Practical Advances in Artificial Intelligence and Computing
Frontier Research Publishing • https://www.frpubs.org
Vol. 4, No. 4, 2025

Risk Aware KV Cache Admission for Batched Large Language Model Inference

Florian Eberhardt[1]*
[1]Cornell University, Ithaca, USA
*Corresponding author: fl.eber3@gmail.com

Abstract: The key value cache of an autoregressive language model grows with generated sequence length, making memory admission sensitive to uncertain output demand. This paper separates cache byte accounting from the probability model used to admit a fixed request prefix. A support clipped Cantelli rule combines a deterministic footprint limit with a moment based risk bound. Exact convolution supplies a reference for independent two point output lengths, while a shared tail mixture tests sensitivity to dependence without changing marginal length probabilities. Numerical analysis uses 1000 reproducible request cohorts. At a 512 MiB cache budget and a target overflow probability of 0.05, admission based on expected demand accepts 11.229 requests on average with mean model overflow risk 0.32171. Exact probabilistic admission accepts 9.095 with mean risk 0.02155; clipped Cantelli accepts 6.717 with mean risk 0.000020, compared with 6.201 for deterministic worst case admission. The bound is conservative, and its guarantee requires the variance of the relevant joint demand distribution. The analysis identifies how marginally unchanged output lengths can alter batch risk and provides a reproducible capacity reference. The analysis evaluates reserved KV cache capacity.

Keywords: large language models; KV cache; memory admission; chance constraints; output length uncertainty

1. Introduction

Autoregressive language model inference retains keys and values from previous tokens so that decoding can reuse attention state. The resulting key value (KV) cache increases as a request progresses. Serving capacity therefore depends on more than the size of the model weights: the same model can require different amounts of cache for different prompts and output lengths. PagedAttention addresses fragmentation and flexible cache allocation [1], but a memory manager still needs a policy for deciding how much uncertain future demand to admit. A mean length estimate does not, by itself, control the probability that a group of requests exceeds its available cache.

Iteration level scheduling, as developed in Orca, allows request execution to be reorganized at decoding boundaries [2]. Admission is a complementary decision. It determines which requests may enter a memory envelope, while scheduling determines how their operations are interleaved. This paper isolates admission for a fixed ordered cohort whose cache blocks are retained until the cohort completes. Under that policy, cache demand grows monotonically and the sum of final per-request footprints determines the cohort maximum. The retention assumption makes the resource event precise.

There are two distinct sources of uncertainty in this decision. First, output length is unknown when admission occurs. Second, lengths within a cohort may be dependent. A workload that occasionally induces long responses across many requests can have the same marginal length probabilities as an independent workload, yet a different probability of exhausting memory. Estimating each request separately is

consequently insufficient to establish a batch guarantee unless the dependence assumptions are justified. This distinction matters even when cache bytes per token are known exactly.

We analyze four prefix rules: expected footprint, deterministic worst case, a support clipped Cantelli bound, and an exact probability reference. All preserve request order and use the same cache representation. A finite two point length model permits exact convolution of aggregate demand and explicit evaluation of a shared tail stress distribution. The study contributes an auditable connection between cache accounting, moment assumptions, and admission decisions. It applies established probability bounds to cache admission.

2. Related Work

PagedAttention organizes attention caches into logical and physical blocks and supports flexible allocation [1]. The amount of persistent cache also depends on attention architecture. Grouped query attention (GQA) uses fewer key and value heads than query heads [3], while multi-query attention shares keys and values across query heads [4]. These architectural choices change cache bytes per token. Their memory implications can be accounted for algebraically. The present study fixes one architecture and examines uncertainty in admitted demand.

Attention computation and persistent cache storage are related but separate. FlashAttention reduces attention intermediate storage and memory traffic through IO aware computation [5]; that does not automatically shrink the persistent keys and values retained for each token. DistServe studies prefill and decoding disaggregation under latency requirements [6], and Sarathi-Serve coordinates these stages through chunked prefill and scheduling [7]. Cache capacity is one resource constraint within these execution systems.

The Llama 3 report documents an 8B architecture with 32 layers, eight KV heads, and a head dimension of 128, and discusses cache reservation for maximum output lengths [8]. These dimensions define the representative cache configuration used in the numerical analysis. Moment inequalities provide the mathematical basis for probabilistic envelopes [9]. FlexGen further shows that memory placement can involve GPU, CPU, and disk resources [10]. Our analysis uses a fixed residual cache budget and excludes offloading. Its exact reference is optimal only among prefixes of a specified queue under the stated length distribution.

Recent serving work further separates cache-aware execution from the admission decision. Online scheduling with KV cache constraints models growing retained demand [11], while FlashInfer supports heterogeneous cache layouts and customizable attention kernels [12]. These mechanisms can determine how an admitted workload is executed; a probabilistic admission gate must additionally specify the uncertain footprint event and the information used to assess it.

The retained representation is another source of variation in cache accounting. AQUA-KV uses dependencies across layers to support adaptive key-value quantization [13], and signal-aligned memory compression for long-context models changes which information is retained [14]. Parameter-efficient instruction tuning instead concerns adaptation of model parameters [15]. Capacity comparisons across such configurations require an explicit accounting model and separate evaluation of their effects on task quality.

Demand estimation supplies information for resource control. Multi-resource cloud forecasting with spatio-temporal graph transformers [16], backend load-trend modeling [17], and attention-enhanced traffic prediction [18] address temporal or structural aspects of demand. Embedding-based scheduling uses estimates of remaining generation length to organize LLM requests [19]. The admission analysis below keeps a forecast's expected value separate from the joint tail event required for an aggregate probability guarantee.

Seesaw changes model sharding between prefill and decoding and uses tiered KV buffering [20]. SLOs-Serve allocates execution work under multiple service requirements [21]. Both provide execution contexts in

which payload occupancy is only one resource constraint. Our fixed-retention cohort isolates payload occupancy.

Distributed resource control includes adaptive throughput scheduling [22], hierarchical microservice orchestration [23], and scheduling under dynamic resource competition [24]. Critical-task workflow scheduling [25], reinforcement learning combined with meta-learning for backend services [26], and adaptive replica allocation [27] broaden the decisions coupled to available resources. A fixed-prefix admission test isolates memory feasibility within these resource-control settings.

A deployment controller can maintain structured state and hierarchical policies [28] and organize actions through task graphs [29]. Evolutionary self-optimization [30] and autonomous exploration with goal discovery [31] extend this decision context as operating conditions change. Such a controller could choose among actions that have already passed an explicit memory-feasibility test. The numerical study below fixes request order so that its admission comparisons depend only on the stated resource and probability assumptions.

3. Cache Demand and Admission Rules

3.1 Cache Accounting and the Resource Event

Consider full attention decoding with L layers, hKV key value heads, head dimension dH, and w bytes per scalar. A token stores both keys and values in every layer. Let P be the number of token positions per cache page, M the usable cache payload budget, and C the resulting number of complete pages. Then

$$k = 2Lh_{KV}d_H w, \quad C = \left\lfloor \frac{M}{kP} \right\rfloor \tag{1}$$

This accounting excludes weights, temporary activations, attention workspaces, block tables, allocator metadata, tensor parallel replication, quantization scales, and shared prefixes. M is the residual payload budget after separate allowances for those resources. Sliding windows and compressed cache representations require different accounting. The numerical setting uses L = 32, hKV = 8, dH = 128, and w = 2, giving 128 KiB per token. With P = 16, a page contains 2 MiB of raw KV payload.

A cohort consists of N requests in a fixed first come, first served order. Request i has a known prompt length and a future output length that takes a short or long value. Let b_i be the page count obtained by rounding the prompt plus short output upward to a multiple of P. Let d_i be the additional pages required by the long output after the same rounding. With long-output probability q_i, its final demand is

$$X_i = b_i + d_i Z_i, Z_i \sim Bernoulli(q_i) \tag{2}$$

Both b_i and d_i are integers, with b_i positive and d_i nonnegative. For an admitted prefix of length n, define S_n as the sum of its final page demands. Overflow is the strict event $S_n > C$; a footprint equal to C fits. The deterministic upper support is $A_n = \Sigma_i(b_i + d_i)$. Since completed requests retain their caches until the cohort ends, final aggregate demand also equals the maximum retained footprint. With earlier cache release, it would instead be a conservative capacity event, and a timing model would be needed to evaluate the actual peak.

3.2 A Support Clipped Moment Bound

Under conditional independence given the cohort inputs, the prefix mean and variance are

$$\mu_n = \sum_{i=1}^{n}(b_i + d_i q_i), V_{0,n} = \sum_{i=1}^{n} d_i^{\,2} q_i(1 - q_i) \tag{3}$$

The subscript zero denotes the independent length model. For any aggregate demand with true mean μ_n and true variance V_n, the one-sided Chebyshev-Cantelli inequality gives, when $C > \mu_n$,

$$Pr(S_n > C) \leq \frac{V_n}{V_n + (C-\mu_n)^2}, \quad C > \mu_n \tag{4}$$

Moment bounds of this kind are classical [9]. For a target overflow probability ε in (0,1), the inequality motivates a threshold μ_n plus a variance-dependent safety margin. That threshold can exceed the largest demand allowed by the finite support. We therefore combine the moment threshold with the deterministic maximum:

$$H_n = min\{A_n, \mu_n + \sqrt{\frac{1-\varepsilon}{\varepsilon}}\sqrt{V_n}\} \tag{5}$$

Proposition 1. Admitting a prefix only when $H_n \leq C$ ensures overflow probability at most ε, provided V_n is its true variance. If $A_n \leq C$, overflow is impossible. Otherwise the moment branch must satisfy the capacity limit, and substitution in Equation (4) gives the bound. If V_n is zero, S_n is deterministic and the mean comparison suffices. The argument does not require independence when a valid aggregate variance is supplied. Independence is used only to obtain $V_{0,n}$ from separate request variances.

Reliability-aware planning with conservative action gating [32] and tail-robust objectives with feasible updates [33] provide related perspectives on restricting decisions under risk. Support information and moment constraints also underpin distributionally robust optimization models [34]. Here, a prefix passes when its deterministic support or valid moment envelope fits within capacity. The guarantee requires aggregate moments and support that remain valid conditional on the information used to propose the candidate prefix.

Risk-constrained cloud-native scaling [35], scheduling under an energy budget [36], and multi-horizon demand forecasting with stochastic resource planning [37] consider decisions within explicit resource envelopes. A cache admission gate likewise needs a defined feasible event. In the present model, finite support and aggregate moments describe retained KV pages; forecasts used to propose a prefix must be assessed against that same event.

Every rule in Table 1 selects the longest admissible prefix, including the empty prefix when no request passes. Clipping the moment threshold guarantees that every prefix accepted by the worst case rule is also accepted by clipped Cantelli. Under the same true distribution, exact probability admission accepts at least as many requests as either valid sufficient rule. Expected demand admission has no corresponding probability guarantee and is not generally an upper bound on the exact admissible prefix. Rare sufficiently large tails can produce a mean above capacity while their probability remains below ε.

Table 1. Prefix admission tests under a common cache budget.

Rule	Acceptable prefix condition	Risk interpretation
Expected demand	$\mu_n \leq C$	Controls the mean only; no chance guarantee
Worst case	$A_n \leq C$	Zero risk within the specified length support
Clipped Cantelli	$H_n \leq C$ using true V_n	Sufficient for conditional risk no greater than ε
Exact probability	$Pr(S_n > C) \leq \varepsilon$	Longest prefix under the specified joint law

A simple demand distribution clarifies the lack of a general ordering with the mean rule. Suppose one request requires either one page with probability 0.99 or 10001 pages with probability 0.01. Its mean is 101 pages. At capacity 100 pages and $\varepsilon = 0.05$, expected demand admission rejects the request, while exact probability admission accepts it because its overflow probability is 0.01. The numerical workload uses the token supports in Section 5.

3.3 Exact Evaluation under Independent Lengths

The finite length model permits direct computation of the aggregate probability mass. Let $f_n(x)$ be the probability that the first n requests use exactly x pages, with $f_0(0) = 1$ and zero mass at other states. The recursion is

$$f_n(x) = (1 - q_n) f_{n-1}(x - b_n) + q_n f_{n-1}(x - b_n - d_n) \quad (6)$$

Summing $f_n(x)$ over $x > C$ gives the exact overflow risk for that prefix. The reference chooses the largest n whose tail probability does not exceed ε. Since every request has nonnegative demand, overflow risk is nondecreasing with prefix length. A scan can therefore stop at the first violation. A dense implementation requires work proportional to the number of requests times the largest page state, using one probability vector at a time. The reference computation is pseudopolynomial in demand measured in pages.

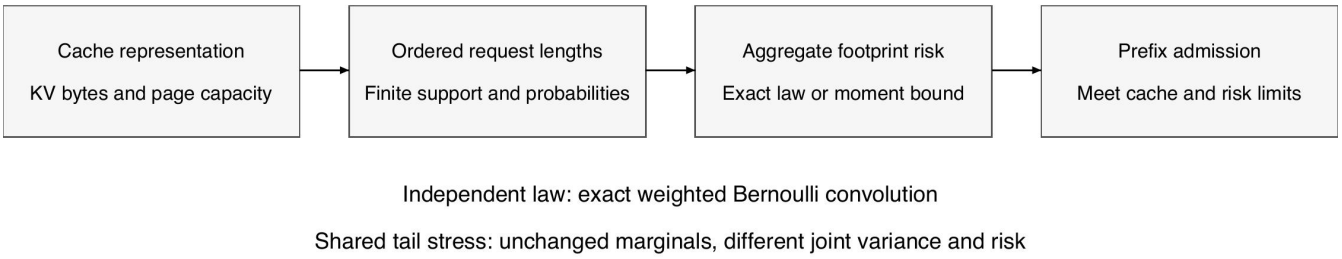

Figure 1. Cache accounting, length assumptions, and ordered admission.

4. Dependence Stress Model

An independence based risk calculation can be exact for the wrong joint distribution. To expose this distinction, construct a common-uniform mode: draw U uniformly on [0,1] and let each long-output indicator be one when $U < q_i$. This preserves every marginal probability q_i while making long outputs occur together. For any fixed prefix, sort the distinct values in $\{0,1,q_1,...,q_n\}$ as interval endpoints τ_r. Within each adjacent interval, a midpoint determines a constant aggregate footprint s_r. Its common-mode overflow risk is

$$R_{com,n}(C) = \sum_r (\tau_{r+1} - \tau_r) 1\{s_r > C\} \quad (7)$$

Endpoints have zero probability, so midpoint evaluation avoids boundary ambiguity. We then select the common mode with probability ρ and the independent mode with probability 1 − ρ. The parameter ρ is a mixture weight, not a common pairwise correlation coefficient. For any fixed admitted prefix,

$$R_{\rho,n}(C) = (1 - \rho) R_{ind,n}(C) + \rho R_{com,n}(C) \quad (8)$$

The mean is unchanged because both modes have identical marginal distributions. Their covariance structure differs, giving

$$V_{\rho,n} = V_{0,n} + 2\rho \sum_{1 \le i < j \le n} d_i d_j [\min(q_i, q_j) - q_i q_j] \quad (9)$$

No additional mixture-of-means variance term appears because the two mode means coincide. A dependence aware Cantelli rule substitutes this true variance into Equation (5) and recomputes the admissible prefix. It retains the conditional risk guarantee. By contrast, a prefix fixed under independence may violate the target under the mixture. Its risk varies linearly with ρ, but the slope can be positive or negative depending on the capacity threshold; positive dependence does not increase every possible upper-tail probability. The stress comparison tests distributional mismatch.

Conditioning on an evolving workload state is a separate modeling task from evaluating a fixed joint distribution. Risk prediction under distribution shift [38], recursive latent-state risk modeling [39], and

adaptive modeling of risk transmission [40] address different aspects of changing risk states. Graph-based risk identification [41] and cross-scale temporal regime classification [42] motivate examining structural and temporal strata when estimating demand. The shared-tail construction provides a controlled way to test one such change while preserving all marginal probabilities.

Causal discovery using financial knowledge graphs [43], cross-period anomaly detection [44], and sequence-based risk detection [45] distinguish temporal state from relational structure. The shared-tail experiment isolates a corresponding modeling choice in aggregate demand: individual output probabilities stay fixed while their joint dependence changes. This separates the effect of dependence assumptions from changes in marginal demand forecasts.

Joint budget-allocation models [46] also highlight how decisions can compete for a common resource envelope. In the cache setting, the cohort shares a finite page budget, and the relevant event concerns the sum of its retained footprints. This resource coupling must be distinguished from statistical dependence among output lengths. The mixture parameter changes the latter without changing the former.

5. Numerical Analysis

5.1 Cohorts and Validation

The study uses 1000 cohorts of 32 requests generated with NumPy default_rng and seed 20250317. For each cohort, the generator first samples prompt lengths uniformly from the integers 64 through 384, then samples short outputs from {32,64}, long outputs from {256,512}, and long-output probabilities from {0.05,0.10,0.20,0.30,0.40}, all with equal choice probabilities within their respective sets. The order of these draws is fixed in the script. Prompts need not be page aligned, and both output endpoints are rounded before their page difference is calculated.

Residual cache budgets are 128, 256, 512, and 768 MiB, corresponding to 64, 128, 256, and 384 pages. Risk targets are 0.01, 0.05, and 0.10. Four rules across all budgets and targets produce 48000 admission records. Every record includes its prefix size, conditional mean demand, exact overflow risk, and expected payload utilization μ_n/C. Utilization is the expected payload occupancy under the specified length model. In particular, a high mean utilization can coexist with a substantial probability of overflow.

Validation checks probability mass conservation, nondecreasing prefix risk, capacity feasibility of deterministic support limits, and the ordering of worst case, clipped Cantelli, and exact admissible prefixes. Independent enumeration of all Bernoulli outcomes checks the first ten prefixes of the first five cohorts. It agrees with convolution to a maximum absolute probability mass difference of 5.55×10^{-17}. Stress tests additionally evaluate $\rho = 0$, 0.25, 0.50, 0.75, and 1 at 512 MiB and $\varepsilon = 0.05$. The original admissions remain fixed when assessing mismatch; only the dependence aware bound recomputes its prefix using the true mixture variance.

All reported overflow quantities are calculated probabilities conditional on a cohort, not frequencies measured from generated responses. We report both average and maximum cohort risks because an average alone cannot establish a per-cohort chance constraint. For admitted counts, the sample standard deviation describes variation across generated cohorts. The support and probability assumptions are supplied by construction, and no length predictor is fitted.

5.2 Admission and Risk Tradeoffs

Table 2. Admission at 512 MiB and ε = 0.05 (1000 cohorts).

Rule	Admitted mean ± SD	Mean risk	Max risk	Mean utilization
Expected	11.229 ± 1.076	0.321714	0.534646	0.953
Worst case	6.201 ± 0.733	0.000000	0.000000	0.527
Clipped Cantelli	6.717 ± 0.982	0.000020	0.000653	0.568
Exact	9.095 ± 1.034	0.021549	0.049980	0.770

Table 2 compares the 512 MiB condition. At ε = 0.05, expected demand admission accepts 11.229 requests on average, but its mean exact overflow risk is 0.32171 and its largest cohort risk is 0.53465. Satisfying a mean demand limit therefore leaves a substantial tail in this instance family. Exact chance constrained admission reduces the mean prefix to 9.095 and its mean risk to 0.02155; its maximum risk is 0.04998. Discrete demand states and integer prefix decisions explain the mean risk below the target.

Worst case admission accepts 6.201 requests on average with zero overflow risk within the finite support. Clipped Cantelli accepts 6.717, adding 0.516 requests on average while retaining the target guarantee. Its mean risk is only 0.000020 and maximum risk 0.000653, indicating substantial conservatism relative to the exact reference. Its mean utilization is 0.56776, compared with 0.76971 for exact admission and 0.95287 for expected demand admission. These differences describe reserved payload capacity.

Support clipping also has a measurable effect on the moment baseline itself. At 512 MiB and ε = 0.05, it increases the admissible prefix by 0.092 requests on average relative to the unmodified Cantelli threshold, benefiting 9.0% of cohorts. Clipping cannot make the rule less safe because it adds only prefixes whose deterministic maximum fits. The improvement remains modest here, and the bound still sacrifices an average of 2.378 admissions relative to exact probability evaluation. Thus the moment rule is best understood as an auditable sufficient test whose conservative cost must be assessed for the workload.

The risk target changes these tradeoffs. At 512 MiB, relaxing ε from 0.01 to 0.10 raises exact mean admissions from 8.251 to 9.579, with mean risks 0.003478 and 0.049472. Clipped Cantelli grows from 6.201 to 7.807 admissions. Expected demand and worst case rules remain unchanged because their tests do not use ε. The exact and moment tests admit nested feasible prefixes as the target is relaxed, but their different uses of distributional information leave different amounts of unused risk allowance.

At ε = 0.05, the exact mean prefix grows from 1.575 at 128 MiB to 14.414 at 768 MiB. The corresponding clipped Cantelli values are 1.218 and 11.188. Figure 2 places this capacity response beside the dependence stress results. Increasing memory enlarges the feasible prefix set for each fixed rule. Changing the joint length model, however, changes the risk assigned to an already admitted prefix even when every individual long-output probability stays the same. The stress curves therefore distinguish the capacity effect from the assumptions that justify the admission decision.

At a shared tail mixture weight of 0.25, the fixed independent exact prefixes still contain 9.095 requests on average, but their mean risk rises to 0.06367; 74.9% of cohorts have calculated conditional risk above 0.05. At weight one, mean risk is 0.19005 and the violating share is 98.9%. Using the true mixture variance, dependence aware Cantelli admits 6.283 requests on average at weight 0.25, with mean risk 0.001013 and maximum risk 0.025001. Every cohort remains within the target. This comparison isolates an assumption mismatch; the dependence aware rule has additional information about the true joint law.

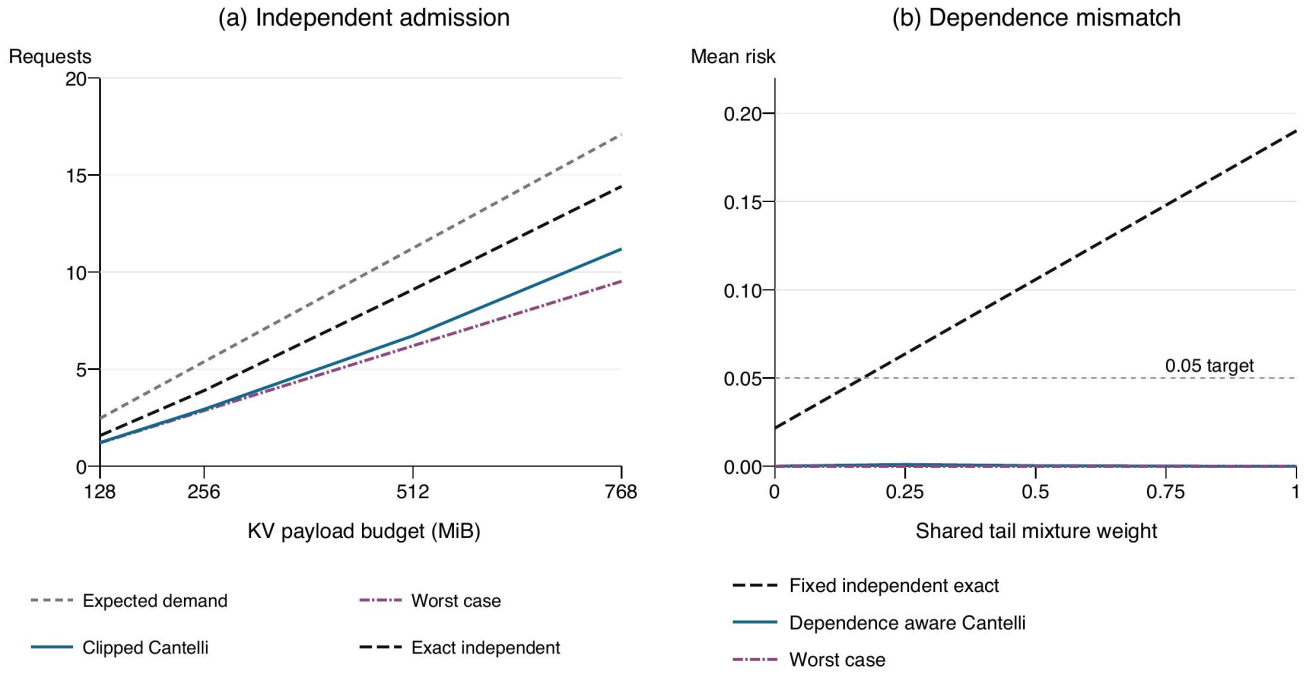

Figure 2. Mean admissions at ε = 0.05 and dependence stress at 512 MiB.

6. Discussion and Conclusions

The central limitation is the availability of a correct conditional demand model. Independence or an aggregate variance cannot be inferred from marginal probabilities alone. In practice, request type, decoding limits, and shared content may affect length dependence. A deployed gate would need validation of the relevant joint tail or a defensible variance bound. Substituting an estimated variance into Equation (5) does not automatically preserve the guarantee. The dependence aware stress rule uses the constructed mixture variance.

Monitoring the conditions under which a fitted demand model is used is therefore part of a deployment evaluation. Distributed anomaly detection with representation alignment [47], dynamic graph modeling of cloud anomalies [48], and temporal dependency modeling for cloud anomaly detection [49] provide complementary monitoring directions. Topology-aware node fault identification [50] and consistency-aware multi-source anomaly detection [51] add structural and cross-source checks. Prediction-aware queueing analysis likewise treats variable service times, evolving KV footprints, and preemption choices as interacting features of LLM inference [52]. These observations motivate rechecking the joint-demand assumptions after a change in workload conditions.

An upstream monitoring layer could consider self-supervised temporal modeling [53], contrastive masked autoencoding [54], adaptive temporal state representations [55], and structural dependency representations [56]. Continual cloud anomaly learning [57] and inference-backend behavior representations [58] provide further contexts for detecting evolving conditions. Distributed estimation also raises secure aggregation [59], cross-cloud anomaly detection [60], and robustness with scarce or noisy observations [61]. The admission guarantee still requires validation of the resulting conditional moments or joint tail.

Backend diagnostics can examine query-plan cost deviations [62], reconstruction-based anomaly signals [63], and microservice call-graph changes [64]. Dynamic dependency graphs [65], temporal evidence for cloud failure diagnosis [66], semantic log modeling [67], and causal root-cause localization [68] offer additional diagnostic views. A deployment evaluation could record such signals alongside predicted and

realized cache footprints, then test whether they identify conditions under which the assumed demand model becomes unreliable.

A finite output support also has operational meaning. If the long endpoint is an enforced generation cap, the deterministic bound remains valid within the accounting assumptions. If it is only a forecasted high length, outputs beyond that endpoint invalidate the claimed support guarantee. A production policy must therefore distinguish a hard limit from a predicted percentile and decide how additional generation is handled. Truncating an answer can protect a memory bound while affecting task completion; neither answer quality nor the acceptability of truncation is evaluated by the admission counts reported here.

The probability target applies to one admitted cohort, rather than an entire service lifetime. Even if each of K cohorts satisfies a valid conditional bound ε, a union bound gives only Kε, capped at one, for the probability of any overflow across that horizon; independence between cohorts is unnecessary for that inequality. A service seeking a horizon-level budget would need to allocate risk across decisions and verify the conditioning used at each admission. The value 0.05 is the comparison parameter for this finite model.

The cache accounting also has a deliberate scope. Prefix sharing, sliding windows, quantization metadata, parallel replication, and early block release can change the payload event. Offloading introduces transfer constraints [10]. Likewise, the execution arrangements studied by DistServe and Sarathi-Serve affect latency and service capacity beyond cache occupancy [6-7]. A memory gate should therefore be combined with compute, bandwidth, and service-level controls. The fixed retention cohort gives a transparent capacity reference, while a continuous batching evaluation would require token timing, arrivals, completions, and the allocator's release policy.

Deployment across heterogeneous GPUs and network conditions adds further resource variation, as considered by ThunderServe [69]. A gate integrated into such a system would need a residual cache budget matched to the selected placement and parallel configuration. Changes in those execution choices should therefore trigger a new footprint calculation before reusing an admission decision.

The two point distribution supports exact checks but is a restricted workload model. Real output lengths can have many values and may change under task mix or generation settings. Finer distributions enlarge the convolution state space, while moment methods remain computationally simple but may become more conservative. Future evaluation should compare predicted and realized cohort footprints, stratify failures by workload condition, and include the cost of estimating length distributions. It should distinguish model misspecification from numerical error and resource exhaustion caused by memory components outside the KV payload budget.

Broader workload evaluation can include long-document classification [70], opinion-target sentiment classification [71], and entity recognition [72]. Keyword-constrained retrieval-augmented generation [73] and structural prompts for fine-tuning [74] provide further task and configuration settings in which to assess demand estimates. Evaluating the admission gate across these settings requires their own prompt and output-length distributions, as well as payload accounting matched to the deployed model.

Knowledge-graph semantic fusion for long-text classification [75], instruction-driven classification with semantic alignment [76], structure-aware financial-report summarization [77], and semantic matching for recommendation [78] provide additional model and task configurations for future resource evaluation. For implementations that use autoregressive generation, cache demand must be defined from the deployed representation and its actual prompt and output distributions.

Multimodal emotional understanding with interpretable feedback [79] and recommendation with temporal and social context [80] extend this evaluation scope to services with different input forms and output requirements. For deployments of such services that use autoregressive generation, a finite-support admission

model must specify which generated tokens contribute to persistent cache demand and whether its output cap is enforced. Application-level quality and task completion should then be assessed alongside the resource and overflow measures.

This study links explicit KV byte accounting to conditional admission risk for ordered batches. Support clipped Cantelli combines deterministic support with a valid moment bound; exact convolution quantifies its conservative cost under independent finite lengths. Across 1000 cohorts, expected footprint admission achieves higher occupancy by accepting risks far above the specified target, while conservative and exact rules expose different admission tradeoffs. A shared tail mixture shows why unchanged marginal lengths do not establish unchanged batch risk. The resulting framework supplies a reproducible cache capacity analysis whose guarantees remain tied to its footprint definition and joint demand assumptions.

References

[1] W. Kwon, Z. Li, S. Zhuang, et al., Efficient Memory Management for Large Language Model Serving with PagedAttention, Proceedings of the 29th Symposium on Operating Systems Principles, pp. 611-626, 2023. doi: 10.1145/3600006.3613165.

[2] 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, 16th USENIX Symposium on Operating Systems Design and Implementation, pp. 521-538, 2022.

[3] J. Ainslie, J. Lee-Thorp, M. de Jong, et al., GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints, Proceedings of EMNLP, pp. 4895-4901, 2023. doi: 10.18653/v1/2023.emnlp-main.298.

[4] N. Shazeer, Fast Transformer Decoding: One Write-Head is All You Need, arXiv preprint arXiv:1911.02150, 2019.

[5] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness, Advances in Neural Information Processing Systems, vol. 35, pp. 16344-16359, 2022.

[6] Y. Zhong, S. Liu, J. Chen, et al., DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving, 18th USENIX Symposium on Operating Systems Design and Implementation, pp. 193-210, 2024.

[7] A. Agrawal, N. Kedia, A. Panwar, et al., Taming Throughput-Latency Tradeoff in LLM Inference with Sarathi-Serve, 18th USENIX Symposium on Operating Systems Design and Implementation, pp. 117-134, 2024.

[8] Llama Team, The Llama 3 Herd of Models, arXiv preprint arXiv:2407.21783, 2024.

[9] D. Bertsimas and I. Popescu, Optimal Inequalities in Probability Theory: A Convex Optimization Approach, SIAM Journal on Optimization, vol. 15, no. 3, pp. 780-804, 2005. doi: 10.1137/S1052623401399903.

[10] Y. Sheng, L. Zheng, B. Yuan, et al., FlexGen: High-Throughput Generative Inference of Large Language Models with a Single GPU, Proceedings of Machine Learning Research, vol. 202, pp. 31094-31116, 2023.

[11] P. Jaillet, J. Jiang, K. Mellou, M. Molinaro, C. Podimata, and Z. Zhou, Online Scheduling for LLM Inference with KV Cache Constraints, arXiv preprint arXiv:2502.07115v4, 2025. doi: 10.48550/arXiv.2502.07115.

[12] Z. Ye, L. Chen, R. Lai, W. Lin, Y. Zhang, S. Wang, T. Chen, B. Kasikci, V. Grover, A. Krishnamurthy, and L. Ceze, FlashInfer: Efficient and Customizable Attention Engine for LLM Inference Serving, Proceedings of Machine Learning and Systems, vol. 7, 2025.

[13] A. Shutova, V. Malinovskii, V. Egiazarian, D. Kuznedelev, D. Mazur, N. Surkov, I. Ermakov, and D. Alistarh, Cache Me If You Must: Adaptive Key-Value Quantization for Large Language Models, Proceedings of the 42nd International Conference on Machine Learning, vol. 267, pp. 55451-55473, 2025.

[14] J. Zhang, Learning What to Remember: Key Signal-Aligned Memory Compression for Long-Context LLMs, 3(6), 2024.

[15] B. Chen, Adaptive Fusion of Low-Rank and Soft-Gated Sparse Updates for Parameter-Efficient Instruction Tuning, 4(6), 2024.

[16] F. Wang, D. Verma, and Y. Chen, Multi-Resource Workload Forecasting in Cloud Data Centers via Spatio-Temporal Graph Transformers, 5(8), 2025.

[17] N. Awate and Z. Chen, Research on Multi-scale Dynamic Evolution Time Series Modeling Methods for Predicting Backend System Load Change Trends, 4(10), 2025.

[18] Z. Qing and F. Huang, Attention-Enhanced Network Traffic Prediction for Intelligent Load Balancing Optimization, 5(8), 2025.

[19] R. Shahout, E. Malach, C. Liu, W. Jiang, M. Yu, and M. Mitzenmacher, Don't Stop Me Now: Embedding Based Scheduling for LLMs, The Thirteenth International Conference on Learning Representations, 2025.

[20] Q. Su, W. Zhao, X. Li, M. Andoorveedu, C. Jiang, Z. Zhu, K. Song, C. Giannoula, and G. Pekhimenko, Seesaw: High-throughput LLM Inference via Model Re-sharding, Proceedings of Machine Learning and Systems, vol. 7, 2025.

[21] S. Chen, Z. Jia, S. Khan, A. Krishnamurthy, and P. B. Gibbons, SLOs-Serve: Optimized Serving of Multi-SLO LLMs, arXiv preprint arXiv:2504.08784, 2025. doi: 10.48550/arXiv.2504.08784.

[22] Y. Wang, Reinforcement Learning Based Adaptive Scheduling Method for Throughput Improvement in Distributed Computing Environments, 5(7), 2025.

[23] C. C. Peng, Joint Scheduling Method for Resource Orchestration Optimization in Distributed Microservices via Hierarchical Reinforcement Learning, 4(9), 2025.

[24] S. Shu, Adaptive Resource Scheduling in Distributed Systems via Reinforcement Learning under Dynamic Competition, 5(7), 2025.

[25] C. He and K. Y. Chen, Value-Iteration Reinforcement Learning for Distributed Cloud Workflow Scheduling: A Critical-Task-Prioritized Execution Optimization Framework, 5(2), 2025.

[26] F. Xue, G. Zhu, and J. Taha, Intelligent Backend Service Scheduling through the Integration of Reinforcement Learning and Meta-Learning, 4(6), 2025.

[27] B. Zhang and I. Abdurahman, Research on a Deep Reinforcement Learning-Based Adaptive Replica Allocation Method for Data Replica Placement Optimization in Distributed Storage Systems, 4(8), 2025.

[28] S. Liu, Structured State Representation and Hierarchical Policy Scheduling for Reinforcement Learning-Based Task Planning, 5(11), 2025.

[29] Q. Yang and K. Y. Lin, Autonomous Planning and Execution Method Based on Task Graph Construction for Large Language Model Agents, 4(7), 2025.

[30] Y. Liu, Evolutionary Strategy-Driven Self-Optimization for Adaptive Intelligent Agents in Dynamic Environments, 4(2), 2025.

[31] Y. Chen, S. Liu, and C. Qian, Unified Modeling of Autonomous Exploration and Goal Discovery for Open-Environment Agents, 5(9), 2025.

[32] Z. Luo, Y. W. Hsu, and R. Qeyum, Safe Tool Use in LLM Agents via Reliability-Aware Planning and Conservative Action Gating, 4(7), 2025.

[33] Y. Wang, Risk-Aware Large Language Model Fine-Tuning with Tail-Robust Objectives and Feasible Gradient Updates, 4(3), 2025.

[34] D. Kuhn, S. Shafiee, and W. Wiesemann, Distributionally robust optimization, Acta Numerica, vol. 34, pp. 579-804, 2025. doi: 10.1017/S0962492924000084.

[35] P. Hu, G. Uziel, and X. Ma, Risk-Constrained Reinforcement Learning for SLO-Driven Cloud-Native Elastic Scaling, 3(1), 2024.

[36] J. Sun, Reinforcement Learning-Based Task Scheduling Under Energy Budget Constraints in Edge-Cloud Systems, 4(4), 2024.

[37] Y. Lin, Attrition-Aware Spatio-Temporal Graph Learning and Stochastic Programming for Multi-Horizon Workforce Demand Forecasting and Labor Cost Planning, 4(1), 2024.

[38] L. Zheng, Enterprise Risk Prediction Under Dynamic Distribution Shift Using Robust Adaptive Learning, 5(10), 2025.

[39] J. Chen and A. Tursun, Deep Latent Representation and State Recursion for Enterprise Audit Risk Detection, 5(2), 2025.

[40] S. Hu, Modeling Dynamic Financial Risk Transmission with Adaptive Multi-Agent Reinforcement Learning, 5(2), 2025.

[41] X. Meng, Deep Graph Learning for Concealed Accounting Fraud Detection in Enterprise Networks, 4(9), 2025.

[42] H. Lu, Cross-Scale Temporal Representation Learning for Short-Term Wind-Solar Energy Generation Regime Classification, 5(3), 2025.

[43] L. Chen and S. Liu, Explainable Cross-Market Causal Discovery via Financial Knowledge Graphs, 3(1), 2024.

[44] Z. Peng, Cross-Period Financial Statement Anomaly Detection via Contrastive Representation Learning, 3(8), 2024.

[45] Z. Su, Microstructure-Aware Risk Detection in High-Frequency Trading Using Embedding and Sequence Modeling, 4(3), 2024.

[46] N. Zhao, Y. Lin, and N. Abulikemu, Deep Reinforcement Learning Framework for Joint Corporate Budget-Revenue Allocation and Labor Cost Expenditure Planning, 4(11), 2025.

[47] K. Yang and Z. Liu, Distributed Data Anomaly Detection Algorithm Integrating Federated Learning and Dynamic Representation Alignment Mechanism, 4(5), 2025.

[48] G. Feng, Dynamic Graph Structure Modeling for Anomaly Detection in Complex Cloud Platforms, 5(7), 2025.

[49] B. Su, A Cloud Computing Anomaly Detection Method Integrating Diffusion Graph Neural Networks with Long- and Short-Term Temporal Dependency Modeling, 5(8), 2025.

[50] F. Xue, K. Wu, and A. Abdulla, Topology-Aware Graph Neural Networks for Distributed Node Fault Identification, 5(3), 2025.

[51] Y. Yang, M. Nagel, and A. Li, Consistency-Aware Unified Modeling for Multi-Source System Anomaly Detection, 4(4), 2025.

[52] M. Mitzenmacher and R. Shahout, Queueing, Predictions, and Large Language Models: Challenges and Open Problems, Stochastic Systems, vol. 15, no. 3, pp. 195-219, 2025. doi: 10.1287/stsy.2025.0106.

[53] K. Yang, Federated Learning and Self-Supervised Temporal Modeling for Collaborative Anomaly Detection in Distributed Tasks, 4(10), 2024.

[54] J. Zhou, Contrastive Masked Autoencoding for Self-Supervised Anomaly Representation Learning in Distributed Task Systems, 4(2), 2024.

[55] Y. Zhan, Adaptive Representation Learning and Temporal State Modeling for Robust Anomaly Detection in Distributed Systems, 3(6), 2024.

[56] T. Ying and C. Ding, Self-Supervised Unified Representation Learning with Structural Dependency Modeling for Early Risk Detection in Distributed Systems, 4(2), 2024.

[57] S. Wang, Continual Learning Algorithm for Online Cloud Anomaly Detection, 4(7), 2024.

[58] T. Ding and Y. Yuan, Behavior Representation Learning for Reliable Monitoring of AI Inference Backends, 4(1), 2024.

[59] Y. Ke, Heterogeneity-Aware Federated Optimization with Secure Aggregation for Distributed Data Mining, 4(4), 2024.

[60] S. Zhang, Secure Parameter Aggregation and Distributed Anomaly Detection in Cross-Cloud Data Center Networks, 3(4), 2024.

[61] Y. Mei, Unified Semi-Supervised and Robust Optimization Framework for Low-Resource Noisy Data Mining, 3(7), 2024.

[62] L. Ren, S. Li, and Z. Liu, Variational Autoencoder-Based Query Plan Anomaly Detection and Cost Deviation Warning for Database Systems, 4(2), 2024.

[63] J. Chen, Discriminative Reconstruction Method for Backend Anomaly Detection Based on Diffusion-Generated Prior Enhancement, 4(2), 2024.

[64] Y. Gong, A. Mehta, and Z. Zhou, Research on a Graph Contrastive Learning-Based Anomaly Detection Method for Backend Microservice Call Graph Structure Change Awareness, 4(2), 2024.

[65] F. Chen, Heterogeneous Dynamic Dependency Graph Learning for Anomaly Detection in Large-Scale Backend Systems, 4(7), 2024.

[66] Z. Hu, Temporal Evidence Aggregation with Causal-Contrastive Transformer Learning for Intelligent Cloud Failure Diagnosis, 3(8), 2024.

[67] Z. Yang, Transformer-Based Semantic Log Modeling for Robust and Explainable Cloud Security Analytics, 4(5), 2024.

[68] L. Wang and C. Y. Chang, Structure-Aware Root Cause Localization for Microservice Backends via Joint Graph Representation Learning and Causal Inference, 3(1), 2024.

[69] Y. Jiang, F. Fu, X. Yao, T. Wang, B. Cui, A. Klimovic, and E. Yoneki, ThunderServe: High-performance and Cost-efficient LLM Serving in Cloud Environments, Proceedings of Machine Learning and Systems, vol. 7, 2025.

[70] Z. Wang, A Hierarchical Encoding and Selective Attention Enhancement Framework Based on Large Language Models for Long-Document Text Classification, 4(11), 2025.

[71] Z. Gu and L. Nie, Knowledge Graph-Enhanced Large Language Models for Opinion Target Sentiment Classification, 4(7), 2025.

[72] J. Zhou, Graph-Based Knowledge Representation and Semantic Alignment for Large Language Model-Driven Entity Recognition, 5(6), 2025.

[73] W. Ma and W. Zhong, Keyword-Constrained Retrieval-Augmented Generation for Large Language Models, 4(8), 2025.

[74] Z. Xiao, C. Moretti, and Q. Li, Frequent Subgraph Mining and Structural Prompt Injection for Large Language Model Fine-Tuning, 5(7), 2025.

[75] Z. Zhu, Knowledge Graph-Augmented Semantic Fusion for Large Language Model-Based Long-Text Classification, 4(10), 2024.

[76] Y. Jiang and Y. L. Peng, Instruction-Driven Language Model for Text Classification via Cross-Attention Fusion and Semantic Alignment, 4(2), 2024.

[77] M. Wang, P. L. Peng, and J. Chen, Financial Report Summarization via Structure-Aware Modeling and Information Fidelity Constraints, 4(7), 2024.

[78] R. Hu, Unified Semantic Matching with Logistic Similarity Learning for Cold-Start Advertising Recommendation, 4(2), 2024.

[79] J. Y. Su, Vision-Language Model-Based Emotional Understanding and Interpretable Feedback for Art Therapy Support, 5(7), 2025.

[80] Z. Ke, MCA-TAN: Cross-Modal Graph Neural Recommendation with Temporal Behavior and Social Network Fusion, 5(10), 2025.