052	0.6%	849/5.1%	16,635/14.9%	1,230/1.1%	268
Nominal, 2×	0.064	0.76%	1,322/6.3%	20,990/18.8%	2,680/2.4%	276

Table 10: Controller variants at 1× nominal load. Brackets are 95% intervals over runs (latency, charge, ECE) or query-bootstrap intervals (executed violations).

Variant	EM/F1	ECE [CI]	Final violation [CI]
Fixed SPARSE	47.2/61.4	–	0.14% [0.11%,0.17%]
Controller-only	50.2/64.3	0.074 [0.067,0.082]	0.5% [0.44%,0.65%]
Point-prediction gate	51.3/65.5	0.118 [0.108,0.129]	1.8% [1.61%,2.01%]
COAST w/o controller	51.4/65.6	–	2.7% [2.45%,2.96%]
COAST	51.4/65.6	0.012 [0.010,0.015]	0.4% [0.36%,0.44%]

Variant	P95 [CI]	P99 [CI]	Charge/1K [CI]
Fixed SPARSE	182 [177,188]	276 [269,284]	0.98 [0.96,1.00]
Controller-only	176 [169,183]	269 [260,278]	1.16 [1.14,1.18]
Point-prediction gate	189 [182,197]	308 [298,319]	1.18 [1.16,1.21]
COAST w/o controller	194 [187,201]	287 [279,296]	1.18 [1.16,1.20]
COAST	156 [149,163]	248 [241,256]	1.18 [1.16,1.20]

Table 11: Controller-comparator paired quality intervals across all slices. Deltas and dataset-stratified bootstrap 95% CIs are relative to fixed DENSE+RR.

Variant/slice	ΔEM [CI]	ΔF1 [CI]
COAST, nominal	+0.3 [0.0,+0.6]	+0.4 [+0.1,+0.7]
COAST, burst	+0.2 [-0.1,+0.5]	+0.3 [0.0,+0.6]
COAST, later	+0.1 [-0.2,+0.4]	+0.1 [-0.2,+0.4]
COAST, 2×	+0.1 [-0.2,+0.4]	+0.2 [-0.1,+0.5]
Controller-only, nominal	-0.9 [-1.2,-0.6]	-0.9 [-1.2,-0.6]
Point gate, nominal	+0.2 [-0.1,+0.5]	+0.3 [0.0,+0.6]
No controller, nominal	+0.3 [0.0,+0.6]	+0.4 [+0.1,+0.7]

The burst slice increases veto activity and fallback use, yet calibrated executed-request violations remain 1.4%. Errors concentrate above the admission threshold, while low-risk bins track realized outcomes closely.

Table 12: Strong same-interface adaptive baseline across load. SUPPORTDR and SUPPORTDR+CTRL share policy class, features, support mask, objective, split, and tuning budget; violation counts aggregate five repetitions.

Method/load	EM/F1	P95	P99	Charge/1K	Viol.	High-support	ESS/n
SUPPORTDR, 1×	51.2/65.3	186	304	1.17	1,560	91.3%	0.49
SUPPORTDR, 1.5×	51.1/65.2	213	321	1.18	2,340	90.6%	0.46
SUPPORTDR, 2×	51.0/65.1	246	338	1.19	3,460	89.8%	0.42
SUPPORTDR+CTRL, 1×	51.3/65.4	158	252	1.19	470	93.0%	0.48
SUPPORTDR+CTRL, 1.5×	51.2/65.3	169	267	1.20	640	94.1%	0.46
SUPPORTDR+CTRL, 2×	51.1/65.2	181	281	1.21	870	95.2%	0.43

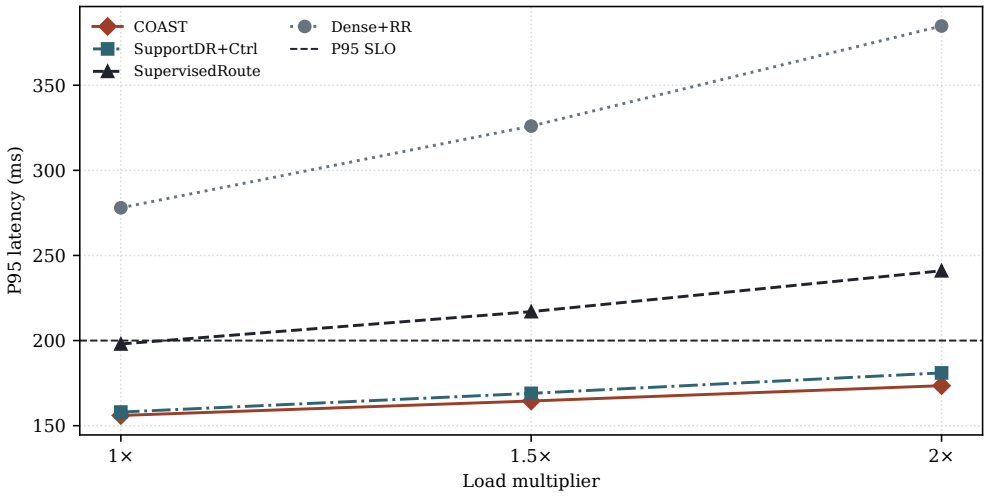

Figure 4: Load stress test. From 1× to 2× load, COAST P95 increases by 11.2%, remains below the 200 ms SLO, and stays below 0.8% retrieval-budget violations; DENSE+RR P95 increases by 38.4%.

A.5 STACK B DETAILS

Table 13 establishes the Stack B high-quality region at base load with the same paired test used on Stack A. Table 14 separately reports run-level tail, charge, violation, and support measurements at 2× load. Every policy and nuisance model is refit for Stack B.

Table 13: Stack B base-load query-hash-disjoint quality test. Deltas and paired 95% bootstrap CIs are relative to DENSE+RR; passing requires both lower bounds to exceed -0.5 points.

Method	ΔEM [CI]	ΔF1 [CI]	NI result
CASCADE+RR	+0.2 [-0.12,+0.47]	+0.2 [-0.09,+0.44]	pass
BEST STATIC 25	-0.4 [-0.69,-0.07]	-0.3 [-0.62,+0.01]	fail
HEURISTICGATE	-0.5 [-0.83,-0.18]	-0.5 [-0.81,-0.16]	fail
SUPERVISEDROUTE	-0.3 [-0.61,+0.02]	-0.2 [-0.51,+0.12]	fail
SUPPORTDR	+0.1 [-0.18,+0.39]	+0.2 [-0.11,+0.48]	pass
SUPPORTDR+CTRL	+0.2 [-0.08,+0.47]	+0.3 [+0.01,+0.56]	pass
COAST w/o controller	+0.2 [-0.07,+0.48]	+0.3 [+0.01,+0.57]	pass
COAST	+0.3 [+0.02,+0.59]	+0.4 [+0.09,+0.68]	pass

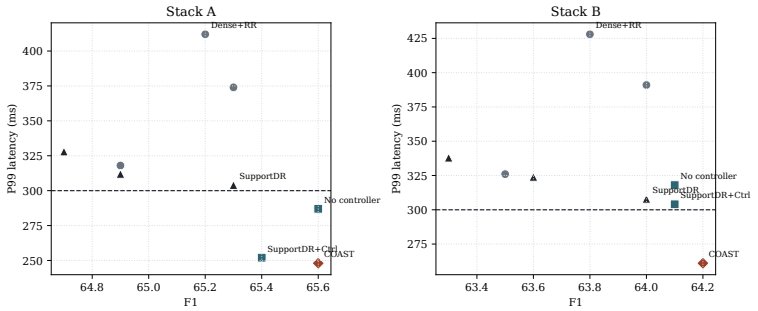

Figure 5: Base-load P99–quality comparisons on Stack A and Stack B. Absolute quality and latency change with the generator and reranker, while controller-backed routes remain left of their offline-only counterparts.

Table 14: Stack B at 320 QPS. Brackets are run-level 95% CIs; counts aggregate five repetitions.

Method	P95 [CI]	P99 [CI]	Charge/1K [CI]	Viol. count/rate	Support	ESS/n
DENSE+RR	403 [390,417]	586 [569,604]	1.49 [1.45,1.53]	20,545/18.4%	100.0%	0.24
CASCADE+RR	333 [323,344]	488 [473,504]	1.33 [1.30,1.36]	15,855/14.2%	100.0%	0.30
BEST STATIC 25	262 [253,271]	385 [373,398]	1.14 [1.11,1.17]	5,805/5.2%	100.0%	0.32
HEURISTICGATE	274 [265,284]	402 [389,416]	1.17 [1.14,1.20]	3,350/3.0%	99.0%	0.56
SUPERVISEDROUTE	239 [231,247]	371 [359,384]	1.13 [1.11,1.15]	2,570/2.3%	86.0%	0.34
SUPPORTDR	221 [214,229]	338 [327,350]	1.15 [1.13,1.17]	2,345/2.1%	89.2%	0.40
SUPPORTDR+CTRL	194 [187,202]	304 [294,315]	1.18 [1.16,1.20]	927/0.83%	94.5%	0.41
COAST w/o controller	244 [236,253]	356 [344,369]	1.14 [1.12,1.16]	4,580/4.1%	92.3%	0.39
COAST	188 [181,195]	288 [279,298]	1.18 [1.16,1.20]	760/0.68%	95.0%	0.42

At doubled load, the matched admission contrast survives the model-stack change: SUPPORTDR+CTRL moves SUPPORTDR from 221/338 to 194/304 ms P95/P99 and reduces violations from 2.1% to 0.83%. Full COAST reaches 188/288 ms and 0.68% violations. The paired quality result above is a separate base-load measurement.

A.6 LOGGED SUPPORT AND OBSERVATION EVIDENCE

The support audit measures the executed distribution rather than only the intended policy. "Exploratory mass" denotes policy execution of catalog actions whose logging coverage originates in the stratified exploration stream; it is distinct from the logger's aggregate exploration probability, and every such action remains subject to the conditional propensity and ESS mask. Table 15 reports the global policy-overlap statistic using $w_i = \pi(a_i \mid x_i)/\tilde{\mu}_i$ over all logged requests, including zero weights for nonmatching actions, and normalizes $(\sum_i w_i)^2 / \sum_i w_i^2$ by the full n. This differs from the within-cell eligibility ESS in Section 5.1. COAST remains above 90% mass on high-support actions at both loads, while SUPERVISEDROUTE enters the exploratory-action tail more often; comparator global ESS ratios range from 0.25 to 0.61, and COAST remains at 0.44 or higher.

Table 15: Executed-mass and importance-weight audit for Stack A.

Policy / load	High-support mass	Exploratory mass	ESS / n	Max weight
COAST @ 1×	92.1%	7.9%	0.47	6.8
COAST @ 2×	94.4%	5.6%	0.44	7.4
HEURISTICGATE @ 1×	98.8%	1.2%	0.61	3.1
HEURISTICGATE @ 2×	99.1%	0.9%	0.59	3.3
SUPERVISEDROUTE @ 1×	89.7%	10.3%	0.45	7.6
SUPERVISEDROUTE @ 2×	86.8%	13.2%	0.39	8.9
DENSE+RR @ 1×	100.0%	0.0%	0.25	4.0
DENSE+RR @ 2×	100.0%	0.0%	0.25	4.0

Table 16: Non-exploratory logging mixture. These weights govern 99.5% of traffic; the remaining 0.5% is stratified exploration over controller-safe actions.

Action	Weight	Action	Weight
NORET	5%	DENSE+RR	25%
SPARSE	20%	CASCADE	15%
DENSE	30%	CASCADE+RR	5%

Table 17: Logging schema and observation boundary. Pre-action fields are available to every router; outcomes are joined only after execution.

Stage	Retained fields	Availability
Admission	Query hash, 18 de-identified features, queue depths, utilization	Pre-action
Router	Supported-action mask, action probabilities, selected action, propensity	Pre-execution
Retrieval	Queue, search, fetch, and reranking latency; retrieved depth	Post-execution
Generation	Prompt/output token counts, first/final-token time, truncation flag	Post-execution
Controller	Quantile/risk predictions, admissible set, veto and fallback decisions	Pre/post decision
Outcomes	Charge for all traffic; EM/F1 for reference-labeled evaluation requests	Post-execution

A.7 LOAD AND DRIFT DIAGNOSTICS

The request-level violation trace complements the reported P95/P99 measurements by showing how often admitted retrieval work crosses the 75 ms sub-budget as load or traffic state changes.

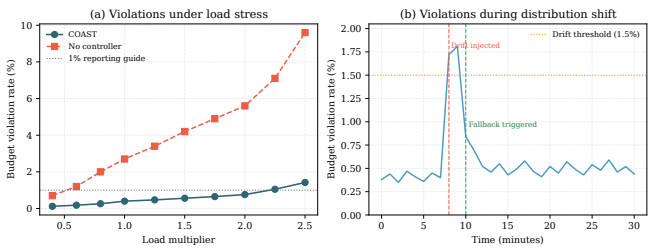

Figure 6: Retrieval-budget violations across load and an injected shift. In panel (a), the 1% line is a reporting guide; it is not the fallback trigger. Panel (b) marks the declared 1.5% activation threshold. Full COAST remains below 0.8% through 2× load, while bypassing admission produces progressively more violations.

The curve aligns with the matched serving tables: the admission-backed policy keeps executed violations bounded through the evaluated $2\times$ regime, whereas bypassing the operator increases both violations and complete-request tails.

A.8 Training and Evaluation Details

Algorithm 1 Serving-time action selection

update the rolling KL and violation monitor
if fallback is active **then**
    set proposal q_t to SPARSE($k = 4$) for high lexical rarity, and NORET otherwise
else
    set $q_t(\cdot \mid x) = \pi_\theta(\cdot \mid x)$
end if
obtain projected $\hat{T}_{\mathrm{ret}}, \hat{p}_v$ and form $\mathcal{A}_{\mathrm{safe}}(x)$ by Equation 9
$\mathcal{A}_{\mathrm{exec}}(x) \leftarrow \mathrm{supp}(q_t) \cap \mathcal{A}_{\mathrm{safe}}(x)$
if $\mathcal{A}_{\mathrm{exec}}(x) = \varnothing$ **then**
    return NORET
end if
renormalize q_t on $\mathcal{A}_{\mathrm{exec}}(x)$ and sample with the request seed

Table 18: Per-method generation truncation counts in one complete 22,328-request repetition. The shared 192-token cap and scoring rule apply unchanged to every method.

Method	Count	Method	Count
NORET	31	BEST STATIC 25	38
SPARSE	33	HEURISTICGATE	36
DENSE	37	SUPERVISEDROUTE	35
DENSE+RR	44	SUPPORTDR	36
CASCADE	39	SUPPORTDR+CTRL	35
CASCADE+RR	42	COAST w/o controller	38
		COAST	34

Table 19: Third-party assets and governing licenses or terms.

Asset	License/terms	Role
2023-12 Wikipedia	CC BY-SA 4.0/GFDL	Retrieval corpus
2024-01 Common Crawl	Common Crawl and source-site terms	Retrieval corpus
Llama-3.1-8B-Instruct	Llama 3.1 Community License	Stack A generator
Mistral-7B-Instruct-v0.3	Apache 2.0	Stack B generator
e5-large-v2	MIT	Dense embeddings
bge-reranker-large	MIT	Stack A reranker
cross-encoder/ms-marco-MiniLM-L-12-v2	Apache 2.0	Stack B reranker
NQ, HotpotQA, TriviaQA	Dataset-specific terms	Reference-labeled evaluation

For request q, the accounting function is $C(q) = r_g t_g(q) + r_c t_c(q) + r_s n_s(q)$, where t_g and t_c sum the GPU- and CPU-stage residence time attributed to that request and n_s counts physical NVMe reads. Concurrent and batched requests are attributed separately, so the aggregate is a normalized per-request service charge rather than physical device busy time or a cloud invoice. Rates are fixed before policy tuning; one-time model/index construction and capacity-isolated shadow traffic are excluded.

The policy is a 2-layer MLP with hidden size 128, trained with Adam at learning rate 3×10^{-4} on up to 2.4 M labeled replay tuples from the retained 14-day window. Forty validation trials sample

Table 20: Audited wall-clock accounting. Each row includes all five independent fits or serving repetitions.

Stage	Hardware	Wall-clock
Quality model and propensity audit	32-core AMD EPYC 9654	71 min
Stack A policy optimization	32-core AMD EPYC 9654	91 min
Latency, risk, and charge models	32-core AMD EPYC 9654	108 min
Stack A nominal execution	2× NVIDIA A100 80GB	12 min
Load, burst, and later-window runs	2× NVIDIA A100 80GB	96 min
Stack B refits and execution	2× NVIDIA A100 80GB	312 min

Table 21: Request-attributed service-charge schedule at the primary COAST operating point. Contributions are rounded after multiplication and sum to \$1.18/1K queries.

Resource	Fixed rate	Usage/1K queries	Contribution/1K
A100 stage time	\$12.00/GPU-h	0.0748 attributed GPU-h	\$0.898
EPYC stage time	\$0.176/core-h	1.37 attributed core-h	\$0.241
NVMe physical reads	\$0.021/M reads	1.95 M reads	\$0.041
Total	–	–	\$1.180

Table 22: Measurement traceability within the evaluation protocol. Each central claim is derived from the listed request-level fields and aggregate.

Claim	Request-level measurement	Aggregate evidence
Standard-split joint operating point	Answers, end-to-end latency, action charge	Tables 2, 11
Controller calibration	Pre-filter risk, paired shadow latency, admission decision	Tables 8, 9
Logged support	Propensity, action, feature stratum, importance weight	Table 15
Temporal stability	Later-window action ranking and serving outcomes	Table 7
Cross-stack transfer	Stack B base-load paired answers; 2× tails, decisions, and support	Tables 13, 14

λ_C log-uniformly over $[0.03, 3.0]$, λ_H over $[0.001, 0.1]$, and policy temperature over $[0.5, 1.5]$. The auxiliary quality, latency, and charge predictors use 256-tree LightGBM models with max depth 8 and learning rate 0.05. Policy training completes in 18 minutes on one 32-core AMD EPYC 9654 machine, and a base-load evaluation pass takes 140 seconds at 160 QPS. The five repetitions rerun nuisance-model fitting, policy optimization, arrival generation, and complete request execution from independent seeds. Adaptive-method latency intervals therefore include both learning and serving variability; fixed-route intervals include independent arrival and serving variability.