Failure-Band Authorization for Filtered Retrieval

Rui Li **Shuang Cao**
Hill Research

Abstract

Filtered vector search in CPU-only retrieval-augmented generation must protect retrieval latency under fixed RAM and SSD budgets. Across recurring workloads, we identify a failure band in which changes in vector–metadata alignment reverse execution-plan rankings and produce 3–6× P95 spread. Within this band, similar predicted winning margins conceal sharply different realized regret, making the cost of a dispatch error central to retrieval control. PACER addresses this instability through runtime authorization: it adjusts four execution plans to a common recall floor, tests top-candidate latency-interval containment and normalized tail width, and vetoes unreliable winners in favor of a regime-local feasible fallback. On a dual-AMD EPYC 7543 server with 128GB RAM, identical-model, identical-action controls isolate the effect: PACER reduces failure-band P95 from 49 to 37ms relative to winner-only dispatch and from 42 to 37ms relative to calibrated confidence, closing 75% of the winner-to-oracle gap at 0.951 Recall@20. Paired evaluation across 25 seeds gives a -12ms P95 difference against winner-only, with a 95% interval of $[-14,-10]$ms. The criterion-defined band recurs on three public workloads, and authorization improves both learned-planner and SSD-backend compositions. Across the enterprise request mixture, P95 falls from 46 to 34ms relative to Qdrant. Recall-adjusted authorization thus connects a recurring workload failure to a transferable dispatch intervention for predictable retrieval.

1 Introduction

Consider two queries on the same CPU-only RAG server, both near 8% selectivity and both under an 80ms retrieval budget. When metadata aligns with embedding neighborhoods, ANN-first expansion is economical; when alignment weakens, the same choice can trigger post-filter expansion and random SSD probes. Switching unconditionally to filter-first can instead spend the budget scanning the wrong slice. The operational challenge is deciding whether the predicted gain justifies committing a query to that execution order.

Filtered vector retrieval is a latency-critical operator in retrieval-augmented generation [10, 4], including pipelines using dense retrievers [7, 8]. Private and on-prem deployments often serve these queries with fixed CPU, RAM, and SSD resources. If generation reserves b_{LLM} of an application deadline T, retrieval has slack $T - b_{\mathrm{LLM}}$; consuming that slack leaves less time for downstream generation. Retrieval-tail control therefore has a direct serving consequence, which we evaluate through retrieval-tail latency under fixed resource budgets.

Our measurements identify a recurring instability regime. Define selectivity as the fraction of vectors satisfying the predicate: a small passing fraction favors filtering first, while a large passing fraction often favors ANN first. Between these extremes, correlation, cache residency, and cold-read amplification can reverse their ranking. Before controller design, we defined a failure band as the contiguous selectivity interval where the ratio of slowest to fastest recall-feasible fixed-plan P95 exceeds $3\times$ at some observed correlation. Calibration identifies 4.6–15.3%; we freeze 5–15% for

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

reporting and recover 5.1–15.8% on held-out queries. Figure 1 shows how this criterion transfers across workloads.

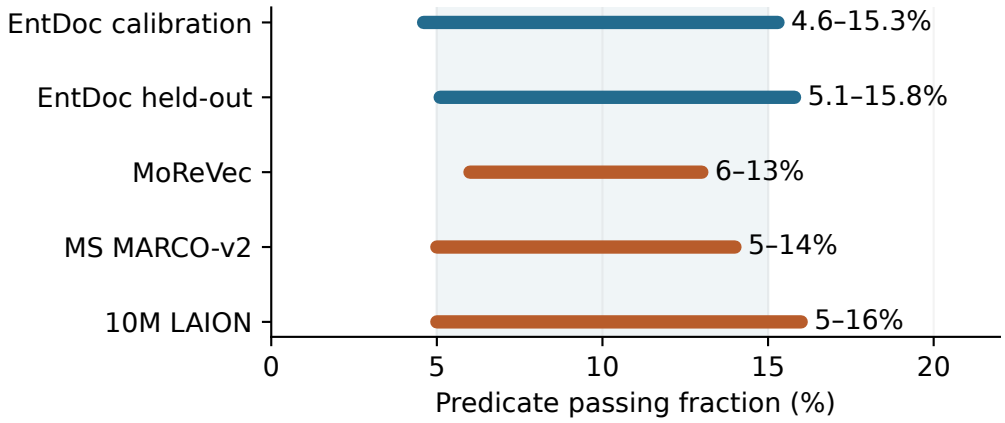

Figure 1: Failure bands under the predeclared criterion: fixed recall-feasible P1–P4 P95 spread exceeds $3\times$ at some correlation in $[.08, .52]$. Shading marks the frozen EntDoc reporting band, not a universal boundary.

The key empirical finding concerns the cost of committing to a ranking. Among queries with the same 20–30% predicted margin, the provisional winner's mean regret is 3.1ms when its latency interval has little overlap with the runner-up, but 19.7ms when overlap exceeds 0.15. A second population has separated candidates but broad conditional tails relative to their predicted P95. A winning margin summarizes relative predicted benefit; these interval features expose different routes to expensive mistakes. This distinction motivates PACER's dual authorization gate: recall-adjust the fanout, filter infeasible candidates, propose the lowest predicted-P95 plan, then test interval containment and normalized tail width before execution. A veto selects a recent feasible plan for the same workload regime. The recall adjustment is integral: additional fanout changes both cold-read exposure and the latency intervals on which authorization operates.

This control boundary complements the surrounding retrieval stack. Gan and Wang's learned planning [3] supplies a candidate ranking; GateANN's SSD traversal [9] reduces cold-tier work; and workload characterization [1] identifies selectivity/correlation effects. PACER connects workload instability to an evaluated dispatch intervention. Adding its gate to the GW* reimplementation with common recall tables reduces P95 from 48 to 39ms. Composing PACER with GateANN* on an I/O-bound server reduces P95 from 74 to 53ms. The asterisks identify reimplementations throughout.

The evaluation covers CPU-only single-server and small-cluster deployments with 64–192GB RAM and 1TB SSD budgets. Four execution families—filter-first, ANN-first, interleaved, and partition-aware—remain fixed throughout controller comparisons. The paper establishes three results:

- A predeclared fixed-plan-spread criterion identifies a recurring failure band on enterprise traces and on MoReVec, MS MARCO-v2, and 10M LAION, with workload-dependent boundaries and $3.6–5.1\times$ public-data peak spread.
- The dispatch mechanism targets both ambiguous rankings and broad conditional tails after recall adjustment. Its two signals yield 44 and 45ms P95 separately and 37ms together. With identical actions and models, the combined gate beats calibrated confidence by 5ms and closes 75% of the winner-to-oracle gap.
- The gain persists across resource profiles and composes with learned planning and SSD traversal. Relative to Qdrant, failure-band P95/P99 fall by 46%/45%; across the whole enterprise request mixture, P95 falls from 46 to 34ms and P99 from 74 to 52ms.

2 Related work and positioning

Filtered search couples metadata selection with approximate-neighbor traversal. HNSW, product quantization, SPANN, and Filtered-DiskANN provide important graph, compression, and storage design points [15, 6, 2, 5]. ACORN extends HNSW with predicate-agnostic graph search [17]; SIEVE constructs a workload-aware collection of indexes for different predicate forms [11]. SeRF organizes graph segments for range predicates, while UNIFY combines range-filter search paths in a unified index [20, 12]. These mechanisms change the cost of feasible actions. The filtered-ANN survey by Lin et al. [13] places them within the vector–scalar search space.

The closest planner comparison is Gan and Wang [3], whose dataset and predicate statistics guide execution-order selection across ANN backends. PACER evaluates a subsequent decision: whether to execute the proposed winner after recall-dependent fanout adjustment. In our matched-budget comparison, GW* with common recall tables reaches 48ms P95 and adding the authorization gate reaches 39ms (Table 3). GateANN [9] separates graph traversal from vector retrieval, routing through nonmatching nodes in memory to avoid their SSD reads. Its traversal changes the cost of an action; PACER's gate selects which feasible action to execute. Their composition is particularly effective on I/O-bound Small+EntDoc. Table 8 separates reimplementation checks from these controlled comparisons.

Amanbayev et al. [1] introduce MoReVec, characterize filtering strategies, and relate global and local selectivity to search performance. Our criterion measures the spread of recall-feasible fixed-plan tails and fixes the evaluation band before controller testing. Applying it across three public workloads establishes where plan choice is consequential; the identical-action comparisons establish how authorization reduces that cost.

Learned query optimization and cardinality estimation motivate model-guided decisions, calibrated uncertainty, and robustness to estimation errors [16, 18, 19, 14]. For filtered retrieval, recall-dependent fanout and asymmetric cold-probe exposure make the magnitude of a dispatch error consequential. Probability of winning and point-margin size summarize different aspects of a decision from conditional tail shape. PACER uses containment and dispersion to select when to consult a regime-keyed feasible fallback. Table 2 holds the models, actions, and fallback ranking fixed; Figure 3 connects the advantage over margin and calibrated-confidence gates to complementary error populations.

3 System design

PACER places a dispatch decision after recall-dependent plan construction. Fanout adjustment changes each plan's predicted work, so authorization operates on the resulting feasible actions and their conditional latency intervals. Top-candidate interval containment tests whether the leading alternatives overlap; normalized tail width tests whether an admitted action has a broad conditional tail relative to its predicted P95 even when the leaders are separated. A regime-local fallback makes this distinction actionable: the controller can exchange a provisional winner for a recently reliable feasible plan. Figure 3 links each signal to its measured decision consequence.

3.1 Query model and architecture

A request is $Q = (q, \Pi, k, \tau, m_{\max})$, with query embedding q, predicate conjunction Π, requested result count k, latency target τ, and per-query memory cap $m_{\max}$. Deployment-level RAM, SSD capacity, and CPU allocations are fixed separately. For each plan $p \in \{\mathrm{P1}, \ldots, \mathrm{P4}\}$, a recall table selects the smallest calibrated fanout k'_p meeting ρ. With feature vector x, define

$$\begin{aligned} \mathcal{F}(Q) &= \{(p, k'_p) : \widehat{R}(p, k'_p \mid x) \geq \rho,\ \widehat{M}(p, k'_p \mid x) \leq m_{\max}\}, \\ \mathcal{A}(Q) &= \{c \in \mathcal{F}(Q) : \widehat{L}_{95}(c \mid x) + \widehat{W}(c \mid x) \leq \tau\}, \quad a = \arg\min_{c \in \mathcal{A}(Q)} \widehat{L}_{95}(c \mid x). \end{aligned} \tag{1}$$

Here $\widehat{W}$ is predicted admission waiting time. Recall and memory define feasible candidates; service P95 plus waiting defines the admitted set under the current load. Ranking and gating both use $\mathcal{A}(Q)$. An empty admission set invokes feasible degraded service under the request policy or rejection. The

observed P95 SLO is evaluated across requests; admission is a forecast, not a deterministic per-request deadline guarantee. Fanout is never reduced below its calibrated recall floor to make a latency forecast fit.

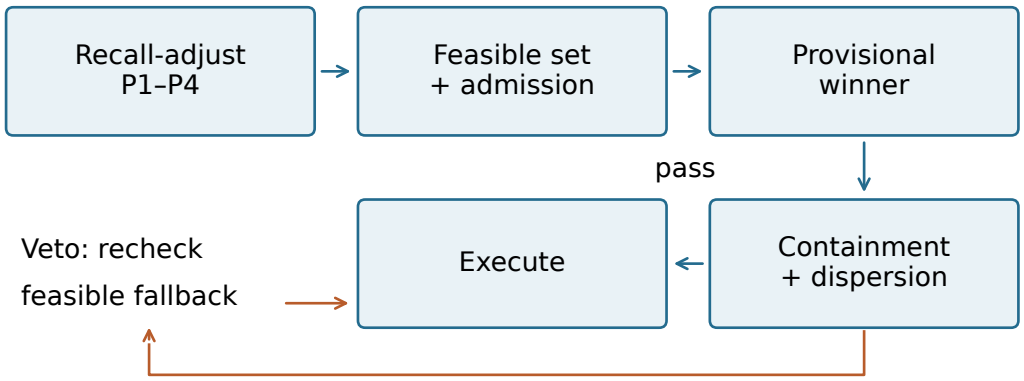

Figure 2: Closed-loop authorization. Recall-adjusted candidates pass resource admission, ranking, and the dual gate. A veto rechecks a regime-local fallback before dispatch. Quantile models belong to the planner.

The frontend normalizes the request, and the estimator produces selectivity and vector–metadata alignment features. The planner's companion quantile models produce the latency intervals, rank candidates, and apply the gate. The execution engine implements the fixed actions over RAM and SSD and returns service telemetry (Table 5). Here, authorization concerns resource dispatch and retrieval reliability; metadata predicate evaluation remains a trusted component.

3.2 Plan space and authorization

Table 1 summarizes the four actions. P4 exploits hierarchical predicates aligned with index partitions, such as region/department/team; P3 streams predicate checks through a bounded traversal frontier. The gate changes the dispatch decision while these execution kernels stay fixed.

Table 1: Execution plans and their resource/recall tradeoffs.

Plan	Dominant work / cold exposure	State and recall response	Favorable regime
P1	Predicate scan and candidate set; cold reads on spill	Candidate-set memory; scan eligible vectors	Small passing fraction
P2	ANN expansion and post-filter; cold reads grow with rejected candidates	Frontier and fanout expand to meet recall	Large passing fraction, aligned or hot neighborhoods
P3	Interleaved traversal and checks; bounded cold probing	Frontier/visited caps; calibrated expansion budget	Failure band and conservative fallback
P4	Partition-local ANN; avoids irrelevant cold partitions	Partition buffers; fanout across eligible partitions	Hierarchy aligned with partitions

For $\Pi = \bigwedge_j P_j$, selectivity is estimated as $\hat{s}_\Pi = \prod_j \hat{s}_{P_j} \prod_{(i,j)\in\mathcal{E}} \hat{\omega}_{P_i,P_j}$, where sketches supply marginal rates and sampled pairs supply correlation corrections. Vector–metadata alignment uses the first 500 ANN candidates: $y_i = \mathbf{1}[\Pi(c_i)]$, $u_i = -i$, and $\hat{r} = \mathrm{Spearman}(y, u)$. Positive $\hat{r}$ indicates that predicate matches concentrate near the query. This candidate probe is retrieval work, distinct from evaluating the controller on its resulting features; it must be included in request-level latency even when candidates are reused by the selected action.

Companion LightGBM quantile regressors predict conditional service-latency quantiles $\widehat{L}_{50}$ and $\widehat{L}_{95}$ from selectivity, correlation, fanout, requested k, plan identity, cache-miss probability, SSD queue depth, and recent fallback activity. Their interval $I_c = [\ell_c, h_c] = [\widehat{L}_{50}(c \mid x), \widehat{L}_{95}(c \mid x)]$ describes conditional tail dispersion. It is not a confidence interval for an estimated P95. Calibration uses held-out query outcomes to align latency predictions and recall-floor lookup with the workload/hardware regime; uncertainty in reported experimental quantiles is assessed separately across runs.

For the admitted candidates $\mathcal{A} \subseteq \mathcal{F}$, define

$$u_c = \frac{h_c - \ell_c}{\max(h_c, 1\,\text{ms})}, \qquad \hat{u} = \max_{c \in \mathcal{A}} u_c, \qquad \text{ov}(a,b) = \frac{\max\{0, \min(h_a, h_b) - \max(\ell_a, \ell_b)\}}{\min\{h_a - \ell_a, h_b - \ell_b\}}, \tag{2}$$

where a, b are the two lowest predicted-P95 admitted candidates. For a zero-width interval, overlap is defined as zero; crossed or nonfinite quantiles invalidate a candidate. This explicit branch avoids a numerical epsilon in the denominator. With one candidate, only its dispersion is tested. Normalization by the narrower interval makes complete containment score one even when the other interval is much wider. Overlap is a decision feature, not a test of equal mean latency; maximizing dispersion over admitted actions supplies a conservative regime-level signal. With overlap threshold $\theta = 0.15$ and dispersion cap $\sigma_{\max} = 0.30$, the gate vetoes if $\text{ov}(a,b) > \theta$ or $\hat{u} > \sigma_{\max}$. These thresholds were selected on a predicate-template-disjoint held-out window and fixed before test evaluation; Section 4.4 quantifies their sensitivity.

Recall cells are keyed by dataset, hardware, workload family, requested k, and selectivity/correlation bin. Sparse cells back off to a coarser key; a candidate without a calibrated floor is unavailable. The floor constrains calibration-cell mean recall, and reported test recall averages per-query recall; neither is an individual-query recall certificate. Fallbacks are ranked within the current regime by zero recent audited recall-floor misses, then lower realized P95, then fewer SSD probes, with ties toward P3. A cache hit is rechecked against the current predicate regime, requested k, recall-adjusted fanout, and admission constraints. An empty or stale cache uses P3 only if its recall-adjusted candidate belongs to $\mathcal{A}(Q)$; otherwise the admission failure path applies. Appendix Table 6 collects the boundary cases.

3.3 Tiered storage and resource accounting

The hot tier uses HNSW [15]; the cold tier uses compressed SSD-backed ANN with PQ-style compression or SPANN-like partitioning [6, 2]. The default Medium configuration keeps 35% of vectors in the hot tier. The 10–70% sensitivity sweep changes this residency fraction under the same 128GB RAM and 1TB SSD caps; vectors, graph state, metadata, and buffers all consume the deployment budget. For an 85M-vector deployment, the two-tier index occupies 94GB RAM and 420GB SSD, versus 153GB RAM for an all-HNSW configuration at comparable recall.

Hot-hit probability and predicted cold probes affect both ranking and authorization. As hot-hit probability rises from 0.4 to 0.9, the observed P2-to-P3 boundary moves about four selectivity percentage points; PACER moves away from P2 beyond roughly 35 predicted cold probes per query. Tier migration uses telemetry, rate limits, and increased hotness for frequently revisited P3 frontiers. Recalibrating the latency models and recall cells after a residency change preserves the gate's meaning (Table 18).

P3 maintains separate routing and result eligibility: a nonmatching node can still connect the traversal to matching neighbors. Predicate failure excludes a node from results, not from routing. Frontier and visited state count toward the memory cap, and duplicate nodes are suppressed. Calibrated fanout and frontier limits determine the search effort; resource exhaustion is recorded as incomplete service, not as successful satisfaction of the recall target. Algorithm 1 gives this traversal invariant. Recall is measured against exact filtered search, as defined in Section 4.5.

3.4 Telemetry and calibration

Normal telemetry records the chosen plan, $(\hat{s}, \hat{r}, \hat{u})$, latency, cache events, frontier size, and veto decisions. Records are aggregated in 5-second windows. A predicted/observed P95 divergence above 15% over the latest 10K queries triggers asynchronous retraining on the latest 50K records, retaining the previous checkpoint for rollback. Headline comparisons freeze the checkpoint within each run; the drift experiment enables adaptation.

Mis-picks require counterfactual plan outcomes, and exact recall requires an oracle. We therefore distinguish the replay audit from ordinary service telemetry: the audit executes recall-feasible alternatives under reconstructed cache state and supplies the mis-pick and recall-miss labels. The one-minute safe-mode statistic is defined over such labeled decisions, not inferred from the latency of the chosen plan alone. Safe mode routes failure-band queries to feasible P3 above 15% mis-picks and exits below 5%; other regimes use threshold heuristics. Audit execution is accounted separately from the 0.18ms feature-to-dispatch controller cost. The main serving comparison includes the selected retrieval path, while the audit supports diagnosis and the controlled drift evaluation.

4 Evaluation

The central test changes the dispatch rule while keeping the predictor, recall adjustment, execution actions, and fallback ranking fixed. We then explain the gain through margin-conditioned regret and separate gate signals, test transfer to the closest planner and backend, and measure impact across workloads and resource profiles.

4.1 Setup and inference

Small uses 2×Intel Xeon Gold 6248R and 64GB RAM; Medium uses 2×AMD EPYC 7543 and 128GB; Cluster uses four 24-core nodes with 48GB each and 25GbE. All use Samsung PM9A3 NVMe SSDs, Ubuntu 22.04, pinned threads, NUMA-aware allocation, and predicate JIT, with GPUs disabled. EntDoc has 50M 768-dimensional vectors, EcomProd 100M 512-dimensional vectors, LegalDoc 20M 1024-dimensional vectors, and SciPaper 30M 768-dimensional vectors. Each has 4–8 metadata attributes; predicates cover temporal, categorical, hierarchical, and multi-constraint forms, 0.1–30% passing fractions, and correlation $r \in [.08, .52]$.

The enterprise mixture contains 15,000 de-identified queries collected over seven days from three RAG deployments. Normalized predicate templates are grouped before splitting into model-fitting, calibration/threshold-selection, and test windows, and before per-plan expansion. Thus all plan outcomes for a template remain in the same partition. SciPaper is evaluated separately. Default results use Medium+EntDoc, the frozen 5–15% band, and Recall@20 floor $\rho = .94$. Each arm uses five runs of 1,000 queries, a 30-second warm-up after page-cache reset, the same query cohort and cache controls, and a frozen planner checkpoint per run. Latencies describe retrieval service through result construction, including planning, predicate checks, ANN work, and SSD access; application generation is outside this boundary. Queue state is a model feature, while admission waiting and service latency are distinct quantities in Section 3.1.

We report the median of run-level P95/P99 estimates and 95% bootstrap intervals from 1,000 resamples of runs. The expanded 25-seed comparison pairs methods within each run; paired intervals resample run-level differences rather than treating pooled requests as independent runs.

4.2 Authorization with identical models and actions

Table 2: Authorization under identical models and P1–P4 actions on Medium+EntDoc. Online arms share recall tables, admission rules, query cohort, and fallback ranking. Brackets are run-level 95% intervals; OH is feature-to-dispatch overhead. Oracle is retrospective.

Controller	P95, ms	P99, ms	Recall	Mis-pick	SSD/q	Veto	OH, ms
Winner-only	49 [47,52]	78 [74,83]	.947	12.4%	41	0%	.12
Margin 10%	45 [43,48]	70 [66,74]	.948	10.1%	36	18.6%	.13
Margin 25%	43 [41,45]	66 [63,70]	.948	9.2%	33	31.2%	.13
Calibrated .70	42 [40,44]	64 [61,68]	.949	8.6%	31	24.8%	.15
PACER	37 [35,39]	56 [53,59]	.951	4.6%	26	21.7%	.18
Oracle	33 [31,35]	50 [47,53]	.956	0%	21	–	–

Table 2 changes only the ranking-to-execution rule. Winner-only dispatches the minimum predicted-P95 admitted action in $\mathcal{A}(Q)$, using the same recall, memory, and latency admission rules as the gated arms. The margin gate vetoes when $(\widehat{L}_{95,b} - \widehat{L}_{95,a})/\widehat{L}_{95,a} < m$, selecting $m = 25\%$ from

$\{5, 10, 15, 20, 25, 30, 40\}\%$. The confidence gate vetoes when isotonic-calibrated $P(L_a \leq L_b) < c$, selecting $c = .70$ from $\{.55, .60, .70, .80, .90\}$, with held-out expected calibration error .021. Selection precedes testing on template-disjoint windows. A mis-pick selects an action other than a measured fastest recall-feasible action for that query; equal-latency best actions count as ties. The retrospective oracle uses these measured outcomes to bound the value of dispatch, while the online arms use predicted admission costs.

Both standard gates improve winner-only. PACER reduces P95 a further 5–6ms with a 21.7% veto rate, compared with 31.2% for the best margin gate and 24.8% for calibrated confidence. Its 12ms gain over winner-only closes $(49 - 37)/(49 - 33) = 75\%$ of the oracle gap, and its 26 SSD reads/-query compare with winner-only's 41. The joint gate achieves lower tail latency with fewer vetoes, identifying the choice of rejected proposals as the useful intervention. Its feature-to-dispatch cost is .18ms, compared with .12–.15ms for the controls; these costs are included in the retrieval timer.

PACER's five-run P95/P99 are 37 [35,39]/56 [53,59]ms, versus 37 [36,38]/57 [55,58]ms over 25 seeds. The latter cohort gives paired P95 differences of -12 [-14,-10]ms against winner-only, -6 [-8,-4] against margin, -5 [-7,-4] against calibrated confidence, and -31 [-34,-28] against Qdrant. These effect intervals provide the primary tail-inference evidence; the distinct P99 estimates retain their respective cohort labels.

The mechanism is visible at comparable predicted margins. Among 1,142 of 5,000 queries with margins of 20–30%, 604 have overlap at most .15 and mean provisional-winner regret 3.1ms; the remaining 538 have higher overlap and mean regret 19.7ms. Regret is realized latency minus that of the fastest recall-feasible action under the same replay conditions. Figure 3 connects this separation at comparable margins to the joint value of the two signals.

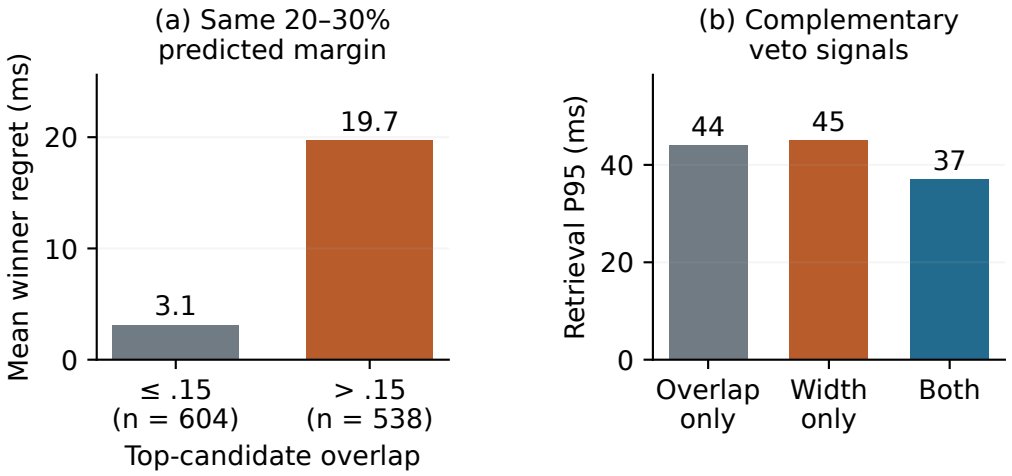

Figure 3: Why the dispatch rule matters on Medium+EntDoc. Left: queries in the same 20–30% predicted-margin range separate by top-candidate overlap into groups with different mean provisional-winner regret. Right: disabling either veto signal increases P95 with the same recall adjustment, action menu, and fallback. Bars show the reported group means (left) and median run-level P95 (right); they are distinct statistics.

4.3 Closest planners and backend composition

In the public-configuration checks, our GW* reimplementation reaches 89.5% plan-selection accuracy and our GateANN* reimplementation reduces SSD reads by 3.1× relative to Filtered-DiskANN. These checks characterize the reimplementations used here; the controlled comparisons below use the common resource budget. Under common recall tables, GW*'s 48ms is close to winner-only's 49ms; adding the gate yields 39ms (Table 3). The improvement transfers to a different learned planner.

Table 3: Planner transfer and backend composition on the same Medium+EntDoc cohort. * denotes a reimplementation; Table 8 separates public-configuration checks from this matched-budget comparison.

Arm	P95, ms	P99, ms	Recall	Mis-pick	SSD/q
GW*	54 [51,57]	84 [80,89]	.944	15.7%	47
GW* + recall tables	48 [46,51]	75 [71,79]	.949	13.1%	40
GW* + tables + gate	39 [37,42]	59 [56,63]	.950	5.9%	28
GateANN*	71 [67,76]	104 [98,111]	.943	–	19
PACER over GateANN*	36 [34,38]	55 [52,58]	.952	4.5%	20
PACER	37 [35,39]	56 [53,59]	.951	4.6%	26

GateANN* uses fewer SSD reads than PACER on Medium, but CPU predicate work leaves its total P95 higher. The composition's 36ms and PACER's 37ms have overlapping intervals. On I/O-bound Small+EntDoc, with 22% hot residency, GateANN* reaches 74 [70,79]ms, PACER 58 [55,62]ms, and their composition 53 [50,56]ms. Backend specialization and dispatch control are most complementary when cold I/O is the bottleneck. The original Medium backend reference is Qdrant at 68 [64,72]/101 [96,107]ms P95/P99 and .942 recall, yielding the 46%/45% headline reduction (Figure 5).

4.4 Band replication and gate robustness

For each selectivity/correlation cell, fixed-plan spread is the largest divided by the smallest run-summary P95 among recall-feasible P1–P4 actions. Applying the pre-controller $> 3\times$ criterion over $r \in [.08, .52]$ gives 4.6–15.3% on calibration and 5.1–15.8% on held-out EntDoc. Table 4 applies the same criterion to MoReVec [1], MS MARCO-v2 (12M vectors, 768 dimensions), and 10M LAION. Both the instability and the gain over winner-only recur; band edges vary with the workload.

Table 4: Criterion-defined public failure bands. Spread is a ratio with a run-level 95% interval; controller columns are P95 in ms, and recall is PACER Recall@20. All use the five-run protocol.

Workload	Band	Peak spread	Qdrant	Winner	PACER	Recall
MoReVec	6–13%	4.3 [3.7,5.0]	71	57	44	.949
MS MARCO-v2	5–14%	3.6 [3.1,4.2]	48	39	31	.957
10M LAION	5–16%	5.1 [4.4,5.9]	66	51	39	.946

Across the 6×5 threshold grid, P95 ranges from 37 to 47ms, recall from .947 to .954, and 22 of 30 settings achieve P95 at most 40ms. The worst setting still improves on Qdrant's 68ms by approximately 31%. Table 10 reports the two intersecting sensitivity slices. Fixed defaults cost at most 1.4ms (3.3%) against per-cell tuning on Small+EntDoc, Cluster+EcomProd, and Medium+LegalDoc. To isolate the two gate signals, overlap-only disables the dispersion veto and dispersion-only disables the overlap veto, preserving recall adjustment and fallback. Their P95 values are 44 and 45ms, compared with 37ms for both (Figure 3). Of 1,085 vetoes, 63.7% trigger overlap alone, 18.1% dispersion alone, and 18.2% both; dispersion-only cases have mean overlap .06 and mean $\hat{u} = .41$. Thus the joint gate handles both overlapping alternatives and separated candidates in a broad-tail regime.

The counterfactual audit covers 5,000 decisions and 1,085 vetoes, rerunning each provisional winner with the same reconstructed cache state. P3 receives 68.2% of fallbacks (Table 12). The original component-removal cohort retains its 50ms minus-fallback result, distinct from the 49ms winner-only cohort; removing P3 or restricting the menu to P1+P2 changes action availability (Table 11).

4.5 Quality, deployment breadth, and whole-mixture impact

The exact filtered oracle ranks all predicate-eligible vectors by exact distance. Per-query recall divides the number of retrieved oracle neighbors by $\min(k, N_\Pi)$; an empty eligible set is evaluated

by correct empty return and excluded from the mean neighbor-recall denominator. Aggregate Recall@20 is .951 in the headline cell. Across the separate predicate-pattern experiment, it spans .943–.978 (rounded .94–.98). Table 14 holds the latency budget fixed and exposes quality degradation; it is distinct from the common-recall-floor latency experiments. Both experiments use the exact filtered oracle as their recall reference.

PACER improves P95 by 22% on Small+EntDoc, 35% on Cluster+EcomProd, 31% on Medium+SciPaper, and 32% on Medium+LegalDoc against each cell's named backend reference (Table 17). On LegalDoc, zero-shot P95 is 66ms versus 51ms in-domain; 5K calibration queries reduce it to 53ms, recovering 22 of the 24ms in-domain advantage over Qdrant. On the hierarchy-aligned slice, P4 reaches 51ms versus P3's 59ms, while the controller reaches 49ms and selects P4 on 73.6% of queries (Tables 15–16).

A deployment cell is feasible when the median of its run-level P95 values is at most 80ms within the 1TB SSD cap. The 85M-vector configuration in Section 3.3 illustrates the resource tradeoff: the two-tier index uses 94GB RAM and 420GB SSD, compared with 153GB RAM for all-HNSW at comparable recall. In the hot-tier sweep, calibrated PACER retains a 44–46% P95 advantage over Qdrant as the band shifts from 4–13% to 7–19%. Reusing the 35% model at 10% residency raises P95 from 49 to 56ms and mis-picks from 6.9% to 12.9%; resource-profile calibration matters more than modest threshold adjustment. Under four-phase drift, recalibration plus fallback caps the hardest transition at 52ms versus 71ms frozen and 58ms with recalibration alone, recovering within 6–8% of steady-state P95 in 22/41/29 seconds after the three transitions (Figure 4).

The unweighted enterprise mixture has 4,338 in-band queries out of 15,000 (28.92%); deployment shares are 24.1/29.6/33.2%, and weighting by request volume gives 31.4%. The corresponding request-weighted veto rate is 8.9%, combining 21.7% in-band and 3.1% out-of-band. Across the whole request mixture, Qdrant-to-PACER P95 falls from 46 [44,49] to 34 [32,36]ms and P99 from 74 [70,79] to 52 [49,55]ms. The whole-mixture P95 improvement is 26%, compared with 46% in the failure band; including out-of-band requests preserves a substantial serving gain while distinguishing population impact from the stress-slice effect.

These experiments distinguish three operating decisions. A hardware or residency change calls for recalibrating the candidate costs and recall cells; a transient ambiguous ranking calls for a feasible fallback; persistent cold-I/O pressure favors composing the controller with a specialized backend. The distinction follows from the stale-profile penalty, joint-gate ablation, and Small+EntDoc composition, respectively. Table 13 maps these decisions to their resource boundaries and measured costs.

The deployment assumption is a recurring CPU workload with enough calibration coverage to select recall-feasible fanouts. Predicate evaluation and resource accounting are trusted, and authorization is evaluated as empirical retrieval reliability. Authorization combines model-based admission with a recent feasible fallback; the measured tail improvements apply to the evaluated resource profiles. Lower retrieval tails preserve downstream serving slack within these operating conditions.

5 Conclusion

The failure band exposes a consequential gap between a predicted winning margin and the realized cost of dispatch. PACER closes much of that gap by evaluating interval containment and normalized tail width after recall adjustment, then using a regime-local feasible fallback. The signals yield 44 and 45ms P95 separately and 37ms together; identical-model winner-only and calibrated-confidence controls reach 49 and 42ms. The gate improves a second learned planner and composes with SSD traversal, while whole-mixture P95 falls from 46 to 34ms against Qdrant. The resulting design principle is to make recall-feasible actions comparable, authorize their execution using both relative separation and tail dispersion, and recalibrate those decisions when the resource profile changes.

Acknowledgments and Disclosure of Funding

This work received no external funding. The authors declare no competing interests.

References

[1] Abylay Amanbayev, Brian Tsan, Tri Dang, and Florin Rusu. Filtered Approximate Nearest Neighbor Search in Vector Databases: System Design and Performance Analysis. *arXiv preprint arXiv:2602.11443*, 2026. doi: 10.48550/arXiv.2602.11443. URL https://arxiv.org/abs/2602.11443.

[2] Qi Chen, Bing Zhao, Haidong Wang, Mingqin Li, Chuanjie Liu, Zengzhong Li, Mao Yang, and Jingdong Wang. SPANN: Highly-efficient Billion-scale Approximate Nearest Neighborhood Search. In *Advances in Neural Information Processing Systems*, volume 34, pages 5199–5212. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/299dc35e747eb77177d9cea10a802da2-Paper.pdf.

[3] Zhuocheng Gan and Yifan Wang. Efficient Filtered-ANN via Learning-based Query Planning. *arXiv preprint arXiv:2602.17914*, 2026. doi: 10.48550/arXiv.2602.17914. URL https://arxiv.org/abs/2602.17914.

[4] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv preprint arXiv:2312.10997*, 2023. doi: 10.48550/arXiv.2312.10997. URL https://arxiv.org/abs/2312.10997.

[5] 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 2023*, pages 3406–3416. ACM, 2023. doi: 10.1145/3543507.3583552. URL https://doi.org/10.1145/3543507.3583552.

[6] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product Quantization for Nearest Neighbor Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/tpami.2010.57. URL https://doi.org/10.1109/tpami.2010.57.

[7] Vladimir Karpukhin, Barlas Oguz, 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)*, pages 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

[8] 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*, pages 39–48. Association for Computing Machinery, 2020. doi: 10.1145/3397271.3401075. URL https://doi.org/10.1145/3397271.3401075.

[9] Nakyung Lee, Soobin Cho, Jiwoong Park, and Gyuyeong Kim. GateANN: I/O-Efficient Filtered Vector Search on SSDs. *arXiv preprint arXiv:2603.21466*, 2026. doi: 10.48550/arXiv.2603.21466. URL https://arxiv.org/abs/2603.21466.

[10] 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*, volume 33, pages 9459–9474, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

[11] Zhaoheng Li, Silu Huang, Wei Ding, Yongjoo Park, and Jianjun Chen. SIEVE: Effective Filtered Vector Search with Collection of Indexes. *Proceedings of the VLDB Endowment*, 18(11):4723–4736, 2025. doi: 10.14778/3749646.3749725. URL https://doi.org/10.14778/3749646.3749725.

[12] Anqi Liang, Pengcheng Zhang, Bin Yao, Zhongpu Chen, Yitong Song, and Guangxu Cheng. UNIFY: Unified Index for Range Filtered Approximate Nearest Neighbors Search. *Proceedings of the VLDB Endowment*, 18(4):1118–1130, 2024. doi: 10.14778/3717755.3717770. URL https://doi.org/10.14778/3717755.3717770.

[13] Yanjun Lin, Kai Zhang, Zhenying He, Yinan Jing, and X. Sean Wang. Survey of Filtered Approximate Nearest Neighbor Search over the Vector-Scalar Hybrid Data. *arXiv preprint arXiv:2505.06501*, 2025. doi: 10.48550/arXiv.2505.06501. URL https://arxiv.org/abs/2505.06501.

[14] Hanwen Liu, Shashank Giridhara, and Ibrahim Sabek. Conformal Prediction for Verifiable Learned Query Optimization. *Proceedings of the VLDB Endowment*, 18(8):2653–2666, 2025. doi: 10.14778/3742728.3742755. URL https://doi.org/10.14778/3742728.3742755.

[15] Yu A. Malkov and D. 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.

[16] Ryan Marcus, Parimarjan Negi, Hongzi Mao, Nesime Tatbul, Mohammad Alizadeh, and Tim Kraska. Bao: Making Learned Query Optimization Practical. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1275–1288. ACM, 2021. doi: 10.1145/3448016.3452838. URL https://doi.org/10.1145/3448016.3452838.

[17] Liana Patel, Peter Kraft, Carlos Guestrin, and Matei Zaharia. ACORN: Performant and Predicate-Agnostic Search Over Vector Embeddings and Structured Data. *Proceedings of the ACM on Management of Data*, 2(3):1–27, 2024. doi: 10.1145/3654923. URL https://doi.org/10.1145/3654923.

[18] Saravanan Thirumuruganathan, Suraj Shetiya, Nick Koudas, and Gautam Das. Prediction Intervals for Learned Cardinality Estimation: An Experimental Evaluation. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 3051–3064. IEEE, 2022. doi: 10.1109/icde53745.2022.00274. URL https://doi.org/10.1109/icde53745.2022.00274.

[19] Jintao Zhang, Chao Zhang, Guoliang Li, and Chengliang Chai. PACE: Poisoning Attacks on Learned Cardinality Estimation. *Proceedings of the ACM on Management of Data*, 2(1):1–27, 2024. doi: 10.1145/3639292. URL https://doi.org/10.1145/3639292.

[20] Chaoji Zuo, Miao Qiao, Wenchao Zhou, Feifei Li, and Dong Deng. SeRF: Segment Graph for Range-Filtering Approximate Nearest Neighbor Search. *Proceedings of the ACM on Management of Data*, 2(1):1–26, 2024. doi: 10.1145/3639324. URL https://doi.org/10.1145/3639324.

A Operational contract and controlled comparisons

The component interfaces in Table 5 separate request normalization, estimation, authorization, and execution. This separation places quantile prediction and gate decisions in the planner while retaining service telemetry at the engine boundary.

Table 5: Inputs and outputs at the request-to-execution interfaces.

Component	Input	Output
Frontend	Query, predicate, SLO	Normalized request
Estimator	Request, sketches, candidate probe	Selectivity and correlation
Planner	Features, recall cells, budgets	Quantiles, gate decision, plan/fanout
Engine	Plan/fanout, resource caps	Results and service telemetry

Table 6 makes the operational checks explicit. Admission precedes both ranking and gating, and a vetoed proposal can only be replaced by a currently feasible action; audit-driven recovery additionally requires labeled decisions.

Table 6: Authorization contract and failure handling.

Stage	Rule and boundary behavior
Recall lookup	Smallest calibrated fanout meeting the cell-mean floor; coarser-key backoff for sparse cells; unavailable floor excludes a candidate.
Feasibility	Equation 1 constructs recall/memory-feasible $\mathcal{F}$ and latency-admitted $\mathcal{A}$; ranking and gating both use $\mathcal{A}$. Deployment RAM/SSD and per-query state are accounted separately.
Quantile handling	Crossed or nonfinite quantiles invalidate a candidate; a zero-width interval has zero overlap by definition. Dispersion is maximized over admitted candidates only.
Gate	Veto when top-two overlap exceeds .15 or maximum dispersion exceeds .30; a single candidate uses only the dispersion test.
Fallback	Recheck current query key, fanout, recall, and budgets. Rank recent feasible plans by audited recall misses, P95, SSD probes; ties favor P3. Empty/stale cache uses P3 only if admitted under Equation 1; otherwise use the admission failure path.
Safe mode	On labeled audit decisions, enter above 15% one-minute mis-picks and exit below 5%. Normal single-plan telemetry does not provide counterfactual labels.
Recalibration	A 15% predicted/observed P95 discrepancy on 10K queries triggers fitting on the latest 50K records; preserve the prior checkpoint for rollback.

A.1 Traversal and comparison boundaries

Algorithm 1 specifies the bounded-state P3 traversal used in the candidate menu. Traversing predicate-ineligible nodes preserves routing paths to eligible results, while explicit frontier, memory, and time checks define how incomplete service is returned.

Algorithm 1 P3: routing through eligible and ineligible nodes with bounded state.

Require: q, Π, k, calibrated expansion/fanout limits, frontier cap, time and memory caps
Initialize frontier H, visited set V, and result heap R; account for all allocated state
Insert seed into H and mark it discovered in V
while $H \neq \emptyset$ and calibrated expansion budget remains **do**
  if time or memory cap would be exceeded **then**
    Mark service incomplete and stop
  end if
  Pop u from H and evaluate its distance
  if $\Pi(u)$ **then**
    Insert u into R if it improves the eligible top-k
  end if
  for each routing neighbor v of u **do**
    if $v \notin V$ and backend search/pruning rule admits v **then**
      Reserve frontier and visited memory; if unavailable, mark service incomplete and terminate traversal
      Mark v in V and insert into the bounded frontier
    end if
  end for
end while
return eligible results R and completion status

Predicate failure affects result eligibility, while routing follows the graph's neighbor relation. The frontier cap and expansion budget are part of the recall-calibrated configuration; a resource stop is not reported as a recall certificate.

Table 7 connects each comparison to the decision it changes. The learned-planner and cold-tier controls separate the value of authorization from improvements to ranking or SSD traversal.

Table 7: Decision mechanisms, controlled evidence, and implications.

System	Decision mechanism	Controlled evidence	Consequence
GW*	Learned ranking	48ms with recall tables; 39ms with gate	Gate improves dispatch.
GateANN*	Routing through non-matches; SSD vector reads	Small: 74ms; composition: 53ms	Cold-tier and dispatch gains compose.
Amanbayev et al.	Workload characterization	Public bands: Table 4	Diagnosis guides dispatch.
PACER	Recall adjustment, dual gate, fallback	Winner/calibrated/dual: 49/42/37ms; single: 44/45ms	Joint signals improve veto selection.

A.2 Baseline controls and realized regret

Table 8 records the baseline configurations and the scope of the two reimplementations. Reimplementation checks characterize their behavior in the public configuration; the common-budget experiment separately applies shared recall and resource controls.

Table 8: Baseline families and reimplementation checks, evaluated separately from the common-budget workload comparison.

Comparator	Configuration and comparison boundary
FF / AF	Filter-first HNSW/IVF versus ANN-first HNSW; AF starts at $k' = 200$ and expands on misses.
Milvus 2.3.4	IVF_FLAT, bitmap/filter scan, $n_{\text{list}} = 4096$.
Qdrant 1.7.3	HNSW with payload filtering. HNSW reference settings: $M = 32$, $\text{ef}_c = 200$, $\text{ef}_s = 128$, with recall-target tuning.
ACORN	Predicate-agnostic HNSW graph search [17]; graph connectivity and filter-aware traversal, with recall-target search tuning.
SIEVE	Workload-aware collection of predicate-serving indexes [11]; collection construction and query routing, under the common memory cap.
GW*	Reimplementation of learned planning [3]. Our public-configuration check gives 89.5% accuracy. Matched-budget evaluation separately adds common recall tables, then the gate.
GateANN*	Reimplementation of SSD traversal [9]. Our public-configuration check gives 3.1× fewer reads than Filtered-DiskANN. Composition substitutes its cold tier beneath the controller.

The comparable-margin groups in Table 9 show why a predicted margin alone is an incomplete dispatch signal. Higher overlap separates requests with much larger provisional-winner regret despite their comparable predicted margins, supporting the use of overlap as an additional authorization signal.

Table 9: Regret under matched replay conditions. The margin subgroups contain 1,142 of 5,000 queries; regret uses the fastest recall-feasible action, with equal-latency actions tied.

Comparable-margin subgroup	Query count	Mean provisional-winner regret, ms
Margin 20–30%, overlap $\leq .15$	604	3.1
Margin 20–30%, overlap $> .15$	538	19.7

A.3 Thresholds and component contributions

Table 10 varies one gate threshold at a time around the selected pair. The nearby settings retain similar latency, complementing the full-grid result reported in the main paper and showing that the benefit is not confined to a single threshold pair.

Table 10: Intersecting threshold-sensitivity slices on the held-out Medium+EntDoc cohort. The full 30-cell sweep spans 37–47ms P95, with 22 settings at most 40ms and recall .947–.954. The shared center is counted once.

θ, fixed $\sigma_{\max} = .30$	.05	.10	.15	.20	.30	.50
P95 (ms)	40	38	37	37	39	43
$\sigma_{\max}$, fixed $\theta = .15$	.15	.20	.30	.40	.60	
P95 (ms)	39	38	37	38	41	

Table 11 separates action-menu changes from controller changes. Removing P3, restricting the menu, or removing fallback each increases P95, demonstrating that both feasible alternatives and the decision to use them contribute to the observed result.

Table 11: Original failure-band component ablations on Medium+EntDoc. Minus-P3 and P1+P2 alter action availability; the heuristic preserves P1–P4. The 50ms minus-fallback result is from this cohort, while Table 2 uses the separate same-menu controller cohort.

Variant	P95 (ms)	Δ	Note
Full (P1–P4 + fallback)	37	–	reference
minus P3	46	+24%	no interleaving
minus fallback	50	+35%	risky mis-picks
P1+P2 only	52	+41%	binary limit
Heuristic (no learned model)	56	+51%	threshold-based

With $\pm 20\%$ selectivity noise, mis-picks are 5.1% and P95 increases 3.6%; at $\pm 40\%$, mis-picks remain below 8% and P95 degradation below 6%. Correlation noise of $\pm 30\%$ yields 5.9% mis-picks and 4.1% P95 degradation.

A.4 Veto accounting and operating boundaries

Table 12 follows the intervention from trigger to fallback and recovery. Explicit trigger and destination counts identify the proposals most often rejected and the alternatives used, while the request-weighted rate distinguishes band-specific intervention from whole-mixture frequency.

Table 12: Veto triggers, destinations, and recovery. Conditional percentages are rounded and use vetoes as their denominator.

Audit quantity	Outcome and denominator
In-band decisions / vetoes	5,000 / 1,085; veto rate 21.7%
Veto outcome ($\pm$2ms neutral)	Correct/neutral/harmful: 78.4/13.9/7.7% of 1,085 vetoes
Trigger	Overlap only 63.7%; dispersion only 18.1%; both 18.2% of vetoes
Vetoed proposal	P2/P4/P1/P3: 61.3/22.9/11.4/4.4% of vetoes
Fallback destination	P3/P1/P4/P2: 68.2/24.5/5.9/1.4% of vetoes
Recent feasible cache	Usable in 97.8% of vetoes; other 2.2% use the P3 fallback
Out-of-band / mixture rate	3.1% of out-of-band requests; 8.9% in the request-weighted mixture
Safe mode	No headline activations; two drift activations lasting 37s and 19s at one-minute mis-pick peaks 17.4% and 15.8%

Table 13 maps these outcomes to the boundary being controlled. It distinguishes query-level ambiguity from deployment-profile changes, linking each perturbation to the relevant evidence and recovery mechanism.

Table 13: Dispatch boundaries and measured outcomes under workload perturbations; admission and authorization follow Equations 1–2.

Boundary and trust assumption	Perturbation and control	Measured outcome
Recall/deadline slack; trusted predicates and recall cells	Correlation shifts; calibrated gate/GW* on template-disjoint queries	.951 recall, .18ms cost, 4.6% mis-picks (Tables 2–3)
Feasible fallback; trusted replay labels and cache key	Noisy estimates; single-signal gates and threshold/noise sweeps	21.7% vetoes with estimate-noise robustness (Tables 10, 12)
RAM/SSD and retrieval SLO; trusted resource accounting	Residency/workload shifts; frozen and recalibration-only controls	44–46% gain over Qdrant; stale profile raises P95 from 49 to 56ms (Table 18, Figure 4)

B Quality, workload transfer, and resource profiles

The fixed-latency comparison in Table 14 isolates retrieval quality across predicate patterns. PACER attains the highest recall in each evaluated pattern; these points answer a different question from the main paper's latency comparison at a common recall floor.

Table 14: Recall@20 on Medium+EntDoc under identical latency budgets. These quality-constrained service points are separate from the common-recall-floor latency comparisons.

Pattern (selectivity)	FF	AF	Qdrant	ACORN	SIEVE	PACER
Temp. (0.5%)	0.971	0.818	0.948	0.962	0.954	0.978
Cat. (5%)	0.938	0.782	0.912	0.928	0.919	0.958
Hier. (12%)	0.891	0.847	0.886	0.912	0.905	0.943
Multi (2%)	0.922	0.811	0.901	0.918	0.908	0.962

Table 15 measures how adaptation recovers performance on LegalDoc. Increasing the calibration set moves both tail latency and recall toward the in-domain result, providing an operational path for transfer from the EntDoc-trained controller.

Table 15: LegalDoc transfer. Calibration uses 1K or 5K adaptation queries; evaluation is on held-out queries. Brackets are 95% run-level intervals.

Condition	P95, ms	P99, ms	Recall	Mis-pick	SSD/q
Qdrant	75 [71,80]	110 [104,117]	.941	–	52
In-domain	51 [48,54]	76 [72,81]	.954	5.2%	29
Zero-shot EntDoc	66 [62,71]	97 [91,104]	.944	14.8%	43
+1K calibration	57 [54,61]	84 [79,90]	.949	9.1%	35
+5K calibration	53 [50,56]	79 [75,84]	.952	6.0%	30

The hierarchy-aligned slice in Table 16 explains the role of P4. The controller selects this action most frequently where it is effective, while retaining other actions to improve the aggregate latency over a fixed-P4 policy.

Table 16: Hierarchy-aligned LegalDoc slice. Fixed-plan mis-pick is undefined; PACER mis-pick is 4.8%. Its selection shares are P4 73.6%, P3 21.8%, and P1 4.6%.

Plan	P95, ms	P99, ms	Recall	SSD/q
P1	68 [64,73]	99 [93,106]	.943	37
P2	91 [85,98]	134 [125,144]	.941	71
P3	59 [56,63]	87 [82,93]	.947	28
P4	51 [48,54]	74 [70,79]	.953	16
PACER	49 [46,52]	71 [67,76]	.955	17

B.1 Deployment breadth and residency

Table 17 reports each deployment with its named backend reference under the common recall floor. The gains span hardware and dataset changes, with the comparator retained explicitly because different deployments stress different serving costs.

Table 17: Deployment comparisons at the common Recall@20 floor of .94. Each backend reference is named; confidence intervals are over runs. LegalDoc appears in Table 15.

Deployment / arm	P95, ms	P99, ms	Recall	SSD/q
Medium+EntDoc / PACER	37 [35,39]	56 [53,59]	.951	26
Medium+EntDoc / Qdrant	68 [64,72]	101 [96,107]	.942	44
Small+EntDoc / PACER	58 [55,62]	87 [82,93]	.946	31
Small+EntDoc / GateANN*	74 [70,79]	111 [105,118]	.940	27
Cluster+EcomProd / PACER	46 [43,49]	69 [65,74]	.943	24
Cluster+EcomProd / Qdrant	71 [67,76]	106 [100,113]	.940	46
Medium+SciPaper / PACER	42 [40,45]	63 [59,67]	.949	22
Medium+SciPaper / ACORN	61 [58,65]	90 [85,96]	.943	38

PACER mis-pick rates on Small+EntDoc, Cluster+EcomProd, and Medium+SciPaper are 6.3%, 5.8%, and 5.5%, respectively.

Table 18 varies hot-vector residency within fixed RAM and SSD caps. Lower residency increases SSD work and veto frequency, while recalibrated dispatch retains the latency advantage across the evaluated resource profiles.

Table 18: Hot-vector residency sweep under 128GB RAM and 1TB SSD caps. Each setting uses its calibrated resource profile.

Hot	PACER P95, ms	Qdrant P95, ms	Band	SSD/q	Veto
10%	49 [46,52]	88 [83,94]	4–13%	44	27.9%
20%	41 [39,44]	76 [72,81]	5–14%	33	24.3%
30%	38 [36,40]	70 [66,74]	5–15%	28	22.4%
35%	37 [35,39]	68 [64,72]	5–15%	26	21.7%
55%	35 [33,37]	64 [60,68]	6–17%	18	19.8%
70%	34 [32,36]	61 [58,65]	7–19%	12	18.1%

B.2 Recovery across workload phases

Table 19 separates transition segments from steady operation. The second transition has the largest temporary latency and mis-pick increase; the subsequent phase returns close to the initial operating point, consistent with the recovery behavior in Figure 4.

Table 19: Four-phase drift: run-level interval summaries for steady and transition segments. One-minute safe-mode peaks in Table 12 are distinct from these segment aggregates.

Segment	P95, ms	P99, ms	Recall	Mis-pick	SSD/q
Phase 1	37 [35,39]	56 [53,59]	.951	4.6%	26
$1 \rightarrow 2$	43 [40,46]	65 [62,69]	.948	7.8%	31
Phase 2	38 [36,40]	58 [55,61]	.950	4.9%	27
$2 \rightarrow 3$	52 [49,55]	79 [75,84]	.944	11.6%	39
Phase 3	39 [37,42]	60 [57,64]	.949	5.6%	29
$3 \rightarrow 4$	45 [42,48]	68 [64,72]	.947	8.9%	34
Phase 4	38 [36,40]	58 [55,62]	.951	4.7%	25

Figure 4 places the segment summaries alongside frozen and recalibration-only controls. The comparison attributes the strongest transition recovery to combining recalibration with feasible fallback.

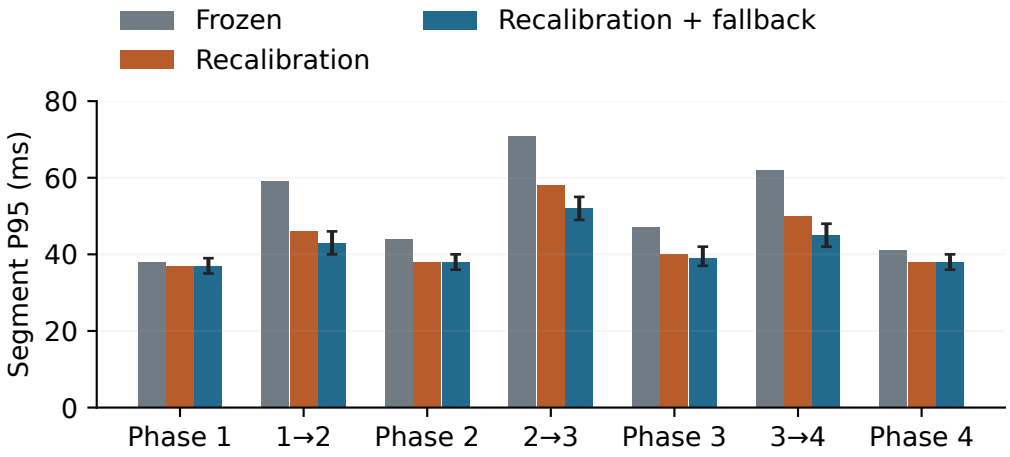

Figure 4: Steady and transition segment P95 under four-phase drift. Error bars show PACER run-level 95% intervals. Recovery after the three transitions takes 22/41/29s; one-minute safe-mode peaks are separate from these segment summaries.

C Additional views of latency and workload prevalence

Figure 5 displays the failure-band tail comparison with uncertainty. The common recall floor makes the latency reduction interpretable as a serving improvement at matched retrieval quality.

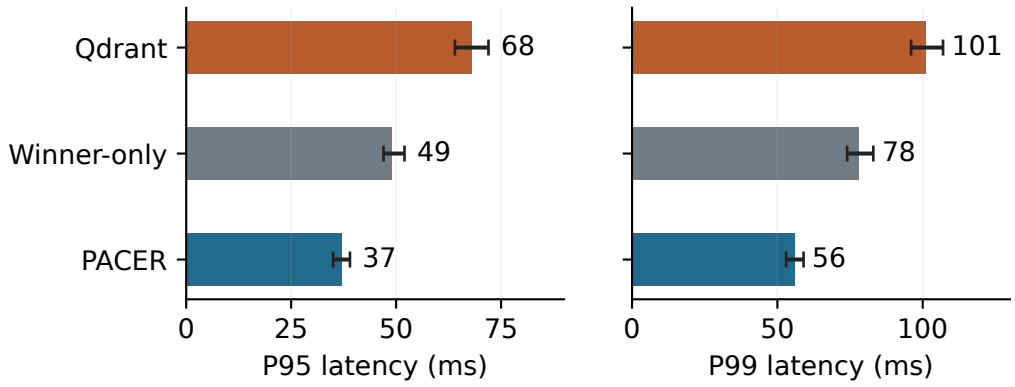

Figure 5: Medium+EntDoc failure-band quantiles at a common Recall@20 floor of .94, with run-level 95% intervals. Bars are aggregate band summaries, not within-band curves. Qdrant is the backend reference for the 46%/45% P95/P99 reductions.

Figure 6 relates this stress regime to the workload population. Request weighting changes prevalence because deployment traffic volumes differ; this distinction motivates reporting the whole-mixture result separately from the band-specific result.

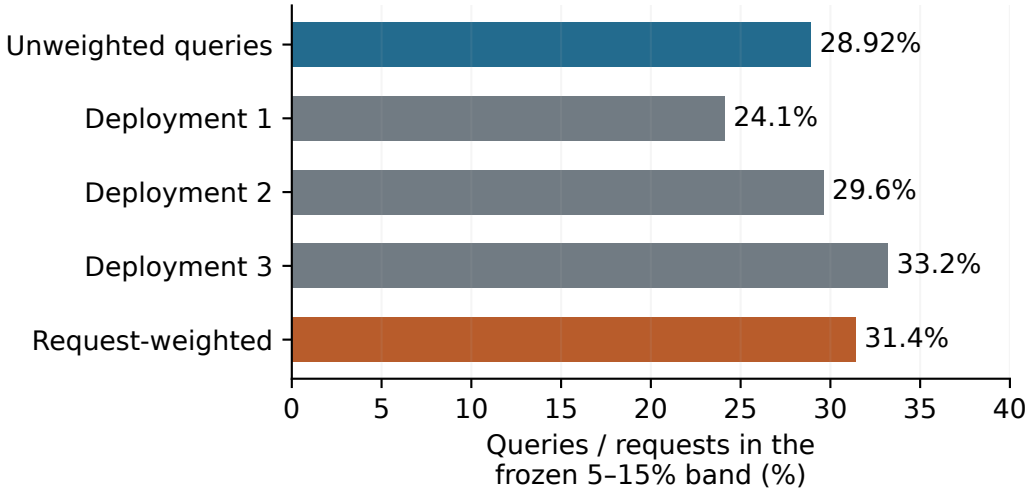

Figure 6: Marginal band prevalence. The unweighted population contains 4,338 in-band queries out of 15,000; request weighting gives 31.4%. Deployment-specific percentages have their own query populations.

Table 20 compares the primary run-level inference with a larger seed cohort and a pooled-query sensitivity view. The stable P95 point estimate across cohorts supports the tail comparison, while run-level resampling preserves the experimental unit used for the headline intervals.

Table 20: PACER tail inference on Medium+EntDoc. Pooled-query resampling is a sensitivity view and does not replace the run-level intervals. Brackets denote 95% intervals.

Cohort / summary	P95, ms	P99, ms
Five-run, run-level bootstrap	37 [35,39]	56 [53,59]
25-seed, run-level bootstrap	37 [36,38]	57 [55,58]
Five-run, pooled-query bootstrap	37 [36,38]	56 [54,58]

Figure 7 resolves the controller's feature-to-dispatch cost. Its measured overhead is included in retrieval latency, so the main service comparison accounts for the cost of the authorization decision itself.

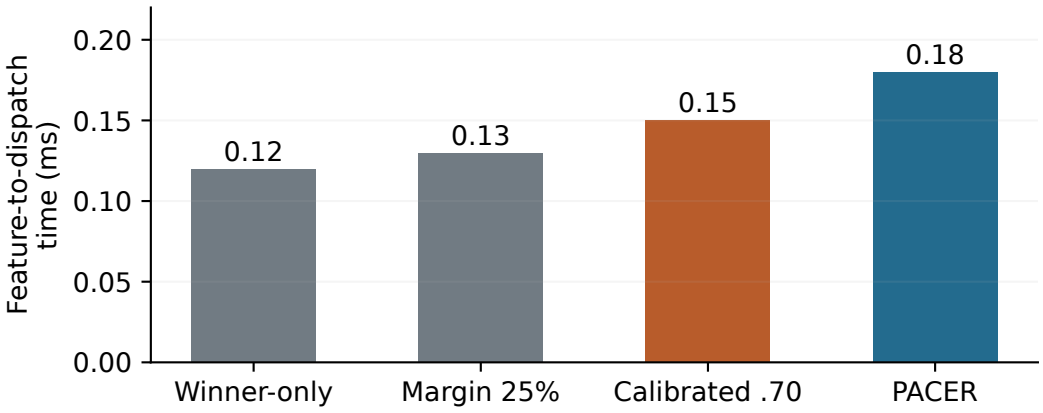

Figure 7: Feature-to-dispatch overhead on Medium+EntDoc. Candidate-probe retrieval and counterfactual audit execution are separate from this controller timer. Veto rates and fallback destinations are reported with explicit denominators in Table 12.

NeurIPS Paper Checklist

1. Claims

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the failure-band criterion, dual gate, identical-model effect, public replication, and whole-mixture result (Sections 1 and 4).

2. Limitations

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The final paragraph of Section 4 states recurring-workload, calibration, and trusted-component assumptions; Section 4.5 reports the stale-profile penalty.

3. Theory assumptions and proofs

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper presents an operational formulation and empirical systems evaluation, without theorem-proof claims.

4. Experimental result reproducibility

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [No]

   Justification: The paper specifies controlled arms, recall floors, split discipline, and aggregate results, but does not fully specify the public-workload construction and model-fitting configuration.

5. Open access to data and code

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Enterprise traces are access-restricted; the manuscript provides experimental protocols and aggregate results.

6. Experimental setting/details

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [No]

   Justification: Hardware, model families, gate thresholds, and split discipline are described; exact fitting hyperparameters and split allocations are not fully specified.

7. Experiment statistical significance

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Sections 4.1 and 4.2 report run-level bootstrap intervals and paired 25-seed differences. Tables retain separate cohorts and confidence intervals.

8. Experiments compute resources

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [No]

Justification: Section 4.1 states hardware, run counts, and warm-up; complete training and experiment durations are not reported.

9. Code of ethics

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses de-identified workload traces and public retrieval workloads; no human-subject intervention is performed.

10. Broader impacts

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The evaluation addresses retrieval efficiency and empirical reliability within the stated operating conditions; societal impacts are not separately analyzed.

11. Safeguards

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The work evaluates retrieval execution control, without releasing a generative model or a new sensitive dataset.

12. Licenses for existing assets

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: Existing retrieval systems and methods are cited in Section 2 and Table 8; the manuscript does not enumerate all third-party asset licenses.

13. New assets

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The submission introduces no separately distributed dataset, model, or software asset.

14. Crowdsourcing and research with human subjects

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This research does not involve crowdsourcing or research with human subjects.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This research does not involve human-subject experiments.

16. Declaration of LLM usage

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not components of the controller, experiments, or evaluation protocol.