JCTA Journal of Computing Theories and Applications ISSN:3024-9104

Research Article

TRIASE-SLO: Risk-Gated and SLO-Aware Routing Between FP16 and INT4 Language-Model Workers for Radiology Impression Serving

Junta Zeniarja [1,2], Guruh Fajar Shidik [1,2,*], Heru Suhartanto [3], and Aris Marjuni [1,2]

[1] Research Center for Intelligent Distributed Surveillance and Security (IDSS), Faculty of Intelligent Technology, Universitas Dian Nuswantoro, Semarang 50131, Central Java, Indonesia; e-mail : junta@dsn.dinus.ac.id; guruh.fajar@research.dinus.ac.id; aris.marjuni@dsn.dinus.ac.id
[2] Faculty of Intelligent Technology, Universitas Dian Nuswantoro, Semarang 50131, Central Java, Indonesia; e-mail : junta@dsn.dinus.ac.id; guruh.fajar@research.dinus.ac.id; aris.marjuni@dsn.dinus.ac.id
[3] Faculty of Computer Science, Universitas Indonesia, Depok 16424, West Java, Indonesia; e-mail : heru@cs.ui.ac.id
* Corresponding Author : Guruh Fajar Shidik

Abstract: Language models can draft radiology impressions, but a 4-bit copy of a model does not always reproduce the output of its full-precision version. TRIASE-SLO routes each request between FP16 and AWQ INT4 copies of one LoRA-adapted Qwen2.5-3B-Instruct model. A frozen rule-based hazard score computed from the FINDINGS text sends every request scoring at least 0.5 to FP16, and a secondary latency-aware score places the remaining requests. We evaluated the design in a locked confirmation in which every arm reserved the same two A100 MIG slices: four arms, seven loads and 12 paired deployment blocks of 450 previously inspected test reports (151,200 requests, none failed). The gate improved agreement with FP16. With full TRIASE-SLO routing, agreement with finding-bearing FP16 references was 99.11–99.49%, compared with 66.15–82.31% under hazard-blind routing, while 34–48% of requests still used INT4. A consistent serving-efficiency advantage was not demonstrated: full routing did not improve tail latency or throughput over two FP16 workers (median p95 change +0.33%; slower at C = 2 after Holm correction), and a consistent speed advantage of INT4 appeared only under hazard-blind routing. The demonstrated benefits of the INT4 copy are a smaller checkpoint and a larger KV-cache capacity. The incremental value of the score remains uncertain, because its prespecified C = 8 throughput benefit over gate-only routing (+1.86%, 95% CI −3.27% to +7.27%) was not confirmed. No INT4-using arm met the zero-drift engineering floor. All quality measures are rule-based FP16-consistency proxies from one public dataset, not evidence of clinical correctness or safety.

Keywords: FP16–INT4 routing; LLM serving; Mixed precision; Multi-Instance GPU; Quantization; Radiology impression generation; Request routing; Service-level objective.

Received: September, 20th 2026
Revised: November, 3rd 2026
Accepted: November, 11th 2026
Published: November, 19th 2026

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) licenses (https://creativecommons.org/licenses/by/4.0/)

1. Introduction

Radiology reports end with an impression that states what the findings mean, and large language models (LLMs) can now draft this passage from the findings text [1], [2]. Serving such a model is a separate problem: requests arrive at uneven rates and tail latency grows with the queue [3], and a common response is a reduced-precision copy of the model. For each request, a dispatcher must then choose between a 16-bit floating-point (FP16) copy and a 4-bit integer (INT4) copy of the same fine-tuned model, knowing that omitting some findings from the generated impression would be costly. The methodological problem is that existing routing approaches select models or precisions to meet latency, cost or average-quality objectives, but they do not impose a domain-specific, input-side constraint that keeps finding-bearing radiology requests away from the reduced-precision copy.

Post-training quantizers such as GPTQ, SmoothQuant and activation-aware weight quantization (AWQ) store weights in four or eight bits [4]–[6], and serving systems co-designed with quantization report higher throughput [7]. A quantized model is nevertheless a different model: its output may remain close to the full-precision output on average, but quality loss depends on the model, bit width and task [8]. Its speed is not constant either. In our serial measurements on the same GPU slice, the INT4 worker had a lower 95th-percentile (p95) latency than FP16 at concurrency 1 but a higher one at concurrency 8 and 16 (Section 5.6). INT4 is therefore a storage-efficient option whose latency benefit depends on load, not a uniformly faster worker.

Two lines of work come close to this decision (Section 2). Serving systems and LLM schedulers such as Clipper, Clockwork, INFaaS, SOLA, DistServe and Sarathi-Serve select model variants or schedule execution to meet latency, accuracy or goodput targets [9]–[14]; INFaaS already chooses among optimized variants, so precision-aware serving is not itself new. Model routers such as FrugalGPT, Hybrid LLM, RouteLLM and kNN routers decide which model answers a query from cascades or from learned predictions of answer quality and cost [15]–[18]. These designs do not address the FP16–INT4 problem studied here for three concrete reasons. First, their objectives are aggregate (a latency percentile, mean quality or cost), whereas the harm of interest is a rare, request-specific omission of a finding. Second, a quality predictor learned from preference or quality labels receives little signal when the two candidates are the same model at two precisions and differ on only a small fraction of requests. Third, none of them encodes a fixed clinical-domain rule stating, before any output exists, that certain inputs must not be served by the cheaper copy. These systems could be extended with such a rule; we claim only that this concrete constraint has not been specified and evaluated.

A simple example shows the intended behavior. A findings text that contains an unnegated mention of pneumothorax receives a hazard score of 1.0 and is always sent to FP16, even when the INT4 worker is idle and would answer sooner. A findings text in which no listed finding is matched as present or uncertain, such as one describing clear lungs and a normal heart size, receives the default score 0.05 and may be served by either worker, whichever the latency-aware score prefers. The two requests are treated differently because an important omission is only possible when the expected impression names a finding. The rule is lexical and can misclassify an input, and the FP16 output is a consistency reference, not clinical ground truth.

TRIASE-SLO combines this input-side gate with a latency-aware score, which accounts for a latency service-level objective (SLO), for requests below the gate. The two workers hold FP16 and AWQ INT4 copies of the same adapted model on two NVIDIA A100 Multi-Instance GPU (MIG) slices [19]. An earlier version of this study compared TRIASE-SLO with a deployment in which only one FP16 worker was active, so its latency and throughput gains could reflect the second worker rather than the routing policy. The present evaluation therefore compares all arms under an equal reservation of two identical slices, in a confirmation whose analysis plan was locked before test inference. Under this equal reservation, the evaluation measures how much agreement with FP16 the gate preserves while traffic still reaches INT4, whether mixed-precision routing changes latency or throughput relative to two FP16 workers, and whether the latency-aware score adds throughput beyond gate-only routing; supporting stress tests probe restarts, a second checkpoint and controlled faults. The contributions of this work are as follows:

1. An input-side precision gate for FP16/INT4 routing of radiology impression requests, specified in full with executable rules (Section 3.2, Table 1), and a two-worker form of the latency-aware score that makes its decision rule explicit (Section 3.3).
2. A locked, equal-reservation confirmation of 151,200 requests showing that the gate preserves consistency with FP16 where hazard-blind routing does not, that mixed-precision routing gives no consistent serving advantage over two FP16 workers, and that the additional benefit of the score is not confirmed.
3. An analysis of the risk–efficiency trade-off and of the residual inconsistency, showing that a consistent speed advantage of INT4 appears only without the gate and that the remaining omissions concentrate in a few low-hazard inputs served by INT4.

4. A transparent account of negative results (an unmet zero-drift floor, zero SLO attainment in a second-checkpoint stress test and losses under controlled faults) with a public result package for independent checking.

Section 2 reviews related work, Section 3 presents the method, Section 4 describes the experimental setup, Section 5 reports and discusses the results, and Section 6 concludes.

2. Related Work

2.1. Impression Generation and Its Evaluation

The Indiana University chest X-ray collection (OpenI) is one of the few public sets that pair radiology reports with images [20]; it has been used mostly for image-to-report generation [21] and also for findings-to-impression summarization [1], and adapted general-purpose models can match or exceed expert summaries in reader studies [2].

Evaluation is harder than generation. ROUGE measures word overlap [22] and can miss a clinically important omission. The CheXpert labeler extracts 14 observations with mention, negation, and uncertainty rules [23], in the tradition of NegEx [24] and NegBio [25]; CheXbert and RadGraph learn labels or entities [26], [27], and GREEN counts clinically significant errors with a language model [28]. Automatic metrics track radiologist error counts unevenly [29], and report text carried most of the signal in a recent chest X-ray alert study in this journal [30]. Our extractor follows the CheXpert taxonomy with simpler cue windows, and every automatic score in this paper is a proxy.

2.2. Adaptation, Quantization, and Serving Engines

Qwen2.5 and Mistral are open-weight families that can be fine-tuned on modest hardware [31], [32], with low-rank adaptation (LoRA) adapters merged into the base model before serving [33]. Among post-training quantizers, GPTQ, SmoothQuant, AWQ and QServe rely on approximate second-order information, activation smoothing, activation-aware channel protection and W4A8KV4 kernel co-design, respectively [4]–[7]; we use AWQ and propose no new quantizer. vLLM pages the key–value (KV) cache and batches continuously [34], following Orca [35], and MIG divides an A100 into hardware-isolated instances [19].

2.3. Latency Targets and Scheduling

Interactive services are judged by their slowest responses [3]. Clipper, Clockwork and INFaaS combine batching, predictable execution and variant selection under latency and accuracy targets [9]–[11]; for language models, SOLA, DistServe and Sarathi-Serve balance time-to-first-token and time-per-output-token, disaggregate prefill and decoding, or chunk prefills to manage goodput and the throughput–latency trade-off [12]–[14]. Our outer dispatcher reproduces none of this scheduling, and with two workers, power-of-two-choices sampling [36] reduces to a least-queue rule. Response time and resources have been treated as joint objectives for healthcare cloud services in this journal [37], and energy reporting follows Green AI [38]; our parent-GPU energy readings are descriptive only.

2.4. Routing Among Language Models

FrugalGPT cascades models [15], Hybrid LLM predicts whether a small model's answer will suffice [16], RouteLLM learns routers from preference data [17], and a well-tuned k-nearest-neighbor router can match learned ones [18]. These methods route between different models on quality-versus-cost grounds and do not show that a particular radiology output preserves a designated finding, while an input-side gate cannot verify the output; we therefore keep input eligibility, consistency with a reference output, and serving performance separate. Our comparators (Supplementary Appendix H9) are adapted re-implementations of selected ideas, labeled "-adapted"; they are not the original systems, and their results are not benchmarks of those systems.

2.5. Reporting Standards and Research Gap

Following CLAIM and TRIPOD+AI [39], [40], we state data provenance, splits and claim limits, and label every acceptance-style number an automated proxy because no radiologist review was performed. The gap evaluated here is operational: whether a frozen input

hazard constraint preserves agreement with FP16 while part of the traffic uses a quantized copy, whether a state-aware score improves on gate-only routing, and whether either yields a serving benefit once hardware is matched. We do not claim the first MIG scheduler, a new quantizer, a validated clinical metric, or that prior routing approaches cannot incorporate risk.

3. Proposed Method

3.1. System Overview

Figure 1 shows the request path. The frozen hazard extractor is run offline on the FINDINGS text of every report, and the resulting score h is stored in the request manifest. At serving time, the CPU dispatcher reads h, applies the gate and, only for requests below the gate, scores the two workers. Hazard extraction and all output scoring therefore lie outside measured latency. Both workers hold the same LoRA-adapted Qwen2.5-3B-Instruct model [31], [33]; the INT4 worker uses its AWQ-quantized derivative [6]. Each worker occupies a full 7g.40gb MIG slice on a separate A100 GPU, so every deployment in the confirmation reserves two GPUs; the design does not pack several models into one GPU. Images are used only to audit report–image linkage, never as model input.

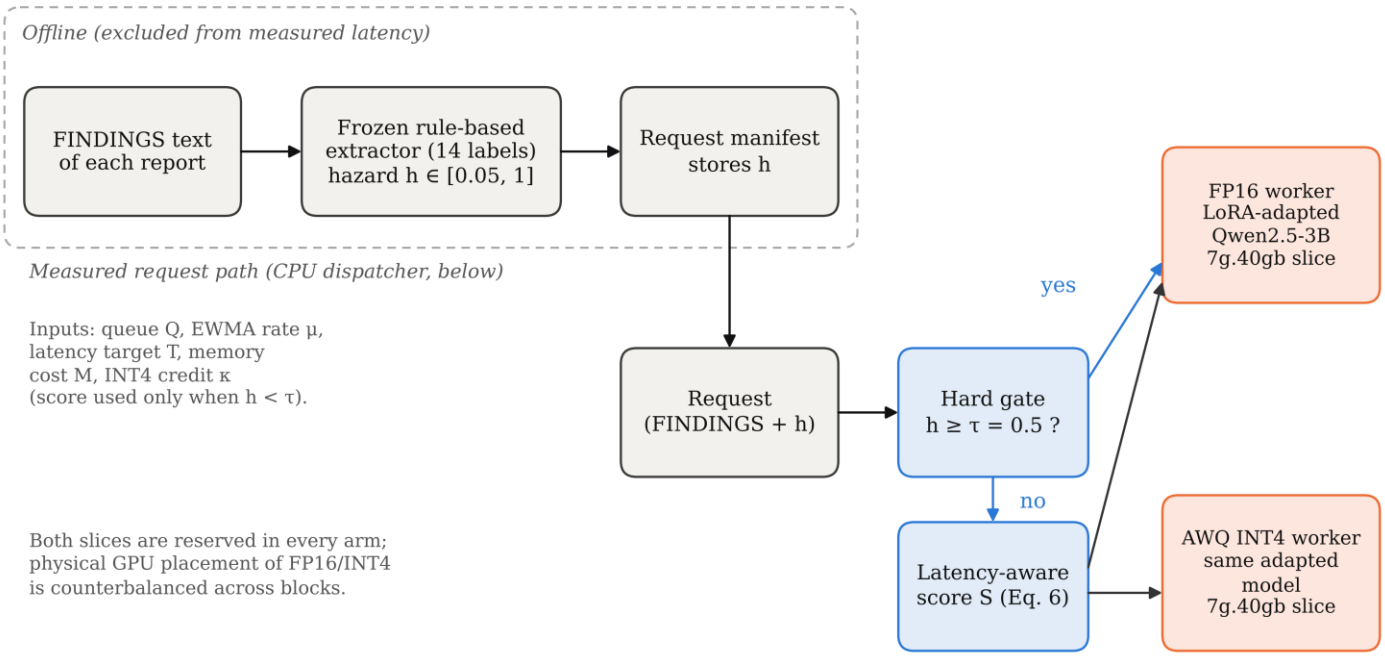


Figure 1. TRIASE-SLO request path. The hazard score h is computed offline by a frozen rule-based extractor and stored with the request. Inputs with h ≥ τ are assigned to FP16; only lower-scoring inputs enter the latency-aware score. The gate is an input-side rule and does not validate the generated output

3.2. Hazard Score and Hard Gate

The extractor follows the 14-observation taxonomy of the CheXpert labeler [23]: No Finding, Enlarged Cardiomediastinum, Cardiomegaly, Lung Opacity, Lung Lesion, Edema, Consolidation, Pneumonia, Atelectasis, Pneumothorax, Pleural Effusion, Pleural Other, Fracture and Support Devices. No Finding is derived (set when no other category is present or uncertain) and does not contribute to h. For each of the other 13 categories, the extractor searches a fixed phrase lexicon and assigns one of four states: present, negated, uncertain, or unmatched.

Matching proceeds in four steps (Supplementary S1.1). (i) The text is lowercased, runs of two or more anonymization characters x are replaced by a space, whitespace (including line breaks) is collapsed, and the text is split into segments at periods and semicolons, so line breaks and commas do not end a segment. (ii) Lexicon phrases are matched as whole words or phrases. (iii) A match is uncertain if one of 12 uncertainty cues occurs as a substring within

the preceding 60 or following 30 characters, otherwise negated if one of 8 negation cues occurs as a whole word within the preceding 60 characters, and otherwise present. (iv) Across matches of one category, present overrides uncertain, which overrides negated, in the spirit of cue-window negation detection [24], [25].

The hazard score aggregates the category states as $h(x) = \max_c s_c$, where $s_c = w_c$ for a present category, $s_c = 0.8w_c$ for an uncertain category and s_c =0 otherwise, with $h =$ 0.05 when no category contributes. Because the smallest weight in Table 1 is 0.50, every category triggers the gate when present, and 8 of the 13 also do so when uncertain. Only inputs without a present or uncertain finding, or whose strongest mention is an uncertain enlarged cardiomediastinum, cardiomegaly, atelectasis, pleural other, or support devices finding, can reach INT4; such uncertain mentions still count as reference categories in Section 4.5. The weights and lexicon are development choices of the authors, neither learned from data nor clinically calibrated. The threshold $\tau = 0.5$ was fixed during development, before the historical test matrix, and was not tuned on test outcomes; it is an engineering cutoff on this weight scale, not a calibrated probability or a clinically validated threshold, and thresholds of 0.25 and 0.75 are sensitivity analyses (Supplementary Appendix H4). The frozen lexicon (SHA-256 e749a387...) added word boundaries and synonyms after an audit of the historical pilot, whose 100 test reports are excluded from the confirmation cohort (Section 4.1). The gate is given in Equation (1).

Table 1. Hazard categories, weights and gate behavior ($\tau = 0.5$). Every category triggers the gate when present; "Gated if uncertain" means $0.8w_c \geq \tau$. The complete lexicon is in Supplementary Table S1

Category	Weight w_c	Gated if uncertain	Example lexicon phrases
Enlarged Cardiomediastinum	0.55	No	widened mediastinum; mediastinal widening (+2)
Cardiomegaly	0.55	No	cardiomegaly; cardiac enlargement (+13)
Lung Opacity	0.65	Yes	opacity; opacities (+4)
Lung Lesion	0.80	Yes	nodule; nodules (+7)
Edema	0.90	Yes	edema; vascular congestion (+5)
Consolidation	0.80	Yes	consolidation; consolidative (+3)
Pneumonia	0.85	Yes	pneumonia; bronchopneumonia (+1)
Atelectasis	0.60	No	atelectasis; atelectatic (+2)
Pneumothorax	1.00	Yes	pneumothorax; pneumothoraces (+3)
Pleural Effusion	0.75	Yes	pleural effusion; pleural effusions (+6)
Pleural Other	0.50	No	pleural thickening; pleural plaque (+2)
Fracture	0.70	Yes	fracture; fractures (+8)
Support Devices	0.60	No	catheter; picc (+5)
No Finding (derived)	—	—	Set when no other category is present or uncertain
No category contributes	h = 0.05	No	Default score

The cue windows do not resolve clinical syntax: a preceding "no" can extend across a comma and "but" to a later finding, a trailing "not seen" is not treated as negation, "no change in" negates a finding that is still present, and "unlikely" is read as uncertain because it contains the cue "likely" (Supplementary S1, Table S3). Because the same extractor gates the inputs and scores the outputs (Section 4.5), a finding that the rules miss on both sides is invisible to our metrics, and no claim of clinically complete detection follows from the gate.

$$\text{route}(x) = \begin{cases} \text{FP16}, & h(x) \geq \tau \\ \arg\min_{i \in \{\text{FP16}, \text{INT4}\}} S_i(x), & \text{otherwise} \end{cases} \quad (1)$$

3.3. Latency-Aware Scoring Below the Gate

For each worker i, the dispatcher keeps the outstanding work Q_i, the summed predicted work of the requests dispatched to i that have not yet returned (including requests queued inside the serving engine), and a smoothed dispatcher-level rate μ_i. The predicted work of request x is given in Equation (2), where $n(x)$ is the number of whitespace-delimited

prompt words, $N_{\max}$ =128 is the generation cap, $N_{\min}$ =16 is an output floor and ρ is the mean ratio of generated tokens to the cap; because w adds a word count to a token estimate, it is measured in work units, not tokens. Both the historical value $\rho = 0.080734$ and the validation-calibrated value ρ =0.10695 give $\rho N_{\max} < 16$, so the floor is active, and recalibration does not change w.

$$w(x) = \max\big(1,\ n(x)\big) + \max(N_{\min},\ \rho\, N_{\max}) \tag{2}$$

Equation (3) gives the expected finish time of x on worker i, where A_i is the set of requests in flight on i. When request j completes successfully after a router-to-worker round-trip time ℓ_j, μ_i is updated by the exponentially weighted moving average in Equation (4) with $\alpha = 0.3$; Q_i is reduced by w_j in every case, but μ_i is not updated after a failure. Initial rates are 40 and 60 work units/s for FP16 and INT4, reset before every measured unit. Because ℓ_j is the time that one request spends in a batching engine, w_j/ℓ_j is a per-request rate, and F_i treats the outstanding requests as if they were served one after another; with several requests in flight, it is a load index in seconds, not a calibrated prediction of the completion time.

$$F_i = \frac{Q_i + w}{\mu_i}, \qquad Q_i = \sum_{j \in A_i} w_j \tag{3}$$

$$\mu_i \leftarrow (1-\alpha)\,\mu_i + \alpha\,\frac{w_j}{\ell_j} \tag{4}$$

Below the gate, each worker receives the score in Equation (6), where the tilde denotes min–max normalization of $z \in \{F, Q, M\}$ across the two workers (Equation (5)). M_i is a fixed memory cost (FP16 = 1.0, $\mathrm{INT4}$ = 0.35), so a lower value favors the smaller $\mathrm{INT4}$ checkpoint; it is a configuration constant, not an online memory measurement. T is the latency target, and the indicator adds λ_S when the expected finish time exceeds T. The last term applies to $\mathrm{INT4}$ only: it adds the hazard score as a penalty with weight λ_H and subtracts a credit $\kappa = 0.25$. All weights λ are equal to 1; the router chooses the worker with the smaller score, and a tie is resolved in favor of FP16. These are engineering settings, not clinical thresholds or monetary costs.

$$\tilde{z}_i = \begin{cases} \dfrac{z_i - \min\limits_k z_k}{\max\limits_k z_k - \min\limits_k z_k}, & \max\limits_k z_k > \min\limits_k z_k \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$S_i = \lambda_F \tilde{F}_i + \lambda_Q \tilde{Q}_i + \lambda_M \tilde{M}_i + \lambda_S\, \mathbb{1}[F_i > T] + \mathbb{1}[i = \mathrm{INT4}]\,(\lambda_H h - \kappa) \tag{6}$$

With two workers, each unequal normalized term is 1 for the larger and 0 for the smaller value, and the normalized memory term is always 1 for FP16 and 0 for INT4. With $d_F = \mathrm{sgn}(F_{\mathrm{FP16}} - F_{\mathrm{INT4}})$, $d_Q = \mathrm{sgn}(Q_{\mathrm{FP16}} - Q_{\mathrm{INT4}})$, and $d_S = 1$ if only the FP16 worker exceeds T, -1 if only the INT4 worker does, and 0 otherwise, the score difference reduces to Equation (7), and INT4 is selected only when $\Delta > 0$. Below the gate, h is 0.05, or 0.40–0.48 for an uncertain mention of a category with weight 0.50–0.60, so with unit weights INT4 is selected unless the state comparisons favor FP16 by a net margin of two (h=0.05) or one (uncertain mention). The memory term and the credit thus act as a constant bias toward INT4, the hazard penalty only shifts the margin, and the score departs from gate-only routing only when the INT4 worker is clearly the more loaded one (verified against the implementation in Supplementary S12).

$$\Delta = S_{\mathrm{FP16}} - S_{\mathrm{INT4}} = \lambda_F d_F + \lambda_Q d_Q + \lambda_S d_S + \lambda_M + \kappa - \lambda_H h \tag{7}$$

The score has no learned parameters. Its additional complexity is justified only if it improves on gate-only routing, which Section 5.5 tests.

3.4. Routing Procedure

Algorithm 1 summarizes the procedure for one request. Step 1 reads the precomputed hazard value; the dispatcher itself performs only the gate, the score, and the state updates. Gated requests also add their work to Q_{FP16}, so the state seen by the score includes the traffic that the gate imposes. Output quality is evaluated after collection and cannot influence routing.

Algorithm 1. Risk-gated, latency-aware routing for one request

INPUT: findings text x; outstanding work Q_i and rate μ_i of both workers; parameters τ, T, α, κ, λ, ρ, $N_{\min}$, $N_{\max}$

OUTPUT: selected worker $i^* \in \{\text{FP16}, \text{INT4}\}$

1: Retrieve the precomputed hazard $h = h(x)$ from the frozen offline request manifest.
2: Compute the predicted work w in work units by Equation (2).
3: If $h \geq \tau$, set $i^* \leftarrow \text{FP16}$ and go to step 8.
4: For each worker i, compute the expected finish time F_i by Equation (3).
5: Normalize F, Q and M across the two workers by Equation (5).
6: Compute S_i by Equation (6) for both workers; the hazard penalty and the credit κ enter S_{INT4} only.
7: Set i^* to the worker with the smaller score; on a tie, $i^* \leftarrow \text{FP16}$.
8: Dispatch the request to i^* and add w to Q_{i^*}.
9: On completion, subtract w from Q_{i^*}; if the request succeeded, update μ_{i^*} by Equation (4). Return i^*.

3.5. What the Gate Does Not Promise

The gate controls which inputs may reach INT4; it does not certify clinical risk or output correctness, because FP16 and INT4 may both omit a finding that the shared rules miss. No radiologist reference is used in this study; clinical evaluation is future work (Section 6).

4. Experimental Setup

4.1. Data and Cohorts

The IU X-Ray/OpenI collection [20] contains 3,955 reports linked to 7,470 images. Requiring non-empty FINDINGS and IMPRESSION sections and an inventoried image yields 3,331 report pairs, split 2,310/334/687 into training, validation, and test sets by seed-42 hashes of the XML report identifier (uId). The sets are disjoint in report identifiers; patient linkage could not be verified, so the split is not claimed to be patient-level. The prompt is "FINDINGS: {findings} IMPRESSION:".

The confirmation cohort was fixed before test inference. Exact SHA-256 prompt matching found 165 of the 687 test reports whose prompts also occur in the training or validation sets. Excluding these and the 100 report identifiers used in the historical latency pilot leaves 450 reports (449 unique prompts; 221 with $h \geq 0.5$). These test reports had been inspected in earlier experiments, so the cohort is a filtered re-evaluation set, not an unseen external test, and exact-prompt filtering does not remove semantic near-duplicates (Supplementary S6).

4.2. Model Adaptation, Quantization, and Serving

Qwen2.5-3B-Instruct is adapted with LoRA [33] (rank 16, alpha 32, dropout 0.05, no bias; q_{proj}, k_{proj}, v_{proj}, o_{proj}, $gate_{proj}$, up_{proj} and $down_{proj}$) for two epochs with fused AdamW (learning rate 2×10^{-4}, cosine schedule, warmup ratio 0.03, weight decay 0.01), per-device batch 2 with gradient accumulation 8, maximum sequence length 256, right padding, FP16 mixed precision and seed 42 for model and data order; prompt and padding tokens are excluded from the loss, and the merged FP16 checkpoint reaches a validation loss of 0.731209 (perplexity 2.077591). AWQ [6] uses 4-bit weights, group size 128, zero point and GEMM serialization (AutoAWQ 0.2.6, torch 2.3.1, Transformers 4.45.2), calibrated on the first 128 validation records in manifest order (prompt plus reference impression, length 256); no test record is used. Serving uses vLLM 0.6.3.post1 [34] with the AWQ-Marlin kernel

for INT4 and the settings of Table 2; Supplementary S2 (Tables S4–S5) lists every setting and package version.

Table 2. Frozen model, serving, and confirmation configuration (historical settings in Supplementary Appendix H11)

Item	Setting
Task	FINDINGS → IMPRESSION generation
Data	3,331 pairs; 2,310/334/687 train/validation/test report UIDs; patient linkage unverified
Confirmation cohort	450 test reports (449 unique prompts; 221 with $h \geq 0.5$)
Workers	Qwen2.5-3B-Instruct + LoRA; merged FP16 and AWQ INT4 (group size 128)
Serving	vLLM 0.6.3.post1; greedy; ≤ 128 new tokens; context 2,048
Hardware	Two A100-SXM4-40GB GPUs; one 7g.40gb slice per worker; CPU dispatcher
Router	$\tau = 0.5$; $T = 0.333788$ s; $\rho = 0.10695$; $\alpha = 0.3$; $\kappa = 0.25$; all $\lambda = 1$
Design	4 arms × 7 loads × 12 paired blocks × 450 reports = 151,200 requests
Proxy floor	Acceptance 1.0 in all 12 blocks (engineering zero-drift requirement)

4.3. Latency-Target Calibration

Two latency targets appear in this paper and are never mixed. The historical target T =0.453084 s and ρ =0.080734 were derived from a pilot of 500 responses (five policies × 100 test reports) and apply only to the historical results in Supplementary Appendix H. For confirmation, a separately locked, validation-only calibration measured all 334 validation reports at both precisions over three service restarts (2,004 responses, 48 warmups excluded). Its primary subset of 240 validation reports (238 unique prompts, no exact prompt overlap with training or test) gave T =0.333788 s, the median over FP16 restarts of p95 latency at concurrency 1, and ρ =0.10695, the mean ratio of completion tokens to 128 over both precisions. The validation set was used earlier for model selection and AWQ calibration, so it is not a fresh cohort (Supplementary S3).

4.4. Arms, Hardware Allocation, and Design

The confirmation compares four arms on the same pair of reserved slices (Table 3): two FP16 workers with least-queue allocation (two-FP16); full TRIASE-SLO; gate-only routing, which sends $h \geq 0.5$ to FP16 and all other requests to INT4; and a no-hazard arm that keeps the score and the INT4 credit but removes the gate and the hazard penalty. The historical policy named "oracle" is the same input-only gate as gate-only and uses no privileged output, so it is deployable. Each arm is run at seven loads: closed-loop concurrency C = 1, 2, 4, 8, and 16, and open-loop Poisson arrivals at 8.0 requests/s and at 49.7 requests/s, the latter derived from a validation capacity probe.

Table 3. Hardware allocation of the baselines: active capacity and reserved hardware are distinct

Deployment	Installed precisions	Active capacity	Reservation	Role in this paper
Historical all_fp16 (Appendix H)	FP16 + INT4	FP16 worker only	Two slices installed; one used	One-active-worker reference; capacity confounded
Confirmation two-FP16	FP16 + FP16	Both workers	Two 7g.40gb slices	Equal-hardware primary comparator and FP16 reference
Confirmation full / gate-only / no-hazard	FP16 + INT4	Both workers	Two 7g.40gb slices	Same reservation; differ in precision and routing

A block consists of one FP16-pair deployment stage and one mixed deployment stage that serves all seven loads with the three mixed arms; twelve blocks were run. The two physical placements of the FP16 worker and the two stage orders were counterbalanced (three blocks per cell), and arm and load orders were shuffled with fixed seeds. Within a block, every arm receives the same 450 reports in the same order with the same arrival times. Router state is reset before each measured unit, warmups are kept separate, and failed requests stay in all

denominators. Arrivals use absolute monotonic deadlines; closed-loop latency runs from admission and open-loop latency from planned arrival to response, so dispatcher waiting is included. Throughput is the number of successful responses divided by the span from the first request start to the last completion. In total, the confirmation measures 4 arms × 7 loads × 12 blocks × 450 reports = 151,200 requests in 336 units.

Reserved GPU-seconds per successful request (2 × makespan / successful responses) is an allocation proxy, not a monetary saving. The two stages of a block run at different times on shared parent GPUs; counterbalancing and telemetry reduce, but do not eliminate, differences in background activity. A preliminary matched study on all 687 test reports (61,830 requests) is reported in Supplementary S5 and is not pooled with the confirmation.

4.5. Outcomes and Reference Definition

To measure consistency with FP16, $H(y)$ denotes the set of categories that the extractor finds as present or uncertain in an output y. For request r, the reference y_r^{FP16} is the output of the two-FP16 arm for the same report in the same block and load, and a policy π commits an omission when its output fails to include a reference category (Equation (8)); $f_r = 1$ marks a failed candidate or reference response, which also counts as an omission. HR-FNF is therefore an FP16-reference omission rate, and proxy acceptance is 1 − HR-FNF. The two-FP16 arm, being the reference, has acceptance 1 by construction, whereas every other arm also carries the run-to-run variability of FP16 between deployment stages (Section 5.4). About 320 of the 450 reference outputs per block contain no listed category, and such references are satisfied by any output; we therefore report agreement on all requests (whole-bank) and on finding-bearing (nonempty) references separately (Supplementary S10). We also report the INT4 share and results by hazard stratum.

$$\text{HR-FNF} = \frac{1}{N}\sum_{r=1}^{N} \mathbb{1}\left[f_r = 1 \ \vee \ H(y_r^{\mathrm{FP16}}) \not\subseteq H(y_r^{\pi})\right] \tag{8}$$

Three levels of discrepancy are distinguished: a textual difference (the outputs differ as strings), a hazard-category difference (the category sets differ in either direction) and a directional omission (a reference category is missing from the candidate, Equation (8)); only the last enters the acceptance measures. None of them is a clinically meaningful error: such differences may be clinically irrelevant, the clinical weight of an omission was not assessed, and errors invisible to the extractor (laterality, severity, unlisted or wrongly added findings) are not measured. Because the FP16 output is itself unvalidated, agreement with FP16 measures consistency, not clinical correctness or safety.

Serving efficiency is measured by successful-response p95 latency, throughput, SLO attainment (the share of requests completed within T) and reserved GPU-seconds per request. Equation (9) defines proxy–SLO goodput, the rate of requests whose candidate and reference both succeed, whose candidate includes the reference categories and whose latency L_r (Section 4.4) is within T, divided by the trace duration D. It is a joint engineering measure, not a clinical outcome or the product of throughput, acceptance and SLO attainment. The strict floor of 1.0 on proxy acceptance, a zero-drift engineering requirement from the original study rather than a clinically validated threshold, is retained unchanged: an arm meets it only when acceptance equals 1 in all 12 blocks. The contribution of the score is measured by comparing full routing with gate-only routing.

$$\text{Goodput}_{\text{proxy}} = \frac{1}{D}\sum_{r=1}^{N} \mathbb{1}\left[f_r = 0 \ \wedge \ L_r \le T \ \wedge \ H(y_r^{\mathrm{FP16}}) \subseteq H(y_r^{\pi})\right] \tag{9}$$

4.6. Statistical Analysis

The statistical unit is the paired deployment block (n=12), conditional on the fixed report cohort. The primary contrast, prespecified to test the score, is the mean of $\log\left(\text{throughput}_{\text{full}}/\text{throughput}_{\text{gate-only}}\right)$ at C = 8, tested one-sided against $\log(1.02)$ at α =0.025; confirmation requires the lower bound of the two-sided 95% paired-t interval to exceed the 2% margin. Sample-size planning assumed a block-level standard deviation of 0.021 for this log ratio (twice that of the preliminary study) and a true 4% effect. Fourteen

secondary two-sided contrasts compare full routing with two-FP16 on log throughput and on p95 at all seven loads, with Holm family-wise correction at 0.05 [41], [42]. All other intervals are unadjusted 95% paired-t intervals and are descriptive. The historical analyses in Supplementary Appendix H used request-level bootstrap [43], Wilcoxon [44] and McNemar [45] tests and are conditional descriptive evidence. No clinical non-inferiority or equivalence is claimed. Contrasts of no-hazard and gate-only routing with two FP16 workers, additional metrics for the contrasts of full routing with no-hazard and gate-only routing, and the decision analysis of Section 3.3 were added after the confirmation in response to review; they are post-hoc, unadjusted, and descriptive (Supplementary S12).

5. Results and Discussion

5.1. Execution

All 336 units completed and all 151,200 measured requests succeeded; no unit was replaced or excluded. Source hashes, worker provenance, and completeness checks are listed in Supplementary S11.

5.2. Serving Performance under Equal Reservation

Table 4 and Figure 2 compare the two-FP16, full, and gate-only arms. Under equal reservation, full routing did not outperform two FP16 workers. Its p95 latency was higher at four of the seven loads (median change across loads +0.33%, range -7.43% to +7.00%), and its throughput was lower at $C = 2$ to 16; the higher throughput at C = 1 (unadjusted $p = 0.004$) did not survive Holm correction ($p = 0.054$). Of the 14 Holm-adjusted secondary tests, only one was rejected: at $C = 2$, full routing was slower in p95 by 0.023 s (Holm $p = 0.0047$). The 9.0% median p95 reduction and the 13–18% throughput gain of the original submission were measured against one active FP16 worker (Supplementary Appendix H1) and reflected additional serving capacity rather than routing. Arm-level results are in Supplementary S7 and paired contrasts in S8.

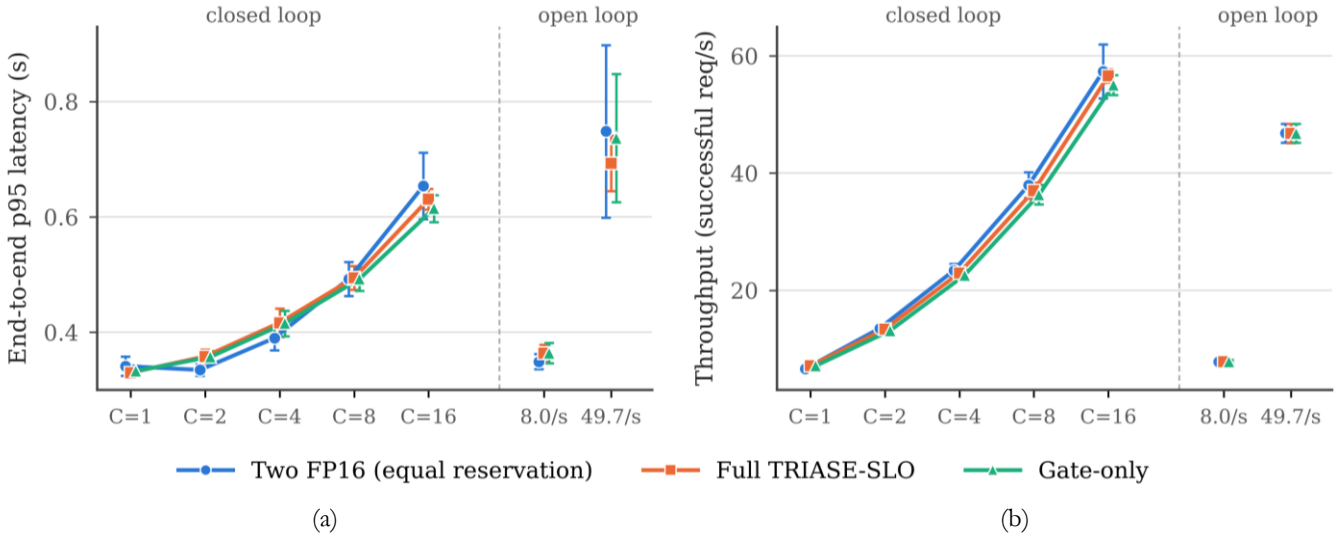
Table 4. Equal-reservation confirmation: means over 12 paired blocks of 450 requests. Each cell lists two-FP16 / full / gate-only (reserved GPU-s/request: two-FP16 / full). p95 uses successful responses and includes dispatcher waiting; SLO uses T = 0.333788 s

Load	p95 (s)	Throughput (req/s)	SLO attainment (%)	Reserved GPU-s/req
C = 1	0.341 / 0.330 / 0.333	6.60 / 7.10 / 7.10	94.7 / 95.1 / 95.0	0.3054 / 0.2822
C = 2	0.335 / 0.358 / 0.357	13.49 / 13.36 / 13.11	95.0 / 93.6 / 93.7	0.1488 / 0.1503
C = 4	0.390 / 0.416 / 0.415	23.40 / 22.95 / 22.57	92.3 / 90.6 / 90.5	0.0860 / 0.0875
C = 8	0.493 / 0.494 / 0.492	37.93 / 36.97 / 36.32	86.2 / 85.3 / 84.9	0.0532 / 0.0543
C = 16	0.653 / 0.631 / 0.614	57.34 / 56.57 / 55.01	76.7 / 77.0 / 74.2	0.0354 / 0.0354
Open 8.0/s	0.349 / 0.363 / 0.364	7.82 / 7.82 / 7.82	93.9 / 93.2 / 93.2	0.2563 / 0.2563
Open 49.7/s	0.748 / 0.693 / 0.736	46.81 / 46.75 / 46.78	70.7 / 70.9 / 66.2	0.0428 / 0.0429

Post-hoc contrasts against two FP16 workers (Supplementary S12, Table S21) show that INT4 can make serving faster: the hazard-blind no-hazard arm had a lower p95 at all seven loads (−0.047 to −0.121 s; intervals below zero at six) and a higher throughput at $C = 1$, 2, and 4 (+9.0% to +32.8%), whereas gate-only routing, like full routing, had a higher throughput only at $C = 1$ and no consistent advantage. The gate fixes the allocation by input content rather than by load: the 221 high-hazard reports of every block stay on FP16, and the score keeps further requests there, so the FP16 worker serves 49–66% of the requests in the gated arms (Table 5), whereas two FP16 workers balance by queue length. With the slower INT4 batches at higher concurrency (Table 8), this is consistent with the absence of a serving gain, although the confirmation was not designed to separate these explanations.

The serving contribution of TRIASE-SLO is therefore not an established latency or throughput improvement. The demonstrated advantages of the INT4 copy, a smaller checkpoint, and a larger KV-cache capacity (Table 8) are worker properties measured in isolation; with two full slices reserved in every arm, the confirmation does not show whether they yield

more served requests or denser packing. In the tested configuration, two FP16 workers were not slower than mixed-precision routing in any Holm-adjusted comparison and exposed no requests to INT4.



(a) (b)

Figure 2. Equal-reservation confirmation: (**a**) p95 latency and (**b**) throughput by load for two FP16 workers, full TRIASE-SLO and gate-only routing. Points are means over 12 paired blocks; error bars are 95% t intervals across blocks. Open-loop throughput is set by the arrival rate.

5.3. Risk Protection and INT4 Use

Table 5 and Figure 3 show what the gate buys. Without the gate, the no-hazard arm placed about half or more of the requests on INT4 and agreed with only two-thirds to four-fifths of the finding-bearing FP16 references. With the gate, full and gate-only routing still served one third to one half of the requests on INT4, and agreement stayed near 99% at every load. Relative to no-hazard routing, full routing raised finding-bearing agreement by 16.8 to 33.3 percentage points, with every paired interval above zero (Supplementary Table S22).

Table 5. INT4 use versus FP16 consistency (means over 12 blocks). Finding-bearing agreement uses only requests whose FP16 reference contains at least one listed category

Load	INT4 share (%): gate / full / no-hazard	Finding-bearing agreement (%): gate / full / no-hazard	Whole-bank acceptance (%): full / no-hazard	Throughput (req/s): full / no-hazard
C = 1	50.9 / 34.1 / 100.0	99.23 / 99.49 / 66.15	99.85 / 90.22	7.10 / 8.75
C = 2	50.9 / 40.4 / 56.0	99.23 / 99.49 / 81.15	99.85 / 94.56	13.36 / 15.61
C = 4	50.9 / 45.6 / 55.1	99.04 / 99.23 / 82.28	99.78 / 94.87	22.95 / 25.46
C = 8	50.9 / 47.1 / 53.9	99.17 / 99.43 / 81.40	99.83 / 94.61	36.97 / 39.59
C = 16	50.9 / 48.0 / 49.4	98.98 / 99.11 / 82.31	99.74 / 94.87	56.57 / 60.90
Open 8.0/s	50.9 / 44.3 / 72.3	99.23 / 99.36 / 75.00	99.81 / 92.78	7.82 / 7.82
Open 49.7/s	50.9 / 48.0 / 49.9	99.10 / 99.36 / 82.14	99.81 / 94.83	46.75 / 46.90

The hazard strata explain the mechanism. Of all finding-bearing FP16 references, 97.7% (127.2 of 130.2 per unit) belong to inputs with $h \geq 0.5$, which the gate keeps on FP16. When the no-hazard arm sent such high-hazard inputs to INT4, the INT4 output omitted at least one reference category in 34.2% of finding-bearing cases, against 0.17% when the same arm sent them to FP16 (Supplementary Table S15).

The protection costs efficiency where INT4 is fastest. Relative to no-hazard routing, full routing had lower throughput at every closed-loop load (−6.7% to −18.8%), a higher p95 at every load (+0.041 to +0.110 s) and lower SLO attainment (−2.7 to −4.3 percentage points; Supplementary Table S22). Its proxy–SLO goodput was also lower at all closed-loop loads despite much higher agreement, because about 71% of references are empty and satisfied by any output, so a fast hazard-blind policy collects most of the goodput while failing finding-

bearing requests. Whole-bank measures therefore understate the value of the gate, and the trade-off has to be read from finding-bearing agreement and INT4 share (Figure 3), not from goodput or constrained throughput.

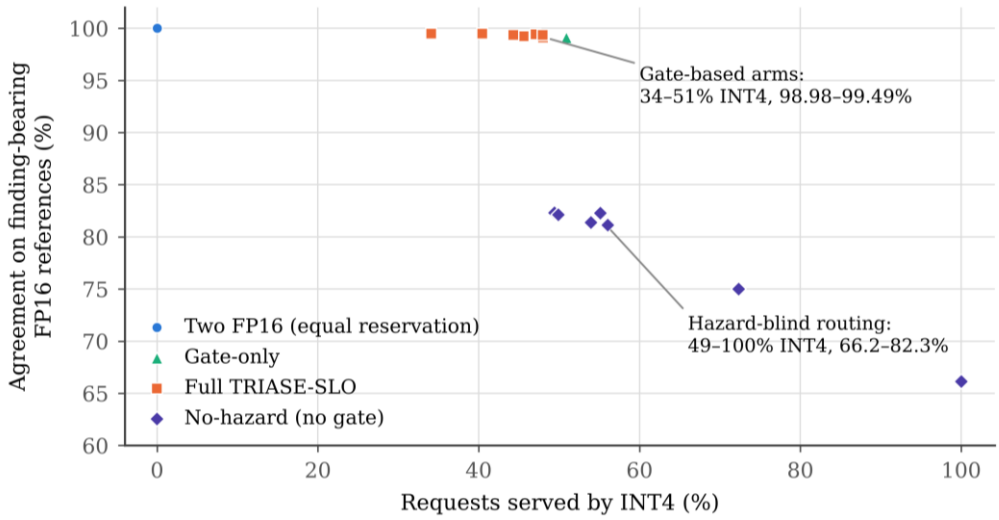


Figure 3. INT4 share versus agreement with finding-bearing FP16 references for each arm and load (means over 12 blocks). The two-FP16 arm is the reference and lies at 100% by construction

5.4. Sources of Residual Inconsistency and the Strict Floor

No arm that used INT4 met the strict floor at any load, while the two-FP16 arm met it by construction (Supplementary Table S7). Table 6 locates the residual omissions over the 84 load–block units. For full and gate-only routing, nearly all omissions occurred on INT4-routed low-hazard requests and omitted a category that was present in all 12 FP16 outputs of the same report and load. Only about 3 of the 229 low-hazard inputs per block have a finding-bearing FP16 reference, and INT4 omitted a reference category in roughly one third of these cases. This is the gate's blind spot: inputs that the rules score as low risk but whose expected impression still names a listed finding.

Table 6. Attribution of directional omissions in the confirmation, summed over 84 load–block units. A stable-reference omission omits a category present in all 12 FP16 outputs of the same report and load. Attribution is conditional on routing, not a randomized precision treatment

Arm	Omissions	On FP16-routed requests	On INT4-routed requests	Omitting a category present in all 12 FP16 outputs	Low-hazard finding-bearing references served by INT4: omitted / total
Two FP16 (reference)	0	0	—	0	—
Full TRIASE-SLO	71	8	63	63	63 / 211
Gate-only	94	10	84	85	84 / 252
No-hazard	2,336	7	2,329	2,323	53 / 160

FP16 itself is not perfectly repeatable across deployments (Table 7): despite greedy decoding with a fixed seed, repeated FP16 outputs for the same report and load differed most often as text, less often in their category sets, and least often by a directional omission. We did not isolate the cause. The FP16-to-FP16 omission rate (at most 0.042% of pairs) is similar to the FP16-routed omission rate of full routing (0.043%) and is an order of magnitude below its INT4-routed rate (0.38%), so the INT4-routed component dominates the residual. It is not attributed to quantization alone: requests reached INT4 through routing, not randomization, and are low-hazard by selection; the INT4 worker also differs in kernel and batch composition; and the two stages ran at different times. The measured quantity is the inconsistency of the INT4 serving path with FP16, of which weight quantization is the most plausible but not the only source.

Table 7. FP16 run-to-run variability in the three studies. Pairs compare the same report and load across repetitions or blocks; they are not independent patients or causal quantization effects

FP16 comparison	Pair observations	Text differences	Category-set differences	Directional omissions (both directions)
Historical main study (3 repetitions)	18,549	30	6	5 / 1
Preliminary matched study (3 restarts)	12,366	26	10	7 / 3
Confirmation (12 blocks)	207,900	503	134	47 / 87

Consequently, the 1.0 floor is infeasible for any INT4-using arm in this setting, because a single systematic omission among 450 reports violates it. The floor is an engineering zero-drift requirement, not a clinically validated threshold: missing it shows that the INT4 path does not reproduce FP16 on the rule-visible categories, not that it is clinically unsafe, and meeting it would not show clinical correctness. We keep the floor unchanged rather than select a lower tolerance after seeing the results; a post-hoc floor sensitivity is given in Supplementary S4, and an operational tolerance would need a prospective and clinically informed justification.

5.5. Contribution of the Latency-Aware Score

The prespecified primary contrast was not confirmed. At $C = 8$, the geometric-mean throughput ratio of full routing to gate-only routing was 1.0186 (+1.86%; 95% interval −3.27% to +7.27%; one-sided $p = 0.522$ against the 2% margin). The observed block-to-block standard deviation of the log ratio was 0.081, about four times the 0.021 assumed in planning, so the power to detect a true 4% effect was only about 11%. The result is therefore inconclusive rather than evidence that the score has no effect.

Equation (7) explains what the score can do in this setting. It departs from gate-only routing only when the state comparisons favor FP16 by a net margin, so its observable effect is to keep 3 to 17 percentage points more requests on FP16 (Table 5). This is also the source of its small consistency gain: full routing served fewer low-hazard finding-bearing references on INT4 than gate-only routing, and INT4 omitted a reference category in a similar fraction of them (29.9% and 33.3%; Table 6). The score therefore reduced the exposure of the blind spot by sending fewer requests to INT4, not by recognizing risky inputs. Its paired gain in finding-bearing agreement over gate-only routing was 0.13 to 0.26 percentage points, with intervals that included zero at four of the seven loads (Supplementary Table S22).

In serving terms, the evidence is mixed. Gate-only routing had the lower p95 at $C = 16$ (paired difference 0.016 s, 95% interval −0.010 to 0.043 s; also in Supplementary Appendix H4), whereas full routing had higher SLO attainment at C = 16 and at 49.7 requests/s (+2.7 and +4.7 percentage points) and higher throughput at C = 4 (+1.7%) (Supplementary Table S22). These contrasts were not prespecified or adjusted for multiplicity and point in different directions. On the current evidence, the score buys a slightly higher agreement with FP16 by using INT4 less, without an established change in efficiency.

We retain the score for reasons that do not depend on a benefit. It belongs to the configuration locked before test inference, so removing it after the results were known would be a post-hoc selection; gate-only routing, its special case, is reported beside it as the simpler alternative; and Equation (7) shows why the score had little room with two workers, leaving its value in larger pools untested rather than refuted. The gate is the component that the current results support. The value added by the score remains uncertain, and for a two-worker deployment like the one tested, gate-only routing gives the same established benefit with less machinery.

5.6. Supporting and Stress-Test Analyses

Table 8 summarizes the supporting studies conducted using the historical configuration, with detailed results provided in Supplementary Appendix H. The historical routing comparisons involving one active FP16 worker remain subject to the capacity confound discussed in Section 5.2 and should not be interpreted as evidence of a serving-efficiency advantage. Overall, these analyses further illustrate the operational limitations of TRIASE-SLO: INT4 performance varies with concurrency, the Qwen-calibrated SLO target was not achieved in the

Mistral stress test, and the fail-closed mechanism does not guarantee service availability under faults. The ablation results also suggest that the hard risk gate plays a more important role in maintaining agreement with FP16, whereas the additional benefit of the latency-aware scoring mechanism remains limited.

Table 8. Supporting and stress-test analyses on the historical configuration (details in Supplementary Appendix H); they probe limits of the design and do not extend its claims

Study	Design	Main result	Limit on interpretation
Serial precision benchmark (H5)	FP16 and INT4 in turn on one slice; 3 restarts × 5 repetitions	INT4 p95 −27% at C = 1, +6.4% / +4.7% at C = 8 / 16; checkpoint −56.3%; KV-cache capacity +19.1%	Parent-GPU power only; worker properties, not serving gains
Restart repeatability (H6)	5 loads × 3 service restarts	Same direction as the historical main matrix; acceptance 99.8–99.9%	Same checkpoint and data; one active FP16 worker
Second-checkpoint stress test (H7)	Mistral FP16/INT4 pair; frozen router; Qwen target; 103,050 requests	SLO attainment 0% for all policies; worst-case acceptance 99.08% (gate-based) vs 95.00% (all-INT4)	One checkpoint (training-mask limitation); not evidence of transfer; no retuning
Controlled faults (H8)	6 scenarios × 3 seeds × 96 requests; fail-closed rule	No request with h ≥ 0.5 served by INT4 (36 units)	Losses 16.3–93.8% under faults, 5.2–14.6% without; not availability.
Ablation and adapted comparators (H4, H9)	25 arms; 463,725 requests	No gate: worst-case acceptance 99.71% → 93.30%; soft terms change p95 ≤2%; no comparator dominates	Re-implementations of selected ideas, not the original systems

5.7. Discussion

The confirmation supports a narrower contribution than the original submission claimed. Under the evaluated conditions (one public dataset, one fine-tuned checkpoint at two precisions, two reserved A100 slices and seven loads), the risk gate improves agreement with FP16 while still allowing a substantial share of requests onto INT4; a consistent serving-efficiency advantage of mixed-precision routing over two FP16 workers was not demonstrated; and the incremental contribution of the latency-aware score remains uncertain. The findings are linked: the gate withholds from INT4 the requests on which it most often disagrees with FP16; the remaining speed advantage does not outperform two freely balanced FP16 workers; and the score, which only moves low-hazard requests, has little left to optimize.

For deployment, this suggests an ordering of choices. Where two FP16 copies fit, two FP16 workers were not slower than mixed precision in any Holm-adjusted comparison and expose no request to INT4. Mixed precision becomes a candidate only when a second FP16 copy cannot be deployed, for example when several models share one GPU through smaller MIG profiles; then the gate is the component that keeps outputs consistent with FP16, and gate-only routing gives the same established benefit as the full score. Such deployments were not tested, and reserved GPU-seconds and checkpoint compression are not monetary savings.

The residual inconsistency lies in a few low-hazard inputs whose impressions nevertheless name a listed finding. The score, which sees only worker state, cannot target them. An output-side check that reruns the extractor on the INT4 answer and regenerates on FP16 when an input category is missing would target them at a latency cost; it is not implemented here.

All agreement results describe consistency with FP16: the FP16 output is not clinical ground truth, the extractor contributes the same errors to both sides of each comparison, the 1.0 floor is an engineering requirement, and the omission counts are rule-visible category differences rather than adjudicated errors. The evidence uses one public collection, previously inspected test reports and report-identifier rather than verified patient splits; the measures are vacuous for about 71% of references; and the intervals are conditional on a fixed cohort and 12 blocks. The results therefore do not establish generalization to other datasets, institutions, models, quantizers, GPU configurations, or larger worker pools. The Mistral experiment is a stress test showing that a latency target calibrated for one checkpoint does not transfer to

another, not evidence of cross-model transfer; the adapted comparators are re-implementations, not benchmarks of the original systems; and the fault study checks one routing invariant, not availability.

6. Conclusions

TRIASE-SLO routes radiology impression requests between FP16 and INT4 copies of one model using a frozen input-hazard gate and a secondary latency-aware score. In a locked confirmation with equal hardware reservation, the risk gate improved agreement with FP16 under the evaluated conditions while one third to one half of the requests still used INT4. A consistent serving-efficiency advantage over two FP16 workers was not demonstrated: the earlier latency and throughput gains reflected added capacity, and a consistent speed advantage of INT4 appeared only when the gate was removed. The incremental contribution of the latency-aware score remains uncertain, because its prespecified throughput benefit over gate-only routing was not confirmed. No INT4-using arm met the zero-drift engineering floor, and most remaining omissions came from a few low-hazard inputs served by INT4.

These are systems and FP16-consistency results on one public dataset and one deployment configuration. They do not establish clinical correctness or safety, general SLO portability, generalization beyond the evaluated data and configuration, or performance benefits beyond those measured. Future work should test an output-side check, memory-constrained deployments, larger worker pools for the score, and clinical validity through a blinded multi-reader study with adjudication on external data with verified patient grouping.

Author Contributions: Conceptualization, J.Z., G.F.S., H.S., and A.M.; Methodology, J.Z., G.F.S., H.S., and A.M.; software, J.Z.; validation, J.Z. and G.F.S.; Formal analysis, J.Z. and H.S.; Investigation, J.Z.; Resources, G.F.S., H.S., and A.M.; Data curation, J.Z.; Writing—original draft preparation, J.Z.; Writing—review and editing, G.F.S., H.S., and A.M.; Visualization, J.Z.; Supervision, G.F.S., H.S., and A.M.; Project administration, G.F.S.; Funding acquisition, G.F.S.

Funding: This work was supported by the Ministry of Higher Education, Science, and Technology of the Republic of Indonesia (Kemdiktisaintek) under Main Contract No. 127/C3/DT.05.00/PL/2025 and derivative contracts No. 028/LL6/PL/AL.04/2025 and No. 118/F.9-05/UDN-09/2025. The funder had no role in the study design, analysis, or writing.

Data Availability Statement: The de-identified IU X-Ray/OpenI collection is available from its original distributor [20]; report texts are not redistributed. Source code (hazard extractor, router, experiment runners and analysis scripts), frozen configurations, tests and the locked result files (CSV/JSON) with SHA-256 manifests are openly available on GitHub at https://github.com/juntazen/TRIASE-SLO and archived on Zenodo at https://doi.org/10.5281/zenodo.23390843 (concept DOI, which resolves to the latest version; version 1.1.0 adds the post-hoc analyses of this revision, and version 1.0.0 is https://doi.org/10.5281/zenodo.23390844; code under the MIT License; result files under CC BY 4.0). The Supplementary Material, available at https://doi.org/10.5281/zenodo.24354962, contains the complete hazard rules, configurations, calibration and confirmation protocols, and all aggregate, per-block and paired results. Model checkpoints are not redistributed and can be rebuilt with the supplied training and quantization scripts.

Acknowledgments: The authors thank the members of IDSS for their feedback and support. Use of AI tools: OpenAI Codex assisted with code inspection, preparation and execution of the revision experiments and analyses, and documentation; Anthropic Claude assisted with auditing the revision packages, restructuring and language editing of the revised manuscript and response letters, preparing figures from the locked result files, and writing and checking the post-hoc analysis script of the second revision. All reported measurements come from executed inference and analysis code; no AI tool supplied simulated measurements or clinical labels. The authors checked the code, evidence, and text and take full responsibility for the submitted version.

Conflicts of Interest: The authors declare no conflicts of interest.

References

[1] Y. Zhang, D. Y. Ding, T. Qian, C. D. Manning, and C. P. Langlotz, "Learning to summarize radiology findings," in *Proc. 9th Int. Workshop on Health Text Mining and Information Analysis (LOUHI)*, Brussels, Belgium, 2018, pp. 204–213, doi: 10.18653/v1/W18-5623.

[2] D. Van Veen *et al.*, "Adapted large language models can outperform medical experts in clinical text summarization," *Nat. Med.*, vol. 30, no. 4, pp. 1134–1142, Apr. 2024, doi: 10.1038/s41591-024-02855-5.

[3] J. Dean and L. A. Barroso, "The tail at scale," *Commun. ACM*, vol. 56, no. 2, pp. 74–80, Feb. 2013, doi: 10.1145/2408776.2408794.

[4] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "GPTQ: Accurate post-training quantization for generative pre-trained transformers," in *Proc. 11th Int. Conf. Learning Representations (ICLR)*, Kigali, Rwanda, 2023, doi: 10.48550/arXiv.2210.17323.

[5] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "SmoothQuant: Accurate and efficient post-training quantization for large language models," in *Proc. 40th Int. Conf. Machine Learning (ICML)*, PMLR, vol. 202, Honolulu, HI, USA, 2023, pp. 38087–38099, doi: 10.48550/arXiv.2211.10438.

[6] J. Lin *et al.*, "AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration," in *Proc. Machine Learning and Systems (MLSys)*, vol. 6, Santa Clara, CA, USA, 2024, pp. 87–100, doi: 10.48550/arXiv.2306.00978.

[7] Y. Lin *et al.*, "QServe: W4A8KV4 quantization and system co-design for efficient LLM serving," in *Proc. Machine Learning and Systems (MLSys)*, vol. 7, Santa Clara, CA, USA, 2025, doi: 10.48550/arXiv.2405.04532.

[8] S. Li *et al.*, "Evaluating quantized large language models," in *Proc. 41st Int. Conf. Machine Learning (ICML)*, PMLR, vol. 235, Vienna, Austria, 2024, pp. 28480–28524. [Online]. Available: https://proceedings.mlr.press/v235/li24bb.html

[9] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, "Clipper: A low-latency online prediction serving system," in *Proc. 14th USENIX Symp. Networked Systems Design and Implementation (NSDI)*, Boston, MA, USA, 2017, pp. 613–627. [Online]. Available: https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/crankshaw

[10] A. Gujarati *et al.*, "Serving DNNs like Clockwork: Performance predictability from the bottom up," in *Proc. 14th USENIX Symp. Operating Systems Design and Implementation (OSDI)*, 2020, pp. 443–462. [Online]. Available: https://www.usenix.org/conference/osdi20/presentation/gujarati

[11] F. Romero, Q. Li, N. J. Yadwadkar, and C. Kozyrakis, "INFaaS: Automated model-less inference serving," in *Proc. 2021 USENIX Annual Technical Conf. (ATC)*, 2021, pp. 397–411. [Online]. Available: https://www.usenix.org/conference/atc21/presentation/romero

[12] K. Hong *et al.*, "SOLA: Optimizing SLO attainment for large language model serving with state-aware scheduling," in *Proc. Machine Learning and Systems (MLSys)*, vol. 7, Santa Clara, CA, USA, 2025. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2025/hash/bc82dbfbfa43232be85b8d9838f49c3e-Abstract-Conference.html

[13] Y. Zhong *et al.*, "DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving," in *Proc. 18th USENIX Symp. Operating Systems Design and Implementation (OSDI)*, Santa Clara, CA, USA, 2024, pp. 193–210. [Online]. Available: https://www.usenix.org/conference/osdi24/presentation/zhong-yinmin

[14] A. Agrawal *et al.*, "Taming throughput-latency tradeoff in LLM inference with Sarathi-Serve," in *Proc. 18th USENIX Symp. Operating Systems Design and Implementation (OSDI)*, Santa Clara, CA, USA, 2024, pp. 117–134. [Online]. Available: https://www.usenix.org/conference/osdi24/presentation/agrawal

[15] L. Chen, M. Zaharia, and J. Zou, "FrugalGPT: How to use large language models while reducing cost and improving performance," arXiv preprint arXiv:2305.05176, 2023, doi: 10.48550/arXiv.2305.05176.

[16] D. Ding *et al.*, "Hybrid LLM: Cost-efficient and quality-aware query routing," in *Proc. 12th Int. Conf. Learning Representations (ICLR)*, Vienna, Austria, 2024, doi: 10.48550/arXiv.2404.14618.

[17] I. Ong *et al.*, "RouteLLM: Learning to route LLMs with preference data," in *Proc. 13th Int. Conf. Learning Representations (ICLR)*, Singapore, 2025, doi: 10.48550/arXiv.2406.18665.

[18] Y. Li, "Rethinking predictive modeling for LLM routing: When simple kNN beats complex learned routers," arXiv preprint arXiv:2505.12601, 2025, doi: 10.48550/arXiv.2505.12601.

[19] NVIDIA Corporation, "NVIDIA Multi-Instance GPU user guide." NVIDIA Docs. [Online]. Available: https://docs.nvidia.com/datacenter/tesla/mig-user-guide/ (accessed Oct. 7, 2026).

[20] D. Demner-Fushman *et al.*, "Preparing a collection of radiology examinations for distribution and retrieval," *J. Am. Med. Inform. Assoc.*, vol. 23, no. 2, pp. 304–310, Mar. 2016, doi: 10.1093/jamia/ocv080.

[21] Z. Chen, Y. Song, T.-H. Chang, and X. Wan, "Generating radiology reports via memory-driven transformer," in *Proc. 2020 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Online, 2020, pp. 1439–1449, doi: 10.18653/v1/2020.emnlp-main.112.

[22] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, Barcelona, Spain, 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013

[23] J. Irvin *et al.*, "CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison," in *Proc. AAAI Conf. Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 590–597, doi: 10.1609/aaai.v33i01.3301590.

[24] W. W. Chapman, W. Bridewell, P. Hanbury, G. F. Cooper, and B. G. Buchanan, "A simple algorithm for identifying negated findings and diseases in discharge summaries," *J. Biomed. Inform.*, vol. 34, no. 5, pp. 301–310, Oct. 2001, doi: 10.1006/jbin.2001.1029.

[25] Y. Peng, X. Wang, L. Lu, M. Bagheri, R. Summers, and Z. Lu, "NegBio: A high-performance tool for negation and uncertainty detection in radiology reports," in *AMIA Jt. Summits Transl. Sci. Proc.*, 2018, pp. 188–196. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC5961822

[26] A. Smit, S. Jain, P. Rajpurkar, A. Pareek, A. Y. Ng, and M. P. Lungren, "Combining automatic labelers and expert annotations for accurate radiology report labeling using BERT," in *Proc. 2020 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, Online, 2020, pp. 1500–1519, doi: 10.18653/v1/2020.emnlp-main.117.

[27] S. Jain *et al.*, "RadGraph: Extracting clinical entities and relations from radiology reports," in *Proc. 35th Conf. Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2021, doi: 10.48550/arXiv.2106.14463.
[28] S. Ostmeier *et al.*, "GREEN: Generative radiology report evaluation and error notation," in *Findings Assoc. Comput. Linguistics: EMNLP 2024*, Miami, FL, USA, 2024, pp. 374–390, doi: 10.18653/v1/2024.findings-emnlp.21.
[29] F. Yu *et al.*, "Evaluating progress in automatic chest X-ray radiology report generation," *Patterns*, vol. 4, no. 9, Art. no. 100802, Sep. 2023, doi: 10.1016/j.patter.2023.100802.
[30] E. Winarno, I. M. Nur, A. Karim, S. Amri, I. E. Wirdati, and P. W. Adi, "An explainable multimodal framework for chest X-ray alert classification using radiology reports and images," *J. Comput. Theor. Appl.*, vol. 3, no. 4, pp. 647–666, May 2026, doi: 10.62411/jcta.16023.
[31] Qwen Team, "Qwen2.5 technical report," arXiv preprint arXiv:2412.15115, 2024, doi: 10.48550/arXiv.2412.15115.
[32] A. Q. Jiang *et al.*, "Mistral 7B," arXiv preprint arXiv:2310.06825, 2023, doi: 10.48550/arXiv.2310.06825.
[33] E. J. Hu *et al.*, "LoRA: Low-rank adaptation of large language models," in *Proc. 10th Int. Conf. Learning Representations (ICLR)*, 2022, doi: 10.48550/arXiv.2106.09685.
[34] W. Kwon *et al.*, "Efficient memory management for large language model serving with PagedAttention," in *Proc. 29th ACM Symp. Operating Systems Principles (SOSP)*, Koblenz, Germany, 2023, pp. 611–626, doi: 10.1145/3600006.3613165.
[35] G.-I. Yu, J. S. Jeong, G.-W. Kim, S. Kim, and B.-G. Chun, "Orca: A distributed serving system for transformer-based generative models," in *Proc. 16th USENIX Symp. Operating Systems Design and Implementation (OSDI)*, Carlsbad, CA, USA, 2022, pp. 521–538. [Online]. Available: https://www.usenix.org/conference/osdi22/presentation/yu
[36] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 10, pp. 1094–1104, Oct. 2001, doi: 10.1109/71.963420.
[37] N. I. Sitlong, A. E. Evwiekpaefe, and M. E. Irhebhude, "Hybrid dynamic programming healthcare cloud-based quality of service optimization," *J. Comput. Theor. Appl.*, vol. 3, no. 2, pp. 115–131, Sep. 2025, doi: 10.62411/jcta.14455.
[38] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green AI," *Commun. ACM*, vol. 63, no. 12, pp. 54–63, Dec. 2020, doi: 10.1145/3381831.
[39] A. S. Tejani *et al.*, "Checklist for artificial intelligence in medical imaging (CLAIM): 2024 update," *Radiol. Artif. Intell.*, vol. 6, no. 4, Art. no. e240300, Jul. 2024, doi: 10.1148/ryai.240300.
[40] G. S. Collins *et al.*, "TRIPOD+AI statement: Updated guidance for reporting clinical prediction models that use regression or machine learning methods," *BMJ*, vol. 385, Art. no. e078378, Apr. 2024, doi: 10.1136/bmj-2023-078378.
[41] S. Holm, "A simple sequentially rejective multiple test procedure," *Scand. J. Stat.*, vol. 6, no. 2, pp. 65–70, 1979, doi: 10.2307/4615733.
[42] R. Bender and S. Lange, "Adjusting for multiple testing—when and how?," *J. Clin. Epidemiol.*, vol. 54, no. 4, pp. 343–349, Apr. 2001, doi: 10.1016/S0895-4356(00)00314-0.
[43] B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Statist.*, vol. 7, no. 1, pp. 1–26, Jan. 1979, doi: 10.1214/aos/1176344552.
[44] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, Dec. 1945, doi: 10.2307/3001968.
[45] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153–157, Jun. 1947, doi: 10.1007/BF02295996.