SWAP-ROPE: PERMUTATION-INVARIANT BLOCK ENCODING FOR LLM JUDGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Pairwise LLM judging repeatedly encodes the same candidates in different positions and contexts. We introduce SWAP-ROPE, which combines shared block coordinates with causal attention restricted to a fixed prefix and each block's own history. We establish that a block's hidden states, keys, and values are independent of its competitors and preserved under token-aligned permutations. These properties permit cache reuse across different comparisons. In an all-pairs tournament over n candidates, fresh candidate encodings decrease from $\binom{n}{2} + n - 1$ to n under fixed-content and cache-retention assumptions; comparison calls remain quadratic. We fine-tune Qwen3 models at 1.7B, 4B, and 8B and evaluate three mathematical benchmarks. Reported reusable key–value (KV) cache ratios increase from 38.5–40.9% under standard RoPE to 73.8–79.2%. Selection accuracy improves over vanilla-position fine-tuning in six of nine settings with four candidates and three of nine with eight candidates. The construction excludes cross-block attention during candidate encoding; verdict consistency and serving speedups require further evaluation.

1 INTRODUCTION

Pairwise LLM judges compare candidate solutions in test-time scaling, model assessment, and preference collection (Snell et al., 2024; Brown et al., 2024; Zheng et al., 2023). These workloads repeatedly present the same candidate beside different alternatives. A candidate's internal token order carries its reasoning, whereas its placement before or after another candidate is a presentation choice. This distinction motivates a computational question: can a judge encode each candidate once and reuse that representation when its position or competitor changes, while retaining a joint computation that compares the candidates?

The distinction matters because the number of comparisons can grow much faster than the number of distinct candidates. In an all-pairs tournament over n candidates, let $T = \binom{n}{2}$ denote the comparison count. The judge processes $2T = n(n-1)$ candidate appearances, each potentially containing a long reasoning trace. Prefix key–value (KV) caching reuses shared instructions and repeated initial candidates, but a candidate following a different competitor requires a fresh encoding. Avoiding this repeated prefill would let the system reuse candidate computation across the tournament while still performing each comparison. Figure 1 illustrates the opportunity: candidate A appears before B, after B, and after C under a fixed shared prefix. Prefix caching encodes A three times; SWAP-ROPE permits one encoding and two cache reuses. The readout that compares the candidates is computed separately for each prompt.

Exact reuse is difficult because cached states depend on both position and context. Rotary positional embeddings (RoPE) encode relative positions through query–key rotations (Su et al., 2024); moving a block can change its distances to the shared prefix and other tokens. Causal attention introduces a second dependency: a later candidate reads earlier candidates, so changing its competitor can change its hidden states throughout the network. Correcting positional coordinates alone therefore does not recover competitor-independent states. Conversely, isolating blocks without aligning their coordinates can leave position-dependent interactions with the prefix. A reusable block encoding must control both dependencies while preserving causal order within each candidate and a mechanism for comparing candidates afterward.

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.

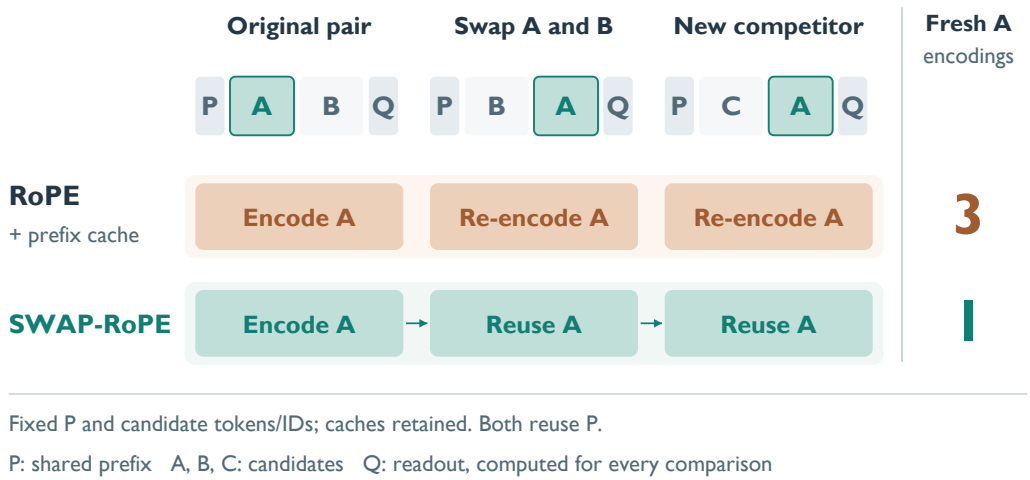


Figure 1: Reusing one candidate across comparisons. With a fixed prefix $\mathbf{P}$, standard RoPE and prefix caching encode candidate $\mathbf{A}$ separately in the three prompts. SWAP-ROPE encodes it once and reuses its states after a swap or competitor change. Candidate tokens and labels are fixed, caches are retained, and the readout $\mathbf{Q}$ is computed for each comparison. The counts describe fresh candidate encodings, not latency.

Existing caching methods address important parts of this problem. PagedAttention and RadixAttention provide efficient storage and shared-prefix reuse (Kwon et al., 2023; Zheng et al., 2024). Prompt Cache extends reuse beyond prefixes through independently encoded modules at schema-assigned positions; its shared-position unions select mutually exclusive alternatives (Gim et al., 2024). Our target instead places candidates simultaneously at a common positional origin, so equal within-block offsets have equal distances to the prefix regardless of presentation order. CacheBlend selectively recomputes cached tokens to recover interactions between text segments (Yao et al., 2025). We formalize a computation that preserves candidate-local states while assigning shared coordinates to co-present blocks. Reuse is exact relative to this computation; removing cross-candidate attention changes the model and requires evaluating the resulting quality tradeoff.

We introduce SWAP-ROPE, whose central insight is to separate candidate encoding from comparison. Every annotated candidate block receives the same positional origin and retains its internal token order. A causal mask restricts each block to the shared prefix and its own history; trailing readout tokens attend to all blocks and perform the comparison. Under the specified computation, we establish *prefix-conditioned block locality*: at every layer, a block's hidden states, rotated keys, and values depend only on its own tokens and the prefix. They are independent of other blocks' order, content, number, and lengths. Permuting unchanged blocks therefore preserves corresponding states after token alignment, and substituting a competitor does not invalidate the candidate's cache. Locality follows from the construction; supervised fine-tuning adapts the judge's task behavior to it.

This guarantee yields a precise reduction in repeated candidate encoding. With fixed candidate tokens and labels, a shared prefix, retained cache entries, and no additional full-block prefix sharing, standard prefix caching requires $T + n - 1$ fresh candidate encodings, whereas SWAP-ROPE requires n. The comparison count remains T, and readout computation and decoding remain necessary. The restriction also has a cost: candidate representations cannot adapt to one another during prefill. Moreover, representation symmetry and final verdict consistency are distinct properties. Observed position bias motivates the latter (Wang et al., 2023a), but a judging interface must preserve candidate identities and satisfy the construction's assumptions before its behavior can inherit that symmetry.

We evaluate Qwen3-1.7B, 4B, and 8B on AIME 2024, AIME 2025, and HMMT 2025. Reported reusable KV ratios rise from 38.5–40.9% under standard RoPE to 73.8–79.2%, gains of 35.3–38.3 percentage points across all nine model–benchmark settings. Selection accuracy relative to vanilla-position fine-tuning improves in six of nine settings with four candidates and three of nine with eight candidates; the largest gain is 13.7 points. These results support broad reuse potential within the

evaluated workloads and a setting-dependent quality tradeoff. Direct verdict-consistency tests and serving benchmarks remain necessary to assess behavioral and end-to-end performance benefits.

Our contributions are:

- **A block encoding with a representation guarantee.** We combine shared positional origins with prefix-conditioned causal masking and prove that every layer's candidate hidden states, rotated keys, and values are independent of competitors. The guarantee covers both token-aligned block permutations and competitor substitutions under explicit assumptions (Sections 3.2–3.3, Proposition 1).
- **An exact analysis of reuse across comparisons.** We derive the fresh-encoding counts $T + n - 1$ for prefix caching and n for SWAP-ROPE, separating candidate computation from the unchanged quadratic comparison count. For eight candidates, this reduces fresh candidate encodings from 35 to 8 under the stated cache assumptions (Section 3.4, Table 2).
- **An empirical characterization of reuse and judging quality.** Across three model sizes and three mathematical benchmarks, we report 35.3–38.3-point increases in estimated KV reuse and evaluate selection with four and eight candidates. The results identify where reuse accompanies accuracy gains and where quality decreases, making the computational benefit and task-quality tradeoff explicit (Sections 4.2–4.3).

2 RELATED WORK

KV cache reuse. PagedAttention and RadixAttention support efficient KV storage and cache sharing for common prefixes (Kwon et al., 2023; Zheng et al., 2024). Prompt Cache extends reuse to independently encoded modules, with a schema assigning their positions and specifying how cached modules are assembled (Gim et al., 2024). Its module-local computation already removes dependencies between separately encoded segments, and its shared-position unions choose one module at a time. It also supports jointly encoded scaffolds when interactions between modules are needed, trading additional cache storage for shared attention. CacheBlend takes a different approach: it selectively recomputes a subset of cached tokens to account for interactions with preceding text (Yao et al., 2025).

SWAP-ROPE builds on the principle of independent block encoding, while assigning the same coordinate range to simultaneously present swappable blocks. Under a fixed shared prefix, its explicit mask makes a candidate's states independent of both placement and competitor substitution. This trades cross-block interaction during prefill for cache compatibility; comparison occurs in the readout. The guarantee concerns the modified computation, rather than exact reconstruction of standard causal attention. Quality and serving performance therefore require evaluation alongside reuse.

Positional encoding and permutation symmetry. Absolute embeddings and relative attention biases encode token order in transformers (Vaswani et al., 2017; Raffel et al., 2020; Press et al., 2022). RoPE rotates queries and keys so that their inner products depend on relative position (Su et al., 2024). Separately, Deep Sets characterizes permutation-invariant set functions and equivariant representations (Zaheer et al., 2018); Set Transformer uses attention to model interactions among set elements (Lee et al., 2019). These results establish symmetry as an architectural property, but symmetry alone does not make each element's representation independent of the others.

Our construction applies symmetry to a collection of ordered token sequences. Causal attention and local coordinates preserve order within each candidate, while shared block coordinates remove its placement among competitors from the encoding. Corresponding states are invariant after token alignment, or equivariant when indexed by physical location. Restricting their context to the prefix and the candidate itself additionally permits reuse when a competitor changes, beyond permutations of a fixed block collection.

Judging and test-time scaling. LLM judges can approximate human preferences but exhibit position, verbosity, and self-preference biases (Zheng et al., 2023). Reordering responses can change rankings (Wang et al., 2023a); option-order sensitivity also appears in multiple-choice reasoning (Pezeshkpour & Hruschka, 2023). Balanced-position calibration aggregates judgments across presentation orders (Wang et al., 2023a). This changes the evaluation protocol without requiring each internal representation to be order-independent. Conversely, representation symmetry

must be combined with consistent candidate identities and a compatible readout to yield an invariant decision.

Test-time scaling allocates computation to repeated generation and selection (Snell et al., 2024; Brown et al., 2024). Self-consistency selects an answer by aggregating sampled reasoning paths, without pairwise judges (Wang et al., 2023b). We focus on all-pairs judging, where candidates recur across comparisons. The target is repeated candidate prefill; candidate generation, comparison calls, and answer aggregation remain separate costs.

3 METHOD

SWAP-ROPE combines shared positional coordinates with causal masking to isolate candidate blocks during prefill. We specify the computation, establish cache compatibility, and separate this guarantee from the judge's decision behavior.

3.1 SETTING AND NOTATION

A *swappable block* is a contiguous token sequence whose presentation order can change without changing the task, provided candidate identities are preserved. Consider

$$\mathbf{x}_\sigma = \mathbf{P} \parallel \mathbf{B}_{\sigma(1)} \parallel \cdots \parallel \mathbf{B}_{\sigma(K)} \parallel \mathbf{Q}, \tag{1}$$

where $\mathbf{P}$ contains shared instructions and the problem, $\mathbf{B}_1, \ldots, \mathbf{B}_K$ are nonempty blocks, $\mathbf{Q}$ is a nonempty readout prompt, $\parallel$ denotes concatenation, and σ permutes $\{1, \ldots, K\}$. Bold symbols denote token sequences, vectors, or matrices; calligraphic symbols denote sets. Vertical bars denote sequence length or set cardinality, according to the argument.

Blocks occupy one contiguous region. Their delimiters, separators, and persistent candidate labels move with them, including across comparisons with different competitors. We mark boundaries with `<Chunk>` and `</Chunk>`; arbitrary intervening context is excluded. Boundaries must be known, and swaps must preserve task semantics without relabeling candidates by display slot.

Table 1: Core notation; sequence identity is distinct from physical token location.

Symbol	Meaning
$n, \mathbf{C}_a$	Tournament pool size $n \geq 2$ and candidate token sequence a.
$K, \mathbf{B}_k$	Number of blocks in one prompt and block k's token sequence.
N, i, j	Prompt length and physical token indices in $\{1, \ldots, N\}$.
$\mathcal{I}_P, \mathcal{I}_k, \mathcal{I}_Q$	Index sets occupied by $\mathbf{P}, \mathbf{B}_k, \mathbf{Q}$.
$m, L_{\max}$	Prefix length and maximum block length.
$\mathrm{off}(i), \mathrm{pos}(i)$	Local zero-based offset and canonical position of token i.
$\mathcal{V}_i$	Visible key-token indices for query token i.

A pool $(\mathbf{C}_a)_{a=1}^n$ yields $T = \binom{n}{2}$ pairwise prompts, one per $a < b$, each with $K = 2$ marked candidate blocks. For a valid, non-tied response, the parsed decision is $J_{ab} = \mathrm{Judge}(\mathbf{P}; \mathbf{C}_a, \mathbf{C}_b) \in \{a, b\}$, with the readout template fixed. An aggregation rule returns $\mathbf{C}^* = \mathrm{Aggregate}((J_{ab})_{1 \leq a < b \leq n})$; the indexed family preserves pair identities and repeated wins.

3.2 CANONICAL COORDINATES AND ATTENTION MASK

Let $N = |\mathbf{P}| + \sum_{k=1}^K |\mathbf{B}_k| + |\mathbf{Q}|$, $m = |\mathbf{P}|$, and $L_{\max} = \max_k |\mathbf{B}_k|$. The index sets partition the prompt and depend on σ; region identities and local offsets do not. For a token in sequence $\mathbf{U}$, $0 \leq \mathrm{off}(i) < |\mathbf{U}|$. Define

$$\mathrm{pos}(i) = \begin{cases} \mathrm{off}(i), & i \in \mathcal{I}_P, \\ m + \mathrm{off}(i), & i \in \mathcal{I}_k, \\ m + L_{\max} + \mathrm{off}(i), & i \in \mathcal{I}_Q. \end{cases} \tag{2}$$

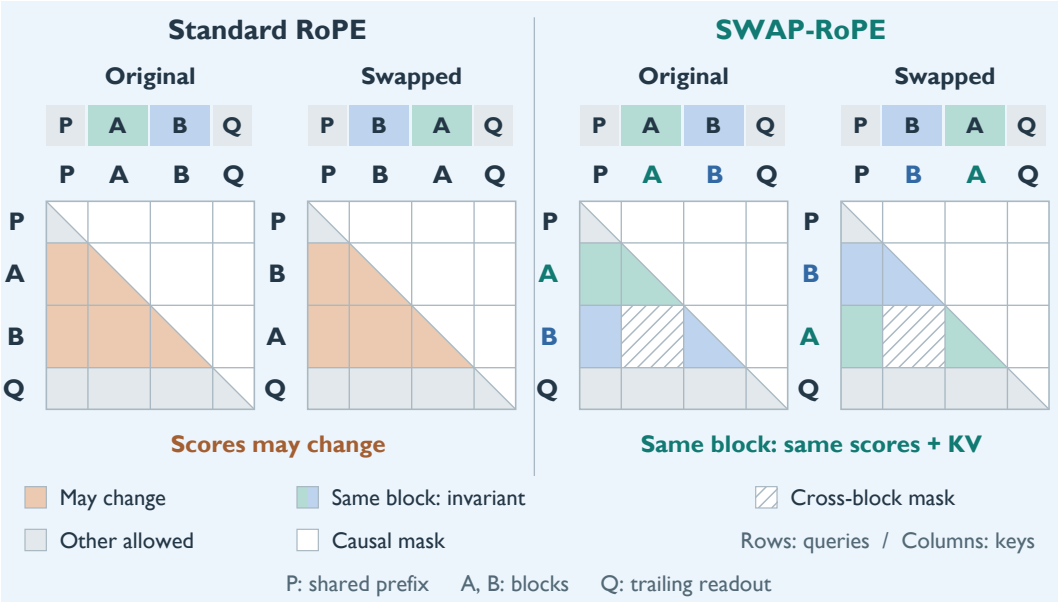


Figure 2: Attention before and after a block swap. Rows are queries and columns are keys; $\mathbf{P}$ is the prefix, $\mathbf{A}, \mathbf{B}$ are blocks, and $\mathbf{Q}$ is the readout. Under SWAP-ROPE, aligned tokens within the same block retain their prefill scores and KV states; hatching marks masked cross-block attention. Colors distinguish blocks rather than equal values across blocks. This is a structural schematic, not measured attention or a verdict guarantee.

Blocks share an origin while retaining internal order. For $K = 0$, set $L_{\max} = 0$ to recover consecutive positions. Shared coordinates alone do not eliminate context dependence, so use

$$\mathcal{V}_i = \begin{cases} \{j \in \mathcal{I}_P : \mathrm{off}(j) \le \mathrm{off}(i)\}, & i \in \mathcal{I}_P, \\ \mathcal{I}_P \cup \{j \in \mathcal{I}_k : \mathrm{off}(j) \le \mathrm{off}(i)\}, & i \in \mathcal{I}_k, \\ \mathcal{I}_P \cup \bigcup_{k=1}^{K} \mathcal{I}_k \cup \{j \in \mathcal{I}_Q : \mathrm{off}(j) \le \mathrm{off}(i)\}, & i \in \mathcal{I}_Q. \end{cases} \tag{3}$$

Each block reads the prefix and its own causal history, including itself. Readout tokens can compare all blocks; blocks cannot read competitors or the readout.

For one head at a fixed layer, queries, keys, and values $\mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$ have head dimension d. Let $\mathbf{R}_z \in \mathbb{R}^{d \times d}$ be the RoPE rotation at coordinate z with fixed frequencies. Set $\bar{\mathbf{q}}_i = \mathbf{R}_{\mathrm{pos}(i)}\mathbf{q}_i$ and $\bar{\mathbf{k}}_i = \mathbf{R}_{\mathrm{pos}(i)}\mathbf{k}_i$. Attention computes

$$s_{ij} = \frac{\bar{\mathbf{q}}_i^\top \bar{\mathbf{k}}_j}{\sqrt{d}} + \mathbf{M}_{ij}, \qquad \mathbf{o}_i = \sum_{j \in \mathcal{V}_i} \alpha_{ij}\mathbf{v}_j, \qquad \alpha_{ij} = \frac{\exp(s_{ij})}{\sum_{r \in \mathcal{V}_i} \exp(s_{ir})}, \tag{4}$$

where the additive mask matrix has $\mathbf{M}_{ij} = 0$ on visible entries and $-\infty$ otherwise; α_{ij} is defined on visible entries. The per-layer cache stores rotated keys $\bar{\mathbf{k}}_i$ and values $\mathbf{v}_i$. Values inherit positional effects through contextual hidden states rather than direct rotation.

The final readout state predicts the first generated token. Emitted token t, numbered from zero, is processed at $m + L_{\max} + |\mathbf{Q}| + t$, where $|\mathbf{Q}|$ is the original suffix length. It sees all prompt tokens and generated history, including itself. Figures 2 and 3 illustrate the mask and coordinates.

3.3 BLOCK LOCALITY AND SWAP INVARIANCE

Assume fixed weights and RoPE frequencies, tokenwise normalization, residuals, and feed-forward operations, and deterministic inference without dropout. No additional physical-index or prompt-length dependence is allowed, including unaccounted-for sliding windows or dynamic position scaling. We analyze exact arithmetic; implementation-level numerical equality needs separate checking.

Example. Let $|\mathbf{P}| = 3$, $|\mathbf{B}_A| = 4$, $|\mathbf{B}_B| = 2$, and $|\mathbf{Q}| = 2$, including delimiters and persistent identifiers.

Region	$\mathbf{P}$	$\mathbf{B}_A$	$\mathbf{B}_B$	$\mathbf{Q}$
Canonical positions	$0, 1, 2$	$3, 4, 5, 6$	$3, 4$	$7, 8$

Swapping $\mathbf{B}_A$ and $\mathbf{B}_B$ preserves these coordinates. Each block reads only $\mathbf{P}$ and its own causal history; $\mathbf{Q}$ reads both blocks.

Figure 3: Canonical coordinates depend on region identity and local token offset. The prefix and block tokens remain fixed across comparisons.

Table 2: Tournament accounting with a fixed prefix, retained caches, stable block tokens, and no additional full-block prefix sharing. Suffix processing and decoding are excluded.

Quantity	Prefix caching	SWAP-ROPE block caching
Comparison prompts	T	T
Candidate appearances	$2T$	$2T$
Fresh candidate encodings	$T + n - 1$	n
Cached candidate appearances	$T - n + 1$	$2T - n$

Proposition 1 (Prefix-conditioned block locality)**.** *For fixed $\mathbf{P}$ and $\mathbf{B}_k$, every block hidden state, rotated key, and value under Eqs.* (2)–(4) *depends only on these two sequences. The states are independent of other blocks' order, number, content, and lengths. Permuting unchanged blocks therefore preserves corresponding prefill states and logits to corresponding visible tokens at every layer and head.*

Proof. Prefix states depend only on prefix history. Corresponding block tokens initially have identical embeddings and coordinates. Assuming prefix and block states agree at a layer's input, Eq. (3) exposes identical prefix and within-block histories. Their coordinates, projections, and masks agree, so Eq. (4) yields identical logits, normalization, and outputs. Tokenwise operations preserve equality. Induction establishes identical hidden states and cached keys and values, regardless of competitor substitution. □

For permutations σ, τ, let $\mathrm{align}_{\sigma\to\tau}(i)$ identify the token with the same region identity and offset in $\mathbf{x}_\tau$. Write $\mathbf{h}_i^{(\ell)}[\mathbf{x}]$ for its hidden state after layer ℓ, with brackets indicating the input sequence. Then

$$\mathbf{h}^{(\ell)}_{\mathrm{align}_{\sigma\to\tau}(i)}[\mathbf{x}_\tau] = \mathbf{h}_i^{(\ell)}[\mathbf{x}_\sigma], \qquad i \in \mathcal{I}_k. \tag{5}$$

Representations are equivariant in physical indices and invariant after identity alignment, for arbitrary K, without learning the symmetry through fine-tuning.

3.4 CACHE REUSE ACROSS COMPARISONS

A cache key must identify prefix and block tokens, including labels and delimiters, model weights/adapters, tokenizer behavior, RoPE configuration, and mask. Matching entries supply block KV states; misses require prefill against the shared prefix. Recompute the suffix for each comparison: both its states and its $L_{\max}$-dependent coordinates can change with competitors.

For the T pairs $a < b$, retain caches and stable block tokens throughout. Prefix caching computes $n-1$ distinct first-candidate prefixes and reuses them on $T-(n-1)$ appearances. The T second-candidate appearances retain competitor-dependent histories. Table 2 gives the resulting counts.

Prefix caching already achieves $\Theta(n^2)$ hits. SWAP-ROPE reduces *fresh* candidate encodings from $\Theta(n^2)$ to $\Theta(n)$ while comparison counts remain quadratic. These block counts are not measured latency: lengths, transfers, batching, readout computation, and decoding determine realized savings.

3.5 JUDGING BEHAVIOR AND SCOPE

With unchanged prefix, suffix, and blocks, $L_{\max}$ is unchanged. Permuting readout keys, values, coordinates, and masks together preserves attention's weighted sum; layerwise induction therefore preserves readout states and next-token logits under the ideal computation above. Slot relabeling, changed prompt text, or additional order-dependent operations fall outside this argument.

For $M > 0$ evaluated pairs, let $\mathcal{E} \subseteq \{1, \ldots, M\}$ index pairs with valid, non-tied verdicts in both orders. Their parsed candidate identities are $J_r^{\rightarrow}$ and $J_r^{\leftarrow}$. For $|\mathcal{E}| > 0$, define

$$D_{\text{swap}} = \frac{1}{|\mathcal{E}|} \sum_{r \in \mathcal{E}} \text{ind}(J_r^{\rightarrow} \neq J_r^{\leftarrow}), \tag{6}$$

where ind equals one when its condition holds and zero otherwise. Report coverage $|\mathcal{E}|/M$, invalid outputs, and ties separately; disagreement is undefined for empty $\mathcal{E}$. Independent stochastic samples can disagree despite identical predictive distributions. Section 4 lacks this direct consistency test, so it does not establish verdict invariance or validate replacing two-order judging.

Physical-order invariance leaves identifier, length, style, and model-origin preferences intact. Independent candidate encoding also prevents representations from adapting to competitors, potentially reducing quality when such interactions matter.

3.6 MODEL ADAPTATION

Supervised fine-tuning (SFT) adapts task quality to the changed computation; it does not establish locality. For each Qwen3 model (1.7B, 4B, 8B), four solutions are generated per problem from a subset of OpenMathReasoning. The same model produces pairwise judgment or aggregation traces, which provide next-token training targets. Appendix A reports training configurations.

4 EXPERIMENTS

We evaluate cache reusability after candidate reordering and aggregation accuracy.

4.1 SETUP AND BASELINES

We fine-tune Qwen3 models with 1.7B, 4B, and 8B parameters on judging and aggregation prompts, using 2,400 examples per model and a maximum sequence length of 32k. Appendix A lists the reported hyperparameters. Evaluation uses AIME 2024, AIME 2025, and HMMT 2025; n denotes the number of candidate solutions.

Van. and **New** denote models fine-tuned on the same data with standard RoPE and SWAP-ROPE, respectively. Their comparison controls for the fine-tuning data when evaluating the full block encoding. **Orig.** is the reported original-model reference; comparisons between the two fine-tuned schemes use Van. and New.

4.2 REUSABLE KV CACHE RATIO IN PAIRWISE JUDGING

For candidates $(\mathbf{C}_a, \mathbf{C}_b)$, we consider reuse between the original prompt and its swapped counterpart. Let N count all prefill tokens and $\mathcal{R}$ contain positions whose complete KV states remain reusable after the swap. Token correspondence follows the shared context and each candidate's internal order, rather than physical indices. The ideal whole-prompt reuse ratio is

$$r_{\text{KV}} = \frac{|\mathcal{R}|}{N}, \tag{7}$$

reported as a percentage. Membership in $\mathcal{R}$ requires valid keys and values at every layer under the swapped prompt's coordinates and mask. Unlike a chunk-only ratio, the denominator includes context, candidates, and the trailing query.

Table 3 reports reuse estimates of 38.5–40.9% under standard RoPE and 73.8–79.2% under SWAP-ROPE. Improvements occur in every model–benchmark combination. On AIME 2024, SWAP-ROPE

yields 77.3–78.4% reuse across the three model sizes, with gains of 37.5–37.7 percentage points. AIME 2025 has the lowest reuse ratio for each model, but still gains 35.3–36.5 points. Thus, the benefit is consistent across the tested settings despite differences in their absolute ratios.

Larger models do not consistently yield higher reuse. On HMMT, the SWAP-ROPE ratio decreases from 79.2% at 1.7B to 76.3% at 8B, while AIME 2024 ratios remain closely grouped. The observed advantage therefore does not require the largest model in the evaluation; these results describe reuse across workloads rather than a monotonic model-scaling trend.

The whole-prompt denominator requires care. Proposition 1 establishes candidate-block locality, while the fully specified ideal swap also preserves prefix and readout states (Section 3.5). Including context and query tokens therefore does not by itself explain reuse below 100%. The aggregate estimates neither verify exact state equality nor identify which assumptions the measured prompts and implementation satisfy. Interpreting the shortfall requires numerical matching diagnostics and a token-type breakdown; it cannot be attributed to particular prompt regions from the table alone.

The practical opportunity is to avoid fresh candidate encodings across repeated comparisons, as analyzed in Section 3.4. The ratio quantifies potentially reusable state rather than elapsed time. Latency and memory use are not measured; realized savings also depend on cache retrieval and assembly, readout computation, and decoding.

Takeaway. SWAP-ROPE increases reported reuse potential by 35.3–38.3 percentage points across all nine settings. These gains do not quantify serving speedup.

Table 3: Reported whole-prompt KV reuse ratio (%) in pairwise judging, including all prefill tokens. Subscripts give percentage-point gains over RoPE; values measure reuse potential rather than serving speedup.

Model	AIME24		AIME25		HMMT25	
	RoPE	SWAP-ROPE	RoPE	SWAP-ROPE	RoPE	SWAP-ROPE
Qwen3-1.7B	39.8	$77.3_{\uparrow 37.5}$	38.5	$73.8_{\uparrow 35.3}$	40.9	$79.2_{\uparrow 38.3}$
Qwen3-4B	40.7	$78.4_{\uparrow 37.7}$	38.7	$75.2_{\uparrow 36.5}$	40.4	$78.1_{\uparrow 37.7}$
Qwen3-8B	40.1	$77.8_{\uparrow 37.7}$	38.6	$75.1_{\uparrow 36.5}$	39.0	$76.3_{\uparrow 37.3}$

4.3 TEST-TIME-SCALING ACCURACY

Aggregation accuracy is the percentage of evaluated instances whose selected candidate has the correct final answer. Tables 4 and 5 report results for $n = 4$ and $n = 8$. The New–Van. comparison holds the fine-tuning data fixed to assess the full block encoding.

At $n = 4$, SWAP-ROPE exceeds Van. in six of nine model–benchmark combinations. The largest gain is on AIME 2024 with Qwen3-1.7B, from 41.3% to 55.0%. Qwen3-4B improves on all three benchmarks, by 2.4–2.6 percentage points. For Qwen3-8B, the gain is concentrated on AIME 2025 (67.8% to 73.8%), with small decreases on AIME 2024 and HMMT. Across models, the three decreases range from 0.6 to 1.7 points. At this budget, AIME 2025 improves at every model size, whereas HMMT improves only at 4B. Thus, the six positive comparisons are distributed unevenly across tasks rather than concentrated at one model scale.

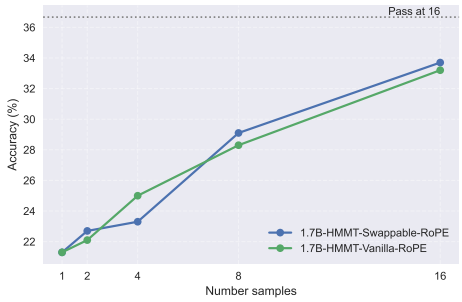


Figure 4: HMMT accuracy for Qwen3-1.7B as candidate count increases. The dotted pass@16 reference measures whether at least one of 16 candidates is correct.

At $n = 8$, SWAP-ROPE exceeds Van. in three combinations and performs worse in six. Gains range from 0.6 to 2.0 percentage points and decreases from 1.2 to 3.8 points. All three models fall behind Van. on AIME 2024, and Qwen3-8B falls behind on every benchmark. HMMT retains gains for

the 1.7B and 4B models, of 0.8 and 2.0 points. The advantage at $n = 4$ therefore does not extend uniformly to a larger pool.

Figure 4 separates absolute scaling from relative performance for Qwen3-1.7B on HMMT. Between four and eight candidates, SWAP-ROPE accuracy rises from 23.3% to 29.1%, while Van. rises from 25.0% to 28.3%, reversing their ordering. Both curves improve further at sixteen candidates. A larger pool can thus improve selection accuracy without giving either encoding a consistent advantage across candidate counts.

One possible explanation for the mixed outcomes is the restriction on competitor-dependent encoding: each candidate is encoded without reading its competitor, leaving comparison to the readout. This is an interpretation, not an isolated causal finding, because coordinates and masks change together. The point estimates establish neither statistical equivalence nor swap-invariant verdicts. Appendix B surveys broader structural opportunities without evaluating these outcomes.

Takeaway. SWAP-ROPE improves accuracy in $6/9$ settings at $n = 4$ and $3/9$ at $n = 8$. The quality tradeoff depends on the model, benchmark, and candidate count.

Table 4: Aggregation accuracy (%) with $n = 4$ candidates. Van. and New denote models fine-tuned with standard RoPE and SWAP-ROPE, respectively; Orig. is the reported original-model reference. Subscripts give percentage-point differences between New and Van.

Model	AIME24			AIME25			HMMT25		
	Orig.	Van.	New	Orig.	Van.	New	Orig.	Van.	New
Qwen3-1.7B	38.8	41.3	55.0$_{\uparrow 13.7}$	36.7	38.7	44.6$_{\uparrow 5.9}$	21.3	25.0	23.3$_{\downarrow 1.7}$
Qwen3-4B	73.3	76.7	79.2$_{\uparrow 2.5}$	65.8	68.7	71.1$_{\uparrow 2.4}$	32.9	36.3	38.9$_{\uparrow 2.6}$
Qwen3-8B	74.6	80.0	79.2$_{\downarrow 0.8}$	65.8	67.8	73.8$_{\uparrow 6.0}$	35.8	38.8	38.2$_{\downarrow 0.6}$

Table 5: Aggregation accuracy (%) with $n = 8$ candidates. Model labels and percentage-point differences follow Table 4.

Model	AIME24			AIME25			HMMT25		
	Orig.	Van.	New	Orig.	Van.	New	Orig.	Van.	New
Qwen3-1.7B	38.8	53.8	50.0$_{\downarrow 3.8}$	36.7	40.2	40.8$_{\uparrow 0.6}$	21.3	28.3	29.1$_{\uparrow 0.8}$
Qwen3-4B	73.3	82.8	79.2$_{\downarrow 3.6}$	65.8	69.9	68.7$_{\downarrow 1.2}$	32.9	37.5	39.5$_{\uparrow 2.0}$
Qwen3-8B	74.6	79.6	76.7$_{\downarrow 2.9}$	65.8	70.4	66.7$_{\downarrow 3.7}$	35.8	41.6	40.2$_{\downarrow 1.4}$

5 CONCLUSION

SWAP-ROPE separates reusable candidate encoding from joint comparison by giving co-present blocks a shared positional origin and restricting their attention to a fixed prefix and their own causal histories. Under explicit model and masking assumptions, this yields locality at every layer: candidate hidden states and cached keys and values remain unchanged under reordering or competitor substitution. For an all-pairs tournament over n candidates, the stated workload assumptions and retained caches reduce fresh candidate encodings from $\binom{n}{2} + n - 1$ to n, while each comparison still requires readout and decoding. Reported KV reuse estimates rise by 35.3–38.3 percentage points, with setting-dependent selection accuracy. The method requires annotated blocks and stable candidate identities; independent prefill may limit interactions needed for judging. Direct verdict-consistency tests, numerical cache validation, and serving benchmarks remain necessary to establish behavioral and practical benefits.

REFERENCES

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench: A bilingual, multitask benchmark for long context understanding, 2024. URL https://arxiv.org/abs/2308.14508.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2024. URL https://arxiv.org/abs/2407.21787.

In Gim, Guojun Chen, Seung seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt cache: Modular attention reuse for low-latency inference, 2024. URL https://arxiv.org/abs/2311.04934.

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, 2023. URL https://arxiv.org/abs/2309.06180.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks, 2019. URL https://arxiv.org/abs/1810.00825.

Jia Li, Xuyuan Guo, Lei Li, Kechi Zhang, Ge Li, Jia Li, Zhengwei Tao, Fang Liu, Chongyang Tao, Yuqi Zhu, and Zhi Jin. Longcodeu: Benchmarking long-context language models on long code understanding, 2025. URL https://arxiv.org/abs/2503.04359.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions, 2023. URL https://arxiv.org/abs/2308.11483.

Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation, 2022. URL https://arxiv.org/abs/2108.12409.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. URL https://arxiv.org/abs/2408.03314.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063. URL https://doi.org/10.1016/j.neucom.2023.127063.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition, 2022. URL https://arxiv.org/abs/2108.00573.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators, 2023a. URL https://arxiv.org/abs/2305.17926.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023b. URL https://arxiv.org/abs/2203.11171.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. Cacheblend: Fast large language model serving for rag with cached knowledge fusion, 2025. URL https://arxiv.org/abs/2405.16444.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets, 2018. URL https://arxiv.org/abs/1703.06114.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

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, 2024. URL https://arxiv.org/abs/2312.07104.

AI USE STATEMENT

Generative AI assisted with writing, notation and argument checks, figures, formatting, and implementation and analysis code. The authors remain responsible for verifying arguments, implementation, and results; writing assistants did not generate the experimental results.

ETHICS STATEMENT

The study uses public benchmarks and open-weight models, with no human subjects or personally identifiable data. Order-invariant encoding does not eliminate other judging biases or establish reliability beyond the evaluated mathematical-solution tasks.

REPRODUCIBILITY STATEMENT

Section 3 specifies the construction and assumptions; Section 4 defines the reported metrics and conditions. Appendix A lists available training settings. Implementation-level invariance and serving performance require separate validation.

A Fine-Tuning Hyperparameters

Table A.1: Reported SFT settings. All models use 2,400 training examples, maximum sequence length 32k, batch size 64, 8 epochs, warmup 0.05, and bfloat16 precision.

Model	Qwen3-1.7B	Qwen3-4B	Qwen3-8B
Learning rate	5×10^{-5}	1×10^{-5}	1×10^{-5}

B Swappable-Block Opportunities

Table B.1 reports preliminary percentages of examples with a swappable-block opportunity in code analysis (LongCode Li et al. (2025)), multiple-choice reasoning (GPQA Rein et al. (2023)), retrieval-augmented generation (MuSiQue Trivedi et al. (2022)), and long-context understanding (LongBench Bai et al. (2024)). Potential blocks are independent code spans, answer options, retrieved passages, and context sections, respectively.

These estimates describe input structure, not demonstrated benefits. Permutations need not preserve task semantics, and chunk-local masking may affect quality; the survey does not validate invariance, accuracy, or reuse beyond the judging experiments.

Table B.1: Preliminary percentages of examples with a swappable-block opportunity in four workloads. These structural estimates do not measure downstream accuracy or cache reuse.

Metric	LongCode	GPQA	MuSiQue	LongBench
Examples (%)	82.8	39.8	43.4	91.6