A DISTRIBUTION-AWARE APPROXIMATE LM HEAD FOR REDUCED-MEMORY DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

The output head of a large language model scores the entire vocabulary at every decoding step, while most decoding methods use only a small set of candidates. This operation is a maximum inner-product search problem in which the rows of the LM head are fixed items and each hidden state is a query. We study when approximate vocabulary search can replace the dense head without changing decoding behavior. Standard search systems, however, optimize top-k recall, a poor target here because it ignores the probabilities of the candidates. We propose the Approximate Head (AH), which searches a compact graph index over the rows of the LM head, optionally reranks the returned candidates and normalizes the scores of those candidates only. At a fixed search budget, AH's time and transient memory do not depend on vocabulary size, and three operating regimes target low latency, low memory, or a balance of the two. On four 7–8B LLMs, rescoring the 64 returned candidates with exact logits keeps AH within 0.1 percentage points of dense multiple-choice accuracy on every language-matched benchmark, whereas decoding from its own approximate scores loses at most 7.1 points. A low-margin rescoring fallback recovers most of that loss for about 6% of serving throughput on Qwen3-8B-FP8, and in vLLM AH cuts the head's share of resident GPU memory, at most 5.5% for the dense head, by at least $2.7\times$.

1 INTRODUCTION

Autoregressive decoding ends each step with a dense projection from the hidden state to every vocabulary row, but the large output $B \times |\mathcal{V}|$ logit tensor is immediately reduced to a much smaller set for greedy decoding, top-k sampling (Fan et al., 2018), or nucleus sampling (Holtzman et al., 2020), so the head computes far more than needed. With vocabularies past 10^5 tokens, and serving engines such as vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024) reducing other activation costs through paged KV caches, continuous batching, and specialized scheduling, the head spends a growing share of time and memory on work that decoding discards. Across the LLMs we evaluate, its fp16 weights take at least 783 MiB of GPU memory, and its logits take at least 6.1 MiB per step at batch size $B = 32$.

The fixed LM head makes each decoding step a maximum inner-product search (MIPS) problem, in which each vocabulary row is an item, each hidden state is a query, and decoding needs only a subset of candidates for the next token. However, top-k recall fits this problem poorly, because it rewards recovering set membership even when the missed tokens carry negligible probability and penalizes misses that leave the decoded distribution unchanged. Put simply, standard MIPS ignores the probabilities that decide whether a candidate token matters. We show that Jensen–Shannon divergence (JSD) between the dense and approximate next-token distributions fits better, because it is bounded, symmetric, and sensitive to changes in the probability distribution.

We propose the Approximate Head (AH), a distribution-aware alternative to the dense LM head that builds a graph index over the rows of the LM head once per checkpoint. At each decoding step, AH searches a compact index, keeps the best candidates it visits, optionally reranks them, and normalizes the final scores over those candidates. We tune it to minimize JSD, which produces a Pareto front from which we name regimes for low latency, low resident memory, and a balance of both. For evaluation, we have a *rescored* arm that keeps AH's 64 candidates but replaces their scores with exact dense logits, so it measures only which tokens are kept and still needs the dense rows at

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

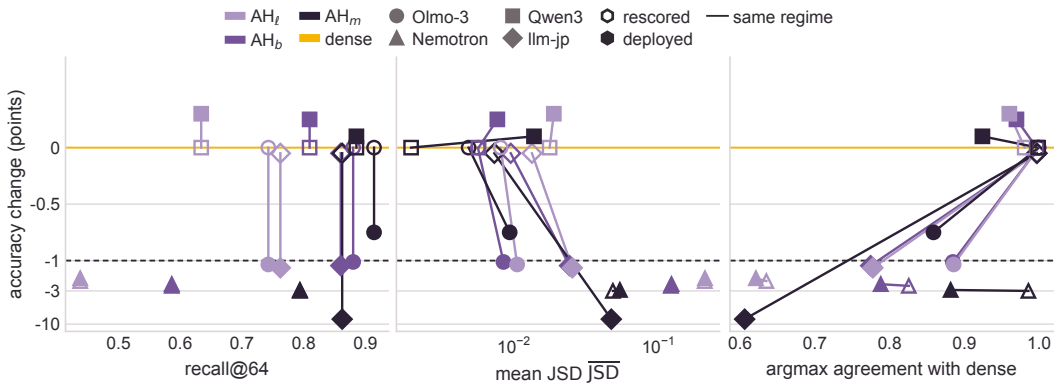

Figure 1. Recall cannot see score error, since a regime's rescored and deployed arms return the same 64 candidates and so share one recall@64, while the deployed arm's own scores shift its mean JSD, argmax agreement and accuracy change against dense. Each point is one model and regime averaged over MMLU and JMMLU, with the rescored arm open, the deployed arm filled and a segment joining the two. The dashed line marks a one-point loss.

decoding time, while a *deployed* arm decodes from AH's own approximate scores. We measure both, together with a *fallback* arm that rescores on low top-two margin steps. Figure 1 summarizes the result: a regime's two arms return the same candidates and so share one recall, yet the deployed arm agrees less with the dense argmax and changes accuracy more, so recall cannot see the score error that JSD and argmax agreement expose. In short, our contributions are:

1. **Per-step certificates.** JSD certifies decoding decisions step by step: we show that the rescored arm's JSD is an exact function of the total dense probability of the tokens it keeps, which certifies 94.9% of AH$_m$'s rescored steps, and bound the deployed arm with the Jensen–Shannon instance of a known f-divergence certificate.
2. **Vocabulary-independent head.** We implement graph search over compact indexes, optional rerank copies and a margin-triggered fallback, which make the head's time and transient memory independent of vocabulary size and remove over 99.9% of the logit memory on four LLMs.
3. **Serving.** AH runs in the vLLM inference engine on five GPU types with the dense weights removed, cutting the head's share of resident memory by at least $2.7\times$.

2 Background

Let $\mathbf{h} \in \mathbb{R}^d$ be one decoder hidden state and $\mathbf{W} \in \mathbb{R}^{|\mathcal{V}| \times d}$ the LM head over the vocabulary $\mathcal{V}$, with one row $\mathbf{w}_i$ per token. A **dense** head forms the logits $\mathbf{z} = \mathbf{W}\mathbf{h}$, with $z_i = \langle \mathbf{w}_i, \mathbf{h} \rangle$, and the distribution $\mathbf{p} = \mathrm{softmax}(\mathbf{z})$. However, greedy decoding needs only the largest logit and top-k decoding normalizes over the k largest, so computing a score for every row before this reduction is wasteful.

We can think of each row $\mathbf{w}_i$ as a database item and $\mathbf{h}$ as a query, making the LM head into MIPS (Shrivastava & Li, 2014). Approximate nearest-neighbor (**ANN**) search answers such a query by visiting a budgeted subset of rows and returning a candidate set $\mathcal{S}$. Graph ANN methods such as HNSW (Malkov & Yashunin, 2020), DiskANN (Subramanya et al., 2019) and CAGRA (Ootomo et al., 2024) store each item as a node with edges to nearby rows, and a query starts from a few entry rows, expands the most promising frontier nodes, scores their neighbors and keeps the best candidates. The head then decodes from the approximate distribution $q_i = \exp(\tilde{z}_i) / \sum_{j \in \mathcal{S}} \exp(\tilde{z}_j)$ for $i \in \mathcal{S}$ and $q_i = 0$ otherwise, where $\tilde{z}_i$ is the approximate score of row i used for decoding.

Thus, approximate search algorithms save memory by storing rows compactly and also by visiting a limited number of rows. The search reads a compressed row $\tilde{\mathbf{w}}_i = \mathrm{enc}(\mathbf{w}_i)$ of each dense row, and these rows form the search **index** $\widetilde{\mathbf{W}}$, which stays resident on the GPU and reduces both resident

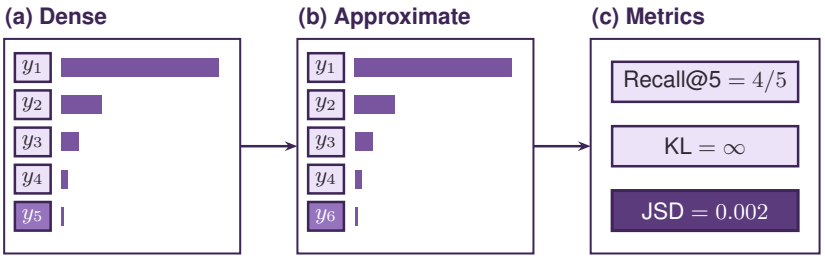

Figure 2. One support substitution, in which the approximate head returns token y_6 in place of the token y_5, whose dense probability is 0.6%. Recall charges a full slot, forward KL is infinite because $\mathbf{q}$ gives y_5 zero probability, and JSD stays small because the distribution barely changes.

memory and memory traffic. The index can store a reduced-precision view of the rows, in bf16, fp16, fp8 (Micikevicius et al., 2022), MXFP8 (Rouhani et al., 2023) or a low-bit vector code. Product quantization replaces each subvector with the identifier of a learned centroid (Jégou et al., 2011), and RaBitQ applies a random rotation and stores a few bits per coordinate together with the row norm (Gao & Long, 2024). When more accurate scores are needed, an optional *rerank copy* $\mathbf{R}$ stores a second, more accurate view of the same rows, which is loaded only after the inexpensive search has narrowed the field, to rerank the pool of k' candidates the search keeps.

In that sense, the quality of an approximate head shows in the shift of its output distribution. As illustrated in Figure 2, top-k recall is not suited for decoding because it penalizes replacing a token by an equal-probability one, while forward KL has the opposite problem, since $q_i = 0$ outside $\mathcal{S}$ means that any omitted token with $p_i > 0$ makes $\mathrm{KL}(\mathbf{p}\|\mathbf{q})$ infinite even when the omitted probability is irrelevant to decoding. On the other hand, **Jensen–Shannon** divergence (Lin, 1991),

$$\mathrm{JSD}(\mathbf{p},\mathbf{q}) = \tfrac{1}{2}\mathrm{KL}\big(\mathbf{p} \,\|\, \tfrac{1}{2}(\mathbf{p}+\mathbf{q})\big) + \tfrac{1}{2}\mathrm{KL}\big(\mathbf{q} \,\|\, \tfrac{1}{2}(\mathbf{p}+\mathbf{q})\big), \tag{1}$$

is symmetric and sensitive to how much probability distribution shifts. Both KL terms compare against the mixture $\frac{1}{2}(\mathbf{p}+\mathbf{q})$, whose support is the union of the two supports, and wherever $p_i > 0$ we have $\frac{1}{2}(p_i + q_i) \geq p_i/2 > 0$, so each term is at most $\log 2$ and the JSD stays finite even when $\mathbf{q}$ assigns zero probability outside $\mathcal{S}$.

3 Method

AH is built once when the model loads, producing a graph over the rows of the head, a compact search index and, optionally, a rerank copy. At each step, AH uses the hidden state as a query to search the graph, scores only the rows it visits and keeps the best of them in a pool, which it may rerank with the rerank copy before normalizing the scores of the final $k = 64$ candidates into the output distribution. Figure 3 compares AH with dense and streaming top-k heads.

We select three **regimes** to weigh latency and memory differently, AH_ℓ for low latency, AH_m for low memory, and AH_b for a balance of latency and memory. AH_ℓ stores an 8-bit RaBitQ index scored with int8 dot products, which the GPU computes with its DP4A instruction, and keeps an fp8 rerank copy of $\mathbf{W}$ with rows $\mathbf{r}_i$, whose scores $\langle \mathbf{r}_i, \mathbf{h}\rangle$ replace the index scores of its returned candidates, since its pool and returned set coincide at $k' = k = 64$, and which makes it the one regime with more resident memory than the dense head. AH_b stores an fp8 view of $\mathbf{W}$ as its index and normalizes the index scores directly with no rerank copy, while AH_m stores a 4-bit RaBitQ index, also with no rerank copy, but counterbalances it by having a larger search budget. All three regimes run the query of Algorithm 1, which keeps a visited set $\mathcal{U}$, so that no row is scored twice, and a pool $\mathcal{C}$ of size k' with the best visited rows under the index scores $\tilde{z}_i = \langle \tilde{\mathbf{w}}_i, \mathbf{h}\rangle$. For a set $\mathcal{A}$ of rows, $\widetilde{\mathbf{W}}_{\mathcal{A}}\mathbf{h}$ and $\mathbf{R}_{\mathcal{A}}\mathbf{h}$ score those rows in one matrix-vector product, $\tilde{z}_{\mathcal{A}}$ collects their scores, and $\mathrm{topk}(\mathcal{A}, r)$ returns the r rows of $\mathcal{A}$ with the largest current scores. The E entry rows are drawn pseudo-randomly from the vocabulary to seed the graph walk, as in CAGRA, while the frontier $\mathcal{F}$ consists of the F highest-scoring rows of $\mathcal{C}$ that have not been expanded yet, and $\mathcal{J}$ holds the neighbors of the frontier that the walk has not scored yet.

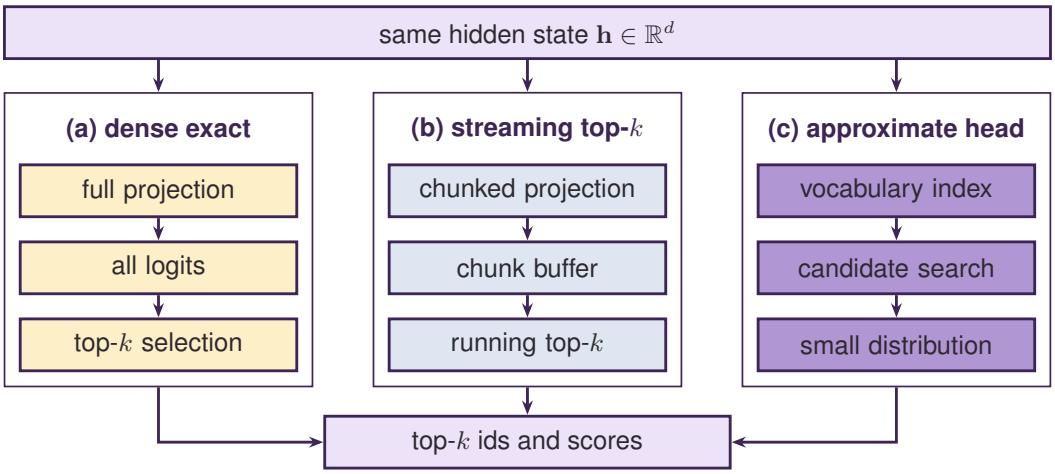

Figure 3. Dense, streaming top-k and AH heads, stage by stage on one hidden state. The dense head materializes the full logit tensor, streaming top-k keeps a small running top-k but scores every row, and AH scores only the rows its search visits.

Algorithm 1 AH query for one hidden state.

Input: hidden state $\mathbf{h} \in \mathbb{R}^d$, neighbor lists $\mathcal{N}$, search index $\widetilde{\mathbf{W}}$, rerank copy $\mathbf{R}$ or None, budget (E, F, T, k'), output size k
Output: returned set $\mathcal{S}$ and probabilities $\mathbf{q}_{\mathcal{S}}$

$\mathcal{U} \leftarrow \mathsf{random.sample}(\mathcal{V}, E)$
$\tilde{z}_{\mathcal{U}} \leftarrow \widetilde{\mathbf{W}}_{\mathcal{U}}\mathbf{h}$
$\mathcal{C} \leftarrow \mathrm{topk}(\mathcal{U}, k')$
$\mathcal{X} \leftarrow \emptyset$
for $_ \in \mathsf{range}(T)$ **do**
  $\mathcal{F} \leftarrow \mathrm{topk}(\mathcal{C} \setminus \mathcal{X}, F)$
  $\mathcal{X} \leftarrow \mathcal{X} \cup \mathcal{F}$
  $\mathcal{J} \leftarrow \{\, j \in \mathcal{N}(i) : i \in \mathcal{F} \,\} \setminus \mathcal{U}$
  $\tilde{z}_{\mathcal{J}} \leftarrow \widetilde{\mathbf{W}}_{\mathcal{J}}\mathbf{h}$
  $\mathcal{U} \leftarrow \mathcal{U} \cup \mathcal{J}$
  $\mathcal{C} \leftarrow \mathrm{topk}(\mathcal{U}, k')$
if $\mathbf{R}$ **is not** None **then**
  $\tilde{z}_{\mathcal{C}} \leftarrow \mathbf{R}_{\mathcal{C}}\mathbf{h}$
$\mathcal{S} \leftarrow \mathrm{topk}(\mathcal{C}, k)$
$\mathbf{q}_{\mathcal{S}} \leftarrow \mathrm{softmax}(\tilde{z}_{\mathcal{S}})$
return $\mathcal{S}, \mathbf{q}_{\mathcal{S}}$

Once returned, the candidates of a query can be scored in two ways, which we call the two **arms** of the head. The rescored arm replaces the scores $\tilde{z}_i$ of the 64 returned candidates with exact logits z_i before normalizing, so it differs from dense only in which tokens it keeps. The deployed arm decodes from $\tilde{z}_i$ as Algorithm 1 returns them, so it also carries the index's score error. How far each arm may move its distribution before decoding changes is a question of stability.

Let $i^\star = \arg\max_i p_i$, assumed unique, and write the JSD token by token as $\mathrm{JSD}(\mathbf{p}, \mathbf{q}) = \sum_i \psi(p_i, q_i)$ with $\psi(u, v) = \frac{1}{2}u \ln \frac{2u}{u+v} + \frac{1}{2}v \ln \frac{2v}{u+v}$. The rescored arm only drops tokens and keeps the dense order, so Proposition 1 writes its JSD as a function of $p(\mathcal{S})$, the total dense probability of the returned tokens.

Proposition 1 (Rescored arm)**.** *Let $q_i = p_i/p(\mathcal{S})$ for $i \in \mathcal{S}$ and $q_i = 0$ otherwise, where $p(\mathcal{A}) = \sum_{i \in \mathcal{A}} p_i$. Then $\arg\max_i q_i = \arg\max_i p_i = i^\star$ exactly when $i^\star \in \mathcal{S}$, and $\mathrm{JSD}(\mathbf{p}, \mathbf{q}) = \phi\big(p(\mathcal{S})\big)$ with*

$$\phi(s) = \tfrac{1}{2}\Big[s \ln \tfrac{2s}{1+s} + \ln \tfrac{2}{1+s} + (1-s)\ln 2\Big] \geq \tfrac{1}{2}(1-s)\ln 2,$$

which strictly decreases in s. Hence $\mathrm{JSD}(\mathbf{p}, \mathbf{q}) < \phi(1 - p_{i^\star})$ implies $\arg\max_i q_i = i^\star$.

Proof. Dividing by $p(\mathcal{S})$ keeps the order of the returned tokens, while $q_{i^\star} = 0$ when $i^\star \notin \mathcal{S}$. By the homogeneity $\psi(cu, cv) = c\,\psi(u, v)$, the returned tokens contribute $p(\mathcal{S})\,\psi\big(1, 1/p(\mathcal{S})\big)$ and each omitted token $\psi(p_i, 0) = \frac{1}{2} p_i \ln 2$, which sum to $\phi(p(\mathcal{S}))$. Since $\phi'(s) = -\frac{1}{2} \ln \frac{1+s}{s} \le -\frac{1}{2} \ln 2$ and $\phi(1) = 0$, and an omitted $i^\star$ leaves $p(\mathcal{S}) \le 1 - p_{i^\star}$, the bound and the certificate follow. □

The deployed arm's own scores can also reorder the returned tokens, so it keeps the argmax only while its distribution stays close enough to $\mathbf{p}$ that no other token overtakes $i^\star$. The information-limited certificate of Dvijotham et al. (2020, Theorem 2), instantiated with JSD on the head's output distribution and the specification that $i^\star$ beats a token j, gives that level exactly, since $\arg\max_i q_i = i^\star$ whenever $\mathrm{JSD}(\mathbf{p}, \mathbf{q}) < \kappa(\mathbf{p})$ with

$$\kappa(\mathbf{p}) = \min_{j \neq i^\star} \min_{0 \le t \le 1/2} \psi(p_{i^\star}, t) + \psi(p_j, t) + \psi(1 - p_{i^\star} - p_j,\, 1 - 2t), \tag{2}$$

the least JSD at which some token j ties $i^\star$, so no level set by $\mathbf{p}$ alone is larger. Pinsker's inequality (Tsybakov, 2009, Lemma 2.5), applied to both terms of Equation (1), gives $\mathrm{TV}(\mathbf{p}, \mathbf{q}) = \frac{1}{2}\|\mathbf{p} - \mathbf{q}\|_1 \le \sqrt{2\,\mathrm{JSD}(\mathbf{p}, \mathbf{q})}$, and a tie needs $\mathrm{TV}(\mathbf{p}, \mathbf{q}) \ge \Delta(\mathbf{p})/2$ for the margin $\Delta(\mathbf{p}) = p_{i^\star} - \max_{j \neq i^\star} p_j$, so $\kappa(\mathbf{p}) \ge \Delta(\mathbf{p})^2/8$, a level that is loosest when the two leading tokens have low probability, $11\times$ below $\kappa(\mathbf{p})$ for leading probabilities 0.05 and 0.04.

Together, the two levels bound each arm per step. Proposition 1 generalizes the divergence of a distribution from a one-hot one (Martins et al., 2020, App. D), needs no margin and covers any decision among a subset of tokens, such as the answer letters of a multiple-choice question, with $p_{i^\star}$ the probability of the dense choice, while an omitted $i^\star$ is a retrieval failure that rescoring cannot repair (Section 4). Both hold per step, so a regime's mean JSD $\overline{\mathrm{JSD}}$, the mean of $\mathrm{JSD}(\mathbf{p}, \mathbf{q})$ over a set of decoding steps, cannot certify steps on its own (Section 4).

The deployed arm's condition is what the **fallback** acts on, moving between the two arms by deciding per step whether to rescore, and because the dense distribution $\mathbf{p}$ is not available at decoding time, it uses the margin $\Delta(\mathbf{q})$ of the head's own distribution, defined like $\Delta(\mathbf{p})$, and rescores exactly when $\Delta(\mathbf{q})^2/8 < \overline{\mathrm{JSD}}$, with $\overline{\mathrm{JSD}}$ taken on the regime's tuning set, which flags the steps whose own margin would not meet the Pinsker level at the regime's mean JSD. Since the dense matrix is removed from the GPU, the fallback keeps it in page-locked host memory and copies only the 64 candidate rows, 512 KiB, for a step that fires, so it can repair the order of the returned candidates but cannot restore a token the search did not return.

Finally, the time and memory of Algorithm 1 are set by its search **budget**, independent of the vocabulary. A query scores the E entry rows and at most G neighbors of each of the F rows it expands in each of its T iterations, where G is the degree of the graph, so it scores $n \le E + FGT$ rows before reranking, fewer after the visited filter, and the rerank adds at most k' more. Each row score is a d-dimensional inner product that stores one scalar per batch item, so a query performs $O((n + k')d)$ work and a batch of B queries holds $O(B(n + k'))$ transient scores, independent of $|\mathcal{V}|$, against $O(|\mathcal{V}|d)$ work and $O(B|\mathcal{V}|)$ transient scores for the dense head. This bound leaves open whether a fixed budget stays sufficient as the vocabulary grows, and the same settings serve all four evaluated vocabularies, from 100,278 to 196,608 rows (Section 4).

4 Experiments

In this section, we evaluate AH on four public models with $d = 4096$, Olmo-3-7B, Nemotron-Nano-8B, Qwen3-8B-FP8 and llm-jp-4-8b, whose vocabularies range from 100,278 to 196,608 tokens. Accuracy is measured at the answer position of up to 1,000 multiple-choice questions per benchmark, on MMLU (Hendrycks et al., 2021) and JMMLU (Yin et al., 2024) for every model and on a wider suite matched to each model's languages, and greedy replay on MATH-500 (Lightman et al., 2024) and AIME-2024 (Mathematical Association of America, 2024) checks at every step whether a head picks the dense token. Every measurement replays cached dense hidden states through the head, so each deviation comes from the head alone, and the three regimes, chosen by an NSGA-II (Deb et al., 2002) search over $\overline{\mathrm{JSD}}$, latency and memory on 32 MATH-500 prompts of Olmo-3-7B, stay fixed throughout. We compare AH with CAGRA (Ootomo et al., 2024) and a streaming top-k head, measure latency at $B = 32$ on an RTX 4090 unless stated otherwise, and serve AH inside vLLM (Kwon et al., 2023) with the dense weights removed, with further details in Appendix A.

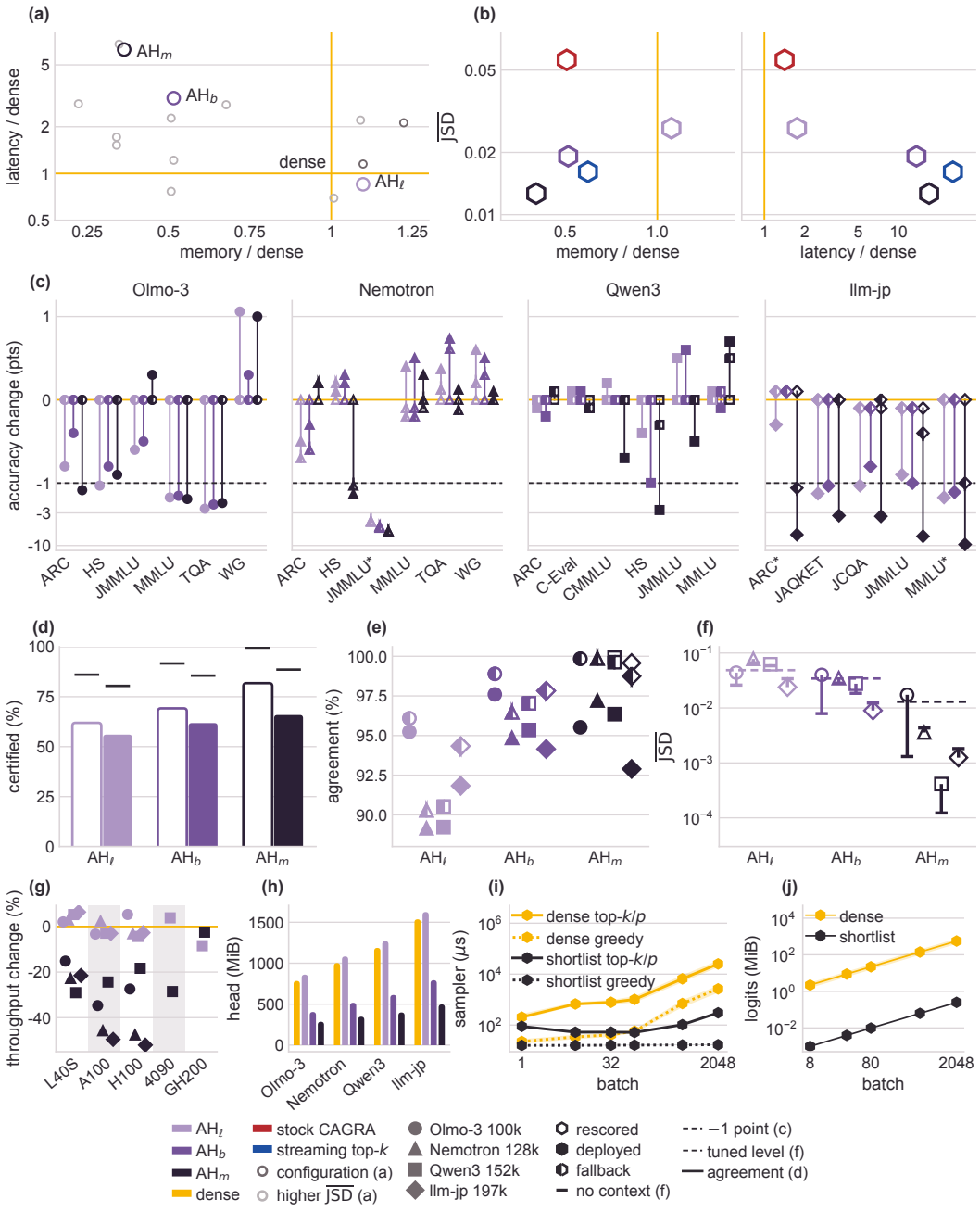

Figure 4. (a) Olmo-3-7B search front, latency and memory relative to dense, lighter circles above the loosest regime's $\overline{\text{JSD}}$. (b) Stock baselines and regimes over MMLU and JMMLU. (c) Accuracy change on every benchmark, one slot per regime with its arms joined, for ARC-Challenge (ARC), HellaSwag (HS), TruthfulQA (TQA), WinoGrande (WG) and JCommonsenseQA (JCQA), controls starred. (d) Answer positions certified by $\Delta(\mathbf{p})^2/8$, 19,905 per regime and arm, under the measured agreement. (e) Greedy-replay agreement and (f) $\overline{\text{JSD}}$ with 4,096 retrieved tokens against none (tick), over MATH-500 and AIME-2024. (g) Served throughput against dense and (h) resident head. (i) Per-step sampler cost and (j) logit tensor, means over the four vocabularies with their range shaded.

We first ask which of recall, JSD and argmax agreement tracks **accuracy** (Figure 1). On the rescored arm accuracy barely moves, and the three metrics order it about the same, but the two arms of one regime separate them, because they return the same candidates and so share one recall, while the deployed arm's distribution moves further from dense, agrees less with the dense argmax and loses more accuracy. Among the baselines (Figure 4(b)), stock CAGRA has the largest JSD and loses a point on average, while streaming top-k matches the JSD of AH at the highest latency, and every configuration of the front that is cheaper than a regime on both costs has a higher JSD (Figure 4(a)). Retuning the search on each GPU finds no configuration that beats a regime on all three objectives in any of the 11 studies with usable data.

Across the fixed **regimes** (Figure 4(c)), the rescored arm stays within 0.1 points of dense on every language-matched benchmark, and only Nemotron's JMMLU control, a benchmark in a language the model does not target, falls below it. The deployed arm loses more, by over a point in 17 of the 54 language-matched regime results of the wider suite, most on llm-jp's Japanese benchmarks and Qwen3's HellaSwag, and the fallback brings all but one of them back within a point, leaving at most 1.1. The losses of llm-jp and of Nemotron's control have different causes. For llm-jp, every returned shortlist, controls included, contains the answer letter dense prefers, so its loss comes from the scores, and a wider search budget cannot help, while the loss is largest under AH_m, which normalizes raw 4-bit scores without a rerank copy. For Nemotron's JMMLU control, the search misses the dense answer on at least 14.5% of the questions in every regime and 24.2% under AH_m, which rescoring cannot repair. Greedy replay on MATH-500 and AIME-2024 orders the arms the same way (Figure 4(e)), with AH_m's rescored arm agreeing with dense on at least 99.5% of steps on every model, its deployed arm on at least 91.2% and the fallback on at least 98.6%. Per token class, numeric tokens never disagree under AH_m, whose worst class, capitalized entities, stays at 0.40%, and under stochastic sampling at temperature 0.7, 22 of 24 rescored cells on 200 questions have a 95% interval of their change against dense that contains zero, and the other two are gains.

Since both results of Section 3 hold per step, we check them on every answer position of the 21 wider-suite cells (Figure 4(d)). The Pinsker level already certifies most steps, and Proposition 1 raises the rescored share to at least 78%, close to the measured agreement and never above it, while certifying more deployed steps would need more of the dense distribution than its margin. Tight margins concentrate on JMMLU, where the Pinsker level certifies 26% of the rescored steps against 71% on MMLU. A regime's mean JSD is a poor proxy for these per-step levels, because the per-step JSD is heavy-tailed, with its median an order of magnitude below its mean.

Beyond replayed hidden states, AH runs as a drop-in head inside vLLM (Kwon et al., 2023), where the dense head weight is removed after the checkpoint loads while paged KV caching and continuous batching stay unchanged, and each served cell runs 8 concurrent requests with greedy decoding and 128 output tokens per request. We **serve** all four models on an L40S, an A100 and an H100, and Qwen3-8B-FP8 also on an RTX 4090 and a GH200 (Figure 4(g)), with AH_ℓ and AH_m, since AH_b has no serving build. The dense head is one $|\mathcal{V}| \times d$ product, so whether AH is faster depends on how fast that product is on each GPU relative to the graph search. AH_ℓ gains on the L40S and the RTX 4090, varies in sign on the A100 and the H100 and loses on the GH200, where Qwen3 runs a fast FP8 product, while AH_m trades throughput for memory. Throughput and time per output token carry these results, since time to first token is a single-run mean over 8 requests, and seven warm passes of the RTX 4090 cell reproduce its throughput gain, 3.4% against 3.8% in the single pass, while its time-to-first-token gain came from a cold start.

The **memory** saving holds on all GPUs, with the dense head taking at most 5.5% of resident GPU memory, and AH_m cuts that share by at least $2.7\times$ and saves at least 63% of the head, from 783 to 285 MiB on Olmo-3-7B (Figure 4(h)), a share that grows slightly with the vocabulary because the RaBitQ rotation is shared across rows. Where the KV cache fills a profiled memory budget, as on the RTX 4090 and the GH200, the cache takes the freed bytes, so total residency stays the same and the cache grows, while the datacenter runs fix the cache at 8 GiB and report the head's share directly. On the activation side, the dense logit tensor grows with batch and vocabulary, to at least 391.7 MiB at $B = 2048$, while AH returns 64 candidates per query, 0.25 MiB at the same batch for every model (Figure 4(j)), and the only transient memory it adds is the one-off index build, which peaks at most 2.8 GiB above the resting allocation and is released before the first request.

The shortlist also serves the **sampler**, which evaluates temperature, min-p, top-k, top-p, argmax and the logprob ranks on the 64 candidates, exactly, since all of them are invariant to columns held at

$-\infty$. Every served run of AH took that path on all 899 sampled steps and avoided 5.6 GB of logit tensor, while requests that need penalties, logit bias, allowed-token ids or wide logprobs take the dense path. Its per-step cost barely varies across the four vocabularies, as the narrow shaded range of Figure 4(i) shows, but greedy sampling at $B = 8$ saves only 12 μs of a 15.7 ms step, so the saving matters under top-k and top-p, where vLLM sorts the whole vocabulary, taking 704 μs dense against 52 μs on the shortlist at $B = 8$ and 26.2 ms against 0.30 ms at $B = 2048$.

The **fallback** was measured on Qwen3-8B-FP8 on the RTX 4090, where over 9,217 sampled steps, AH_m's trigger fires on 33.9% of steps and finds 96% of the token changes that rescoring every step would make, at a cost of 5.5% in time per output token and 5.9% in throughput, from 361 to 340 tokens per second against 508 for dense, and on llm-jp's MMLU control it recovers 8.6 of the 9.6 points the deployed arm loses. Exact sequence identity is hard for greedy decoding, since a single flip rewrites the rest of a continuation, and the dense head against itself on another GPU and vLLM build matches only 3 of 8 sequences.

Finally, prepending up to 4,096 lexically retrieved tokens to every MATH-500 and AIME-2024 prompt changes the hidden states while the index stays fixed, yet it moves each regime's JSD only slightly and leaves the order of the regimes unchanged (Figure 4(f)), and in its worst run AH_m's rescored arm agrees with dense on 97.2% of steps with the retrieved context against 99.2% without it. In free generation with the dense head removed, AH_m decodes 80 math prompts at 2,487 tokens per second on a GH200, and the gold answer is among the boxed answers of all 16 continuations that box one within 512 tokens.

5 Related work

Approximate search and vector compression. AH builds on approximate MIPS, including product quantization (Jégou et al., 2011) and its optimized variant (Ge et al., 2014), graph indexes such as HNSW (Malkov & Yashunin, 2020), NSG (Fu et al., 2019), DiskANN (Subramanya et al., 2019) and CAGRA (Ootomo et al., 2024), and direction-preserving codes such as RaBitQ (Gao & Long, 2024) and its extension (Gao et al., 2025), with descendants such as SymphonyQG (Gou et al., 2025), which fuses codes into graph traversal, and GPU-native IVF-RaBitQ (Shi et al., 2026). Those systems are evaluated with top-k recall, and AH, CAGRA and streaming top-k baselines of Section 4 use the same primitives, although the set AH returns feeds a softmax at every decoding step.

Approximate softmax and sub-vocabulary heads. Many works approximate softmax, like adaptive softmax (Grave et al., 2017), SVD-softmax (Shim et al., 2017), MIPS formulations that score only screened candidates, such as asymmetric LSH (Shrivastava & Li, 2014), MIPS-based inference (Mussmann & Ermon, 2016) and learned screening (Chen et al., 2019), graphs over the vocabulary (Zhang et al., 2018), and clustered shortlists (Amer et al., 2022), with Liu et al. (2020) already noting that recall-tuned MIPS mismatches output layers. For modern LLMs, HiRE (Samaga B L et al., 2024) estimates high-recall top-k logits with a compressed head, and CSV-Decode (Liu et al., 2025) certifies pruning with per-cluster logit bounds but keeps the dense head resident for scoring and fallback. Certified robustness of smoothed classifiers bounds the divergence at which a top class can change, with Rényi certificates in Li et al. (2019) and the general f-divergence form that Section 3 applies to the head's own output distribution.

Decoding and serving efficiency. Top-k (Fan et al., 2018) and nucleus sampling (Holtzman et al., 2020) sample from a small set of high-probability tokens, so an approximate support is viable. Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) cuts target-model passes, as do Medusa (Cai et al., 2024) and EAGLE (Li et al., 2024b), and FR-Spec (Zhao et al., 2025), VocabTrim (Goel et al., 2025) and DynaSpec (Zhang et al., 2025) shrink the drafter vocabulary, but all run the dense target head at each scored position, which AH can replace. Margin-gated fallbacks to exact computation appear in early-exit decoding (Schuster et al., 2022) and in selective higher-precision recomputation of the head (Du et al., 2026), while ours takes its threshold from Equation (2). Fused kernels such as Cut Cross-Entropy (Wijmans et al., 2025) and Liger-Kernel (Hsu et al., 2025) avoid materializing the logit tensor for the training loss, as, concurrently, FlashSampling (Ruiz et al., 2026) does for decode-time sampling, and weight-only quantization such as GPTQ (Frantar et al., 2023), AWQ (Lin et al., 2024) and QuaRot (Ashkboos et al., 2024) shrinks parameter bytes for the same

checkpoint, while AH combines search, compression and normalization to shrink both the logit tensor and the resident head. vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024) leave the head dense, and AH runs inside vLLM as a drop-in head.

6 Conclusion

Limitations. The rescored arm needs the dense rows of its candidates, loaded on demand or kept in memory at a lower precision, which keeps the transient saving but raises resident memory. The fallback keeps the dense matrix in host memory and repairs the order of the returned candidates, but it cannot recover a token the search never returns, as on Nemotron's JMMLU control, and its serving cost was measured only on Qwen3. The three regimes were selected on one model and one dataset at $B = 32$, all four models have 7 to 8B parameters and the same hidden width, every serving run uses a single GPU with 8 concurrent requests, and the quality evidence covers answer positions, greedy replay and short free generation on math.

AH treats the LM head as approximate vocabulary search tuned by the divergence of the next-token distribution, and that divergence certifies decoding step by step. For the rescored arm, the JSD is an exact function of the dense probability the returned tokens keep, which certifies 94.9% of AH_m's rescored steps, and the f-divergence certificate bounds the deployed arm. Across four 7–8B models, the rescored arm stays within 0.1 points of dense on every language-matched benchmark. The deployed arm loses up to 7.1 points, from score error on shortlists that already contain the answer, and the margin-triggered fallback leaves at most 1.1 points for about 6% of throughput. In vLLM, AH cuts the head's share of resident memory by at least $2.7\times$ and its per-step logit memory by over 99.9%, while its throughput depends on how fast each GPU runs the dense product it replaces. Evaluations of approximate heads should therefore report the head a server actually runs, beside the rescored ceiling.

Future work can shard the index across GPUs, with one index per shard and a cross-device merge of the candidates at every step, and replace the fallback's Pinsker trigger with the exact flip cost of the head's own distribution, which would fire less often. It can also extend the regimes to larger models and wider hidden states, and test the head on fine-tuned or LoRA-adapted backbones, in SGLang and at higher concurrency, and on long-form, summarization and code generation.

Reproducibility Statement

Section 3 and Algorithm 1 specify the head, its search space and its tuning objectives and state its proposition with its assumptions and a complete proof, and Table 2 gives the index, rerank copy and measured cost of each regime. Section 4 states the models, benchmarks, question counts, hardware and batch sizes of each evaluation, and Appendices A and D to F give the prompts, answer keys, robustness checks, serving protocol and kernel determinism study behind them. Every model checkpoint and benchmark we use is public.

Ethics Statement

This work studies the efficiency of the LM output head and uses only public model checkpoints and public benchmarks, with no human subjects or personal data. An approximate head can change a model's outputs, so we report the accuracy it loses as deployed alongside the accuracy it keeps with exact rescoring.

AI use statement

In this work, we used generative AI tools for writing and checking proofs, implementing some experiments, and parsing and plotting result data. We have not used generative AI tools for generating synthetic datasets, formulating hypotheses or mathematical claims, or qualitative data analysis, and translation and dataset cleaning are not applicable to this work. Additionally, we used generative AI tools for writing the initial draft and formatting references. We have reviewed all AI-assisted work. The authors checked every reported number against the measurement that produced it and checked

every claim and proof by hand. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Hossam Amer, Young Jin Kim, Mohamed Afify, Hitokazu Matsushita, and Hany Hassan Awadallah. Fast vocabulary projection method via clustering for multilingual machine translation on GPU. In *Proceedings of the 15th Biennial Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pp. 58–69, Orlando, USA, September 2022. Association for Machine Translation in the Americas. URL https://aclanthology.org/2022.amta-research.5/.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 100213–100240. Curran Associates, Inc., 2024. doi:10.52202/079017-3180.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 5209–5235. PMLR, 2024. URL https://proceedings.mlr.press/v235/cai24b.html.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023. doi:10.48550/arXiv.2302.01318.

Pei-Hung (Patrick) Chen, Si Si, Sanjiv Kumar, Yang Li, and Cho-Jui Hsieh. Learning to screen for fast softmax inference on large vocabulary neural networks. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=ByeMB3Act7.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. doi:10.48550/arXiv.1803.05457.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002. doi:10.1109/4235.996017.

Tobias Domhan, Eva Hasler, Ke Tran, Sony Trenous, Bill Byrne, and Felix Hieber. The devil is in the details: On the pitfalls of vocabulary selection in neural machine translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1861–1874, Seattle, United States, July 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.naacl-main.136.

Gaoyuan Du, Anam Nawaz Khan, Rex Zhou, Xiaoyang Liu, Deepayan Chakrabarti, Fnu Suya, and Xueping Li. Greedy decoding is not precision-invariant: Cross-precision output divergence in LLM inference. *Transactions on Machine Learning Research*, September 2026. URL https://openreview.net/forum?id=QDOKyg7a5e.

Krishnamurthy (Dj) Dvijotham, Jamie Hayes, Borja Balle, J. Zico Kolter, Chongli Qin, András György, Kai Xiao, Sven Gowal, and Pushmeet Kohli. A framework for robustness certification of smoothed classifiers using F-divergences. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=SJlKrkSFPH.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi:10.18653/v1/P18-1082.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS. Widely known as GPTQ, the title of its preprint arXiv:2210.17323.

Cong Fu, Chao Xiang, Changxu Wang, and Deng Cai. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proceedings of the VLDB Endowment*, 12(5):461–474, January 2019. doi:10.14778/3303753.3303754.

Jianyang Gao and Cheng Long. RaBitQ: Quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search. *Proceedings of the ACM on Management of Data*, 2(3):1–27, May 2024. doi:10.1145/3654970. SIGMOD 2024.

Jianyang Gao, Yutong Gou, Yuexuan Xu, Yongyi Yang, Cheng Long, and Raymond Chi-Wing Wong. Practical and asymptotically optimal quantization of high-dimensional vectors in Euclidean space for approximate nearest neighbor search. *Proceedings of the ACM on Management of Data*, 3(3): 1–26, June 2025. doi:10.1145/3725413. SIGMOD 2025, known as Extended RaBitQ.

Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):744–755, April 2014. doi:10.1109/TPAMI.2013.240.

Raghavv Goel, Sudhanshu Agrawal, Mukul Gagrani, Junyoung Park, Yifan Zao, He Zhang, Tian Liu, Yiping Yang, Xin Yuan, Jiuyan Lu, Chris Lott, and Mingu Lee. VOCABTRIM: Vocabulary pruning for efficient speculative decoding in LLMs. In *ES-FoMo III: 3rd Workshop on Efficient Systems for Foundation Models at ICML 2025*, 2025. doi:10.48550/arXiv.2506.22694. URL https://openreview.net/forum?id=1xrOnsEJyB.

Yutong Gou, Jianyang Gao, Yuexuan Xu, and Cheng Long. SymphonyQG: Towards symphonious integration of quantization and graph for approximate nearest neighbor search. *Proceedings of the ACM on Management of Data*, 3(1):1–26, February 2025. doi:10.1145/3709730. SIGMOD 2025.

Édouard Grave, Armand Joulin, Moustapha Cissé, David Grangier, and Hervé Jégou. Efficient softmax approximation for GPUs. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1302–1310. PMLR, 2017. URL https://proceedings.mlr.press/v70/grave17a.html.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Pin-Lun Hsu, Yun Dai, Vignesh Kothapalli, Qingquan Song, Shao Tang, Siyu Zhu, Steven Shimizu, Shivam Sahni, Haowen Ning, Yanning Chen, and Zhipeng Wang. Liger-Kernel: Efficient Triton kernels for LLM training. In *CODEML: Championing Open-source Development in Machine Learning Workshop at ICML 2025*, 2025. URL https://openreview.net/forum?id=36SjAIT42G.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline Chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 36, pp. 62991–63010. Curran Associates, Inc., 2023. doi:10.52202/075280-2749.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, January 2011. doi:10.1109/TPAMI.2010.57.

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. JGLUE: Japanese general language understanding evaluation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC)*, pp. 2957–2966, Marseille, France, June 2022. European Language Resources Association. URL https://aclanthology.org/2022.lrec-1.317/.

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

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from Transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/335cd1b90bfa4ee70b39d08a4ae0cf2d-Abstract.html.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. CMMLU: Measuring massive multitask language understanding in Chinese. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11260–11285, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-acl.671.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28935–28948. PMLR, 2024b. URL https://proceedings.mlr.press/v235/li24bt.html.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html. The MATH-500 subset is released as the Hugging Face dataset `HuggingFaceH4/MATH-500`.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 6, pp. 87–100, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html.

Jianhua Lin. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, January 1991. doi:10.1109/18.61115.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.229.

Dong Liu, Shu Wang, Yanxuan Yu, Haisheng Wang, and Ben Lengerich. CSV-Decode: Certifiable sub-vocabulary decoding for efficient large language model inference. *arXiv preprint arXiv:2511.21702*, 2025. doi:10.48550/arXiv.2511.21702.

Zichang Liu, Zhaozhuo Xu, Alan Ji, Jonathan Li, Beidi Chen, and Anshumali Shrivastava. Climbing the WOL: Training for cheaper inference. *arXiv preprint arXiv:2007.01230*, 2020. doi:10.48550/arXiv.2007.01230.

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

Pedro Henrique Martins, Zita Marinho, and André F. T. Martins. Sparse text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4252–4273. Association for Computational Linguistics, 2020. doi:10.18653/v1/2020.emnlp-main.348.

Mathematical Association of America. 2024 American Invitational Mathematics Examination (AIME). Competition problems, distributed as the Hugging Face dataset `Maxwell-Jia/AIME_2024`, MIT license, 2024. URL https://huggingface.co/datasets/Maxwell-Jia/AIME_2024. Accessed September 2026.

Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, Naveen Mellempudi, Stuart Oberman, Mohammad Shoeybi, Michael Siu, and Hao Wu. FP8 formats for deep learning. *arXiv preprint arXiv:2209.05433*, 2022. doi:10.48550/arXiv.2209.05433.

Stephen Mussmann and Stefano Ermon. Learning and inference via maximum inner product search. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2587–2596. PMLR, 2016. URL https://proceedings.mlr.press/v48/mussmann16.html.

Hiroyuki Ootomo, Akira Naruse, Corey Nolet, Ray Wang, Tamas Feher, and Yong Wang. CAGRA: Highly parallel graph construction and approximate nearest neighbor search for GPUs. In *Proceedings of the 40th IEEE International Conference on Data Engineering (ICDE)*, pp. 4236–4247, Utrecht, The Netherlands, 2024. IEEE. doi:10.1109/ICDE60146.2024.00323.

Bita Darvish Rouhani, Nitin Garegrat, Tom Savell, Ankit More, Kyung-Nam Han, Ritchie Zhao, Mathew Hall, Jasmine Klar, Eric Chung, Yuan Yu, Michael Schulte, Ralph Wittig, Ian Bratt, Nigel Stephens, Jelena Milanovic, John Brothers, Pradeep Dubey, Marius Cornea, Alexander Heinecke, Andres Rodriguez, Martin Langhammer, Summer Deng, Maxim Naumov, Paulius Micikevicius, Michael Siu, and Colin Verrilli. OCP microscaling formats (MX) specification. Specification Version 1.0, Open Compute Project, September 2023. URL https://www.opencompute.org/documents/ocp-microscaling-formats-mx-v1-0-spec-final-pdf.

Tomas Ruiz, Zhen Qin, Yifan Zhang, Xuyang Shen, Yiran Zhong, and Mengdi Wang. FlashSampling: Fast and memory-efficient exact sampling. *arXiv preprint arXiv:2603.15854*, 2026. doi:10.48550/arXiv.2603.15854.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, April 2020. doi:10.1609/aaai.v34i05.6399.

Yashas Samaga B L, Varun Yerram, Chong You, Srinadh Bhojanapalli, Sanjiv Kumar, Prateek Jain, and Praneeth Netrapalli. HiRE: High recall approximate top-k estimation for efficient LLM inference. *arXiv preprint arXiv:2402.09360*, 2024. doi:10.48550/arXiv.2402.09360.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17456–17472. Curran Associates, Inc., 2022. doi:10.52202/068431-1269.

Jifan Shi, Jianyang Gao, James Xia, Tamás Béla Fehér, and Cheng Long. GPU-native approximate nearest neighbor search with IVF-RaBitQ: Fast index build and search. *Proceedings of the VLDB Endowment*, 19(11):3454–3467, July 2026. doi:10.14778/3836663.3836701.

Kyuhong Shim, Minjae Lee, Iksoo Choi, Yoonho Boo, and Wonyong Sung. SVD-Softmax: Fast softmax approximation on large vocabulary neural networks. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/4e2a6330465c8ffcaa696a5a16639176-Abstract.html.

Anshumali Shrivastava and Ping Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/hash/c98e7c4b8f20d384e3ad857d0ee226cc-Abstract.html.

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

Masatoshi Suzuki, Jun Suzuki, Koji Matsuda, Kyosuke Nishida, and Naoya Inoue. JAQKET: Construction of a Japanese QA dataset on the subject of quizzes. In *Proceedings of the 26th Annual Meeting of the Association for Natural Language Processing (NLP2020)*, pp. 237–240. The Association for Natural Language Processing, March 2020. URL https://www.anlp.jp/proceedings/annual_meeting/2020/pdf_dir/P2-24.pdf. In Japanese.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer, New York, NY, 2009. ISBN 978-0-387-79051-0. doi:10.1007/b13794.

Erik Wijmans, Brody Huval, Alexander Hertzberg, Vladlen Koltun, and Philipp Krähenbühl. Cut your losses in large-vocabulary language models. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, pp. 68174–68193, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/aa963ac256590bb7ad5fc26c68229a3a-Abstract-Conference.html.

Ziqi Yin, Hao Wang, Kaito Horio, Daisuke Kawahara, and Satoshi Sekine. Should we respect LLMs? A cross-lingual study on the influence of prompt politeness on LLM performance. In *Proceedings of the Second Workshop on Social Influence in Conversations (SICon 2024)*, pp. 9–35, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.sicon-1.2. Introduces the JMMLU benchmark, released as the Hugging Face dataset `nlp-waseda/JMMLU`.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi:10.18653/v1/P19-1472.

Jinbin Zhang, Nasib Ullah, Erik Schultheis, and Rohit Babbar. DynaSpec: Context-aware dynamic speculative sampling for large-vocabulary language models. *arXiv preprint arXiv:2510.13847*, 2025. doi:10.48550/arXiv.2510.13847.

Minjia Zhang, Xiaodong Liu, Wenhan Wang, Jianfeng Gao, and Yuxiong He. Navigating with graph representations for fast and scalable decoding of neural language models. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/366f0bc7bd1d4bf414073cabbadfdfcd-Abstract.html.

Weilin Zhao, Tengyu Pan, Xu Han, Yudi Zhang, Ao Sun, Yuxiang Huang, Kaihuo Zhang, Weilun Zhao, Yuxuan Li, Jie Zhou, Hao Zhou, Jianyong Wang, Zhiyuan Liu, and Maosong Sun. FR-Spec: Accelerating large-vocabulary language models via frequency-ranked speculative sampling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3909–3921, Vienna, Austria, July 2025. Association for Computational Linguistics. doi:10.18653/v1/2025.acl-long.198.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583. Curran Associates, Inc., 2024. doi:10.52202/079017-2000.

A Experimental setup

We evaluate four models, Olmo-3-7B ($|\mathcal{V}| = 100{,}278$), Nemotron-Nano-8B ($|\mathcal{V}| = 128{,}256$), Qwen3-8B-FP8 ($|\mathcal{V}| = 151{,}936$) and llm-jp-4-8b ($|\mathcal{V}| = 196{,}608$), all with $d = 4096$. Multiple-choice accuracy is read at the answer position, the step at which the model writes the letter of its chosen option, on up to 1,000 questions per benchmark, with differences in absolute percentage points. MMLU and JMMLU run on every model, and the wider suite adds ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020) and TruthfulQA (Lin et al., 2022) in English, CMMLU (Li et al., 2024a) and C-Eval (Huang et al., 2023) in Chinese, and JCommonsenseQA (Kurihara et al., 2022) and JAQKET (Suzuki et al., 2020) in Japanese, for a wider suite of 21 cells, 18 matched to the model's languages and 3 controls in a language it does not target, Nemotron's JMMLU and llm-jp's MMLU and ARC-Challenge, with 817 questions for TruthfulQA MC1. Together with Olmo's and Qwen3's JMMLU cells, these are the 23 cells of Figure 4(c), which Table 1 lists. Greedy replay feeds each model its own dense greedy trajectory on MATH-500 and AIME-2024 and records whether a head picks the dense token at every step.

Across all of them, every quality and latency measurement replays cached dense hidden states through the head, so a deviation comes from the head alone. The regimes of Table 2 stay fixed for every model and benchmark. Measurements run on one RTX 4090 unless stated otherwise, while the configuration sweep, the query shift and free generation ran on a GH200, and head latency is timed at $B = 32$ after warm-up. The baselines are stock CAGRA search from NVIDIA's cuVS library (Ootomo et al., 2024) and a streaming top-k that scores every row with a low-bit RaBitQ index and keeps a running top-k.

The prompts follow each model's own format. Nemotron runs through its chat template with the system instruction `detailed thinking off`, with answer letters tokenized for that boundary, and its dense accuracies are 0.531 on MMLU, 0.713 on ARC-Challenge and 0.304 on JMMLU. TruthfulQA lists the correct answer first on every row, so we shuffle the choices deterministically and permute the key with them, which gives dense accuracies of 0.606 for Nemotron and 0.536 for Olmo. Nemotron's greedy replay uses the same template, while its retrieved-context study uses plain completion prompts, and a 16-prompt check through the template raises its mean JSD by at most 0.008 without changing the order of the regimes.

Both arms are read from one search built with the vLLM integration's builder and truncated to the 64 candidates the deployed arm returns, so they are directly comparable. The multiple-choice search runs at batch 256, and because the graph walk is not batch-invariant, AH_m's eight MMLU and JMMLU cells were run again at batch 8, the served width, which left accuracy, agreement and trigger rate unchanged and moved mean JSD and recall only in the fifth decimal (Table 3).

B Regime search

The three regimes are configurations $(G, E, F, T, k', \text{index}, \text{rerank})$ that NSGA-II (Deb et al., 2002) chose against three objectives, the mean JSD after exact rescoring, the head latency and the resident memory, with recall recorded only as a diagnostic, on the prompt-final hidden states of 32 MATH-500 prompts on Olmo-3-7B at $B = 32$ (Table 2). The regimes keep their parameters on the GH200 front and on the front searched again on an RTX 4090 after the kernel fix of Appendix F, and their mean JSD moves from 0.044, 0.036 and 0.016 to 0.049, 0.034 and 0.013. Retuning with 60-trial NSGA-II studies per GPU and model, warm-started from the regimes, leaves every regime on the hardware-local front in all 11 studies with usable data, since no retuned trial dominates a regime and the search adds at most 0.02% hypervolume.

C Arm comparison

Tables 4 and 5 give MMLU, JMMLU and greedy replay for every model and regime under both arms and the fallback, behind Figures 4(c) and 4(e).

Table 6 pools the per-step certificates of Section 4 over the 21 wider-suite cells. The per-question records keep the dense margin $\Delta(\mathbf{p})$ but not $p_{i^\star}$, so the rescored arm is read at $\phi(1 - \Delta(\mathbf{p}))$, since $p_{i^\star} \geq \Delta(\mathbf{p})$ and ϕ decreases, and the deployed arm at the least inner minimum of Equation (2) over

every distribution with a gap of at least $\Delta(\mathbf{p})$, tabulated on 2,001 margins and read at the grid point at or below each step's margin. That worst case exceeds $\Delta^2/8$ by 0.3% at $\Delta = 0.1$ and by 8% at $\Delta = 0.5$, which is why the deployed arm gains little from it. In all 63 regime cells the certified share stays at or below the measured agreement. The margin does not predict the deployed loss either, since the Spearman correlation between the share of answer positions that fail the margin condition and the deployed accuracy loss is 0.068 over the 63 cells and -0.012 over the 54 whose dense accuracy exceeds chance by more than 0.10.

D SERVING

These tables give the values behind Section 4: every measured serving cell (Table 7), the online fallback and its warm passes (Tables 8 and 9) and the 128-token continuations (Table 10). Head shares compare only within the datacenter runs, which fix the KV cache at 8 GiB, since the RTX 4090 and GH200 runs fill a profiled budget, and the GH200 row is a single pass from before the kernel fix with vLLM's full-vocabulary sampler. In free-running serving, one of eight continuations per cell matches dense token for token and the rest diverge within 20 tokens, a per-step flip rate near 4%, inside the per-step disagreement of AH_m's deployed arm in greedy replay.

E ROBUSTNESS CHECKS

Stochastic sampling preserves accuracy as well, since sampling at temperature 0.7 with 16 samples per question keeps 22 of 24 model–dataset–regime cells within a paired-bootstrap interval containing zero, and the two exceptions are gains (Table 11).

Beyond accuracy, vocabulary shortlists can drop one category of token while aggregate metrics stay flat (Domhan et al., 2022), so we audit disagreement per token class on Olmo-3-7B's rescored replay traces, $9{,}558$ steps per regime. A candidate miss always forces a disagreement, so these rates bound the miss rate from above. Numeric tokens never disagree under AH_m, whose worst class stays at 0.40%, the disagreements concentrate in capitalized entities and code identifiers, reaching 11.8% under AH_ℓ, and whitespace and punctuation are nearly exact (Table 12).

Fine-tuning the backbone while leaving the head untouched moves only the queries, so query shift is the one channel that can degrade AH without rebuilding its index. Each MATH-500 and AIME-2024 prompt receives lexically retrieved in-domain documents within a 4,096-token budget, and dense greedy trajectories are replayed through the unchanged regimes in the rescored arm, 24 cells and 47,982 step evaluations, where the worst-case JSD rises by about 0.01 (Table 13). AH_ℓ's eight cells were measured again after the kernel fix on an RTX 4090 (Table 14), and a same-machine control on the 716 steps of the Olmo MATH-500 cell shows that the race had cost accuracy, since the fix moves recall@64 from 0.699 to 0.832 and agreement from 0.9455 to 0.9511, with the order of the regimes unchanged.

Finally, beyond teacher forcing, we decoded 80 MATH-500 and AIME-2024 prompts to 512 tokens each under AH_m with the dense head removed, at 80-way concurrency on the GH200. Throughput held at $2{,}487$ tokens per second, and the gold answer appears among the boxed answers in all 16 continuations that reached one, where 14 end on it and 2 box a different value after it.

F KERNEL DETERMINISM

Before the fix, AH_ℓ's int8 DP4A scoring kernel reused one word of shared memory first for the query's maximum absolute value and then as an integer accumulator, with no barrier between the two, so some warps quantized their query slice to zero and each launch walked the graph from a different frontier. Adding one `__syncthreads()` barrier makes every stage bit-identical across seven repetitions at unchanged latency and throughput, 15.19 ms per output token and 524 tokens per second before and after, while AH_m and AH_b never take that path. Toggling each of AH_ℓ's three differences from AH_m on 64 identical queries isolates the kernel, since only turning off DP4A scoring restores repeatability, and the rerank, the one stage that sums floating-point products, returns byte-identical scores in every repetition (Table 15). Every AH_ℓ accuracy result was measured after

Table 1. Accuracy on every benchmark cell of Figure 4(c), dense accuracy and the change of the rescored (res.), deployed (dep.) and fallback (fb.) arms against it in points, where an asterisk marks a control in a language the model does not target.

			AH_ℓ			AH_b			AH_m		
Model	**Benchmark**	**dense**	**res.**	**dep.**	**fb.**	**res.**	**dep.**	**fb.**	**res.**	**dep.**	**fb.**
Olmo-3-7B	ARC-Challenge	0.787	+0.0	−0.8	+0.0	+0.0	−0.4	+0.0	+0.0	−1.3	+0.0
	HellaSwag	0.406	+0.0	−1.1	+0.0	+0.0	−0.8	+0.0	+0.0	−0.9	+0.0
	JMMLU	0.345	+0.0	−0.6	+0.0	+0.0	−0.5	+0.0	+0.0	+0.3	+0.0
	MMLU	0.571	+0.0	−1.7	+0.0	+0.0	−1.6	+0.0	+0.0	−1.8	+0.0
	TruthfulQA	0.536	+0.0	−2.6	+0.0	+0.0	−2.2	+0.0	+0.0	−2.1	+0.0
	WinoGrande	0.584	+0.0	+1.2	+0.0	+0.0	+0.3	+0.0	+0.0	+1.0	+0.0
Nemotron-Nano-8B	ARC-Challenge	0.713	+0.0	−0.5	−0.7	+0.0	−0.3	−0.6	+0.0	+0.2	+0.2
	HellaSwag	0.422	+0.0	+0.1	+0.2	+0.0	+0.3	+0.2	+0.0	−1.5	−1.1
	JMMLU*	0.304	−4.1	−4.2	−4.1	−4.9	−5.2	−5.0	−5.9	−6.1	−5.9
	MMLU	0.531	−0.1	+0.4	−0.2	−0.1	+0.5	−0.2	−0.1	+0.3	+0.0
	TruthfulQA	0.606	+0.0	+0.4	+0.1	+0.0	+0.7	+0.6	+0.0	+0.1	−0.1
	WinoGrande	0.528	+0.0	+0.6	+0.2	+0.0	+0.5	+0.3	+0.0	+0.1	+0.0
Qwen3-8B-FP8	ARC-Challenge	0.910	+0.0	−0.1	+0.0	+0.0	−0.2	+0.0	+0.0	+0.1	+0.1
	C-Eval	0.767	+0.0	+0.1	+0.0	+0.0	+0.1	+0.0	+0.0	−0.1	−0.1
	CMMLU	0.671	+0.0	+0.2	+0.0	+0.0	+0.0	+0.0	+0.0	−0.7	+0.0
	HellaSwag	0.640	+0.0	−0.4	+0.0	+0.0	−1.0	+0.0	+0.0	−2.7	−0.3
	JMMLU	0.627	+0.0	+0.5	+0.0	+0.0	+0.6	+0.0	+0.0	−0.5	+0.0
	MMLU	0.747	+0.0	+0.1	+0.0	+0.0	−0.1	+0.1	+0.0	+0.7	+0.5
llm-jp-4-8b	ARC-Challenge*	0.857	+0.1	−0.3	+0.1	+0.1	+0.1	+0.1	+0.1	−6.7	−1.2
	JAQKET	0.897	+0.0	−1.5	+0.0	+0.0	−1.1	+0.0	+0.0	−3.3	+0.0
	JCommonsenseQA	0.941	−0.1	−1.1	−0.1	−0.1	−0.8	−0.1	−0.1	−3.4	+0.0
	JMMLU	0.569	−0.1	−0.9	−0.1	−0.1	−1.0	−0.1	−0.1	−7.1	−0.4
	MMLU*	0.659	+0.0	−1.7	+0.0	+0.0	−1.4	+0.0	+0.0	−9.6	−1.0

Table 2. Three regimes on Olmo-3-7B at $B = 32$ on an RTX 4090, where JSD is the mean over the tuning set with the returned candidates rescored exactly. Resident memory counts the graph, the index, the rerank copy where present, and the centroid and rotation that RaBitQ shares across rows, relative to the dense fp16 head.

Regime	**JSD**	**latency (ms)**	**resident (MiB)**	**against dense**	**index**	**rerank**
AH_ℓ	0.049	0.88	860	109.8%	RaBitQ 8-bit, int8 DP4A	fp8 copy
AH_b	0.034	3.17	404	51.6%	fp8-e5m2 copy	none
AH_m	0.013	6.51	285	36.3%	RaBitQ 4-bit	none

the fix, and only the configuration sweep of Figure 4(b), the AH_ℓ row of Table 13 and the GH200 serving row predate it.

Table 3. AH_m's eight MMLU and JMMLU cells, 24 rows over both arms and the fallback, searched at batch 8 against batch 256, as the largest absolute difference over the 24 rows.

Quantity	**range over the 24 rows**	**largest change at batch 8**
answer-position accuracy	0.2440 to 0.7540	0
argmax agreement with dense	0.6000 to 1.0000	0
fallback trigger rate	0.0000 to 0.9570	0
mean JSD	0.0005 to 0.1179	3.3×10^{-7}
recall@64	0.7347 to 0.9315	1.6×10^{-5}

Table 4. Multiple-choice accuracy on MMLU and JMMLU, 1,000 questions per cell, by arm and with the fallback, with the share of steps the fallback trigger fires on.

				by arm				
Model	**Benchmark**	**Regime**	**dense**	**rescored**	**deployed**	**+ fallback**	**fires on**	**run**
Olmo-3-7B	MMLU	AH_ℓ	0.571	0.571	0.554	0.571	63%	B
Olmo-3-7B	MMLU	AH_b	0.571	0.571	0.555	0.571	60%	B
Olmo-3-7B	MMLU	AH_m	0.571	0.571	0.553	0.571	47%	B
Olmo-3-7B	JMMLU	AH_ℓ	0.345	0.345	0.339	0.345	98%	A
Olmo-3-7B	JMMLU	AH_b	0.345	0.345	0.340	0.345	97%	A
Olmo-3-7B	JMMLU	AH_m	0.345	0.345	0.348	0.345	94%	A
Nemotron-Nano-8B	MMLU	AH_ℓ	0.531	0.530	0.535	0.529	36%	B
Nemotron-Nano-8B	MMLU	AH_b	0.531	0.530	0.536	0.529	44%	B
Nemotron-Nano-8B	MMLU	AH_m	0.531	0.530	0.534	0.531	47%	B
Nemotron-Nano-8B	JMMLU	AH_ℓ	0.304	0.263	0.262	0.263	82%	R
Nemotron-Nano-8B	JMMLU	AH_b	0.304	0.255	0.252	0.254	86%	R
Nemotron-Nano-8B	JMMLU	AH_m	0.304	0.245	0.243	0.245	79%	R
Qwen3-8B-FP8	MMLU	AH_ℓ	0.747	0.747	0.748	0.747	26%	B
Qwen3-8B-FP8	MMLU	AH_b	0.747	0.747	0.746	0.748	27%	B
Qwen3-8B-FP8	MMLU	AH_m	0.747	0.747	0.754	0.752	19%	B
Qwen3-8B-FP8	JMMLU	AH_ℓ	0.627	0.627	0.632	0.627	48%	A
Qwen3-8B-FP8	JMMLU	AH_b	0.627	0.627	0.633	0.627	44%	A
Qwen3-8B-FP8	JMMLU	AH_m	0.627	0.627	0.622	0.627	39%	A
llm-jp-4-8b	MMLU	AH_ℓ	0.659	0.659	0.642	0.659	72%	B
llm-jp-4-8b	MMLU	AH_b	0.659	0.659	0.645	0.659	70%	B
llm-jp-4-8b	MMLU	AH_m	0.659	0.659	0.563	0.649	49%	B
llm-jp-4-8b	JMMLU	AH_ℓ	0.569	0.568	0.560	0.568	85%	B
llm-jp-4-8b	JMMLU	AH_b	0.569	0.568	0.559	0.568	83%	B
llm-jp-4-8b	JMMLU	AH_m	0.569	0.568	0.498	0.565	65%	B

Runs: rows come from three runs of the same protocol, A, R and B in order of date.

Table 5. Greedy-replay argmax agreement with the dense head, by arm, over MATH-500 and AIME-2024, where dense against itself is 1.0 by construction.

				by arm			
Model	**Benchmark**	**Regime**	**dense**	**rescored**	**deployed**	**+ fallback**	**fires on**
Olmo-3-7B	MATH-500	AH_ℓ	1.0000	0.9626	0.9543	0.9626	23%
Olmo-3-7B	MATH-500	AH_b	1.0000	0.9885	0.9767	0.9885	20%
Olmo-3-7B	MATH-500	AH_m	1.0000	0.9985	0.9734	0.9985	16%
Olmo-3-7B	AIME-2024	AH_ℓ	1.0000	0.9591	0.9505	0.9591	30%
Olmo-3-7B	AIME-2024	AH_b	1.0000	0.9893	0.9753	0.9893	28%
Olmo-3-7B	AIME-2024	AH_m	1.0000	0.9984	0.9370	0.9982	22%
Nemotron-Nano-8B	MATH-500	AH_ℓ	1.0000	0.9020	0.8911	0.9020	24%
Nemotron-Nano-8B	MATH-500	AH_b	1.0000	0.9611	0.9455	0.9611	21%
Nemotron-Nano-8B	MATH-500	AH_m	1.0000	0.9986	0.9705	0.9984	16%
Nemotron-Nano-8B	AIME-2024	AH_ℓ	1.0000	0.9036	0.8919	0.9036	25%
Nemotron-Nano-8B	AIME-2024	AH_b	1.0000	0.9682	0.9516	0.9682	22%
Nemotron-Nano-8B	AIME-2024	AH_m	1.0000	0.9987	0.9737	0.9987	16%
Qwen3-8B-FP8	MATH-500	AH_ℓ	1.0000	0.9055	0.8933	0.9055	20%
Qwen3-8B-FP8	MATH-500	AH_b	1.0000	0.9703	0.9559	0.9703	18%
Qwen3-8B-FP8	MATH-500	AH_m	1.0000	0.9995	0.9650	0.9980	12%
Qwen3-8B-FP8	AIME-2024	AH_ℓ	1.0000	0.9049	0.8911	0.9049	19%
Qwen3-8B-FP8	AIME-2024	AH_b	1.0000	0.9703	0.9510	0.9703	16%
Qwen3-8B-FP8	AIME-2024	AH_m	1.0000	0.9987	0.9620	0.9945	11%
llm-jp-4-8b	MATH-500	AH_ℓ	1.0000	0.9542	0.9361	0.9542	22%
llm-jp-4-8b	MATH-500	AH_b	1.0000	0.9814	0.9544	0.9814	20%
llm-jp-4-8b	MATH-500	AH_m	1.0000	0.9952	0.9456	0.9884	15%
llm-jp-4-8b	AIME-2024	AH_ℓ	1.0000	0.9326	0.9005	0.9326	34%
llm-jp-4-8b	AIME-2024	AH_b	1.0000	0.9750	0.9286	0.9750	31%
llm-jp-4-8b	AIME-2024	AH_m	1.0000	0.9964	0.9122	0.9865	23%

Table 6. Per-step certificates over the 21 wider-suite cells, per regime and arm, with the JSD pooled over steps. The exact level is Proposition 1 on the rescored arm and Equation (2) on the deployed arm, both read at their worst case over the recorded margin, and agreement is the measured share of steps whose argmax matches dense.

			JSD		certified by		
Regime	**Arm**	**steps**	**mean**	**median**	$\Delta^2/8$	**exact**	**agreement**
AH_ℓ	rescored	19,905	0.0676	0.0052	61.9%	78.3%	86.0%
	deployed	19,905	0.0710	0.0106	56.0%	56.5%	80.4%
AH_b	rescored	19,905	0.0411	0.0028	69.2%	85.1%	91.6%
	deployed	19,905	0.0453	0.0068	61.8%	62.3%	85.6%
AH_m	rescored	19,905	0.0073	0.0006	81.8%	94.9%	99.6%
	deployed	19,905	0.0190	0.0065	65.8%	66.3%	88.6%

Table 7. Measured serving cells with 8 concurrent requests, greedy decoding and 128 output tokens per request, with percentages against the dense cell of the same GPU and model.

GPU	Model	Head	TTFT (ms)	TPOT (ms)	throughput (tok/s)	head share
L40S	Olmo-3-7B	dense	52.1	21.19	323.5	3.08%
		AH_ℓ	34.2 (-34.5%)	20.96 (-1.1%)	330.1 (+2.0%)	2.93%
		AH_m	39.6 (-24.1%)	28.03 (+32.3%)	274.3 (-15.2%)	1.00% (3.1×)
L40S	Nemotron-Nano-8B	dense	41.1	22.44	354.0	3.64%
		AH_ℓ	34.2 (-16.8%)	21.87 (-2.6%)	364.1 (+2.9%)	3.51%
		AH_m	40.0 (-2.6%)	29.19 (+30.0%)	273.2 (-22.8%)	1.20% (3.0×)
L40S	Qwen3-8B-FP8	dense	36.8	16.46	481.1	5.53%
		AH_ℓ	33.0 (-10.5%)	15.66 (-4.9%)	506.2 (+5.2%)	5.16%
		AH_m	39.4 (+7.1%)	23.30 (+41.5%)	341.4 (-29.0%)	1.77% (3.1×)
L40S	llm-jp-4-8b	dense	61.4	23.17	340.8	5.30%
		AH_ℓ	43.6 (-29.1%)	21.90 (-5.5%)	362.4 (+6.3%)	5.10%
		AH_m	50.9 (-17.1%)	29.74 (+28.4%)	267.5 (-21.5%)	1.69% (3.1×)
A100	Olmo-3-7B	dense	56.4	15.88	433.3	3.07%
		AH_ℓ	40.7 (-28.0%)	16.16 (+1.8%)	419.0 (-3.3%)	2.93%
		AH_m	46.5 (-17.5%)	27.12 (+70.8%)	282.9 (-34.7%)	1.01% (3.1×)
A100	Nemotron-Nano-8B	dense	51.8	14.75	512.2	3.64%
		AH_ℓ	37.7 (-27.3%)	15.08 (+2.2%)	524.0 (+2.3%)	3.51%
		AH_m	48.9 (-5.6%)	28.62 (+94.0%)	277.8 (-45.8%)	1.20% (3.0×)
A100	Qwen3-8B-FP8	dense	44.5	18.91	418.3	5.52%
		AH_ℓ	45.2 (+1.6%)	19.48 (+3.0%)	406.2 (-2.9%)	5.15%
		AH_m	56.5 (+26.9%)	25.03 (+32.3%)	316.4 (-24.4%)	1.77% (3.1×)
A100	llm-jp-4-8b	dense	48.2	14.84	529.4	5.30%
		AH_ℓ	38.6 (-19.9%)	15.38 (+3.7%)	513.4 (-3.0%)	5.10%
		AH_m	51.8 (+7.3%)	29.76 (+100.6%)	267.1 (-49.5%)	1.69% (3.1×)
H100	Olmo-3-7B	dense	31.8	10.64	642.0	3.06%
		AH_ℓ	35.1 (+10.3%)	10.15 (-4.6%)	675.9 (+5.3%)	2.91%
		AH_m	30.8 (-3.4%)	16.45 (+54.7%)	465.9 (-27.4%)	1.00% (3.1×)
H100	Nemotron-Nano-8B	dense	28.6	8.66	905.9	3.61%
		AH_ℓ	28.5 (-0.4%)	8.96 (+3.4%)	877.0 (-3.2%)	3.48%
		AH_m	27.6 (-3.4%)	16.76 (+93.4%)	474.7 (-47.6%)	1.19% (3.0×)
H100	Qwen3-8B-FP8	dense	38.8	14.18	556.3	5.48%
		AH_ℓ	47.1 (+21.5%)	14.77 (+4.2%)	532.1 (-4.4%)	5.11%
		AH_m	53.9 (+39.1%)	17.31 (+22.1%)	454.3 (-18.3%)	1.74% (3.1×)
H100	llm-jp-4-8b	dense	29.3	8.47	925.2	5.26%
		AH_ℓ	28.6 (-2.5%)	8.72 (+3.0%)	899.9 (-2.7%)	5.05%
		AH_m	29.5 (+0.5%)	17.92 (+111.6%)	443.8 (-52.0%)	1.67% (3.2×)
RTX 4090	Qwen3-8B-FP8	dense	36.3	15.57	508.3	5.20%
		AH_ℓ	30.8 (-15.0%)	15.03 (-3.5%)	527.7 (+3.8%)	5.68%
		AH_m	36.1 (-0.4%)	21.91 (+40.7%)	363.2 (-28.5%)	1.89% (2.7×)
GH200	Qwen3-8B-FP8	dense	75.1	28.27	279.2	1.42%
		AH_ℓ	103.2 (+37.4%)	30.71 (+8.6%)	255.7 (-8.4%)	1.52%
		AH_m	84.1 (+12.0%)	28.93 (+2.3%)	272.4 (-2.5%)	0.48% (3.0×)

Table 8. Online dense fallback on the RTX 4090, Qwen3-8B-FP8, over 9,217 sampled steps. The trigger fires when the head's own margin fails the Pinsker level $\Delta^2/8$ at the regime's mean JSD measured on the GH200, 0.044 for AH_ℓ and 0.016 for AH_m. A token change is a step whose top token changes under exact rescoring of the same shortlist, as in the rescored arm, and the margin column is the threshold on the head's own top-two margin below which the trigger fires.

			share of steps			cost against no fallback	
Regime	**trigger**	**margin threshold**	**fires on**	**changes token**	**of fired**	**TPOT**	**throughput**
AH_ℓ	at $\overline{\mathrm{JSD}}$	0.5922	41.3%	4.00%	9.7%	+9.5%	-9.2%
AH_ℓ	always	1.0000	96.6%	4.00%	4.1%	+11.9%	-11.2%
AH_m	at $\overline{\mathrm{JSD}}$	0.3607	33.9%	9.61%	28.4%	+5.5%	-5.9%
AH_m	always	1.0000	93.9%	9.99%	10.6%	+7.7%	-7.8%

Table 9. RTX 4090 cell measured again with the shortlist sampler on Qwen3-8B-FP8, with warm medians over seven passes and their interquartile range in brackets. The single pass is the original measurement from before the kernel fix.

	TTFT (ms)		TPOT (ms)		throughput (tok/s)		tensor avoided
Head	**warm median**	**single pass**	**warm median**	**single pass**	**warm median**	**single pass**	**GB**
dense	22.3 [21.8, 24.6]	36.3	15.68 [15.64, 15.68]	15.57	508.5	508.3	0.0
AH_ℓ	23.3 [23.1, 23.7]	30.8	15.15 [15.14, 15.19]	15.03	525.7	527.7	5.6
AH_m	25.3 [25.0, 27.2]	36.1	22.12 [22.10, 22.13]	21.91	361.2	363.2	5.6

Table 10. Greedy 128-token continuations against the same session's dense run, 8 prompts, where a dash marks a prompt that never diverged.

Regime	**fallback**	**identical**	**first divergence, per prompt**
AH_m	off	1 of 8	0, 16, 9, 0, 12, —, 0, 9
AH_m	at $\overline{\mathrm{JSD}}$	2 of 8	4, 43, —, 8, 22, —, 48, 35
AH_m	always	3 of 8	4, 111, —, —, 22, —, 104, 35
AH_ℓ	off	1 of 8	4, 1, 6, 0, 1, —, 0, 0
AH_ℓ	at $\overline{\mathrm{JSD}}$	1 of 8	4, 1, 6, 0, 1, —, 5, 9

Table 11. Stochastic answer-position sampling at temperature 0.7, 16 samples per question, in percentage points against the dense head with 95% paired-bootstrap intervals, for the rescored arm on 200 questions per benchmark, fewer than the 1,000-question runs of Table 4.

Model	**Benchmark**	$\mathbf{AH}_\ell$	$\mathbf{AH}_b$	$\mathbf{AH}_m$
Olmo-3-7B	MMLU	$+0.01$ $[-1.56, +1.59]$	$+0.64$ $[-1.00, +2.31]$	$+1.01$ $[-0.66, +2.72]$
	JMMLU	$+0.44$ $[-1.72, +2.66]$	$+0.54$ $[-1.44, +2.50]$	$+0.19$ $[-1.94, +2.28]$
Nemotron-Nano-8B	MMLU	-0.13 $[-1.34, +1.13]$	$+0.87$ $[-0.47, +2.22]$	$+0.49$ $[-0.75, +1.75]$
	JMMLU	-0.32 $[-2.25, +1.66]$	-0.12 $[-2.03, +1.75]$	$+0.51$ $[-1.44, +2.44]$
Qwen3-8B-FP8	MMLU	$+0.22$ $[-0.81, +1.28]$	$+0.70$ $[-0.44, +1.78]$	$+0.38$ $[-0.94, +1.66]$
	JMMLU	-0.11 $[-1.37, +1.16]$	$+0.46$ $[-0.72, +1.62]$	$+2.00$ $[+0.63, +3.38]$
llm-jp-4-8b	MMLU	-0.37 $[-2.31, +1.56]$	-0.47 $[-2.37, +1.41]$	-0.60 $[-2.53, +1.41]$
	JMMLU	$+1.12$ $[-0.69, +2.94]$	$+2.19$ $[+0.25, +4.09]$	$+0.83$ $[-1.31, +2.97]$

Table 12. Greedy-replay disagreement with the dense head by token class on Olmo-3-7B, pooled over MATH-500 and AIME-2024, in percent of the steps in that class. The classes overlap by construction, so one step can count in several of them. AH_ℓ was measured again after the kernel fix, so it has its own step counts.

	after the kernel fix		original traces		
Token class	**steps**	**AH_ℓ**	**steps**	**AH_b**	**AH_m**
numeric	1,032	0.29%	1,015	0.10%	0.00%
symbol or punctuation	2,781	1.44%	2,768	0.47%	0.04%
whitespace	726	0.41%	743	0.27%	0.00%
non-ASCII	44	4.55%	42	2.38%	0.00%
code identifier	4,992	6.89%	5,052	1.98%	0.16%
capitalized entity	738	11.79%	757	5.81%	0.40%

Table 13. Retrieved-context query shift, worst run per regime over four models and two datasets. The without columns measure the same cell without the retrieved prefix, and all rows come from the GH200 study, so the AH_ℓ row predates the kernel fix.

		worst mean JSD		worst argmax agreement	
Regime	**steps**	**without**	**with**	**without**	**with**
AH_ℓ	15,994	0.070	0.080	0.888	0.877
AH_b	15,994	0.029	0.041	0.952	0.936
AH_m	15,994	0.005	0.018	0.992	0.972

Table 14. AH_ℓ's eight retrieved-context cells measured after the kernel fix on an RTX 4090. The within-level column is the share of steps whose dense margin meets the Pinsker level $\Delta(\mathbf{p})^2/8$ at the regime's own mean JSD.

Benchmark	**Model**	**steps**	**mean JSD**	**agreement**	**recall@64**	**within level**
MATH-500	Olmo-3-7B	716	0.0328	0.9511	0.8316	0.8366
MATH-500	Nemotron-Nano-8B	1,953	0.0720	0.8817	0.7402	0.7624
MATH-500	Qwen3-8B-FP8	2,048	0.0549	0.9141	0.7528	0.8872
MATH-500	llm-jp-4-8b	2,048	0.0189	0.9727	0.6820	0.8950
AIME-2024	Olmo-3-7B	361	0.0550	0.9030	0.8179	0.7258
AIME-2024	Nemotron-Nano-8B	1,811	0.0842	0.8564	0.7079	0.7090
AIME-2024	Qwen3-8B-FP8	2,048	0.0701	0.8926	0.7344	0.8564
AIME-2024	llm-jp-4-8b	2,048	0.0290	0.9600	0.7023	0.8818

Table 15. Seven repetitions of five head variants on 64 identical queries, before and after the barrier. The walks columns compare the returned row ids and the shortlists columns the final 64 the head returns.

	before the barrier			after	
Variant	**walks**	**shortlists**	**rows differing**	**walks**	**shortlists**
AH_ℓ	1 of 7	1 of 7	50.1%	7 of 7	7 of 7
AH_ℓ without DP4A	7 of 7	7 of 7	0.0%	7 of 7	7 of 7
AH_ℓ without the fp8 rerank copy	1 of 7	1 of 7	52.8%	7 of 7	7 of 7
AH_ℓ with a 256-wide search	1 of 7	1 of 7	53.4%	7 of 7	7 of 7
AH_m	7 of 7	7 of 7	0.0%	7 of 7	7 of 7