BEYOND INDEPENDENT TOP-k: PAIRWISE ENERGY ROUTING FOR FROZEN MIXTURE-OF-EXPERTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse mixture-of-experts (MoE) models activate only K of N experts per token, yet conventional Top-K routing ranks experts individually without explicitly accounting for interactions within the selected subset. The highest-scoring experts therefore need not form the most effective subset. We propose PAIRQ, a parameter-efficient method for adapting pretrained MoEs through token-conditioned pairwise routing. PairQ combines frozen native-router scores with learned interactions between expert-selection variables, assigning a quadratic energy to each subset. Only the added routing parameters are trained; the backbone, experts, native router, and expert-mixing rule remain unchanged. An energy-weighted surrogate over candidate subsets propagates language-model loss gradients while preserving hard selection in the forward pass. The selection objective admits a quadratic unconstrained binary optimization (QUBO) formulation. Evaluations on Qwen3-30B-A3B and GPT-OSS-20B show gains over native routing across five mathematical reasoning benchmarks. A complete autoregressive inference run demonstrates that a physical coherent Ising machine (CIM) can decode the routing QUBO.

1 INTRODUCTION

Sparse mixture-of-experts (MoE) models increase total capacity while activating only K of N experts for each token (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022). At a fixed activation budget, a larger expert pool expands the space of possible subsets, $\binom{N}{K}$. For example, Qwen3-30B-A3B selects eight of 128 experts, admitting approximately 1.43×10^{12} possible subsets per token (Yang et al., 2025). Conventional Top-K routing selects among these subsets by ranking experts individually. This efficient rule does not explicitly account for complementarity or redundancy among the selected experts, so the highest-scoring experts need not form the most effective subset.

Evidence from pretrained MoEs motivates considering such interactions. Counterfactual analyses have identified expert subsets with lower token loss than the native route under the same computation budget (Yoon et al., 2026), while studies of expert behavior report structured co-activation and collaboration (Tang et al., 2025; Nguyen-Nhat et al., 2025). Router-only adaptation further motivates improving selection while reusing pretrained experts (Li et al., 2026). We ask whether token-conditioned pairwise interactions can improve expert selection beyond learned individual scores while keeping pretrained MoE parameters frozen. This question calls for a subset-level objective in which the preference for an expert depends on which other experts are selected.

We propose PAIRQ, a parameter-efficient routing method that combines frozen native-router scores with learned pairwise preferences. A compact query network and trainable expert embeddings generate token-conditioned interactions between binary expert-selection variables, defining a quadratic energy over subsets. The quadratic terms act on joint expert selection, not on input features within an individual expert's score. Routing seeks a low-energy subset of K experts. Only the added routing parameters are trained; the backbone, experts, native router, and expert-mixing rule remain unchanged. Figure 1 summarizes the architecture and routing computation.

Learning these preferences requires training through discrete expert selection. We construct an energy-weighted surrogate over candidate expert subsets. A straight-through estimator preserves the selected subset in the forward pass and supplies gradients through the candidate mixture in the backward pass. Language-model loss thus trains the routing preferences without requiring labels for

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.

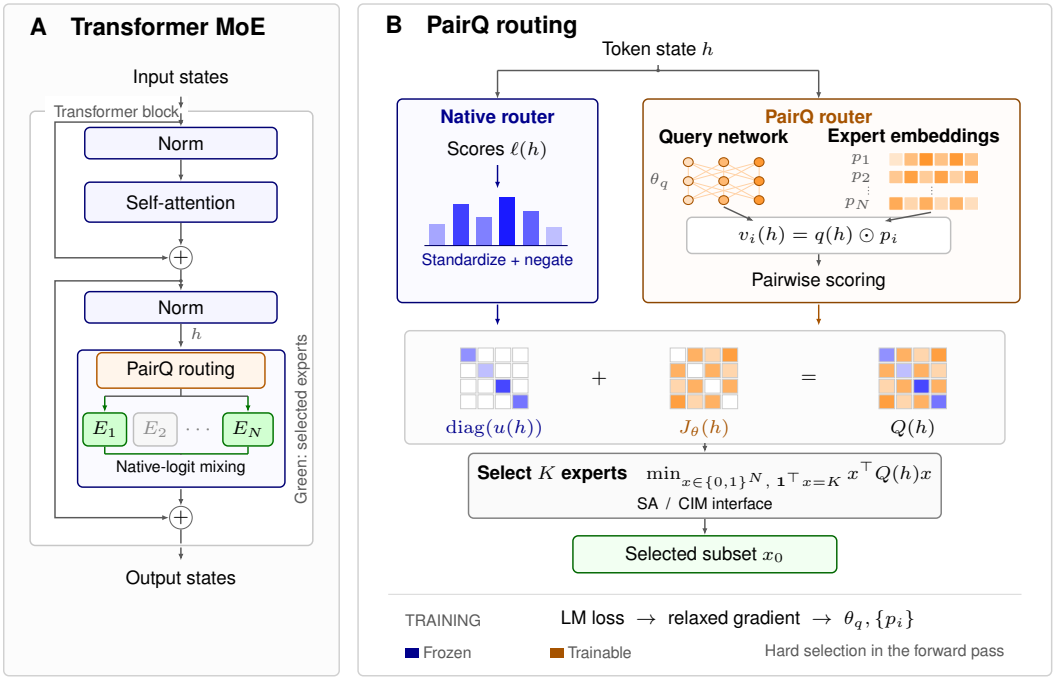


Figure 1: Overview of PairQ. **A:** Selected experts (green) use native-logit mixing in a frozen Transformer MoE block. **B:** A trainable query network and expert embeddings generate token-specific pairwise costs $J_\theta(h)$, which combine with frozen unary costs as $Q(h) = \mathrm{diag}(u(h)) + J_\theta(h)$. Subset selection uses SA by default or a CIM interface; an energy-weighted candidate surrogate supplies training gradients.

expert pairs or differentiation through the discrete solver. At inference, the candidate relaxation is removed and only the K selected experts are evaluated.

The quadratic objective connects expert selection to combinatorial optimization. Encoding the activation-budget constraint as a quadratic penalty yields a quadratic unconstrained binary optimization (QUBO) problem (Glover et al., 2018). We use GPU-batched simulated annealing (SA) as the default solver (Kirkpatrick et al., 1983), with feasibility handling to retain exactly K experts. The equivalence between QUBO and Ising formulations also allows the routing objective to be mapped to a coherent Ising machine (CIM), a specialized photonic quantum computing device for Ising/QUBO optimization (Wei et al., 2026). In this hybrid workflow, conventional hardware computes token representations, routing energies, and expert outputs, while the CIM performs discrete subset search.

Evaluations on Qwen3-30B-A3B and GPT-OSS-20B show gains over native routing across multiple reasoning benchmarks. Comparisons with exact enumeration assess the quality of the approximate routing solver. In addition, a complete autoregressive inference run with CIM-based routing demonstrates the feasibility of the hardware interface.

Our contributions are:

- We construct the PairQ router from a token-query network and trainable expert embeddings, introducing token-conditioned interactions between expert-selection variables while preserving pretrained MoE parameters and the number of active experts.
- We formulate fixed-cardinality MoE expert selection as a QUBO and train its energy with an energy-weighted surrogate over candidate subsets while retaining hard selection in the forward pass.
- We show gains over native baselines on multiple mathematical reasoning benchmarks, characterize approximate search using exact enumeration, and demonstrate CIM-based routing in a complete autoregressive inference run.

2 PROBLEM SETUP

Consider an MoE layer with N experts and K active experts per token. For a token representation $h \in \mathbb{R}^H$, the frozen native router produces logits $\ell(h) \in \mathbb{R}^N$. A route is represented by $x \in \{0,1\}^N$, where $x_i = 1$ selects expert i. The feasible routes are

$$\mathcal{X}_K = \{x \in \{0,1\}^N : \mathbf{1}^\top x = K\}. \tag{1}$$

For a feasible route, the MoE output is

$$o(x;h) = \sum_i \frac{x_i \exp(\ell_i(h))}{\sum_j x_j \exp(\ell_j(h))} f_i(h), \tag{2}$$

where $f_i(h)$ is the output of expert i.

The native Top-K route can equivalently be characterized as

$$x_{\mathrm{nat}}(h) \in \arg\min_{x \in \mathcal{X}_K} -\ell(h)^\top x. \tag{3}$$

This is a modular set objective: the marginal routing cost of expert i is always $-\ell_i(h)$, regardless of the other selected experts. This restriction concerns only the router's subset score, not the nonlinear computation performed by the experts or downstream Transformer layers. Our goal is to learn a token-conditioned interaction score over expert pairs while preserving the feasible set $\mathcal{X}_K$. Consequently, the number of active experts, the backbone, the native router, the experts, and the restricted-softmax mixing rule in Equation 2 remain unchanged. Only the parameters introduced by the pairwise selection mechanism are trainable.

3 PAIRWISE ENERGY ROUTING

PAIRQ changes only expert selection. For each token, it combines a fixed native first-order energy with a learned pairwise energy and selects an exact-K route. Expert computation and the mixing rule in Equation 2 remain unchanged; Figure 1 summarizes the construction.

3.1 NATIVE FIRST-ORDER ENERGY

We convert the frozen native logits into a first-order energy $u(h) \in \mathbb{R}^N$:

$$u_i(h) = -\frac{\ell_i(h) - \mu(h)}{\max\{\sigma(h), \varepsilon\}}, \tag{4}$$

where $\mu(h)$ and $\sigma(h)$ are the mean and population standard deviation of $\{\ell_i(h)\}_{i=1}^N$, and $\varepsilon = 10^{-6}$. The negative sign converts preference maximization into energy minimization, and standardization balances its scale against the learned interaction. Over $\mathcal{X}_K$, centering adds a constant and the positive scaling preserves route order. Thus, minimizing $u(h)^\top x$ recovers $x_{\mathrm{nat}}(h)$ up to boundary ties. We detach $u(h)$, making it a fixed pretrained anchor.

3.2 TOKEN-CONDITIONED PAIRWISE ENERGY

To make expert interactions token dependent, a trainable LayerNorm–MLP with parameters θ_q maps h to a query $q(h) = \mathrm{MLP}_{\theta_q}(\mathrm{LN}(h)) \in \mathbb{R}^d$. Each expert i has a trainable embedding $p_i \in \mathbb{R}^d$, initialized from the corresponding frozen native-router weight:

$$p_i^{(0)} = \mathrm{LN}_0(A w_i^{\mathrm{router}}), \qquad v_i(h) = q(h) \odot p_i, \tag{5}$$

where $w_i^{\mathrm{router}} \in \mathbb{R}^H$ is the router weight for expert i, $A \in \mathbb{R}^{d \times H}$ is a one-time Xavier-initialized projection, LN_0 is non-affine LayerNorm, and $\odot$ denotes elementwise multiplication. We discard A after initialization. The query $q(h)$ represents the token, p_i represents expert identity, and $v_i(h)$ combines them coordinate-wise.

Given trainable coordinate weights $a \in \mathbb{R}^d$, the raw interaction matrix $R(h) \in \mathbb{R}^{N \times N}$ is

$$R_{ij}(h) = \begin{cases} (a \odot v_i(h))^\top v_j(h)/\sqrt{d}, & i \neq j, \\ 0, & i = j. \end{cases} \tag{6}$$

The shared a makes $R(h)$ symmetric, and the zero diagonal leaves all explicit first-order preference in $u(h)$. We normalize the interaction scale and modulate it by a trainable scalar gain $g \in \mathbb{R}$:

$$J(h) = \tanh(g)\frac{R(h)}{\max\{\mathrm{RMS}(R(h)), \varepsilon\}}, \tag{7}$$

where $\mathrm{RMS}(R(h)) = \sqrt{N^{-2}\sum_{i=1}^{N}\sum_{j=1}^{N} R_{ij}(h)^2}$. At initialization, $g = 0$ gives $J(h) = 0$, leaving only the native first-order term. A negative $J_{ij}(h)$ favors selecting experts i and j together, whereas a positive entry discourages it. These are learned routing preferences, not direct measurements of expert synergy. Each modified layer has independent trainable parameters $\theta = \{\theta_q, p_1, \ldots, p_N, a, g\}$; all pretrained weights remain frozen. Router activations use bfloat16 where supported, while normalization statistics, energies, and the CLM loss are accumulated in float32.

3.3 Quadratic Routing Objective

The first- and second-order terms define

$$\begin{aligned} Q(h) &= \mathrm{diag}(u(h)) + J(h), \\ E(x;h) &= x^\top Q(h) x = \sum_{i=1}^{N} u_i(h) x_i + 2 \sum_{1 \le i < j \le N} J_{ij}(h) x_i x_j. \end{aligned} \tag{8}$$

For binary x, the diagonal of $Q(h)$ contributes first-order costs and its off-diagonal entries contribute pairwise costs. The ideal route minimizes $E(x;h)$ over $\mathcal{X}_K$. At $g = 0$, exact minimization recovers $x_{\mathrm{nat}}(h)$ up to boundary ties.

To use an unconstrained binary solver, we penalize deviations from K:

$$\begin{aligned} E_\lambda(x;h) &= E(x;h) + \lambda(\mathbf{1}^\top x - K)^2 \\ &= x^\top \left[Q(h) + \lambda \mathbf{1}\mathbf{1}^\top - 2\lambda K I\right] x + \lambda K^2, \end{aligned} \tag{9}$$

for binary x, where $\lambda > 0$ is chosen from the matrix scale (Appendix B) and I is the identity matrix. The constant does not affect the minimizer, yielding a QUBO. We approximately minimize it with simulated annealing; feasibility filtering and, if needed, repair ensure $x_0 \in \mathcal{X}_K$. Appendix A details read selection.

3.4 Hard-Forward Straight-Through Surrogate

The discrete map from $Q(h)$ to x_0 is non-differentiable. We retain x_0 in the forward pass and relax only the backward pass. For each token, the ordered candidates are the solver route x_0, the native route $x_1 = x_{\mathrm{nat}}(h)$, and M random routes $x_2, \ldots, x_{M+1} \in \mathcal{X}_K$.

Let $o_c = o(x_c; h)$ be the output of x_c under Equation 2. We detach the routes, native logits, and expert outputs, but recompute candidate energies from the live $Q(h)$:

$$e_c = \frac{x_c^\top Q(h) x_c}{K}, \qquad \pi_c = \frac{\exp(-e_c/\tau)}{\sum_{r=0}^{M+1}\exp(-e_r/\tau)}, \tag{10}$$

where $\tau > 0$ is the temperature. We then define

$$\begin{aligned} o_{\mathrm{soft}} &= \sum_{c=0}^{M+1} \pi_c o_c, \\ o_{\mathrm{ST}} &= o_0 + o_{\mathrm{soft}} - \mathrm{sg}(o_{\mathrm{soft}}), \end{aligned} \tag{11}$$

where sg is the identity in the forward pass and has zero derivative. Consequently, $o_{\mathrm{ST}} = o_0$ numerically, preserving hard routing, while the backward pass follows $\mathcal{L}_{\mathrm{CLM}} \to o_{\mathrm{soft}} \to \pi_c \to e_c \to Q(h)$. The task loss therefore changes the relative energies of complete candidate routes through their output differences. No gradient passes through simulated annealing, repair, candidate sampling, the native router, or candidate expert outputs.

This estimator is biased and candidate-local; it does not differentiate the solver itself. Candidate construction and the gradient derivation appear in Appendices C and D.

4 EXPERIMENTS

We evaluate PAIRQ on two MoE backbones and five mathematical reasoning benchmarks. Comparisons with frozen native routing and continued training of the native router assess downstream accuracy; a parameter-matched unary control in Appendix E probes the role of pairwise interactions. We also analyze solver fidelity, CIM inference, expert use, and the training surrogate.

4.1 EXPERIMENTAL SETUP

Models. We evaluate PAIRQ on two open-weight MoE language models that differ in expert count and routing granularity: GPT-OSS-20B (OpenAI, 2025), with $N = 32$ experts and $K = 4$ active, and Qwen3-30B-A3B (Yang et al., 2025), with $N = 128$ experts and $K = 8$ active. Exact enumeration is feasible for GPT-OSS but not for Qwen3; Section 4.3 evaluates SA against exact enumeration for GPT-OSS.

Datasets & Metrics. We report exact-match accuracy on five mathematical reasoning benchmarks; Appendix Table 7 lists their domains and evaluation sizes. We compare parsed numeric answers for GSM8K and AIME, and selected options for AMC23 and MathQA, against the references; unparseable responses are incorrect. Routing diagnostics use total variation distance (TVD) and expert-utilization statistics defined in Section 4.4.

Training and evaluation settings. Both backbones remain frozen except for the routing parameters at the final MoE layer (23 for GPT-OSS and 47 for Qwen3). We train for one epoch on 10M MathPile tokens (Wang et al., 2023); PairQ uses the surrogate in Section 3.4. For B uncorrupted sequences of length T, with token $s_{b,t}$ at position t, the mean next-token objective is

$$\mathcal{L}_{\mathrm{CLM}}(\theta) = -\frac{1}{B(T-1)} \sum_{b=1}^{B} \sum_{t=2}^{T} \log p_\theta(s_{b,t} \mid s_{b,<t}). \tag{12}$$

Logits at positions $1, \ldots, T-1$ are paired with targets at $2, \ldots, T$. The loss is evaluated at every next-token position in float32, with model weights loaded in bfloat16. No auxiliary routing, load-balancing, or distillation loss is added. The router and optimization settings are collected in Table 8.

Inference constructs $Q(h)$, decodes a feasible x_0, evaluates only its K selected experts, and applies native restricted mixing weights; the candidate and straight-through branches are absent. Pairwise scoring and decoding add inference overhead, while training evaluates all experts for the surrogate. Evaluation uses hard routes, the default chat template, and at most 4,096 generated tokens; further settings appear in Appendix F.

4.2 DOWNSTREAM ACCURACY: PAIRQ VS. BASELINES

Table 1 compares the pretrained model without further training (*frozen native router*), continued training of its native gate (*native-router CPT*), and PAIRQ. The trained methods share the data, schedule, and evaluation protocol within each backbone; both adapt routing at only one MoE layer ($L = 1$).

On Qwen3-30B-A3B, PAIRQ exceeds both baselines on all five benchmarks. Native-router CPT matches the frozen native router on AIME25 and AMC23 but falls below it on GSM8K, AIME24, and MathQA; continued training of the original gate does not reproduce PAIRQ's gains. On GPT-OSS-20B, PAIRQ exceeds native-router CPT on all five benchmarks, but its changes relative to the frozen native router vary: AIME24 rises by 6.66 percentage points while AIME25 falls by 3.34 points. GSM8K and AMC23 also improve, whereas MathQA declines. The two AIME sets contain 30 questions each, so these contrasting outcomes do not establish a consistent gain over the pretrained model.

Table 1: Downstream accuracy (%) of the Frozen native router, Native-router CPT, and PairQ. Higher is better.

Method	GSM8K	AIME24	AIME25	AMC23	MathQA
GPT-OSS-20B					
Frozen native router	85.44	36.67	**36.67**	72.29	**87.54**
Native-router CPT	85.82	36.67	26.67	68.67	86.20
PAIRQ	**86.13**	**43.33**	33.33	**73.49**	86.77
Qwen3-30B-A3B					
Frozen native router	85.20	26.70	16.70	61.45	88.14
Native-router CPT	84.80	23.30	16.70	61.45	87.20
PAIRQ	**86.60**	**30.00**	**20.00**	**62.65**	**88.17**

Note: Best results within each model block are bolded; second-best results are underlined.

Parameter-matched control. PAIRQ has more trainable parameters than native-router CPT on both backbones: 1,498,305 versus 92,192 on GPT-OSS-20B, and 3,283,969 versus 262,144 on Qwen3-30B-A3B. To complement this comparison, we include a parameter-matched unary routing experiment in Appendix E.

4.3 DISCRETE SOLVER FIDELITY AND CIM INFERENCE

Because PAIRQ requires a discrete solution at every routing decision, we ask how closely SA approaches exact selection and whether a physical CIM can replace SA throughout autoregressive inference. At the tested scale, SA returns an exact or one-expert-different route in 64.29% of decisions, while the CIM supplies every route needed to complete a full generation. On GPT-OSS-20B AIME24, exact enumeration of the $\binom{32}{4} = 35{,}960$ feasible subsets provides a same-step reference: SA matches an optimum in 23.56% of $111{,}719$ routing decisions and shares exactly three experts with one in another 40.73% (Figure 2, left). Subsequent tokens always follow the SA-selected route, not the enumerated optimum.

The physical-solver test keeps the trained router and experts unchanged. In a complete GPT-OSS-20B GSM8K inference run, the CIM returns a feasible four-expert route at all 214 decisions, with no SA fallback or cardinality repair. The resulting answer to the robe question is "3 bolts", matching the reference (Figure 2, right). This establishes a working physical-solver interface for token-level MoE routing and lays the groundwork for exploring larger-scale MoE systems.

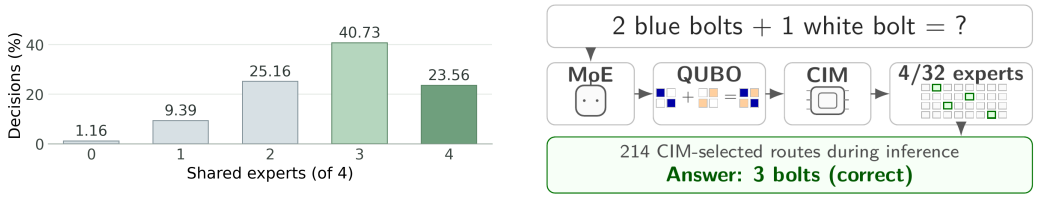


Figure 2: Solver-level evidence. Left: SA–exact expert overlap on GPT-OSS-20B AIME24. Right: a worked GSM8K inference example with CIM decoding.

4.4 ROUTING ANALYSIS: LOAD BALANCE AND SELECTION OVERLAP

To clarify how PAIRQ changes expert selection beyond its aggregate accuracy, we ask whether its behavior is explained by simply activating more experts or by redistributing and reweighting the selected experts. We therefore compare aggregate usage, per-token concentration, and selection overlap with the frozen native router.

Routing metrics. For expert e, let c_e^{base} and c_e^{pair} count selections by the frozen gate and PairQ, respectively, over N experts. The normalized aggregate selection distributions are

$$P_e = \frac{c_e^{\text{base}}}{\sum_{j=1}^{N} c_j^{\text{base}}}, \qquad Q_e = \frac{c_e^{\text{pair}}}{\sum_{j=1}^{N} c_j^{\text{pair}}}, \tag{13}$$

where each denominator is the number of routed tokens times K. Their total variation distance is

$$\text{TVD}(P, Q) = \frac{1}{2} \sum_{e=1}^{N} |P_e - Q_e|. \tag{14}$$

TVD ranges from zero for identical aggregate distributions to one for disjoint distributions. For the Qwen3 Top-8 statistics, let p_e be expert e's share of all selections and $w_{t,j}$ the eight selected weights renormalized to sum to one at token t. Active experts have $p_e > 0$; effective experts are $\exp(-\sum_e p_e \log p_e)$; and Top-8 mass sums the eight largest p_e. Routing entropy averages $-\sum_{j=1}^{8} w_{t,j} \log w_{t,j} / \log 8$; mean Top-1 weight averages $\max_j w_{t,j}$.

On Qwen3 AMC23, PAIRQ activates more experts than the frozen router (128 vs. 120; Table 2). However, its lower effective-expert count and greater Top-8 selection mass (29.53% to 35.58%; see Appendix H) reveal more concentrated aggregate usage. Lower per-token routing entropy and higher mean Top-1 weight likewise indicate sharper weighting among the selected experts. Together with a TVD of 0.384 and a four-of-eight Top-8 overlap, these results show that PAIRQ redistributes expert use rather than simply balancing load.

On GPT-OSS MathQA, the smaller TVD of 0.163 and the increase in effective experts from 23.94 to 26.17 indicate a more modest redistribution relative to the frozen router.

Table 2: Expert-routing statistics on AMC23 (Qwen3-30B-A3B; N=128, K=8).

Method	Acc. (%)	Active experts	Eff. experts	Top-8 mass (%)	Routing entropy	Mean Top-1 weight (%)
Frozen native router	61.45	120/128	69.79	29.53	0.933	25.52
PAIRQ	**62.65**	128/128	64.71	35.58	0.793	36.17

4.5 ABLATION: STRAIGHT-THROUGH SURROGATE

For each token, we consider six routes: the simulated-annealing (SA) route, the native route, and four random routes. Each has an energy (routing score) and an expert output. We compare two backward pairings:

Aligned: Each route's energy is paired with its own expert output.

Cyclic: Each energy uses the previous route's output (the first uses the sixth).

Both use the same hard SA route and expert output in the forward pass. We train three paired GPT-OSS-20B runs (seeds 123–125), matching initialization, data order, budget, and solver settings; only the backward pairing differs. We compare held-out loss and estimated loss changes from replacing a route. Appendix F.3 gives the protocol.

Part 1: Does correct pairing help after training? On 128 held-out MathPile documents (32,640 next-token targets), aligned pairing has lower NLL (mean next-token loss) in all three seeds under hard SA routing (Table 3). Here ΔNLL is aligned minus cyclic NLL, so negative values favor aligned. The exact two-sided seed sign-flip test gives $p = 0.25$ (two of eight possible sign patterns are at least as extreme).

Table 3: Held-out hard-SA NLL for aligned and cyclic GPT-OSS-20B checkpoints (three seeds; mean ± SD).

Seed	Aligned NLL	Cyclic NLL	ΔNLL
123	1.621756	1.779226	−0.157469
124	1.615797	1.771258	−0.155461
125	1.619690	1.778249	−0.158559
Mean ± SD	1.6191±0.0030	1.7762±0.0043	−0.1572±0.0016

Part 2: Does correct pairing point toward better routes? We now hold the three *aligned* final checkpoints fixed. At eight token positions in each of 32 held-out documents, we consider six candidate routes. For each alternative to the SA route, we estimate the loss change by taking the dot product of the current loss gradient with the change in expert output. We then run that replacement through the final normalization and language-model head to measure its actual NLL. At the same checkpoints and candidate routes, we compare whether the estimates get the loss direction and pairwise ordering right; the cyclic column does not use cyclic-trained checkpoints. Aligned pairing agrees with measured loss direction in 86.41% of comparisons versus 44.30% for cyclic, and with candidate ordering in 88.66% versus 47.28% (Table 4).

Table 4: Fixed-candidate loss agreement at aligned GPT-OSS checkpoints (%; three seeds, mean ± SD). Duplicates are merged and ties excluded; see Appendix F.5.

Agreement metric	Aligned correspondence	Cyclic correspondence
Loss-change sign	86.41 ± 0.73	44.30 ± 0.99
Candidate-pair ordering	88.66 ± 0.45	47.28 ± 0.17

5 Related Work

Sparse MoE routing and router adaptation. Sparsely gated MoEs increase model capacity through conditional expert activation (Shazeer et al., 2017). GShard and Switch Transformer established large-scale Top-k and Top-1 routing paradigms (Lepikhin et al., 2021; Fedus et al., 2022), while ST-MoE studied routing stability and transferability (Zoph et al., 2022). Expert Choice reverses the assignment direction by allowing experts to select tokens under a fixed capacity (Zhou et al., 2022). Subsequent work adapts routing decisions to tokens or layers: Ada-K varies the number of active experts per token (Zhao et al., 2025), Alloc-MoE allocates an activation budget across layers and tokens (Liu et al., 2026a), and RMoE incorporates information from earlier routing decisions (Qiu et al., 2025). In frozen MoEs, counterfactual analysis has identified equal-compute subsets with lower token loss than the native route (Yoon et al., 2026); router-only adaptation likewise shows that routing can be improved without updating the experts (Li et al., 2026). PairQ considers this frozen-expert setting while retaining the native router's unary scores and a fixed activation cardinality. It changes the objective used to select a subset by adding token-conditioned pairwise terms.

Optimization of discrete routes. Hard expert selection limits gradient flow through routing decisions. DSelect-k constructs a differentiable sparse gate (Hazimeh et al., 2021), Soft MoE uses continuous token–expert assignments (Puigcerver et al., 2024), and SMEAR and Lory use differentiable expert merging (Muqeeth et al., 2023; Zhong et al., 2024). ReMoE replaces discontinuous Top-k+Softmax routing with a ReLU-based router (Wang et al., 2025). Other methods retain sparse forward execution while improving the gradient signal through estimation or denser backpropagation (Liu et al., 2023; Panda et al., 2024). PairQ also retains a hard route: an approximate solver selects an exact-K subset in the forward pass, while an energy-softmax over complete candidate subsets supplies a backward surrogate. The surrogate is local to those candidates and does not differentiate through the solver.

Expert interactions and structured selection. Independent Top-k scores cannot directly assign different values to an expert depending on which other experts are selected. Prior work has examined

expert collaboration (Tang et al., 2025), specialization (Nielsen et al., 2025), graph-based expert relations (Nguyen-Nhat et al., 2025), and router–expert alignment or coupling (Li et al., 2026; Lv et al., 2026). These studies motivate considering structure in routing, but differ in how that structure affects the selection objective. Quadratic gating applies second-order functions to the input within individual expert gates (Akbarian et al., 2024); PAIRQ instead places pairwise terms on the binary variables that jointly select experts. Its fixed-cardinality objective admits a QUBO formulation and classical annealing-based search (Glover et al., 2018; Kirkpatrick et al., 1983). Hybrid quantum–classical MoEs have explored quantum gating with classical experts on classification tasks (Heddad & Bouanane, 2025), while photonic MoE hardware has implemented optical experts for image classification (Liu et al., 2026b). In contrast, a physical CIM decodes our token-level routing QUBO within an autoregressive language-model MoE. The contribution lies in learning token-conditioned subset scores anchored to a frozen pretrained router and demonstrating their physical decoding, rather than in the QUBO formulation or solver itself.

6 Limitations and Discussion

While PAIRQ shows the potential of quadratic expert selection in MoE models with frozen experts, two limitations remain. First, while we demonstrate CIM-based routing in a complete autoregressive inference run, its efficiency and scalability remain to be established. Evaluating practical benefits requires accounting for hardware mapping, communication overhead, and end-to-end latency across broader workloads and larger expert pools. Second, PAIRQ trains only the new router while keeping pretrained experts frozen. This isolates the effect of routing and preserves parameter efficiency, but prevents routing and expert representations from adapting jointly. Joint optimization, informed by a better understanding of expert collaboration, redundancy, and reuse, may enable more effective expert selection and MoE architectures.

7 Conclusion

In this work, we investigate whether MoE routing can benefit from modeling interactions among experts beyond conventional independent Top-k selection. We introduce PAIRQ, a parameter-efficient routing framework that formulates expert selection as a token-conditioned quadratic optimization problem, enabling pairwise expert interactions while keeping pretrained experts and routing components largely unchanged. Experiments on open-weight MoE language models show task-dependent gains over native routing. Beyond PAIRQ, our work highlights the potential of viewing MoE routing as a structured combinatorial optimization problem and provides a foundation for future exploration of scalable optimization algorithms, expert adaptation strategies, and hardware-efficient routing systems.

AI use statement

In this work, we used generative AI tools for methodological feedback, literature search, manuscript drafting, and language editing. We also used these tools to assist with data processing, code development, experiment execution, and the aggregation and interpretation of experimental results. The authors reviewed and verified the resulting code, analyses, claims, and manuscript, and take responsibility for the final content of this work.

Reproducibility statement

Equations 5–11 specify the router and training surrogate; Appendix D gives the candidate-energy and routing-matrix gradients. Appendix F.1 records audited hyperparameters; Appendix A specifies read selection.

References

Pedram Akbarian, Huy Nguyen, Xing Han, and Nhat Ho. Quadratic gating functions in mixture of experts: A statistical insight. *arXiv preprint arXiv:2410.11222*, 2024. URL https://arxiv.org/abs/2410.11222v1.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://arxiv.org/abs/2101.03961.

Fred Glover, Gary Kochenberger, and Yu Du. A tutorial on formulating and using QUBO models. *arXiv preprint arXiv:1811.11538*, 2018. URL https://arxiv.org/abs/1811.11538.

Hussein Hazimeh, Zhe Zhao, Aakanksha Chowdhery, Maheswaran Sathiamoorthy, Yihua Chen, Rahul Mazumder, Lichan Hong, and Ed H. Chi. DSelect-k: Differentiable selection in the mixture of experts with applications to multi-task learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://arxiv.org/abs/2106.03760.

Reda Heddad and Lamiae Bouanane. Hybrid quantum-classical mixture of experts: Unlocking topological advantage via interference-based routing. *arXiv preprint arXiv:2512.22296*, 2025. URL https://arxiv.org/abs/2512.22296.

Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. doi: 10.1126/science.220.4598.671.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2006.16668.

Zhongyang Li, Ziyue Li, and Tianyi Zhou. Routing manifold alignment improves generalization of mixture-of-experts LLMs. In *International Conference on Learning Representations*, 2026.

Baihui Liu, Kaiyuan Tian, Wei Wang, Zhaoning Zhang, Linbo Qiao, and Dongsheng Li. AllocMoE: Budget-aware expert activation allocation for efficient mixture-of-experts inference. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9653–9667, 2026a. doi: 10.18653/v1/2026.acl-long.437. URL https://aclanthology.org/2026.acl-long.437/.

Liyuan Liu, Jianfeng Gao, and Weizhu Chen. Sparse backpropagation for MoE training. *arXiv preprint arXiv:2310.00811*, 2023. URL https://arxiv.org/abs/2310.00811.

Wencan Liu, Zhenghang Zhang, Peng Meng Chan, Run Sun, Yutong He, Yuhao Wang, Caihua Zhang, Sigang Yang, Tingzhao Fu, Yuyao Huang, Chaoran Huang, and Hongwei Chen. Photonic mixture-of-experts for scalable multi-task on-chip optical neural networks. *Nature Communications*, 17:7145, 2026b. doi: 10.1038/s41467-026-73983-4. URL https://www.nature.com/articles/s41467-026-73983-4.

Ang Lv, Jin Ma, Yiyuan Ma, and Siyuan Qiao. Coupling experts and routers in mixture-of-experts via an auxiliary loss. In *International Conference on Learning Representations*, 2026.

Mohammed Muqeeth, Haokun Liu, and Colin Raffel. Soft merging of experts with adaptive routing. *arXiv preprint arXiv:2306.03745*, 2023.

Minh-Khoi Nguyen-Nhat, Rachel S. Y. Teo, Laziz Abdullaev, Maurice Mok, Viet-Hoang Tran, and Tan Minh Nguyen. Modeling expert interactions in sparse mixture of experts via graph structures. *arXiv preprint arXiv:2510.16411*, 2025. URL https://arxiv.org/abs/2510.16411.

Stefan Nielsen, Rachel Teo, Laziz Abdullaev, and Tan Nguyen. Tight clusters make specialized experts. In *International Conference on Learning Representations*, 2025.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025. URL https://arxiv.org/abs/2508.10925.

Ashwinee Panda, Vatsal Baherwani, Zain Sarwar, Benjamin Therien, Sambit Sahu, Stephen Rawls, Supriyo Chakraborty, and Tom Goldstein. Dense backpropagation improves routing for sparsely-gated mixture-of-experts. In *Proceedings of the 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, volume 262 of *Proceedings of Machine Learning Research*, pp. 81–101, 2024. URL https://proceedings.mlr.press/v262/panda24a.html.

Joan Puigcerver, Carlos Riquelme, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. In *International Conference on Learning Representations*, 2024.

Zihan Qiu, Zeyu Huang, Shuang Cheng, Yizhi Zhou, Zili Wang, Ivan Titov, and Jie Fu. Layerwise recurrent router for mixture-of-experts. In *International Conference on Learning Representations*, 2025.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1701.06538.

Yuanbo Tang, Yan Tang, Naifan Zhang, Meixuan Chen, and Yang Li. Unveiling hidden collaboration within mixture-of-experts in large language models. *arXiv preprint arXiv:2504.12359*, 2025. URL https://arxiv.org/abs/2504.12359.

Zengzhi Wang, Xuefeng Li, Rui Xia, and Pengfei Liu. MathPile: A billion-token-scale pretraining corpus for math. *arXiv preprint arXiv:2312.17120*, 2023. URL https://arxiv.org/abs/2312.17120.

Ziteng Wang, Jun Zhu, and Jianfei Chen. ReMoE: Fully differentiable mixture-of-experts with ReLU routing. In *International Conference on Learning Representations*, 2025.

Hai Wei, Chengjun Ai, Putuo Guo, Bingjie Jia, Lixin Yuan, Hanquan Song, Shaobo Chen, Chongyu Cao, Jie Wu, Chao Ju, Yin Ma, Jintao Fan, Minglie Hu, Chuan Wang, and Kai Wen. A versatile coherent Ising computing platform. *Light: Science & Applications*, 15:74, 2026. doi: 10.1038/s41377-025-02178-1.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Youngsik Yoon, Siwei Wang, Wei Chen, and Jungseul Ok. When are experts misrouted? counterfactual routing analysis in mixture-of-experts language models. *arXiv preprint arXiv:2605.07260*, 2026. URL https://arxiv.org/abs/2605.07260.

Zijia Zhao, Longteng Guo, Jie Cheng, Xuange Gao, Hua Huang, and Jing Liu. Ada-K routing: Boosting the efficiency of MoE-based LLMs. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/df22a19686a558e74f038e6277a51f68-Abstract-Conference.html.

Zexuan Zhong, Mengzhou Xia, Danqi Chen, and Mike Lewis. Lory: Fully differentiable mixture-of-experts for autoregressive language model pre-training. In *Conference on Language Modeling*, 2024.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M. Dai, Zhifeng Chen, Quoc V. Le, and James Laudon. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*, volume 35, pp. 7103–7114, 2022. URL https://arxiv.org/abs/2202.09368.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. ST-MoE: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022. URL https://arxiv.org/abs/2202.08906.

A READ SELECTION

Each annealing read produces a binary mask and its penalized energy. Among feasible reads, we select the mask with the lowest original routing energy. If none is feasible, the bounded repair procedure constructs an exact-K mask from the lowest-cost reads. This decoder is used only for the hard forward route, so its approximation quality is evaluated separately from the training surrogate. Appendix B gives the implementation details.

B SA IMPLEMENTATION

We implement the SA decoder with a batched Triton kernel. Algorithm 1 gives the procedure used for the reported routes. For each token, the adapter adds the exact-K penalty to its routing matrix and converts the result to dense Ising fields and couplings. If the penalty is not specified, its value is the number of experts times the mean absolute matrix entry, with a 10^{-6} numerical offset. A geometric inverse-temperature schedule is constructed separately for each token from its fields and couplings.

Algorithm 1 SA decoding of a batch of routing QUBOs

Require: Token matrices $\{Q_t\}$, expert count N, target K; $R = 64$ reads, $B = 1{,}000$ inverse-temperature points

for all tokens t **do**
  If unset, set $\lambda_t \leftarrow N\big(\mathrm{mean}_{ij}\,|Q_{t,ij}| + 10^{-6}\big)$
  Add the cardinality penalty to Q_t and convert to Ising (h_t, J_t)
  Build the geometric schedule $\beta_{t,1:B}$ from (h_t, J_t)
  for $r = 1, \ldots, R$ **in parallel do**
    Initialize random spins $s \in \{-1, +1\}^N$ and cached flip costs Δ_j
    for $b = 1, \ldots, B$ **do**
      for $p = 1, \ldots, N$ **do** ▷ one sweep; indices sampled with replacement
        Sample spin index j uniformly and $u \sim \mathrm{Uniform}(0, 1)$
        if $\Delta_j \le 0$ **or** $u < \exp(-\beta_{t,b}\Delta_j)$ **then**
          Flip s_j and update the cached Δ values
        end if
      end for
    end for
    Decode s to a binary read $x_{t,r}$
  end for
  $F_t \leftarrow \{r : \sum_i x_{t,r,i} = K\}$
  if $F_t \neq \varnothing$ **then**
    Set x_t to the read in F_t with the lowest original routing energy
  else
    Repair the eight lowest-penalty-energy reads; set x_t to the repaired mask with the lowest original routing energy
  end if
end for
return the selected expert indices for all x_t

Each token–read pair is an independent Triton program, so a token batch is processed in one kernel launch. The reported configuration makes one sweep at each of the 1,000 schedule points. Accepted spin flips update cached flip costs rather than recomputing every trial energy from scratch. When repair is needed, the adapter removes or adds the expert that gives the lowest original routing energy until exactly K remain, then repeatedly applies the best improving one-for-one swap. The returned indices are used for hard expert selection; no gradient passes through the decoder.

C CANDIDATE-BASED STRAIGHT-THROUGH TRAINING

This appendix records the implementation choices behind Equations 10 and 11. For each token, the ordered candidate list contains the decoded hard route x_0, the native route $x_1 = x_{\mathrm{nat}}(h)$, and M

random exact-K routes $x_2, \ldots, x_{M+1}$; reported runs use $M = 4$. In the GPT-OSS implementation, a random route takes the first K indices of a random permutation; a draw identical to x_0 is retried at most 32 times. The Qwen implementation instead takes the Top-K indices of independent random scores, without rejecting a draw equal to x_0. Neither implementation otherwise deduplicates candidate slots. Thus equal subsets can occupy multiple softmax slots and receive separate probability mass.

The candidate masks, native logits, and expert outputs are detached. GPT-OSS computes all N expert outputs once per training token chunk and reuses them across candidates; Qwen evaluates the selected K experts separately for each candidate without tracking expert-output gradients. Both form o_c using the restricted native-logit mixing rule in Equation 2. Candidate energies, by contrast, are recomputed from the live $Q(h)$ so their softmax probabilities remain differentiable. Since every candidate has exactly K active experts, its cardinality penalty is constant and is omitted. Reported runs use $\tau = 1$; GPT-OSS clamps a configured temperature below at 10^{-6}, whereas Qwen uses the configured positive temperature directly.

Equation 11 returns o_0 numerically in the forward pass; its backward branch differentiates only the candidate-energy mixture. Neither candidate sampling nor annealing, repair, native logits, or expert activations receives a gradient. At inference this branch is absent, and only the selected K experts are evaluated.

D Gradient Derivation

For one token, treat candidate routes and outputs as constants and let $\delta = \partial \mathcal{L}_{\mathrm{CLM}} / \partial o_{\mathrm{ST}}$ be the upstream gradient evaluated at the hard forward output. Differentiating the candidate softmax gives

$$r_c \equiv \frac{\partial \mathcal{L}_{\mathrm{CLM}}}{\partial e_c} = -\frac{\pi_c}{\tau} \langle \delta,\, o_c - o_{\mathrm{soft}} \rangle \,. \tag{15}$$

Hence a candidate whose output is locally favorable receives a gradient that lowers its energy under gradient descent. The r_c sum to zero, although shared router parameters need not move candidate energies independently. This is the derivative of the surrogate, not of the discrete solver.

Because $e_c = x_c^\top Q(h) x_c / K$ and x_c is detached, the signal to the live routing matrix is

$$\frac{\partial \mathcal{L}_{\mathrm{CLM}}}{\partial Q(h)} = \frac{1}{K} \sum_{c=0}^{M+1} r_c x_c x_c^\top . \tag{16}$$

The native first-order term is detached, so only $J(h)$ receives trainable gradients. For a symmetric off-diagonal coefficient $J_{ij} = J_{ji}$, its derivative is

$$\frac{\partial \mathcal{L}_{\mathrm{CLM}}}{\partial J_{ij}(h)} = \frac{2}{K} \sum_{c=0}^{M+1} r_c x_{c,i} x_{c,j}, \qquad i < j. \tag{17}$$

The factor of two follows from the symmetric energy convention.

Finally, $J(h) = \tanh(g) R(h) / \rho(h)$ with $\rho(h) = \max\{\mathrm{RMS}(R(h)), \varepsilon\}$. At $g = 0$, the task gradients to the embeddings, coordinate weights, and query network are zero, but the gain can receive a nonzero gradient because $R(h)$ is already defined. Once g moves from zero, automatic differentiation propagates Equation 16 through the normalization and factorization to these parameters. Identical candidate outputs provide no discriminative routing gradient, even to g.

E Parameter-Matched Unary Routing Control

This control matches PairQ's 3,283,969 trainable parameters without pairwise expert interactions. On Qwen3-30B-A3B, we compare the original L1 PairQ checkpoint with this unary router, both trained with seed 123 on approximately 10M MathPile tokens while adapting only the final MoE layer. The unary router retains the query network, expert embeddings, coordinate weights, and scalar gain. For query $q(h) \in \mathbb{R}^d$, expert embedding p_i, and coordinate weights a, its correction is

$$r_i(h) = \frac{1}{\sqrt{d}} \sum_{m=1}^{d} a_m q_m(h)^2 p_{i,m}, \qquad c_i(h) = \tanh(g) \frac{r_i(h)}{\max\{\sqrt{N^{-1} \sum_j r_j(h)^2}, \epsilon\}}. \tag{18}$$

It minimizes $E_{\mathrm{U}}(x;h) = \sum_i [u_i(h) + c_i(h)]x_i$ subject to $\sum_i x_i = K$, where u_i is the frozen, standardized native-router cost. With no off-diagonal coupling, the unary optimum is the K experts with the lowest individual costs. Zero gain recovers native Top-K routing. Selected experts retain the native router's mixing weights. In contrast, PAIRQ parameterizes pairwise interactions using the same learned components and decodes joint energy with simulated annealing. The control therefore tests a complete unary routing alternative at the same parameter count; its exact decoder and the PairQ solver are different.

Table 5 compares the frozen native router, PairQ L1, and Unary L1. All three use the same 128 held-out MathPile documents (32,640 predicted tokens) for token-weighted NLL and PPL. Downstream scores are zero-shot accuracy under the same Qwen3 prompting and generation protocol.

Table 5: Qwen3-30B-A3B L1 unary-versus-pairwise comparison. NLL and PPL are lower-is-better; benchmark entries are accuracy in percent. One training seed (123) is reported.

Router	NLL ↓	PPL ↓	GSM8K	AIME24	AIME25	AMC23
Frozen native	1.557991	4.749272	85.216	26.667	16.667	61.446
Unary	1.549735	4.710219	85.216	26.667	**23.333**	60.241
PairQ	**1.547129**	**4.697963**	**86.581**	**30.000**	20.000	**62.651**

The paired-document bootstrap interval for Unary minus PairQ NLL is $[-0.001701, 0.006737]$. The 30-question AIME sets change by 3.33 points per answer.

F TRAINING AND INFERENCE

F.1 BACKBONE AND REFERENCE CONFIGURATIONS

Table 6 records the supplied single-layer cross-backbone settings. The GPT-OSS parameter count is specific to the $D = 512$, $d = 32$ configuration; values are not extrapolated to other sweeps.

Table 6: Model-specific configurations for the supplied single-layer cross-backbone runs.

Configuration	GPT-OSS-20B	Qwen3-30B-A3B
Transformer / decoder layers	24	48
Experts / active experts	32 / 4	128 / 8
Total / activated parameters	20.9B / 3.6B	30.5B / 3.3B
Modified layer	Final MoE layer (23)	Final MoE layer (47)
Training tokens	10M	10M
Optimizer steps	2,442	2,442
Trainable parameters	1,498,305	3,283,969

F.2 DOWNSTREAM BENCHMARKS

Table 7 lists the five benchmarks used in the main downstream comparison. AIME answers are integers in $[0, 999]$; answer extraction and exact-match scoring are described in Section 4.1.

Table 7: Mathematical reasoning benchmarks and the size of each evaluation set.

Dataset	Domain	# Samples	Metric
GSM8K	grade-school arithmetic word problems	1,319	exact match
AIME24	competition mathematics, 2024	30	exact match
AIME25	competition mathematics, 2025	30	exact match
AMC23	multiple-choice competition mathematics, 2023	83	exact match
MathQA	multiple-choice quantitative reasoning	2,985	exact match

F.3 GPT-OSS Straight-Through Ablation Protocol

This diagnostic adapts only GPT-OSS-20B's final MoE layer ($N = 32$, $K = 4$). The backbone, experts, and native router are frozen; the PairQ router has $d = 32$, $D = 512$, and 1,498,305 trainable parameters. For seeds 123, 124, and 125, paired arms start from identical router weights and share data order, optimizer, schedule, and SA settings. Each uses 9,997,057 tokens from 474 complete MathPile documents under a 10M-token ceiling, sequence length 256, effective batch size 16, AdamW (10^{-4} learning rate, 0.01 weight decay), and gradient-norm clipping at 1.0. Training ends at the predeclared step 2,441 checkpoint rather than a validation-selected step. SA uses 64 reads and 1,000 sweeps; candidate temperature is 1.

The six ordered candidates are the SA route, native route, and four random size-4 subsets. Let e_c and o_c denote candidate energy and detached expert output, with $c = 0$ the SA route and $\pi_c = \mathrm{softmax}(-e/\tau)_c$. The aligned surrogate is $s = \sum_c \pi_c o_c$; the negative control uses the fixed cyclic mismatch $s_{\mathrm{cyc}} = \sum_c \pi_c o_{(c-1) \bmod 6}$. In each arm, the returned activation is $o_0 + s - \mathrm{sg}(s)$, substituting s_{cyc} for the control. The cyclic permutation is deterministic, uses no extra random draws, and is reproduced during gradient-checkpoint recomputation. Identical inputs and parameters thus produce identical hard forward values but different surrogate Jacobians. Routes and candidate sets need not remain identical once training trajectories diverge.

The first document crossing the training-token budget is excluded from both splits. Held-out evaluation uses the first 256 tokens from each of the next 128 distinct documents (32,640 next-token targets), with hard SA and matched evaluation seeds within each pair. NLL is token-weighted; PPL is $\exp(\mathrm{NLL})$. A normalized full-document hash audit found no exact duplicates across splits, but does not rule out shorter overlaps. Table 3 reports the per-seed endpoints. A crossed seed/document bootstrap gave a descriptive 95% interval $[-0.1719, -0.1428]$ for aligned minus cyclic NLL; the seed-level sign-flip test is reported in Section 4.5.

F.4 Reference Hyperparameters

Table 8 gives the reference router and optimization settings used for the reported checkpoints.

Table 8: Reference router and autoregressive-training configuration.

Setting	Reference value
Query dimensions (d, D)	$(32, 512)$
Candidates (M, τ)	$(4, 1)$
Candidate energy	$x_c^\top Q(h) x_c / K$
SA reads / schedule steps	64 / 1000
Batch / token chunk	16 / 256
Sequence length / stride	256 / 256
Optimizer	AdamW, 10^{-4}, decay 0.01
Clipping / seed / epochs	1.0 / 123 / 1

F.5 Fixed-Candidate Counterfactual Protocol

This diagnostic uses only the three aligned step-2,441 checkpoints from Appendix F.3; it performs no additional training. We take the first 32 of that experiment's fixed held-out 256-token document blocks. Within every block, zero-based positions $15, 47, 79, 111, 143, 175, 207, 239$ predict their respective next tokens. These are the same document identities and positions across seeds. At each checkpoint and position, we fix the six ordered candidates: the SA route ($c = 0$), native route, and four random size-4 subsets. Random-candidate draws are reproducible by document. This gives 256 evaluated token states per checkpoint, not 256 independent training runs.

Let r be the cached final-layer residual and o_c the candidate's native-weighted expert output. We evaluate each finite replacement through the actual final normalization and language-model head,

$$\ell_c = \mathrm{CE}(W_{\mathrm{LM}} \, \mathrm{Norm}(r + o_c), y_{t+1}) \, .$$

The implementation preserves the full-sequence normalization shape before selecting the inspected positions. It checks that $r + o_0$ equals the original final-normalization input and that reconstructed

hard-route logits match the full forward pass within maximum absolute error 10^{-5}. With $\delta = \nabla_{o_0} \ell_0$, define

$$d_c = \delta^\top (o_c - o_0), \qquad d_c^{\mathrm{cyc}} = \delta^\top \left(o_{(c-1) \bmod 6} - o_5 \right).$$

The cyclic diagnostic changes only the correspondence used to construct this first-order score; the checkpoint, candidates, ground-truth labels, and measured finite losses ℓ_c remain fixed.

Loss-change sign agreement compares d_c (or d_c^{cyc}) with $\ell_c - \ell_0$. Candidate-pair agreement compares $d_c - d_j$ with $\ell_c - \ell_j$ for unordered pairs, and analogously for the cyclic scores. Repeated unordered expert sets are merged before either metric; scores within each duplicate group are averaged, including cyclic scores. After merging, sign comparisons re-center both predicted and measured differences on the merged group containing the SA route. Comparisons with absolute predicted or measured difference at most 10^{-6} are excluded as ties. Each document contributes its own ratio of concordant to non-tied comparisons; we average these ratios across documents and then across the three seeds. The six slots contain an average of 5.8867 unique expert sets.

We report seed SD and descriptive 95% intervals from 10,000 document bootstrap draws (seed 9241). Each draw resamples the same document IDs for all three fixed checkpoints. The sign-agreement intervals are $[84.85, 87.92]\%$ for aligned and $[42.36, 46.19]\%$ for cyclic correspondence; candidate-pair intervals are $[87.25, 90.05]\%$ and $[45.68, 48.87]\%$, respectively. These intervals condition on the three trained checkpoints and do not represent uncertainty over a population of training seeds.

For context, ranking candidates by QUBO energy alone agrees with their measured loss ordering in 56.69% of non-tied pairs. The SA candidate's mean excess loss over the best of the six measured candidates is 0.1612 nats. Replacing SA with the native candidate changes NLL by -0.0064 nats, with a document-bootstrap interval $[-0.0169, 0.0030]$. The diagnostic therefore does not identify SA as the loss-optimal candidate or show a reliable native-versus-SA loss difference. It tests a fixed candidate set and an output-space local approximation, not an optimizer step followed by fresh SA decoding.

G Interpretation and Limitations

G.1 Pairwise and First-Order Identifiability

Zeroing the diagonal removes explicit trainable unary coefficients from J, but fixed-cardinality routes do not make unary and pairwise effects uniquely identifiable. For arbitrary $b_1, \ldots, b_N$, define

$$\Delta J_{ij} = \frac{b_i + b_j}{2} \; (i \neq j), \qquad \Delta J_{ii} = 0.$$

For every $x \in \mathcal{X}_K$,

$$x^\top \Delta J x = \sum_{i<j} (b_i + b_j) x_i x_j \tag{19}$$

$$= (K-1) \sum_i b_i x_i. \tag{20}$$

Thus, a node-additive component of the pairwise matrix is equivalent to a first-order correction on $\mathcal{X}_K$. Accuracy gains alone therefore do not prove reliance on irreducible pairwise structure; a parameter-matched unary control or an explicit projection is required for that claim.

G.2 Candidate Coverage and Duplicate Slots

The surrogate compares only $M+2$ candidates among $\binom{N}{K}$ feasible routes, so its gradient is candidate-local. If one route occurs m times with the same energy and output, its aggregate unnormalized weight is

$$m \exp(-e/\tau) = \exp[-(e - \tau \log m)/\tau].$$

Duplicates therefore change the relaxation rather than being neutral. If all candidate outputs are identical, then $o_c - o_{\text{soft}} = 0$ for every c and the CLM gradient to the energies vanishes. These limitations motivate reporting candidate coverage and solver-budget sensitivity in future evaluations.

H Additional Routing-Selection Analysis

H.1 Cumulative Selection Mass

Figure 3 compares concentration rather than expert identity because each method ranks experts independently.

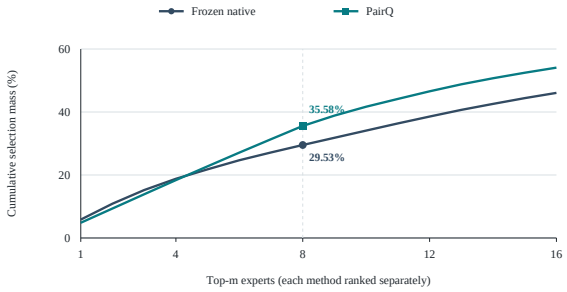


Figure 3: Cumulative selection mass of the m most-used experts on AMC23. Each method is ranked separately by aggregate expert-selection frequency; the values at $m = 8$ match the Top-8 mass in Table 2.