EXPERTS WORK TOGETHER: LEARNING MOE ROUTING FROM COOPERATIVE GAMES

Anonymous authors
Paper under double-blind review

ABSTRACT

MoE models select a small number of experts for each token, and an expert's contribution depends on which experts are activated together. We model MoE expert interactions through cooperative games: under a fixed Top-k constraint, we treat two compatible expert replacement actions as players and define coalition utility as the loss reduction of a complete expert combination relative to natural routing. Marginal contributions and the pairwise interaction identify complementarity and conflict missed by per-action evaluation. We then propose GameMoE, which constructs an interaction-weighted reward from coalition utilities. Router updates account for single-action utility and pairwise interaction, with the interaction scaled by the other replacement's current selection probability. Training combines this coalition supervision with cross-entropy on the natural forward pass and constrains relative routing score changes among experts with respect to the initial router; inference keeps the native Top-k without search or utility evaluation. Theoretical analysis characterizes how loss curvature gives rise to interactions and how they enter routing updates. GameMoE achieves the best greedy-decoding results among the evaluated methods on social commonsense reasoning and both multi-hop and single-paragraph reading comprehension. These results support using cooperative games to guide expert selection and make better use of existing expert capacity under a fixed activation budget.

1 INTRODUCTION

MoE models activate only a subset of experts for each token, leveraging a large pool of parameters within a fixed compute budget (Lepikhin et al., 2021; Fedus et al., 2022; Muennighoff et al., 2025). For task adaptation, prior work improves performance through routing gradient estimation, task-guided set preferences, and selective expert fine-tuning (Liu et al., 2023; Yao et al., 2026; Yoon et al., 2026; Wang et al., 2024), but mainly evaluates the overall effect of experts or expert sets. However, whether an expert can reduce the task loss also depends on which other experts are activated together with it; a training signal for the router must therefore distinguish whether a combination's effect comes from the members' individual contributions or from their complementarity and conflict when acting together.

Cooperative games provide a way to analyze this problem: evaluate coalitions by task performance and examine a participant's marginal contributions to different coalitions (Shapley, 1953). Intervention experiments show that, when experts in the Top-k set chosen by the current router (the natural routing set) are replaced with inactive experts, at least one replacement reduces the answer loss in 76.2% of 256 questions. Two compatible replacements also exhibit complementarity: each replacement alone increases the loss, yet executing both reduces it. When the outgoing expert is selected by single-removal damage, the initial model corrects 5 previously wrong answers among 128 questions (Appendix B.1).

Accordingly, we keep the number of activated experts fixed, treat two replacement actions as players, and define coalition utility as the loss reduction of each of the four coalitions relative to natural routing. The difficulty is that a conventional router scores experts individually: applying softmax to the summed expert scores of the four coalitions yields a distribution that factorizes over the two actions. This distribution cannot express the conditional preferences induced by interactions.

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.

We propose GameMoE, which directly optimizes an expected interaction-weighted reward instead of fitting a coalition distribution. Coalition supervision guides the router to adjust a replacement's selection probability using its single-action utility and pairwise interaction; the interaction's influence depends on the other replacement's current selection probability. An interaction weight further strengthens both complementarity and conflict signals. Coalition utilities are re-measured as the model updates and used to construct coalition supervision. GameMoE constructs these actions through weak-member replacement and uses the Centered Routing Anchor to constrain score changes relative to the initial router, while inference keeps the native Top-k without extra search or utility evaluation. Our main contributions are as follows:

(1) We model complementary and conflicting interactions in MoE expert selection from a cooperative-game perspective, define coalition utility as the loss reduction achieved by an expert combination, characterize how loss curvature gives rise to interactions, and use intervention experiments to identify complementary combinations missed by per-action evaluation.

(2) We propose GameMoE, which turns marginal contributions and interactions from cooperative games into routing supervision. The router learns to select a replacement based on both its individual benefit and its complementarity or conflict with another replacement; inference keeps native routing without search or utility evaluation.

(3) GameMoE improves task adaptation for social commonsense reasoning, multi-hop reading comprehension, and single-paragraph reading comprehension, outperforming the evaluated baselines on multiple metrics. These results support using expert interactions as a source of supervision for router training.

2 Related work

2.1 Sparse routing and task-guided expert selection

GShard and Switch Transformer support large-scale sparse training with Top-2/Top-1 routing, capacity constraints, and load balancing (Lepikhin et al., 2021; Fedus et al., 2022), and BASE Layers and Expert Choice Routing allocate computation through a global linear assignment and by letting experts choose tokens, respectively (Lewis et al., 2021; Zhou et al., 2022). These works control the scale and distribution of expert usage, but do not compare the task loss of different combinations under the same activation count. SparseMixer constructs gradient estimates for sparse routing (Liu et al., 2023), DenseMixer additionally computes all expert outputs during training to improve routing gradients (Yao et al., 2026), and Default MoE uses historical expert outputs to supply information for inactive branches (Panda et al., 2025). In native routing, the scores of unselected experts also receive gradients, but these gradients do not contain their current outputs; we swap an inactive expert in and compute the task loss of the resulting combination, directly evaluating the effect of changing expert membership.

Another line of work uses task performance to guide routing. RoMA supervises the current routing with routing patterns from successful answers on similar inputs (Li et al., 2026), TEXAS increases the cross-entropy (CE) weight when failed instances activate success-related experts (Deng et al., 2026b), and EPO measures the task loss of alternative expert sets and trains the router with set preferences (Yoon et al., 2026). GameMoE further compares a replacement's marginal contribution with and without the other replacement, and uses the interaction in routing updates according to current selection probabilities. AAR and LogitRouter introduce attention features and cross-token context into the routing input, respectively (Kosmopoulou et al., 2026; Rodriguez & Mendoza, 2025); GameMoE keeps the native routing input. RPB builds a prior from the base router to constrain routing changes during adaptation (Lee et al., 2026b); GameMoE uses CRA to constrain relative score changes.

2.2 Task adaptation of MoE

Expert adaptation changes experts' task-specific capabilities. ESFT fine-tunes selected experts (Wang et al., 2024); Router Lens analyzes experts related to context faithfulness, and CEFT performs selective adaptation accordingly (Bai et al., 2025). Building on LoRA (Hu et al., 2022), PERFT studies embedded, routed, and shared adapter modules for MoE (Liu et al., 2026), while IA^3

and Prompt Tuning learn scaling vectors that modulate internal activations and continuous prompts at the input side, respectively (Liu et al., 2022; Lester et al., 2021). Under a limited adaptation budget, ACE, DR-LoRA, MoE^2-LoRA, and MoE-Sieve respectively merge expert adapters by functional grouping, expand their effective rank, share them across layers, and allocate them by routing frequency (Lee et al., 2026a; Deng et al., 2026a; Yang et al., 2026; Manzoni, 2026). Advancing Expert Specialization encourages expert division of labor with orthogonality and variance objectives (Guo et al., 2025). These methods study how adaptation parameters are allocated and organized, as well as training objectives that promote expert specialization, whereas GameMoE focuses on the training signal for the router; because expert adaptation changes expert outputs, coalition utilities are recomputed as the model updates.

2.3 Cooperative games and expert interactions

Cooperative games allow a coalition's value to be higher or lower than the sum of individual values, and can therefore describe complementarity and conflict. The Shapley value averages marginal contributions over joining orders (Shapley, 1953), and has been applied to data valuation (Wang et al., 2025) and reward allocation over parallel reasoning paths (Zhang et al., 2026). SHAPE defines an empirical utility from the activation frequency of expert combinations in recorded routing logs and performs Shapley-style contribution evaluation for expert pruning (Zhang, 2026). Unlike pruning, task-adaptive routing must choose a combination for the current input. Under a fixed Top-k budget, we take replacement actions as players and define coalition utility by the reduction in measured task loss. The expected interaction-weighted reward guides the router in learning whether to select a replacement, with the interaction scaled by the other replacement's current selection probability, rather than using the Shapley allocation as a fixed label.

3 Expert replacements as a cooperative game

3.1 Sparse routing under a fixed activation budget

We call the computation position determined jointly by the input, the prediction position, and the layer a routing position. Suppose the MoE layer under consideration has E experts, indexed by $e, j \in \{1, \dots, E\}$, with input hidden representation $h \in \mathbb{R}^d$. The routing matrix $W_{\mathrm{r}} \in \mathbb{R}^{E \times d}$ produces a score z_e for every expert, and a softmax over all experts gives the mixture weight p_e:

$$z = W_{\mathrm{r}} h, \qquad p_e = \frac{\exp z_e}{\sum_{j=1}^{E} \exp z_j}, \qquad S_0 = \mathrm{Top}_k(z). \tag{1}$$

S_0 contains the indices of the k highest-scoring experts and is called the natural routing set. Let $f_e(h)$ be the feed-forward output of expert e. Given a set S of experts participating in the computation, the mixed output is

$$m(S) = \sum_{e \in S} p_e f_e(h). \tag{2}$$

The natural forward pass takes $S = S_0$. We build on OLMoE-1B-7B-0924: 16 MoE layers, each with $E = 64$, $k = 8$, and $d = 2048$; the native routing normalizes over all experts and then selects Top-8, without renormalizing the selected weights (Muennighoff et al., 2025). The replacements below therefore keep $|S| = k$, with each expert retaining its mixture weight from Eq. (1).

3.2 Compatible replacements and coalition utility

Cooperative games describe the outcome of participants acting together through coalition values. Under fixed Top-k routing, adding one expert requires swapping out a selected one, so we take replacement actions that keep the activation count unchanged as players. A replacement action $a = (i \to e)$ removes expert i from the natural routing set and adds expert e from outside; two replacement actions $a = (i \to e)$ and $b = (j \to f)$ are compatible when $i \neq j$ and $e \neq f$, meaning they can be executed together while keeping the activation count unchanged. The executed actions

form an action coalition T, whose possible values form the coalition set $\mathcal{C}$:

$$\begin{aligned} \mathcal{C} &= \{\varnothing, \{a\}, \{b\}, \{a, b\}\}, \\ S(\varnothing) &= S_0, \\ S(\{a\}) &= S_a = (S_0 \setminus \{i\}) \cup \{e\}, \\ S(\{b\}) &= S_b = (S_0 \setminus \{j\}) \cup \{f\}, \\ S(\{a, b\}) &= S_{ab} = (S_0 \setminus \{i, j\}) \cup \{e, f\}. \end{aligned} \tag{3}$$

Here, $S(T)$ is the expert set obtained after executing the actions in coalition T. Two compatible replacements are the minimal construction that separates individual effects from the joint effect, and the four coalitions cover all ways of executing them. Each coalition corresponds to a complete set of k experts; in our Top-8 setting, the six shared experts form the cooperation context, and the interaction between a swapped-in expert and these members is already part of the coalition utility.

At the prediction position of the current reference token, we fix the model and the reference prefix. Let $F(S)$ be the CE loss on this token after computing the last-layer MoE output with expert set S, i.e., the negative log-likelihood (NLL) of this token (computation in Appendix D.4; conventions in Appendix D.3). We define coalition utility as the loss reduction relative to natural routing:

$$B(T) = F(S_0) - F(S(T)), \qquad B(\varnothing) = 0. \tag{4}$$

$B : \mathcal{C} \to \mathbb{R}$ is the characteristic function of this replacement game; a positive utility means the task loss decreases, and the utility of executing a single action is called the single-action utility.

3.3 Marginal contributions, interaction, and Shapley allocation

A marginal contribution describes the value change a participant brings when joining an existing coalition, and therefore depends on the cooperation context before joining. Let $B_a = B(\{a\})$, $B_b = B(\{b\})$, and $B_{ab} = B(\{a, b\})$ denote the utilities of executing a alone, b alone, and both. When b is not executed, the marginal contribution of a is B_a; when b is already executed, it is $B_{ab} - B_b$. The difference between the two defines the pairwise interaction:

$$J = (B_{ab} - B_b) - B_a = B_{ab} - B_a - B_b. \tag{5}$$

$J > 0$ means executing b strengthens the contribution of a, i.e., complementarity; $J < 0$ means executing b weakens the contribution of a, i.e., conflict; $J = 0$ means the two action utilities are additive.

The Shapley value (Shapley, 1953) averages the marginal contributions under the two joining orders with equal weights:

$$\phi_a = \tfrac{1}{2}\big[B_a + \big(B_{ab} - B_b\big)\big] = B_a + \tfrac{1}{2}J, \qquad \phi_b = B_b + \tfrac{1}{2}J. \tag{6}$$

Each action thus receives its own single-action utility plus half of the interaction, and $\phi_a + \phi_b = B_{ab}$, i.e., the joint utility is fully allocated (the efficiency property of the Shapley value), reflecting each action's average contribution to the joint outcome. Although the output displacements of two replacements add linearly in the expert mixture, the subsequent output normalization and CE can make coalition utility non-additive; Appendix A.1 characterizes the source of the pairwise interaction through the mixed curvature along the two replacement directions. Section 4.2 keeps the utilities of the three non-empty coalitions in training and gives the connection between routing updates and the Shapley allocation.

4 GameMoE: Learning routing from coalition utility

We turn the coalition evaluations in Section 3 into a training signal for the native router. GameMoE constructs weak-member replacements and computes coalition utilities (4.1), updates routing scores with interaction-weighted coalition supervision (4.2), and uses CRA to constrain relative score changes across inputs caused by shared routing parameters (4.3). The training objective combines these terms with CE on the natural forward pass, termed natural CE (4.4). At inference, the adapted router is used directly.

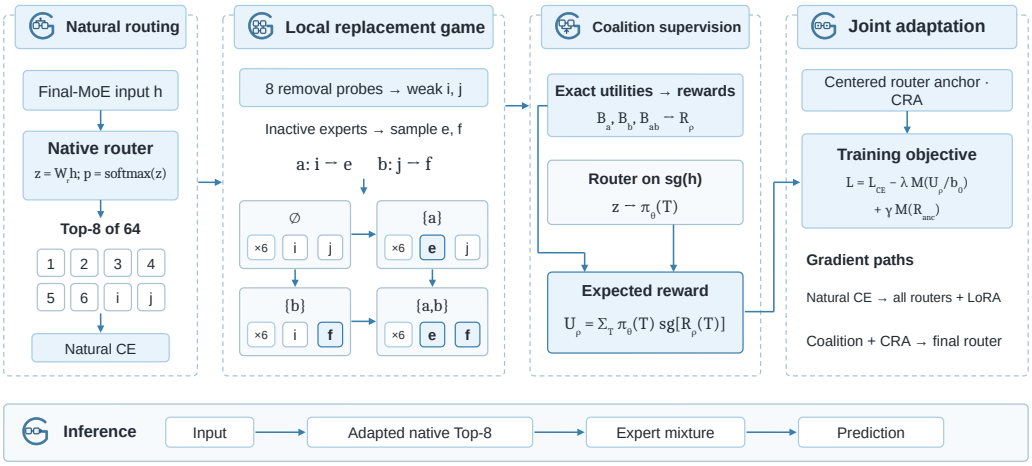

Figure 1: **Overview of GameMoE.** Single and joint weak-member replacements provide coalition utilities for interaction-weighted supervision. Coalition supervision and the CRA update the last-layer router; natural CE updates all routers and LoRA parameters. Inference retains native Top-8 routing without candidate search or utility evaluation.

4.1 Weak-member replacement and coalition evaluation

We construct replacements and evaluate the prediction loss of the current reference token at the last MoE layer. No attention or MoE follows this layer, so the loss of each coalition only requires recomputing the expert mixture, the final normalization, and the vocabulary mapping (Appendix D.4). The nine-document comparison in Appendix B.2 also shows that, at matched prediction positions and within the same single-replacement candidate space, the last layer offers the largest average current-token loss reduction.

Pre-trained routing has already formed input-dependent expert combinations. We start from the natural routing set, preferentially keep the members supported by the task loss, and then explore whether inactive experts can provide a more favorable supplement. To this end, we measure for each selected expert the damage of removing it alone while the other members are kept:

$$D_i = F(S_0 \setminus \{i\}) - F(S_0). \tag{7}$$

The smaller D_i, the less the loss increases after removing expert i. The eight D_i values are each computed relative to the same natural set S_0, and the experts with the two smallest values are selected as the weak members to be swapped out; weak members are determined by the task loss rather than routing scores. Two swap-in experts are then sampled uniformly without replacement from the $E - k = 56$ unselected experts and paired with the two weak members in sampling order, giving actions a, b and the four coalitions of Eq. (3) (Figure 3 in Appendix A). Removal damage provides a task-related exit criterion, while randomly sampling swap-in experts keeps exploration over inactive experts. All coalitions reuse the same hidden representation and all-expert mixture weights (Appendix D.4), and Eq. (4) gives B_a, B_b, B_{ab}; the discrete operation of selecting expert identities does not participate in backpropagation.

4.2 Interaction-weighted coalition supervision

An action's contribution changes with whether the other action is executed, so updating scores by single-action utility alone would miss this dependence; yet the native router scores experts individually and cannot assign an independent score to each cooperation context. To let complementarity and conflict jointly affect routing updates while keeping per-expert scoring, we turn the problem into an expected-reward optimization over the four action coalitions. Concretely, summing the scores of the expert combination $S(T)$ of coalition T gives the coalition score, and normalizing with a fixed

temperature $\tau > 0$ gives the coalition selection probabilities:

$$s_\theta(T) = \sum_{e \in S(T)} z_e, \qquad \pi_\theta(T) = \frac{\exp(s_\theta(T)/\tau)}{\sum_{T' \in \mathcal{C}} \exp(s_\theta(T')/\tau)}. \tag{8}$$

Here, θ denotes the parameters of the last-layer routing matrix; the hidden representation h and the replacement actions are fixed in this computation, and the scores of shared experts cancel in the comparison. π_θ is only used to weight rewards and guide score updates; the natural forward pass still uses the native Top-k and the weights of Eq. (1). Because the scores are additive, π_θ factorizes into independent selection probabilities of the two actions and cannot fully fit the joint preferences induced by interactions, but in terms of expressiveness it can still rank a target coalition first (Appendix A.2).

Let $\rho \geq 0$ be a fixed hyperparameter controlling the interaction weight. Define the interaction-weighted reward $R_\rho(T)$ and the expected interaction-weighted reward U_ρ:

$$R_\rho(T) = \begin{cases} 0, & T = \varnothing, \\ B_a, & T = \{a\}, \\ B_b, & T = \{b\}, \\ B_a + B_b + \rho J, & T = \{a, b\}, \end{cases} \qquad U_\rho = \sum_{T \in \mathcal{C}} \pi_\theta(T)\, \mathrm{sg}[R_\rho(T)]. \tag{9}$$

sg denotes stop-gradient: the reward is treated as a constant in backpropagation, and gradients reach the routing parameters only through the coalition selection probabilities; minimizing $-U_\rho$ constitutes coalition supervision. For the joint replacement, the reward reduces to the sum of single-action utilities when $\rho = 0$ and equals the joint utility B_{ab} when $\rho = 1$; we fix $\rho = 2$, which additionally rewards complementarity and penalizes conflict on top of the joint utility. We add this weight because the interaction's influence on a replacement's selection probability is also scaled by the other replacement's current selection probability. For replacements that are harmful alone but beneficial together, a low probability of selecting the other action can leave the complementarity signal too weak to offset the negative single-action utility.

Let $u = \pi_\theta(\{a\}) + \pi_\theta(\{a, b\})$ and $v = \pi_\theta(\{b\}) + \pi_\theta(\{a, b\})$ be the selection probabilities of actions a, b, and $\xi = z_e - z_i$ the score difference between the swap-in and swap-out experts of action $a = (i \to e)$. Fixing the utilities, temperature, replacement actions, and the score difference of action b, substituting Eq. (8) into Eq. (9) gives

$$\begin{aligned} U_\rho &= uB_a + vB_b + \rho uvJ, \\ \frac{\partial U_\rho}{\partial(\xi/\tau)} &= u(1-u)(B_a + \rho vJ) \\ &= u(1-u)\left[\phi_a + \left(\rho v - \tfrac{1}{2}\right) J\right]. \end{aligned} \tag{10}$$

The sign of $B_a + \rho vJ$ determines whether coalition supervision encourages increasing or decreasing the score difference ξ between the incoming and outgoing experts, thereby influencing the probability of selecting replacement a. The other replacement's selection probability v scales the interaction's influence. When $\rho = 1$, this term is $(1 - v)B_a + v(B_{ab} - B_b)$, i.e., the marginal contributions of a under the two cooperation contexts, weighted by the selection probability of b; the Shapley value averages these two contributions with equal weights and coincides with it at $v = 1/2$. All three coalition utilities thus participate in training: rather than using the Shapley allocation as a fixed label, GameMoE uses marginal contributions according to the current action selection probabilities and adjusts the influence of interactions through ρ (Figure 4 in Appendix A). Appendix A.2 derives this gradient and gives the condition under which the update direction reverses.

4.3 Centered Routing Anchor (CRA)

Routing parameters are shared across inputs, so coalition supervision also changes expert scores on other inputs (Appendix A.3). To constrain the shift of expert selection preferences, we take the frozen last-layer routing matrix before adaptation, W_{r}^0, as a reference and compare the score difference $(W_{\mathrm{r}} - W_{\mathrm{r}}^0)h$ on the same hidden representation h. When all expert scores increase by the same constant, the softmax weights and Top-k membership do not change, so we penalize only

relative score changes among experts: first subtract the mean of the score differences over all E experts, then compute the mean square of the remaining components, which gives the CRA:

$$P = I_E - \frac{\mathbf{1}\mathbf{1}^\top}{E}, \qquad \mathcal{R}_{\mathrm{anc}}(h) = \frac{1}{E}\left\|P(W_{\mathrm{r}} - W_{\mathrm{r}}^0)\,\mathrm{sg}(h)\right\|_2^2. \tag{11}$$

Here, I_E is the E-dimensional identity matrix, $\mathbf{1} \in \mathbb{R}^E$ is the all-ones vector, and the centering matrix P performs the mean subtraction. CRA penalizes relative score shifts while still allowing the activated set to adjust with the task; $\mathrm{sg}(h)$ makes it update only the last-layer routing matrix. Appendix A.3 analyzes the relation between relative score changes and Top-k membership preservation.

4.4 Training objective and native inference

The training objective combines three terms: natural CE $\mathcal{L}_{\mathrm{CE}}$ updates the adaptation parameters and all routers, while coalition supervision and CRA update only the last-layer router. All three are aggregated as averages over all supervised tokens: the reference answer tokens plus the end-of-sequence token (EOS). To control the cost, coalition supervision and CRA uniformly sample at most 8 supervised positions without replacement in each question and weight them by the inverse sampling probability, so as not to change the weights of answers of different lengths; we denote this aggregation by $\mathcal{M}(\cdot)$ (defined in Appendix D.1). Let $\lambda, \gamma \geq 0$ control the strength of the two terms relative to natural CE, and let $b_0 > 0$ be a fixed constant that scales the reward; the full training objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} - \lambda\,\mathcal{M}(U_\rho/b_0) + \gamma\,\mathcal{M}(\mathcal{R}_{\mathrm{anc}}). \tag{12}$$

The negative sign before the coalition supervision term corresponds to maximizing U_ρ. All coefficients are fixed during training (values in Appendix D.1), and the full procedure is given as Algorithm 1 in Appendix D.2. Model updates change hidden representations and expert outputs, so subsequent steps reconstruct the actions and recompute coalition utilities to keep the supervision consistent with the current model. At inference, the adapted router and experts are used directly, keeping the native Top-8 and its weight computation, without constructing action coalitions, computing coalition utilities, or CRA, and without increasing the number of activated experts.

5 Experiments

We test whether coalition supervision improves task adaptation, isolate the role of interaction weighting, and examine combinations missed by per-action evaluation.

5.1 Experimental setup

Models and data. We fine-tune OLMoE-1B-7B-0924 (Muennighoff et al., 2025) separately on SocialIQA (Sap et al., 2019), HotpotQA (Yang et al., 2018), and SQuAD (Rajpurkar et al., 2016). HotpotQA uses the given-context version of Router Lens (Bai et al., 2025). Data splits and sample counts are given in Appendix D.3.

Baselines. We compare with Prompt Tuning (Lester et al., 2021), IA^3 (Liu et al., 2022), DenseMixer (Yao et al., 2026), CEFT-8 (Bai et al., 2025), PERFT-E (Liu et al., 2026), and ACE (Lee et al., 2026a). All methods use the same task-specific data and evaluation protocols, train for one epoch, and retain native Top-8 inference. Baseline adaptation scopes are listed in Appendix D.1.

Training setup. GameMoE freezes the original attention and expert weights, adds LoRA ($r = 8, \alpha = 16$) to these modules in all 16 layers, and directly trains all routers. Natural CE updates all trainable parameters; coalition supervision and CRA update only the last-layer router. We set $\lambda = \gamma = 0.1$ and $\rho = 2$.

Evaluation protocol. Greedy decoding is evaluated by option accuracy (Acc.) on SocialIQA and exact match (EM) and word-level F1 on the QA tasks. With eight sampled answers per question, Mean@8 measures average answer accuracy and Pass@8 measures the fraction answered correctly at least once. Percentages and differences are computed before rounding to one decimal place. Sampling and scoring protocols are in Appendices C and D.3.

Table 1: Task adaptation after one epoch (%; higher is better). Acc. and EM/F1 use greedy decoding; Mean@8 and Pass@8 use eight samples. Bold marks column bests before rounding.

Method	SocialIQA			HotpotQA				SQuAD			
	Acc.	Mean@8	Pass@8	EM	F1	Mean@8	Pass@8	EM	F1	Mean@8	Pass@8
Prompt Tuning	71.0	69.0	86.6	62.4	78.4	61.8	77.1	72.8	82.7	65.8	87.7
IA³	70.5	66.2	89.5	63.0	78.8	63.1	79.5	69.6	76.1	60.3	76.1
DenseMixer	77.2	73.2	91.9	65.4	80.9	64.8	78.4	72.5	75.1	64.0	79.1
CEFT-8	77.9	74.7	92.0	65.8	81.1	64.8	79.4	82.6	90.4	74.1	90.8
PERFT-E	75.7	72.8	91.8	65.6	81.0	64.2	77.0	81.7	89.3	**76.2**	**91.3**
ACE	74.6	70.4	90.8	64.9	80.4	65.0	79.5	82.6	87.3	60.9	80.2
GameMoE	**78.4**	**74.8**	**92.4**	**66.3**	**81.7**	**65.8**	**80.1**	**85.0**	**91.4**	69.0	88.5

(a) SQuAD ablation

Setting	EM	F1
GameMoE	**85.0**	**91.4**
CE only	82.4	89.8
No interaction ($\rho = 0$)	76.3	85.7
No upweighting ($\rho = 1$)	75.4	84.9

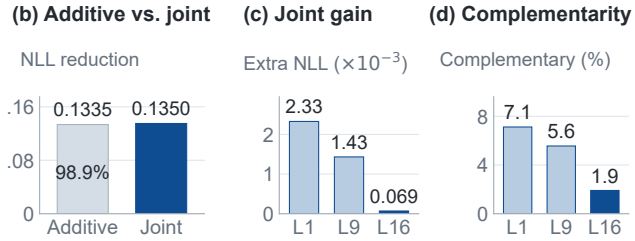

Figure 2: **Training ablations and fixed-model interventions.** (a) SQuAD EM/F1 (%). (b) Additive versus joint selection (12 pairs). (c,d) Extra NLL reduction and frequency of individually harmful but jointly beneficial pairs (36 documents). Interventions use full-answer mean NLL; L1/L9/L16 denote layers 1/9/16 (Appendix E.1).

5.2 Main Results

GameMoE achieves the best greedy-decoding results across the three tasks (Table 1). SocialIQA accuracy reaches 78.4%, exceeding the strongest baseline by 0.5 percentage points. Gains over the strongest baseline for each metric also extend to reading comprehension: 0.5 EM and 0.6 F1 on HotpotQA, and 2.4 EM and 1.0 F1 on SQuAD. These results support using coalition supervision to improve task adaptation while retaining native Top-8 inference.

Under repeated sampling, GameMoE also leads on SocialIQA and HotpotQA, improving both average answer accuracy and coverage over eight attempts. On SQuAD, however, PERFT-E leads on both sampling metrics. Coalition utility measures token loss under reference prefixes, whereas sampling metrics evaluate freely generated answers; improved greedy performance therefore need not imply better sampled answers (Appendix C).

5.3 The Role of Coalition Supervision and Interaction

On SQuAD, we hold the set of trainable parameters, data, and training duration fixed (Figure 2a). The CE-only baseline removes coalition supervision and CRA; $\rho = 0$ removes the interaction term, and $\rho = 1$ uses the original coalition utility without extra interaction weighting. Both ρ variants retain CRA.

GameMoE improves over the CE-only baseline by 2.6 EM and 1.7 F1 points, linking the gain to the added objective rather than adaptation capacity. Removing interaction or its extra weighting lowers EM from 85.0 to 76.3 or 75.4, respectively; both fall below the CE-only baseline (82.4).

These results support retaining interactions and strengthening their contribution to routing supervision. In Eq. (10), single-action utility and $\rho v J$ jointly guide a replacement's selection probability, with v determined by the other replacement's current selection probability. The ablations connect this cooperation-dependent signal to task performance: the gain depends on how coalition utilities enter the training objective.

5.4 Cooperation revealed by coalition evaluation

The ablations test whether interactions are useful for learning. We next examine what joint evaluation reveals by comparing single and joint replacements in a fixed model, at the same input and intervention position (Figure 2b–d). These diagnostics use full-answer mean NLL rather than the current-token loss used in training.

Across 36 documents, 7.1%, 5.6%, and 1.9% of pairs at layers 1, 9, and 16 are individually harmful but jointly beneficial. Such pairs show why a replacement's marginal contribution depends on its cooperation context. Additive selection already recovers 98.9% of joint selection's NLL reduction in the separate 12-pair diagnostic; joint evaluation identifies complementarity beyond these single-action gains.

Earlier-layer replacements yield larger full-answer NLL gains and propagate through subsequent layers. GameMoE evaluates the current-token loss at the last layer, reusing preceding computation; matched-position diagnostics support improvement opportunities there (Appendix B.2). Weak-member selection and generation diagnostics further connect candidate construction to loss reductions and answer changes (Appendix E).

6 Conclusion

We studied MoE expert selection from a cooperative-game perspective, treating complementarity and conflict as information for router training. Coalition utility characterizes the marginal contributions and pairwise interaction of expert replacements; loss curvature explains the source of these interactions, and expert-replacement interventions reveal beneficial combinations missed by per-action evaluation. GameMoE constructs coalition supervision through an expected interaction-weighted reward, guiding the router to select combinations using both the individual and joint effects of replacements. The trained model retains native Top-8 routing, with no search or utility evaluation at inference. Task-adaptation and ablation experiments support this mechanism's role in improving answer quality. Cooperative games thus connect the evaluation of expert combinations to learning how to select them: they explain why experts need to be considered together and provide a way to use those relationships in routing. This offers a training approach for organizing and using existing expert capabilities within a fixed activation budget.

AI use statement

We used AI tools to assist with literature organization, checking arguments and derivations, summarizing experimental materials, and drafting text.

References

Jun Bai, Minghao Tong, Yang Liu, Zixia Jia, and Zilong Zheng. Understanding and leveraging the expert specialization of context faithfulness in mixture-of-experts llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21927–21942. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.emnlp-main.1114/.

Guanzhi Deng, Bo Li, Ronghao Chen, Xiujin Liu, Zhuo Han, Huacan Wang, Lijie Wen, and Linqi Song. Dr-lora: Dynamic rank lora for fine-tuning mixture-of-experts models, 2026a. URL https://arxiv.org/abs/2601.04823.

Guanzhi Deng, Haibo Wang, Kuan Wu, Xiangru Jian, Shing Yin Wong, Sichun Luo, Zhuoran Wang, and Linqi Song. Texas: Task-expert-aware supervision for downstream mixture-of-experts llm adaptation, 2026b. URL https://arxiv.org/abs/2608.06396.

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://jmlr.org/papers/v23/21-0998.html.

Hongcan Guo, Haolang Lu, Guoshun Nan, Bolun Chu, Jialin Zhuang, Yuan Yang, Wenhao Che, Xinye Cao, Sicong Leng, Qimei Cui, and Xudong Jiang. Advancing expert specialization for better moe. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/4598de7d243d528e38eb0c5d8155fb52-Abstract-Conference.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Despoina Kosmopoulou, Anastasios Tsetsilas, Efthymios Georgiou, Giannis Karamanolakis, Swastik Roy, and Alexandros Potamianos. Attention-aware routing: Coupling routing and attention in moes, 2026. URL https://arxiv.org/abs/2609.20974.

Ahin Lee, Sehyun Yun, Joonha Park, and Taesik Gong. Ace: Adapter consolidation across experts for parameter-efficient fine-tuning of moe llms, 2026a. URL https://arxiv.org/abs/2609.06072.

Jaedeok Lee, Keonwoo Kim, Dongyoon Han, Sangdoo Yun, Yera Choi, and Haanju Yoo. Router prior bias: Preserving base routing structure in moe post-training, 2026b. URL https://arxiv.org/abs/2609.08115.

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.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.emnlp-main.243/.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. In *Proceedings of the 38th International Conference on Machine Learning, PMLR 139*, pp. 6265–6274, 2021. URL https://proceedings.mlr.press/v139/lewis21a.html.

Zhongyang Li, Ziyue Li, and Tianyi Zhou. Routing manifold alignment improves generalization of mixture-of-experts llms. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3lskwxB653.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 1950–1965, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/0cde695b83bd186c1fd456302888454c-Abstract-Conference.html.

Liyuan Liu, Jianfeng Gao, and Weizhu Chen. Sparse backpropagation for moe training. In *NeurIPS 2023 Workshop on Advancing Neural Network Training (WANT)*, 2023. URL https://arxiv.org/abs/2310.00811.

Yilun Liu, Yunpu Ma, Yuetian Lu, Shuo Chen, Zifeng Ding, and Volker Tresp. Parameter-efficient routed fine-tuning: Mixture-of-experts demands mixture of adaptation modules. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 4439–4457, 2026. doi: 10.18653/v1/2026.findings-eacl.232. URL https://aclanthology.org/2026.findings-eacl.232/.

Andrea Manzoni. Moe-sieve: Routing-guided lora for efficient moe fine-tuning, 2026. URL https://arxiv.org/abs/2603.24044.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. Olmoe: Open mixture-of-experts language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/9b224ace8963c9385ad5e2b5c9039b97-Abstract-Conference.html.

Ashwinee Panda, Vatsal Baherwani, Zain Sarwar, Benjamin Therien, Sambit Sahu, Tom Goldstein, and Supriyo Chakraborty. Dense backpropagation improves training for sparse mixture-of-experts. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=jTUIo4Kl6u.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392. Association for Computational Linguistics, 2016. URL https://aclanthology.org/D16-1264/.

Yi Ren and Danica J. Sutherland. Learning dynamics of llm finetuning. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2407.10490.

Felipe Rodriguez and Marcelo Mendoza. Logitrouter: a novel attention variant for reducing myopic routing in mixture of experts. In *Proceedings of the 18th International Natural Language Generation Conference*, pp. 499–510, 2025. URL https://aclanthology.org/2025.inlg-main.30/.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 4463–4473. Association for Computational Linguistics, 2019. URL https://aclanthology.org/D19-1454/.

L. S. Shapley. A value for n-person games. In *Contributions to the Theory of Games (AM-28), Volume II*, pp. 307–317. Princeton University Press, 1953. doi: 10.1515/9781400881970-018. URL https://doi.org/10.1515/9781400881970-018.

Jiachen T. Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2406.11011.

Zihan Wang, Deli Chen, Damai Dai, Runxin Xu, Zhuoshu Li, and Yu Wu. Let the expert stick to his last: Expert-specialized fine-tuning for sparse architectural large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 784–801. Association for Computational Linguistics, 2024. URL https://aclanthology.org/2024.emnlp-main.46/.

Qingyu Yang, Haonan He, Minglei Li, Jingqi Ye, Tao Chen, Lei Bai, and Peng Ye. Moe^2-lora: When moe models meet moe-style low-rank adaptation, 2026. URL https://arxiv.org/abs/2607.21978.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380. Association for Computational Linguistics, 2018. URL https://aclanthology.org/D18-1259/.

Feng Yao, Junxia Cui, Ruohan Zhang, Liyuan Liu, Shibo Hao, Li Zhang, Chengyu Dong, Churan Zhi, Shuohang Wang, Yelong Shen, Jianfeng Gao, and Jingbo Shang. Densemixer: Improving moe post-training via precise router gradient. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2026. To appear.

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

Wentao Zhang, Haoyu Zhang, Xinke Jiang, Yuxuan Cheng, Yuhan Pan, Miao Li, Zhipeng Qiao, Tao Feng, Zhen Tao, and Dengji Zhao. Fishing out free riders: Shapley-based reward attribution for parallel reasoning via reinforcement learning, 2026. URL https://arxiv.org/abs/2607.18979.

Yuhao Zhang. Shape: Coalition-aware expert pruning for sparse mixture-of-experts llms, 2026. URL https://arxiv.org/abs/2606.09886.

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://proceedings.neurips.cc/paper_files/paper/2022/hash/2f00ecd787b432c1d36f3de9800728eb-Abstract-Conference.html.

A Additional method figures and theoretical analysis

This appendix illustrates weak-member replacement and interaction-weighted coalition supervision (Figures 3 and 4), explains how loss curvature gives rise to pairwise interactions, derives the routing gradient of the interaction-weighted reward and its connection to the Shapley allocation, and analyzes shared routing updates and CRA.

A.1 Native routing gradients and curvature interaction

Let $\mathbf{1}\{\cdot\}$ be the indicator function that equals 1 when the condition holds and 0 otherwise, and let $m = m(S_0)$ denote the natural mixed output. In a region where the Top-k membership is fixed, $\partial p_e/\partial z_j = p_e(\mathbf{1}\{e = j\} - p_j)$ and Eq. (2) give

$$\frac{\partial m}{\partial z_j} = p_j\big[\mathbf{1}\{j \in S_0\} f_j(h) - m\big]. \tag{A.1}$$

The derivative for an unselected expert j is $-p_j m$, which is generally nonzero but does not contain its not-yet-executed output $f_j(h)$. Only after this expert is swapped in and its output is computed can its actual effect on the mixed output and the task loss be measured.

The linear mixture of expert outputs and the additivity of coalition utility are two different properties. Fixing the hidden representation and the all-expert mixture weights at this routing position, the output displacements of two compatible replacements are $d_a = p_e f_e(h) - p_i f_i(h)$ and $d_b = p_f f_f(h) - p_j f_j(h)$, and the joint displacement equals $d_a + d_b$. The replaced mixed output is added to the residual branch and then passes through normalization, vocabulary prediction, and CE; this nonlinear mapping makes the joint utility deviate from the sum of the two single-action utilities. Below we write this deviation as an integral of the loss curvature.

Proposition 1 (Curvature representation of interaction). A fixed computation graph means that all computations use the same operations and discrete routing branches. Let $m_0 = m(S_0)$ be the natural mixed output, and let $\mu, \nu \in [0, 1]$ be the interpolation coefficients along the two replacement displacements. If the real-valued loss F on this graph is twice continuously differentiable in a neighborhood of $\{m_0 + \mu d_a + \nu d_b : \mu, \nu \in [0, 1]\}$, then

$$J = -\int_0^1 \int_0^1 d_a^\top \nabla^2 F(m_0 + \mu d_a + \nu d_b) d_b \, d\mu \, d\nu. \tag{A.2}$$

To prove Eq. (A.2), let $\Psi(\mu, \nu) = F(m_0 + \mu d_a + \nu d_b)$. Applying the fundamental theorem of calculus twice gives

$$F(m_0 + d_a + d_b) - F(m_0 + d_a) - F(m_0 + d_b) + F(m_0) = \int_0^1 \int_0^1 \partial_{\mu\nu} \Psi(\mu, \nu) \, d\mu \, d\nu. \tag{A.3}$$

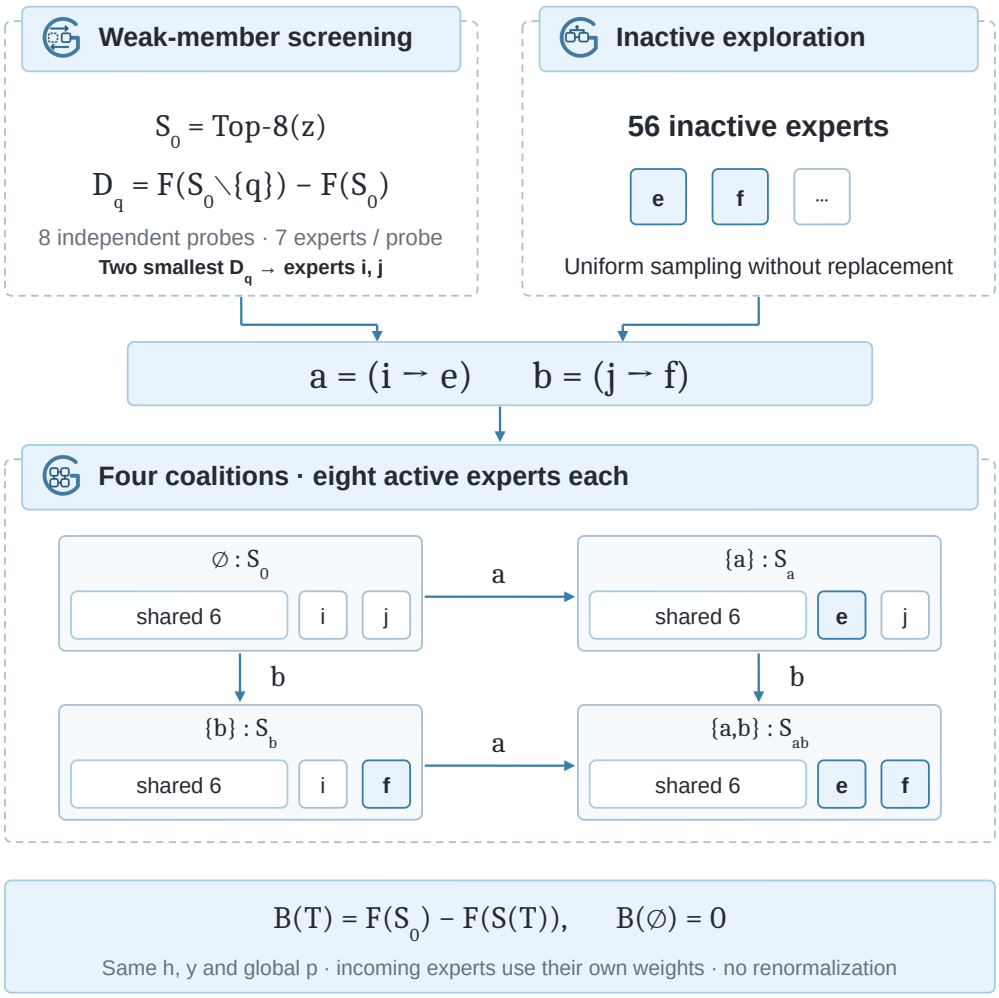

Figure 3: **Weak-member replacement.** Two members with the smallest single-removal damage on the natural set are paired with distinct inactive experts. Each coalition corresponds to an eight-expert combination, with six experts shared across all four combinations; their current-token losses define coalition utilities. Seven-expert removal probes and eight-expert replacement evaluations reuse the same global mixture weights.

Taking the negative sign and substituting $\partial_{\mu\nu}\Psi = d_a^\top \nabla^2 F(m_0 + \mu d_a + \nu d_b) d_b$ yields Eq. (A.2). If the operator norm of the Hessian (its largest amplification of a unit vector) is at most L, then $|J| \le L\|d_a\|\|d_b\|$.

As a geometric example, take $F(m) = m^\top H m/2$ with $H = \left[\begin{smallmatrix} 1 & c \\ c & 1 \end{smallmatrix}\right]$, $0 < |c| < 1$, $d_a = (1, 0)$, and $d_b = (0, 1)$. The two displacements are orthogonal, yet the interaction is $J = -c$. If the vocabulary logits are an affine map $Am + c_0$, where A is a linear map matrix and c_0 a bias vector, the Hessian of the CE with respect to the mixed output is $A^\top[\mathrm{diag}(p_{\mathrm{vocab}}) - p_{\mathrm{vocab}} p_{\mathrm{vocab}}^\top]A$. Here p_{vocab} is the vocabulary prediction probability vector, and diag places a vector on the diagonal of a diagonal matrix. When RMSNorm, which rescales the hidden vector by its root mean square, is included, the full Hessian also contains the second-order term of this mapping itself.

The mixed curvature in Eq. (A.2) describes how the rate of change of the loss along one replacement direction changes when the output moves along the other. Let $m_0 = m(S_0)$ and $H = \nabla^2 F(m_0)$; replacing the curvature over the region with the Hessian at the natural output gives the approximation

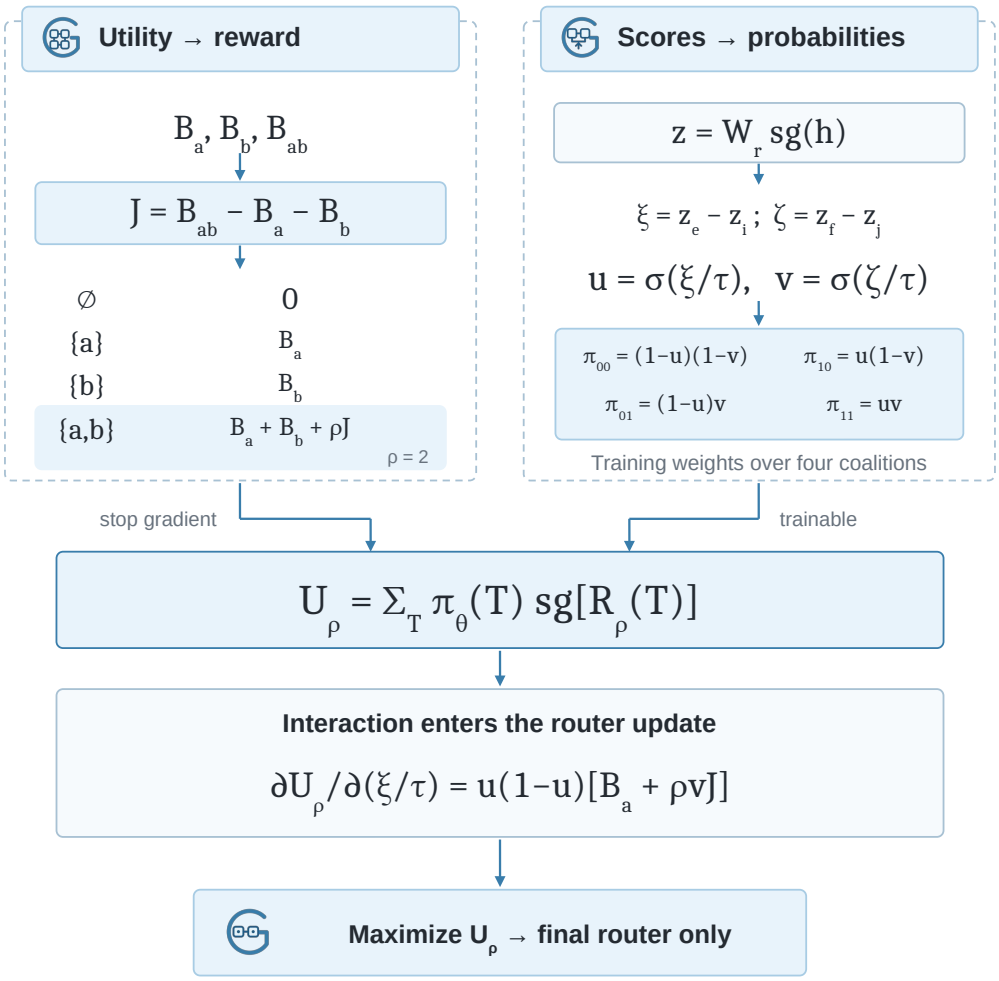

Figure 4: **Interaction-weighted coalition supervision.** Coalition utilities define rewards, and router scores define coalition selection probabilities. Maximizing the expected reward updates the last-layer router through $B_a + \rho v J$, where v is the other action's selection probability. Gradients are stopped through the rewards and hidden representations; coalition probabilities do not alter the natural forward route.

$\widetilde{J} = -d_a^\top H d_b$ of the pairwise interaction. This approximation serves to explain where interactions come from; in training, GameMoE computes the pairwise interaction by Eq. (5) from the coalition utilities of single and joint replacements. The orthogonal-displacement example above shows that the interaction is determined jointly by the output changes and the task loss, and cannot be judged from the similarity of expert outputs alone.

This result concerns a real-valued smooth loss on a fixed computation graph. The main method recomputes normalization and vocabulary prediction at the last layer; if an early- or middle-layer intervention changes subsequent discrete routing, it needs to be analyzed on the corresponding piecewise computation graph.

As a numerical example of utilities, when $B_a = 0.2$, $B_b = 0.3$, and $B_{ab} = 0.4$, we have $J = -0.1$. The two actions interact negatively, yet the joint replacement is still better than natural routing and either single replacement. Whether the joint replacement reduces the task loss is checked by $B_{ab} = B_a + B_b + J > 0$; whether it is strictly better than the other three combinations requires $B_{ab} > \max(0, B_a, B_b)$.

A.2 PRODUCT DISTRIBUTIONS AND INTERACTION-DEPENDENT UPDATES

Coalition utilities can contain interactions, but the coalition scores in Eq. (8) are still obtained by summing per-expert scores. Let $\xi = z_e - z_i$ and $\zeta = z_f - z_j$ denote the score changes caused by the two replacements. In the order of no replacement, only a, only b, and both, subtracting the common natural score gives $(0, \xi, \zeta, \xi + \zeta)$. Denote the corresponding probabilities by $\pi_{00}, \pi_{10}, \pi_{01}, \pi_{11}$, where the subscripts indicate whether each action is executed.

Proposition 2 (Product-family constraint). Let $\sigma(x) = 1/(1+e^{-x})$ be the sigmoid function. The selection probabilities of actions a, b induced by Eq. (8) are $u = \sigma(\xi/\tau)$ and $v = \sigma(\zeta/\tau)$; these probabilities do not condition on whether the other action is selected. The probabilities of the four coalitions satisfy

$$\pi_{00} = (1-u)(1-v), \quad \pi_{10} = u(1-v), \quad \pi_{01} = (1-u)v, \quad \pi_{11} = uv. \tag{A.4}$$

Eq. (A.4) shows that the probabilities of the four action coalitions equal products of two independent Bernoulli action probabilities, so $\pi_{00}\pi_{11} = \pi_{10}\pi_{01}$.

After removing the common natural score, the softmax denominator (partition function) of Eq. (8) factorizes as

$$1 + e^{\xi/\tau} + e^{\zeta/\tau} + e^{(\xi+\zeta)/\tau} = (1 + e^{\xi/\tau})(1 + e^{\zeta/\tau}), \tag{A.5}$$

which gives the product form of Eq. (A.4). This parameterization has no independent interaction score; GameMoE uses joint evaluation for routing updates through the interaction-weighted reward.

To examine whether this parameterization can fit joint preferences, we convert the measured coalition utilities with a temperature $\tau_{\mathrm{B}} > 0$ into a target distribution for fitting analysis, called the teacher distribution $Q(T) \propto \exp(B(T)/\tau_{\mathrm{B}})$, where $Q_{00}, Q_{10}, Q_{01}, Q_{11}$ correspond to no replacement, only a, only b, and the joint replacement. The log odds ratio measuring the dependence between the two actions satisfies $\log(Q_{00}Q_{11}/Q_{10}Q_{01}) = J/\tau_{\mathrm{B}}$: when $J \neq 0$, the teacher distribution Q does not satisfy the product identity above. Q is used here only to analyze distribution fitting; the training objective of GameMoE directly uses the reward of Eq. (9).

We measure the difference between the teacher distribution and the coalition selection probabilities with the KL divergence. Minimizing $\mathrm{KL}(Q\|\pi)$ is equivalent to matching the selection probabilities of the two actions separately, giving $u = Q_{10} + Q_{11}$ and $v = Q_{01} + Q_{11}$. This projection may change which coalition ranks first.

In the order of no replacement, the two single replacements, and the joint replacement, take synthetic utilities $(0, -0.01, -10, 0.1)$ and $\tau_{\mathrm{B}} = 1$, which give $Q \approx (0.323074, 0.319859, 0.0000147, 0.357052)$. The two marginals are $u \approx 0.676911$ and $v \approx 0.357067$, so the largest probability among the four coalitions falls on the single replacement a (state 10), whose utility is -0.01, whereas the joint replacement has the highest utility, 0.1.

Distribution fitting and combination selection have different goals: additive scores cannot fully represent the joint distribution induced by interactions, but can still rank a given target coalition first. GameMoE therefore keeps per-expert scoring and directly maximizes the expected interaction-weighted reward. Substituting Eq. (A.4) into Eq. (9) and collecting the single-action terms gives $U_\rho = B_a u + B_b v + \rho J uv$. Fixing the coalition utilities, temperature, and replacement actions, and keeping ζ/τ unchanged, the partial derivative with respect to ξ/τ gives Eq. (10) in the main text.

In Eq. (10), $u(1-u) > 0$, so the sign of $B_a + \rho J v$ decides whether maximizing the reward raises or lowers the selection probability of replacement action a. The single-action utility gives an independent term, and the other action's current probability carries the interaction term into the update. For $B_a < 0$, $J > 0$, and $\rho > 0$, this direction is positive when $v > -B_a/(\rho J)$; if this threshold is at least 1, no v under finite routing scores can satisfy the condition. The conclusion for b is symmetric.

When $J < 0$ and $\rho > 0$, $\rho J v = -\rho|J|v$ contributes a negative gradient component for action a. With other quantities fixed, the higher the other action's probability v, the stronger this suppression. If $B_a > 0$, the total gradient turns to lowering the selection probability of a when $v > B_a/(\rho|J|)$; if this threshold is at least 1, the reversal does not occur within the reachable probability range. A negative interaction can thus weaken the tendency to jointly select an action that is beneficial alone, while whether it changes the overall update direction still depends on the single-action utility.

From the game perspective, when $\rho = 1$, the term $B_a + Jv = (1 - v)B_a + v(B_{ab} - B_b)$ in Eq. (10) is exactly the marginal contribution weighted by the current selection probability of b. The Shapley value averages the two cooperation contexts with equal weights, and the two coincide at $v = 1/2$. Both use the marginal contributions under these two cooperation contexts: the Shapley allocation summarizes an action's average effect, whereas the routing gradient adjusts the weights of the cooperation contexts according to the current policy.

The other action's probability v carries the interaction into the update: increasing ρ strengthens the effect of complementarity and also strengthens the suppression from negative interactions; the overall update direction is still determined jointly by the single-action utility and the weighted interaction.

When $B_a < 0$, $J > 0$, and $\rho > 0$, increasing ρ lowers the threshold on v required for the positive update above. This explains how interaction weighting supports complementary replacements whose individual utilities are negative. The distribution still belongs to the product family of Eq. (A.4); what changes is the training criterion, which lets the interaction act through coupled reward gradients.

This update can also be written as a weighting of cooperation-context-dependent marginal contributions:

$$B_a + \rho v J = (1 - v)B_a + v(B_{ab} - B_b) + (\rho - 1)vJ.$$

The first two terms correspond to the marginal contributions when the other action is not executed and when it is executed, and the last term is the additional interaction weighting. Using Eq. (6), the full objective can also be written as

$$U_\rho = u\phi_a + v\phi_b + J\left(\rho uv - \frac{u + v}{2}\right).$$

Directly optimizing $u\phi_a + v\phi_b$ therefore differs from the GameMoE objective: the latter keeps the contribution of the interaction that varies with the current action probabilities.

A.3 Shared updates and CRA

Section 4.3 uses CRA to constrain changes in relative routing scores, motivated by the fact that coalition supervision affects other inputs through shared parameters; such cross-input learning dynamics have been characterized in fine-tuning research (Ren & Sutherland, 2025). To analyze this propagation concretely, we first consider a single SGD update with fixed hidden representations. Let $\mathcal{L}_{\mathrm{loc}}(z)$ be a scalar routing training loss on the current input, e.g., the coalition supervision term formed by $-U_\rho$; let $g_z = \nabla_z \mathcal{L}_{\mathrm{loc}}$ and let $\eta > 0$ be the step size. By the chain rule, $\nabla_{W_{\mathrm{r}}} \mathcal{L}_{\mathrm{loc}} = g_z h^\top$, so minimizing this loss changes the routing matrix by $\delta W_{\mathrm{r}} = -\eta g_z h^\top$ in one step. For the fixed hidden representation h' of another input, let $\Delta z(h') = \delta W_{\mathrm{r}} h'$ be the score change caused by this update; then

$$\Delta z(h') = -\eta g_z \langle h, h' \rangle. \tag{A.6}$$

Eq. (A.6) shows that the inner product between the features of the two inputs determines the strength and sign with which this update propagates to the other input. A positive inner product propagates the score change in the same direction, and a zero inner product makes the one-step effect on this fixed feature vanish; the final task effect also depends on whether this direction benefits the other input.

Shared updates change scores on other inputs, so CRA constrains relative score changes with the initial router as the reference. Below we write the anchor as a parameter constraint under the current feature distribution, and then discuss how this constraint relates to Top-k membership preservation. Let $\Delta W = W_{\mathrm{r}} - W_{\mathrm{r}}^0$ denote the cumulative change relative to the initial router, as distinct from the single-step change δW_{r} in the derivation of Eq. (A.6), and let $\Sigma_h = \mathbb{E}[hh^\top]$ be the second moment of the hidden representations at the selected routing positions. Taking the expectation over the representation distribution at these positions gives

$$\mathbb{E}[\mathcal{R}_{\mathrm{anc}}(h)] = \frac{1}{E} \operatorname{tr}(P \Delta W \Sigma_h \Delta W^\top P), \tag{A.7}$$

where tr denotes the matrix trace. Eq. (A.7) shows that CRA penalizes changes in relative routing scores according to the current feature distribution: directions with a larger feature second moment

receive larger constraint weights. Letting $M = P\Delta W$, the identity follows from $\mathbb{E}\|Mh\|^2 = \mathrm{tr}(M\Sigma_h M^\top)$.

Relative scores also affect membership selection through the Top-k boundary. On the same fixed input, let the gap between the k-th and $(k+1)$-th scores of the reference router be $d_{\mathrm{gap}} > 0$. The change in any inside-minus-outside score difference is at most twice the largest absolute value of the centered perturbation, so keeping this largest absolute value below $d_{\mathrm{gap}}/2$ suffices to preserve the Top-k membership. CRA softly constrains this perturbation through a mean-square penalty; the per-input boundary condition characterizes when the set is actually preserved.

B Additional mechanism measurements

B.1 Motivating intervention protocols

The two interventions summarized in the Introduction both use the initial OLMoE. The first, when predicting the first answer token, randomly selects an activated expert at the last layer and replaces it with each of four inactive experts, comparing the reference-answer content NLL of each replaced complete combination with that of natural routing; in 76.2% of 256 questions, at least one replacement reduces the loss. The second first selects the expert with the smallest single-removal damage, then chooses among four swap-in options the one with the lowest reference-answer loss, applies the replacement at the same position, and generates freely; answers to 5 of the 128 questions change from incorrect to correct. The complementarity between two compatible replacements is further shown in Section 5.4 and Appendix E.1.

B.2 Current-token loss reduction across layers

On nine held-out documents, we examine layers 1, 9, and 16 at the same prediction position of each document. At each layer, we enumerate the 448 single-expert replacements between the eight selected experts and the 56 unselected experts, select the best replacement within the candidates by the NLL of the current predicted token, and average its loss reduction across documents. This comparison directly corresponds to the prediction target that GameMoE uses to define coalition utility.

Table 2: Mean maximum current-token NLL reduction over candidate replacements at each examined layer.

Layer	Mean maximum NLL reduction ↑
Layer 1	0.08450
Layer 9	0.11632
Layer 16	0.26819

In this candidate comparison with the same inputs and prediction positions, the last layer offers the largest average room for current-token improvement. Together with the structural property that no attention or MoE follows the last layer, so that candidate losses can reuse the preceding computation, this result supports the choice of layer in Section 4.1.

C Repeated-sampling evaluation

Repeated sampling is performed on each training source of the QA and commonsense tasks: one question is taken from each of 512 parent-question groups, and 8 answers are sampled independently per question with temperature 0.7 and top-p 0.95. Let $c_{ij} \in \{0, 1\}$ be the correctness of the j-th answer to question i: using the same answer normalization and scoring rules as greedy evaluation, it equals 1 when the generated answer exactly matches any reference answer on the QA tasks or when the selected option is correct on SocialIQA, and 0 otherwise. The two metrics are

$$\mathrm{Mean@8} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{8}\sum_{j=1}^{8} c_{ij}, \qquad \mathrm{Pass@8} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left\{\sum_{j=1}^{8} c_{ij} > 0\right\}, \tag{C.1}$$

where $n = 512$ and $\mathbf{1}\{\cdot\}$ is the indicator function. Mean@8 is the average accuracy over all answers, and Pass@8 is the fraction of questions answered correctly at least once in 8 attempts; greedy EM and accuracy measure the single answer of deterministic decoding. Repeated-sampling results for all methods are given in Table 1.

GameMoE reaches 74.8% Mean@8 and 92.4% Pass@8 on SocialIQA, and 65.8% and 80.1% on HotpotQA. On SQuAD, its 69.0% and 88.5% trail PERFT-E's 76.2% and 91.3%, despite higher greedy EM and F1. Greedy decoding selects the most probable token at each step, while sampling draws answers from the generation distribution. Coalition utility is defined by token loss under reference prefixes, so gains in this training signal need not improve both decoding regimes equally. These aggregate metrics do not establish how the generation distribution changed.

D Implementation and training procedure

D.1 Training and evaluation configuration

The expert feed-forward function f_e consists of three projections: gate, up, and down. The gate projection is a different module from the router that selects experts:

$$f_e(h) = W_{e,\text{down}}\left[\text{SiLU}(W_{e,\text{gate}}h) \odot W_{e,\text{up}}h\right]. \tag{D.1}$$

Here the three W are linear projection matrices inside the expert, $\odot$ denotes element-wise multiplication, and SiLU is the element-wise activation $\text{SiLU}(x) = x\sigma(x)$ with $\sigma(x) = 1/(1+e^{-x})$. The gate projection provides input-dependent feature gating, the up projection maps to the expert's internal dimension, and the down projection maps the result back to the hidden dimension. GameMoE freezes the original projection weights and uses LoRA (Hu et al., 2022) to update the Q/K/V/O projections of attention in all 16 layers and the gate/up/down projections of all experts. For any adapted projection matrix W, the effective weight is

$$W^{\text{eff}} = W^0 + \frac{\alpha}{r} V_W U_W. \tag{D.2}$$

During training, all three expert projections in Eq. (D.1) use the effective weight of Eq. (D.2), and the Q/K/V/O projections of attention use the same parameterization. W^0 is the frozen original weight, U_W, V_W are trainable low-rank factors, r is the rank, and α/r controls the update scale. If the input and output dimensions of W are d_{in} and d_{out}, then $U_W \in \mathbb{R}^{r\times d_{\text{in}}}$ and $V_W \in \mathbb{R}^{d_{\text{out}}\times r}$. All routing matrices are updated directly by natural CE without low-rank parameterization.

Both attention and expert adapters use $r = 8$ and $\alpha = 16$ without dropout; this configuration is denoted QKVO-A-r8. U_W uses Kaiming initialization and V_W is initialized to zero, so adaptation starts from the original model. The fixed reward scale in Eq. (12) is $b_0 = 0.09755275751847554$, the coefficient of natural CE is 1, and the coefficients of coalition supervision and CRA are $\lambda = 0.1$ and $\gamma = 0.1$; the interaction weight is $\rho = 2$, and the temperature of the coalition selection probabilities is $\tau = 1$. These coefficients are fixed during training.

The aggregation operator $\mathcal{M}(\cdot)$ in Eq. (12) is defined as follows. Let the set of questions in one optimizer update (the effective batch) be $\mathcal{B}$, and let n_x be the number of supervised positions of question x (the reference answer tokens plus EOS). Natural CE is computed at all positions; coalition supervision and CRA uniformly sample $K_x = \min(8, n_x)$ positions without replacement in each question, with index set $\mathcal{T}_x$, and the coalition losses of Appendix D.4 are computed only at these positions. Let q_{xt} be the scalar to aggregate at question x, position t, and define

$$\mathcal{M}(q) = \frac{\sum_{x\in\mathcal{B}} (n_x/K_x) \sum_{t\in\mathcal{T}_x} q_{xt}}{\sum_{x\in\mathcal{B}} n_x}. \tag{D.3}$$

For the coalition supervision term, q_{xt} is U_ρ/b_0 at that position; for the CRA term, it is $\mathcal{R}_{\text{anc}}$.

The position-sampling weights in Eq. (D.3) are based on the following consideration: after position sampling, the sampled positions cannot be averaged directly, which would change the weights of answers of different lengths. For example, if two questions of different lengths, both with more than eight supervised positions, each have eight positions sampled, a direct average gives the two questions equal weight, whereas when averaging over all tokens the longer answer should carry

more weight. Each position is sampled with probability K_x/n_x, so the sampled values are weighted by its inverse n_x/K_x.

One optimizer update uses an effective batch of 64 questions, implemented through micro-batches and gradient accumulation. The input window is capped at 4096 tokens, the model forward pass uses BF16, and key probabilities, cross-entropies, and utilities are computed in FP32. The learning rate is 6×10^{-5}, Adam uses $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, and weight decay is zero; the gradients of all losses are accumulated, clipped jointly, and applied in one update.

Each training set is traversed once, and we report the model saved after one epoch: 313 updates for SQuAD, 1,083 for HotpotQA, and 523 for SocialIQA.

Table 3: Parameter update scopes for the compared methods.

Method	Update scope
Prompt Tuning	Continuous prompts at the input
IA^3	Scaling vectors in attention and expert feed-forward modules
DenseMixer	Routers and expert adaptation parameters; sparse forward with dense routing gradients
CEFT-8	Experts selected by context faithfulness
PERFT-E	Embedded adapter modules in experts
ACE	Consolidated group-shared adapters
GameMoE	All routers; rank-8 adapters on Q/K/V/O in all layers and on expert gate/up/down

These methods differ in adaptation scope; ablations with the same trainable parameters are reported in Section 5.3. DenseMixer retains the authors' sparse forward pass and dense routing gradients. CEFT-8 and PERFT-E are configurations of the cited CEFT and PERFT methods, while IA^3 and Prompt Tuning update scaling vectors and continuous prompts, respectively. All methods use the same backbone, task-specific data, and one-epoch training budget.

All methods use the same prompts, generation length limits, and scoring rules within a task; each task follows its own evaluation protocol.

D.2 Training algorithm

Algorithm 1 GameMoE Training Step

1: **Input:** an effective batch, the current model, the frozen initial last-layer routing matrix, and the random state for position and action sampling.
2: Run the natural forward pass and compute the CE of the answer/EOS; sample from the supervised positions of the reference answer and EOS, and cache the hidden representations, residual-branch hidden representations, and routing scores at the corresponding last-layer routing positions.
3: At each routing position, compute the loss change after removing each of the eight selected experts and determine the two experts to swap out; sample two distinct unselected experts to pair with them, and build the four action coalitions and their expert combinations.
4: Reusing the current hidden representation, residual, and all-expert mixture weights, compute the losses of the three non-empty coalitions by Eq. (D.4) and obtain the coalition utilities by Eq. (4).
5: Compute the pairwise interaction by Eq. (5), then construct the stop-gradient interaction-weighted rewards by Eq. (9).
6: On the stop-gradient hidden representations, compute the differentiable selection probabilities of the four coalitions and the CRA through the current router.
7: Aggregate coalition supervision and CRA by Eqs. (D.3) and (12), accumulate their gradients together with natural CE, clip them jointly, and perform one optimizer update.

D.3 Data and diagnostic protocols

Combination interventions use independent source paragraphs of SQuAD 1.1 (Rajpurkar et al., 2016): 16 paragraphs for engineering calibration and 256 for diagnostics, with one question per paragraph; the weak-member analysis uses a predefined subset of 128 questions. These interventions use the initial OLMoE without task adaptation. Training, calibration, and diagnostics use their own predefined data splits, and diagnostic candidates are not selected by main test results.

Combination search compares random removal, lowest-routing-score removal, and weak-member removal. The activation count is the same before and after replacement, and the natural Top-8 is the no-replacement candidate. The oracle evaluates each candidate combination using the reference-answer loss and selects the lowest-loss candidate, measuring the improvement opportunity within this candidate space. Comparisons fix the model parameters, input, reference prefix, intervention layer, and position; after early- or middle-layer interventions, the subsequent network continues to run. Generation diagnostics apply the selected replacement and then generate freely, whereas main-task testing directly uses the adapted native routing.

The three main training sources use their own fixed training, validation, and test question lists. We randomly sample 20,000 examples from the SQuAD training split (Rajpurkar et al., 2016) for training. Because its official test set is not publicly available, we randomly sample 1,000 and 2,000 examples from its validation split for our validation and test sets, respectively; the main results and ablations share this test set. HotpotQA (Yang et al., 2018) uses the given-context QA version of Router Lens (Bai et al., 2025), with 69,274 training and 5,901 test questions, and is not evaluated as a full retrieval system. SocialIQA (Sap et al., 2019) uses 33,410 training and 2,224 test questions in our answer-supervision format.

When computing the task loss, the input text and the reference answer tokens before the prediction position serve as the reference prefix; it is the text condition visible to the model, whereas the hidden representation is the internal representation the model computes from it. Let the current reference token be y with predicted probability $P(y \mid \text{prefix})$; its CE is $-\log P(y \mid \text{prefix})$, i.e., the NLL of this token, and a sequence NLL sums or averages over a specified token range. Main training applies next-token CE supervision to the answer and EOS.

NLL evaluation distinguishes the supervision scope from the aggregation. Content NLL excludes EOS, while full-answer NLL includes it; question-averaged NLL first averages the loss within each question and then across questions, while token-weighted NLL divides the total loss by the number of all supervised tokens. Greedy decoding picks the most probable token at each step. EM checks, after answer normalization, whether the answer exactly matches any reference answer, and F1 computes word-level overlap under the same normalization. Generation results of combination interventions are paired per question, recording fixes (wrong to correct), harms (correct to wrong), and changes in answer strings. Compute cost is measured as elapsed wall-clock time.

Main training computes natural CE on all reference answer tokens and EOS. Coalition supervision and CRA uniformly sample at most eight of these target positions without replacement per question; when the target token has sequence index j, its prediction is evaluated with the last-layer representation at position $j-1$. EOS is sampled together with the other target positions, so for long answers it is not necessarily sampled at every step. The definitions and results of repeated sampling are given in Appendix C.

D.4 Last-layer coalition evaluation

This section gives the computation of the loss $F(S)$ in Section 3.2 and of the coalition evaluation in Section 4.1. For the last MoE layer, let r_h be the hidden representation of the residual branch added to the MoE output. Fixing r_h and the reference token y, let Norm be the final normalization, W_{head} the output matrix mapping hidden representations to vocabulary scores, and CE the function computing the CE of the reference token from these scores. The loss when using expert set S is

$$F(S) = \mathrm{CE}(W_{\mathrm{head}}\,\mathrm{Norm}(r_h + m(S)), y)\,. \tag{D.4}$$

No attention or MoE follows this layer, so the loss of each combination can reuse the preceding hidden representations and only recompute the expert mixture and the subsequent normalization and vocabulary mapping. This allows training to continuously obtain coalition utilities that directly correspond to the current prediction.

The expert combinations of all coalitions reuse the same hidden representation h, and each expert retains its mixture weight from the current natural forward pass: swap-in expert e uses p_e, and retained experts keep their weights, e.g., $m(S_a) = m(S_0) - p_i f_i(h) + p_e f_e(h)$. Because OLMoE scores and normalizes over all experts, the weights need not be renormalized within the replaced set.

At each selected position, the outputs of the eight naturally activated experts are reused and the outputs of the two swap-in experts are computed. Weak-member selection compares the task losses after the eight single-expert removals; once the two actions are determined, the losses of the three non-empty coalitions are computed.

E Coalition evaluation and generation diagnostics

This appendix supplements joint evaluation, weak-member removal, and generation diagnostics, showing how the loss reductions of expert combinations reveal opportunities for better selection. Task results of the final training method are given in Table 1 and Figure 2a of the main text.

E.1 Joint selection and weak-member removal

Figure 2b–d in the main text compares how summing single-action utilities and joint evaluation affect combination selection. Table 4 gives the full values of the independent 36-document experiment, separating the average gain of joint selection from the complementarity missed by per-action judgment.

This 36-document diagnostic computes coalition utility as the reduction in full-answer mean NLL, with the subsequent network executed after early- and middle-layer interventions. Joint selection uses joint utility; additive selection uses the sum of single-action utilities. Complementary-pair fractions are computed per document and then averaged. The 12-pair panel uses a different candidate protocol: selection by the sum of single-action utilities reduces NLL by 0.133487, while selection by joint utility reduces it by 0.135030, a ratio of 98.9%. These observations measure gains from selection within the candidate space, not the training gains evaluated in Section 5.3.

The comparisons fix the model, input, reference prefix, intervention position, and activation count, using natural Top-8 routing as the reference. Pairs with $B_a < 0$, $B_b < 0$, and $B_{ab} > 0$ reverse the sign of each replacement's marginal contribution when the other replacement is present. Earlier layers exhibit larger joint-selection gains and more such pairs. Their interventions propagate through subsequent attention and MoE layers and may affect later predictions, whereas a single-position last-layer intervention directly affects the current prediction. This full-answer comparison therefore differs from the matched-position current-token diagnostic in Appendix B.2, which supports the last-layer evaluation used in training.

Table 4: Joint-selection gains and complementary-pair fractions across 36 documents, evaluated by full-answer mean NLL in a fixed model.

Layer	joint − additive NLL reduction ↑	Both single replacements increase the loss, joint replacement reduces it
Layer 1 (early)	0.002328	7.1%
Layer 9 (middle)	0.001433	5.6%
Layer 16 (last)	0.00006873	1.9%

Joint utility distinguishes complementarity and conflict between given replacements, while the choice of which members to start replacing determines which combinations can be found. If an expert has small removal damage in the current combination, replacing it first provides a task-related search direction. We therefore test whether this choice improves the efficiency of finding low-loss combinations per unit time. On 128 questions, we fix the four inactive experts used for swapping in and compare three rules for choosing the expert to swap out: random choice among the activated experts, the lowest routing score, and the smallest increase in content NLL after single removal. The third rule is called weak-member removal; it measures an expert's role in the current combination by the actual loss change.

On the initial OLMoE, last-layer weak-member removal achieves an NLL reduction of 1.10377 per minute, 54.7% higher than random removal. This single-removal experiment provides empirical support for the task-based screening in Section 4.1; GameMoE selects the two members with the smallest removal damage on the same natural set and then evaluates single and joint replacements. Search timing includes the loss computations of natural routing, eight single-expert removals, and four expert replacements.

E.2 Combination interventions and generated answers

Section 5.4 measures the utility of actions by comparing the reference-answer NLL before and after replacement, but a loss reduction can affect the generated answer only if it changes token choices during decoding. We therefore select a replacement by reference-answer content NLL and apply it at the same position for greedy generation, recording whether the answer changes from wrong to correct or from correct to wrong.

The fixed-removal protocol uses 256 questions: one expert to swap out is chosen at random, each of four inactive experts is swapped in, and the replacement with the lowest content NLL is selected. The weak-member removal protocol uses the predefined subset of 128 questions: the expert whose removal increases the loss the least is chosen first, and the four swap-in options are compared in the same way.

On 128 questions, the weak-member removal protocol on the initial OLMoE produces 5 fixes and no harms, showing that combination replacements guided by the reference-answer loss can change generated answers. NLL measures the probability of the reference sequence, whereas greedy decoding depends on the relative order of candidate tokens, so final task performance is still evaluated by metrics after decoding.