GAME-THEORETIC DATA ALLOCATION FOR EXPERT SPECIALIZATION AND ROUTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Can the allocation used to train experts also predict which expert will perform best on a prompt? Such a connection would make the intended division of labor inspectable before training and allow the allocation itself to guide inference. We introduce INFLENS, a mixture of LoRA experts whose training weights and inference routes come from the same location game in a fixed prompt-trait space. Players choose positions to maximize their expected shares of the prompt distribution, without using expert losses. We derive a kernel-general criterion for when co-located positions become locally unstable to separation, with an explicit Gaussian boundary determined by player count, kernel geometry, and trait covariance. In a transductive evaluation on a corpus assembled from seven safety-related benchmarks, the highest-share expert has the lowest held-out response negative log-likelihood (NLL) on approximately 90% of prompts across four kernels under a sharpened simplex representation, compared with approximately 15% after permuting allocations across prompts. In the principal seven-adapter configuration, token-level probability mixing improves NLL by 0.105 nats over a pooled adapter. Across four data splits, game shares outperform validation-trained linear gates in expected NLL when sampling one expert, while completed-sequence mixtures perform similarly. Controls show that the correspondence between allocations and expert identities matters, and that fixing solved positions retains performance at the principal operating point. These results connect an analyzable allocation of training data to useful expert specialization and routing without fitting a gate to expert performance.

1 INTRODUCTION

In many mixture-of-experts systems, expert specialization develops as the gate and experts are optimized together (Shazeer et al., 2017). We ask whether this division of labor can be specified through a training allocation that also predicts where each expert will be useful. An explicit allocation would let a designer inspect which experts receive training signal from particular kinds of prompts, such as privacy-sensitive requests or attempted jailbreaks, before training the adapters. It could also supply a rule for selecting among the trained experts. Assigning an expert more training weight in a region, however, does not guarantee that it will outperform its competitors there.

We introduce INFLENS, a mixture of LoRA experts (Hu et al., 2022) coordinated through a location game over the prompt distribution. A fixed representation maps prompts to trait vectors. Players choose positions in this space, and an influence kernel determines each player's share of a prompt relative to its competitors. Each player seeks to maximize its expected share of the prompt distribution. Shares sum to one, making the game constant-sum, so a player's gain comes at its competitors' expense. The shares weight the associated adapters' training losses and determine their inference allocation. We solve the position game numerically before adapter training, then allow positions to adapt through game-utility gradients. Neither the initial solve nor the subsequent updates use adapter losses (Figure 1).

Clustering-based methods also use data geometry to organize expert training and inference (Gururangan et al., 2023; Gou et al., 2026). In INFLENS, positions follow from individual competitive payoffs. We derive a local stability criterion for a symmetric stationary configuration in which all players occupy the same position (Section 2). The criterion applies across influence kernels and

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.

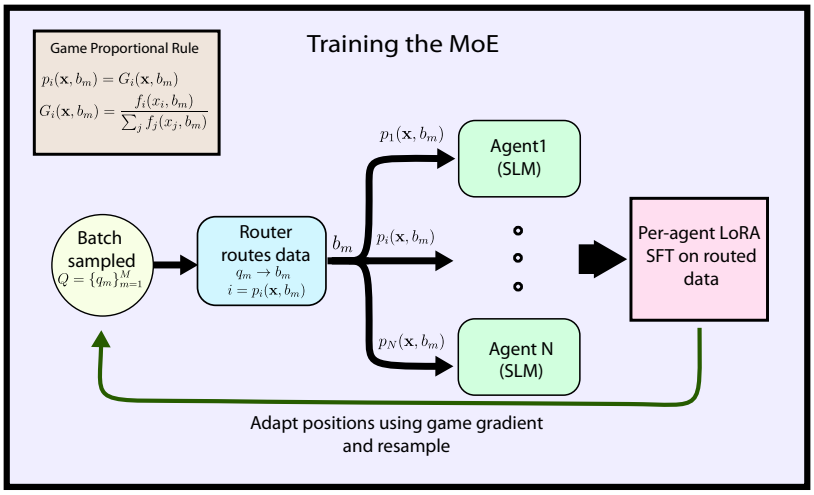

Figure 1: INFLENS makes the allocation of training data explicit through a location game. Prompt traits and player positions determine allocation shares, which weight adapter training and route inference. Initial positions are solved before adapter training. Subsequent position updates depend on prompt traits and game utilities, without feedback from adapter losses.

gives an explicit Gaussian boundary determined by player count, kernel geometry, and prompt-trait covariance. This boundary supplies a reference scale for kernel reach and identifies when co-location becomes unstable to separation. The positional analysis is available before training any adapter. We then test whether the induced allocation predicts the relative response loss of the trained experts.

We evaluate INFLENS on a heterogeneous corpus assembled from seven safety-related benchmarks, using a fixed, benchmark-informed trait representation. The evaluation is transductive: the trait map and resource distribution use the full prompt population, while test examples are excluded from adapter optimization and gate fitting. Under a sharpened simplex representation, the highest-share expert has the lowest held-out response negative log-likelihood (NLL) on approximately 90% of prompts across four kernels, compared with approximately 15% after permuting allocations across prompts.

Separate experiments test whether this allocation improves predictive performance. In the principal seven-adapter configuration, token-level probability mixing improves NLL by 0.105 nats over a pooled adapter. Holding the trained experts fixed, game-weighted mixing also improves completed-sequence mixture NLL over uniform weighting, showing that the prompt-dependent allocation contributes to the gain. Across four data splits, the game allocation achieves lower expected NLL than validation-trained linear gates when sampling one expert, while completed-sequence probability mixtures perform similarly. Holding numerically solved positions fixed during adapter training also retains performance at the principal operating point, showing that a useful allocation need not depend on continued position updates.

The game gives us a way to study expert organization through choices about training data, competitive positions, and influence. Its local stability analysis explains when positions begin to separate, and the experiments establish a connection between the resulting allocation and learned expert quality. This makes the division of training data both inspectable and testable through its predictions about expert performance. The same formulation offers a starting point for studying how experts might reorganize as new data arrive, by updating the game alongside continued adapter training.

1.1 RELATED WORK

Sparse mixtures and routing. Classical adaptive mixtures and modern sparse Transformer MoEs conventionally learn an input-dependent gate jointly with their experts (Jacobs et al., 1991; Shazeer et al., 2017; Fedus et al., 2022). Alternative assignment schemes show that such joint learning is not always necessary: token hashing and stochastic expert selection can rival learned routers in the

settings studied, whereas StableMoE learns a router in an initial stage and then freezes it to limit assignment drift (Roller et al., 2021; Zuo et al., 2022; Dai et al., 2022). Capacity balancing introduces a separate trade-off. BASE guarantees equal per-batch expert loads by constrained assignment (Lewis et al., 2021), while computing load-balancing losses within each micro-batch can spread domain-specific sequences across experts and inhibit domain specialization (Qiu et al., 2025). Audits also find architecture-dependent semantics: OpenMoE routes largely by token identity, Mixtral shows little topical separation but syntactic and temporal locality, and OLMoE reports some domain and vocabulary specialization. A recent preprint further relates expert-use similarity to hidden-state geometry (Xue et al., 2024; Jiang et al., 2024; Muennighoff et al., 2025; Wang et al., 2026). INFLENS instead defines prompt allocation through a location game in a trait space. Its allocations depend on competitive positions rather than adapter performance. The initial allocation and the local stability of co-located positions can therefore be analyzed before fine-tuning, and the experts' coordinates determine their subsequent routes.

Expert libraries, composition, and post-hoc routing. Modular systems differ in when they determine specialization and how they recombine experts. BTM specializes full-model branches on metadata-defined domains and combines their predictions with a likelihood-derived domain posterior. c-BTM replaces metadata with corpus clusters whose centers also weight experts at inference. BTX first trains domain experts and then learns a token-level gate after converting them into an MoE (Li et al., 2022; Gururangan et al., 2023; Sukhbaatar et al., 2024). At adapter scale, MoCLE similarly clusters instruction embeddings and conditions LoRA activation on cluster similarity (Gou et al., 2026). Once adapters have been trained, AdapterSoup averages selected weights. MoLE and PHATGOOSE learn gates over trained adapters. Arrow obtains training-free routing prototypes from adapter weights. LoraRetriever learns input-aware adapter retrieval (Chronopoulou et al., 2023; Wu et al., 2024; Muqeeth et al., 2024; Ostapenko et al., 2024; Zhao et al., 2024). At the whole-model level, RouteLLM learns prompt-level choices from preference data, while recent work directly compares routing, merging, and ensembling parameter-efficient experts (Ong et al., 2025; Lotfi et al., 2026). The closest non-game analogs are c-BTM and MoCLE, because cluster geometry organizes both specialization and inference. INFLENS uses competitive positions in a trait space to determine training exposure and inference allocation. Positions are solved before adapter training, and their shares provide routes without fitting a gate to expert losses.

Game-theoretic and competitive mixtures. Prior work differs chiefly in which entity plays and what it controls. Nash-MTL treats tasks as players bargaining over a shared update (Navon et al., 2022), whereas mean-field and congestion formulations make tokens the players and trade routing quality against expert load (Sen, 2026; Mouzouni, 2026). Strategic-predictor models instead let learners compete for accurately served data (Ben-Porat & Tennenholtz, 2019). Within MoEs, Mixture of Bidders routes batches by VCG auctions (Vyas, 2025). NAMEx bargains over expert-merging weights, and a gate-free vision MoE uses marginal contribution to regulate expert participation (Nguyen et al., 2026; Agarap et al., 2026). CompeteSMoE trains a router to reproduce assignments to the highest-response experts (Nguyen et al., 2025). A related softmax-routing model yields a pitchfork in expert load, rather than in expert positions (Kiselev, 2026). In INFLENS, experts choose positions in a frozen data trait space and earn expected training signal share. A numerical game solve supplies the initial positions, and a local stability analysis characterizes positional separation. The resulting shares weight adapter training and route inference.

Spatial competition and positional separation. Hotelling's location model motivated minimum differentiation, although its original price–location equilibrium was later shown to be invalid (Hotelling, 1929; d'Aspremont et al., 1979). With random-utility demand, sufficient consumer heterogeneity can restore co-location (de Palma et al., 1985). The closest structural precedent is the exposure game of Hron et al. (2023), in which producers choose embedding-space positions and receive softmax-normalized shares of a fixed attention distribution. Hard nearest-target shares also arise in position-optimization games (Frongillo et al., 2026). Related supply-side and heterogeneous-data games characterize when heterogeneous demand yields producer or model specialization (Jagadeesan et al., 2023; Xu et al., 2025), while feedback between participation and retraining can specialize or segment competing learners (Ginart et al., 2021; Dean et al., 2024). EM and deterministic annealing offer an algorithmic analogy. Responsibilities softly weight prototype updates, and coincident prototypes split at critical temperatures as a common objective loses stability (Dempster et al., 1977; Rose et al.,

1990; Ueda & Nakano, 1998). In our game, players maximize individual shares rather than a common likelihood or free energy. The softmax coupling that can be a statistical liability in MoEs (Nguyen et al., 2024) defines the payoff. We derive a kernel-general local instability criterion, including a Gaussian boundary explicit in player count and resource covariance, and use the numerically solved positions and allocation shares to initialize the system, weight adapter training, and route inference.

2 THE INFLUENCER'S GAME

Here we introduce the influencer game. Let a data point's frozen trait vector be $b \in \mathbb{B}$, drawn from a resource distribution B. Player $i \in \{1, \dots, N\}$ chooses position $x_i \in \mathbb{B}$. Then for a positive influence kernel $f_\kappa(x, b; \sigma)$ with parameter σ, a player's share and expected payoff are

$$G_i(\mathbf{x}, b) = \frac{f_\kappa(x_i, b; \sigma)}{\sum_{j=1}^{N} f_\kappa(x_j, b; \sigma)}, \qquad u_i(\mathbf{x}) = \mathbb{E}_B[G_i(\mathbf{x}, b)]\,. \tag{1}$$

For every resource point $\sum_i G_i = 1$, hence $\sum_i u_i = 1$ at every configuration, indicating the game is constant-sum. Importantly, being constant-sum implies that one player's gain comes at the expense of the other player(s). The constant-sum nature means that specialists will be competing for the same resource signal and are therefore encouraged to explore new niches when a niche is crowded and their influence kernel is not sufficiently wide. Importantly, the utility of the player is based on their training signal share, and their accuracy is an emergent property of the training on that signal. In turn, this means that a player's received training signal is tractable through the history of the game, allowing us to predict specialists without explicitly testing their accuracy during training.

2.1 EQUILIBRIUM AND INSTABILITY

We start by assuming that players are at a symmetric profile $x_1 = \cdots = x_N = x^\circ$, meaning they share a position in the trait space. We let the first derivative of a player's log-influence at the trait point $s(b)$ be $s(b) = \nabla_x \log f_\kappa(x^\circ, b; \sigma)$. Then an interior symmetric stationary point obeys $\mathbb{E}_B[s(b)] = 0$. Define restoring log-curvature and score scatter by

$$K = -\mathbb{E}_B\left[\nabla_x^2 \log f_\kappa(x^\circ, b; \sigma)\right], \qquad C = \mathbb{E}_B\left[s(b)s(b)^\top\right]. \tag{2}$$

When K is positive definite on the relevant tangent space, the antisymmetric, player-separating mode has instability depth

$$D_\kappa = \frac{N-2}{N-1}\,\lambda_{\max}\left(K^{-1/2} C K^{-1/2}\right). \tag{3}$$

For $D_\kappa < 1$ the symmetric profile is locally stable, while $D_\kappa > 1$ gives a player-separating linear instability; $D_\kappa = 1$ is the stability boundary and, under a transversal parameter crossing, the first local bifurcation. Importantly, the log-concavity of a kernel indicates there is always a kernel parameter regime where the symmetric equilibrium is stable and unique. Additionally, for two effective players, the prefactor vanishes, so there is no separating-mode bifurcation and the symmetric equilibrium is always locally stable. Appendix B derives these claims from Equation 1.

We study Gaussian, hyperbolic, and exponential-power kernels whose parameters determine the kernel's spread from its mode (standard deviation, etc.), as well as kernels whose parameters determine the kernel's shape, including the simplex-native Dirichlet kernel and a coordinate-wise product-Beta kernel. We denote a kernel's reach by σ, where Gaussian log influence scales as σ^{-2}, hyperbolic as σ^{-1}, and exponential-power as $\sigma^{-\beta}$; the shape kernels place inverse reach inside their density parameters and normalizers. To simplify comparisons across kernel types, we compare kernels by their depth in Equation 3 and by their allocation sharpness (Section 4); Appendix E gives every implemented closed form.

3 METHODS

The game router needs to have a trait assigned to every data point so that experts have a space to compete within for training signal. Beyond the requirement of a trait space, other properties of the data distribution do not limit the applicability of INFLENS, such as dimensionality or labels. Naturally,

many empirical data sets have high-dimensional trait spaces when using raw embeddings, such as the 4096-dimensional embeddings of Qwen3-Embedding-8B (Zhang et al., 2025), which likely leads to very sparse resource distributions with limited data sets, this in itself does not limit the effectiveness of INFLENS but can make it more difficult to demonstrate the nuances of its impacts. To address this, we transform the 4096-dimensional embeddings of Qwen3-Embedding-8B to 7 distinct axis labels assigned from safety-relevant data sets; however, these could have been learned axis labels instead of assigned with prior knowledge. We generate a mixed dataset from seven benchmarks on distinct safety-relevant properties: harm, hallucination, jailbreak, privacy, over-refusal, policy violation, and prompt injection (Appendix A). Each safety-relevant property then defines an axis label that we use.

To generate the trait space, we use the frozen Qwen3-Embedding-8B encoder throughout, and for each axis we fit a linear direction that separates that benchmark from the rest, orthogonalized against the axes already fitted. Projecting an embedding onto the seven directions gives seven raw scores, and a per-axis empirical CDF maps them into the unit hypercube. Thus each coordinate is a percentile within the corpus and the seven marginals are uniform by construction. A temperature softmax over the same coordinates gives the simplex variant. Both the simplex and the cube can be seen in Figure 7 in Appendix D where the construction is given in full. The 28,355 prompts in the combined corpus form a distribution in the domain, which serves as the resource distribution the experts compete for as learning signals.

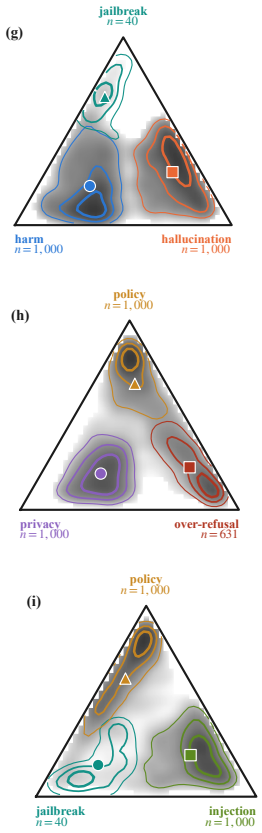

Figure 2: **Trait Space example**; held-out prompt density in three conditional simplex views at $T = 0.35$. Grey is the combined density; colored contours mark benchmark origin. Panels (g)–(i) are the bottom row of Figure 7, which also gives the cube and $T = 1$ views.

The principal experiment places fourteen clones of the base model at seven co-located pairs of positions and trains one rank-16 LoRA adapter per pair. The pairing of the clones is motivated by two game properties: a symmetric two-player game with positive log-concave influence kernels has no player-separating instability (Section 2.1), and duplicating every solved location into an equal co-located pair leaves the effective allocation exactly unchanged (Appendix B). In effect, this implies that the pairs behave as one agent when updating their positions, with the added benefit of increased diversity at a lower depth than individual initialization. The pair is therefore the routing and adapter-training unit: its members' shares sum to the allocation a single agent would receive at that position in the seven-player game, but at a much lower depth. This is a cost-saving mechanism, as it effectively doubles the learning signal individual agents would receive; if we were concerned with rational agents' decision-making, we could keep the individual adapters of pairs at double the training cost. We keep the even split between members only in the position dynamics, where each clone follows its own utility gradient in the fourteen-player game. Equal co-located clones receive identical updates, so pairing is an invariant of the update rather than an explicit copy operation (Appendix C). Solving the eight-player position game directly shows the same invariance, as players started in four co-located pairs remain paired at every reach below $0.7\sigma^*$ (Figure 6).

In each experiment, the pipeline completes 12 rounds, consisting of the following steps: it projects a training batch to traits, computes pair shares by Equation 1, and assigns each prompt to its top-k pairs, with retained shares renormalized as loss weights. The dense $k = N$ arm retains all N pairs, whereas $k = 1$ is a hard allocation. In the dynamic-position arm, after allocation, each clone takes a simultaneous projected ascent step on its game payoff, using the score of the chosen influence kernel over the whole round batch. A fixed-position control disables every position-update path while leaving assignment and adapter training otherwise unchanged. Appendix G separately ablates fan-out, sampling, and loss weighting, showing why the main experiments retain the full share-weighted allocation.

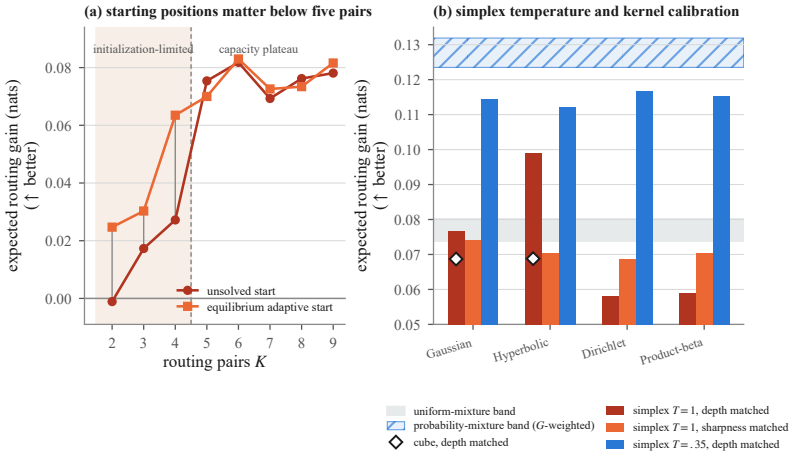

Figure 3: Initialization matters when routing capacity is scarce, whereas induced allocation sharpness matters more than kernel identity. (a) Expected routing gain is pooled-generalist NLL minus the share-weighted NLL of sampling one pair (higher is better). The equilibrium adaptive start improves on the unsolved start for two, three, and four pairs, but the two curves largely converge once five or more pairs are available. (b) Matching bifurcation depth leaves a large gain spread across four kernels (dark red); matching prompt-level sharpness narrows it (orange), and sharpening the simplex projection to $T = 0.35$ (dispersed) raises and nearly equalizes the gains (blue). Hollow diamonds are cube controls. The grey and hatched bands are sequence-level probability mixtures, so they provide deployment references rather than like-for-like comparisons with the single-pair bars.

At evaluation, final pair positions yield $G_p(b)$ without fitting a gate to held-out expert losses. We distinguish expected NLL $\sum_p G_p(b)\ell_p(b)$ (the loss of sampling one pair), argmax commitment, and the NLL of mixing pair probabilities over a completed sequence, which can exploit probability averaging that a single-expert decision cannot. In each experiment, we use a hindsight best-expert oracle to supply an upper bound on recoverable routing headroom. To show how a single model compares, a pooled comparator trains on the corresponding union of routed batches, matching the training signal the seven experts receive in aggregate. Therefore, an expert and pooled adapter each receive the same 1,248 optimizer steps.

3.1 Experimental setup

Appendix A gives the source, retained counts, and annotation signal for each of the seven benchmarks in Table 2. The resulting 28,355 records are partitioned independently within each source using a persisted seeded manifest: 19,848 train, 2,836 validation, and 5,671 test records; a (70/10/20) split. Every arm reuses those exact indices for their corresponding experiment. We fit the trait axes, quantile normalization, and resource KDE to the full 28,355-record population before any adapters are trained; this resource KDE is only used for calculating the theoretical equilibrium of the distributions. "Test" therefore means held out from adapter optimization, gate fitting, and expert-loss feedback, but not from resource-map estimation. This allows us to test if the theoretical equilibrium of the entire data set is stable in the training update steps and how the theoretical equilibrium impacts the agent's performance. The principal arm starts from Qwen2.5-1.5B-Instruct (Yang et al., 2024), trains rank-16 LoRA adapters for twelve rounds, uses soft routing with weights equal to the proportion of influence over data (Appendix G), and scores a 5,671-prompt held-out pool. We do both an expert model family and size ablation in Appendix I

4 Results

4.1 Game Controls:Equilibrium Positioning, Influence Kernel, and Resource Geometry

Our starting experiment in Figure 3.a with a spherical multivariate Gaussian influence kernel demonstrates that the equilibrium adaptive start helps most when player capacity is scarce, compared with an

Table 1: How well the game router predicts expert quality, per influence kernel and simplex temperature T, which sets how concentrated the resource distribution is. Each cell is one seven-pair training run scored on the shared 5,671-prompt test pool; higher is better for every measure. Top 1, top 2 and top 3 are the percentages of held-out prompts whose best expert receives the largest share, one of the two largest or one of the three largest. Each null is the same measure after shuffling which prompt each allocation belongs to, averaged over 200 shuffles. Spearman is the mean within-prompt rank correlation between allocation and expert quality.

		Top 1 (%)		Top 2 (%)		Top 3 (%)		Spearman	
T	Kernel	obs.	null	obs.	null	obs.	null	obs.	null
1.00 (dense)	Gaussian	86.8	15.1	95.5	25.8	97.9	38.5	0.735	0.223
	Hyperbolic	87.3	15.0	96.4	27.0	98.5	41.1	0.650	0.148
	Dirichlet	82.6	15.1	93.0	25.8	96.2	38.2	0.737	0.216
	Product-beta	83.2	15.0	93.6	25.8	96.7	38.2	0.742	0.223
0.35 (dispersed)	Gaussian	90.4	15.4	96.6	28.6	98.1	44.1	0.495	0.053
	Hyperbolic	89.2	15.2	96.8	27.5	98.5	42.2	0.554	0.102
	Dirichlet	90.1	15.3	96.9	28.0	98.2	42.7	0.476	0.054
	Product-beta	90.8	15.3	96.9	28.0	98.3	42.7	0.480	0.050

unsolved start, which begins from an approximate, non-converged solve of the game. In the measured two-, three-, and four-pair arms, starting at equilibrium improves routed NLL by 0.0258, 0.0130, and 0.0363 nats, respectively; at five or more pairs, its average effect is approximately zero. However, this is also confounded by the fact that larger pair counts are quicker to arrive at theoretical equilibria than the smaller ones, indicating that the gradient magnitude of agent updates varies significantly across player counts as seen in Figure 10.

In Figure 3.b, four kernels and three trait-space geometries show that trait-space geometry shifts how much kernel identity matters. Matched on bifurcation depth on the simplex at $T = 1$ (dense), expected gain spans 0.0407 nats across the four kernels and tracks their sharpness almost exactly: where the sharpest kernel is best, and the most diffuse is worst. Matching sharpness by shifting kernel parameters collapses that spread to 0.0055 nats. Meanwhile, sharpening the simplex projection to $T = 0.35$ (dispersed) lifts every kernel to roughly 0.115 nats with a spread of 0.0046, effectively eliminating any difference between the kernels when depth matched. Table 1 shows the same collapse in how well the allocation predicts expert quality: top-1 agreement spans 82.6–87.3% across kernels at $T = 1$ and 89.2–90.8% at $T = 0.35$, against a shuffled-allocation null of about 15%. This indicates that kernel choice and resource distribution impact the predictive power of the game router.

Importantly, in Table 1 the game router is approximately 6x more predictive in what expert data should be routed to than a random chance, indicating how specialization is derived from INFLENS. Consequently, the results show higher router–oracle agreement and mixture capture on the simplexes than the cube counterpart, but a bidirectional frozen-expert control shows that each expert family prefers the geometry in which it trained (Appendix D). This shows that experts trained on their geometry do not transfer to experts on another geometry of data.

4.2 KERNEL PARAMETERS

Kernel parameters control how broadly an expert competes for training signal and can determine the depth of the influence kernel past its bifurcation threshold. To test this effect on INFLENS, in Figure 4 we limit to 7 pairs and use the first local bifurcation σ^* as a reference scale for a spherical multivariate Gaussian influence kernel's reach parameter σ, followed by a sweep of actual reach values and measuring the resulting routing response. We compare two starts, and positions adapt during training under both: an equilibrium adaptive start, which begins at the solved empirical-game equilibrium and tends to stay near it, and a pinned adaptive start, which begins every reach on a data split from the same unsolved configuration.

Initialization acts on the shape of the reach response and leaves its level unchanged. Mean per-split roughness (excluding the degenerate 0.70 endpoint) is 0.483 for the equilibrium adaptive start and

1.266 for the pinned start over three data splits each, and the per-split ranges do not overlap (0.43–0.53 against 1.07–1.45).

Both starts peak at reach 0.40 or 0.45, so a shared start fixes where the response peaks, but the equilibrium adaptive start is 2.6 times smoother and makes the choice of which configuration to pin less arbitrary. Mean gain does not separate them, at 0.0764 nats for the pinned and 0.0747 nats for the equilibrium adaptive start, and the paired difference, -0.0018 ± 0.0082 nats across splits, is small next to its own spread (Figure 4).

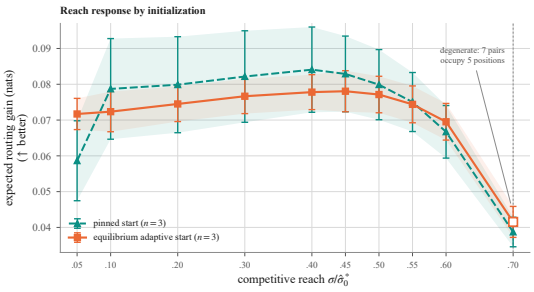

Figure 4: The equilibrium adaptive start stabilizes the response to competitive reach without changing mean gain. Each point is a within-data-seed contrast against its pooled generalist; bands and error bars are sample standard deviations across three data splits per start.

The pinned start runs had not settled by round 11: they closed 43% of their distance to the equilibrium adaptive runs on average, and continuing the position update past training carries only 15 of the 27 non-degenerate reach–split cells to the same fixed point (Appendix F), indicating there is some choice of equilibria that needs to be made.

4.3 Routing evidence and controls

With the seven-pair expert family fixed, the standard token-level routed mixture gains 0.1049 nats over pooling. The completed-sequence estimator used for the same-expert nulls gives a slightly more optimistic 0.1093–0.1095 nats. On that same estimator, uniform mixing retains a 0.0661-nat gain, whereas shuffled expert identities lose 0.0671 nats and uniform random hard routing loses 0.4144. Here we only change the combiner, but in the appendix we show the impact of changing the way the adapters are trained. (Figure 5). The best choice is to demonstrate the dense top-$k = N$ case, where N is the number of pairs.

To compare to a learned gate, we fit linear prompt-conditioned gates on validation expert losses, freezing all adapters. Across four data splits, the game allocation beats both the full-NLL soft-target gate and a best-expert classification gate under single-expert commitment, by 0.071–0.090 nats. Under full probability mixing, the sign divides two-to-two across splits and the largest difference is only 0.0017 nats, at the measured run-to-run noise floor. These results support that a gate-free allocation for hard routing is sufficient for expert routing; however, we did not test a nonlinear, token-level, and nearest-neighbor gate.

4.4 Supplementary additional experiments

Experimenting with model sizes including Qwen sizes 1.5B, 3 B, 7 B, 14 B, and 31 B shows that increasing model sizes each improves over their own pooled comparator while oracle-headroom capture declines. The allocation effects also transfer to two other model families across Qwen, Gemma, and Llama. These tests serve as portability checks and show nuances in training parameters, such as fine-tuning method; Appendix I provides both ladders and their distinct protocols.

Additionally, our method performs very well within a ten-family baseline audit including but not limited to a frozen gate, same-expert nulls, balanced c-BTM, BASE-style assignment, MoLoRA, KnOTS, and a pooled seed deep ensemble; all of which we have collected in Appendix J. In the most influential experiment of these, on the same seed-0 split and estimator, the game mixture nearly doubles c-BTM's gain over pooling (NLL 1.9057, +0.1095 vs. 1.9564, +0.0588), while balanced c-BTM degrades to 1.9647, reinforcing the benefit of proportional allocation when compared to its closest mechanistic and directly comparable baseline. Other results demonstrated similar gains in performance.

Finally, in Appendix K we test the impacts of the training on safety benchmark performance. While the game did outperform the pooled on many benchmarks, the base model ultimately far outperformed

all trained models on the benchmarks. This indicates that fine-tuning itself negatively affects model alignment and does not decisively point to the impact INFLENS has.

5 LIMITATIONS AND DISCUSSION

The headline claim of INFLENS is the influencer's game makes specialization of experts an explicit process rather than an empirically derived one. Through experiments, we have seen that even without training a gate on experts' losses, INFLENS predicted the best expert 90% of the time as its top choice and the best expert 96% of the time in its top-2 choices when routing data. This has many consequences for interpretability, since we may isolate how data is routed and to which experts. For safety, this means that we may target experts with specific safety data, or if the MOE experts perform poorly on a specific data point in the trait space, identify which expert is explicitly responsible.

Importantly, there are many levers that researchers can pull in order to shape the behavior of the MoE through the game structure, including the number of experts, the data distribution representation, and kernel choice and its parameters. Perhaps more useful are the equilibria given by the influencer's game component, which provide targets for models adapting to an unknown data distribution or possibly a changing one. If the data distribution is known, then the socialists' equilibrium positions can be calculated before time and used as stable initializations for expert specialization.

There are several limitations. First and foremost, INFLENS relies on symmetric and log-concave kernels. This implies that experts cannot have multiple central points of expertise and that their influence cannot decay asymmetrically. While the first can be addressed by a larger number of specialists, there is a real cost to training more adapters. The second cost is more limiting because the theory fundamentally relies on symmetric decay of influence, but the need for asymmetric kernels remains uncertain.

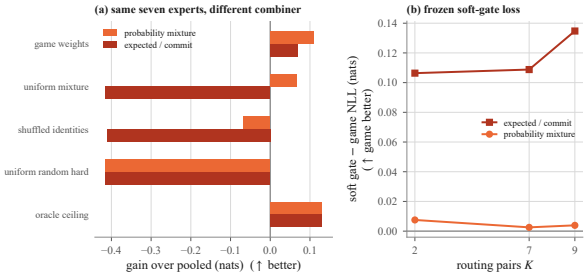

Figure 5: The game allocation carries expert-specific routing information. (a) Holding all seven trained pairs fixed, the game combiner improves over the pooled model, whereas shuffling position–expert identities destroys the gain and uniform random commitment is substantially worse. Orange bars mix sequence probabilities; dark bars report sampling/commitment semantics, and the oracle is a hindsight ceiling (higher gain is better). (b) Extra NLL of the validation-trained linear soft gate over the game allocation, for the equilibrium adaptive start on one data seed; positive values favor the game. The gate loses under commitment and is nearly tied under probability mixing. The four-data-seed replication is reported in the text.

Perhaps the most promising future direction is online adaptation to new data streams. Given that there is no trained gate there is no risk of router collapse. The game defines a position payoff for any observed resource distribution, so an updated prompt distribution implies new position incentives and a new candidate equilibrium. In the present fixed-distribution experiments, pairs from the pinned start move closer to the equilibrium adaptive runs in all 27 non-degenerate reach–split cells, closing 43% of their initial distance by round 11 on average, which shows that the implemented position dynamics can correct an initial position mismatch without testing distribution shift. A direct test would change the prompt distribution over time, compare predicted and observed position movement, and continue adapter training under the revised allocations.

6 CONCLUSION

INFLENS treats expert coordination as a location game over a trait space of a data distribution. Its equilibrium supplies a starting position before any adapter is trained, and its allocation is a falsifiable prediction about held-out expert quality rather than a fitted gate. The components of INFLENS are not limited to the language model setting nor by choices like encoder and data type, and may contribute to other areas of work. The open question is how this method will apply to online settings where the data distribution is not fixed.

7 AI USE DISCLOSURE

Generative AI tools, including OpenAI Codex, assisted with literature discovery and citation checking, manuscript organization and language editing, and software implementation and debugging. The authors reviewed all retained prose and code, checked citation-dependent claims against primary sources, tested AI-assisted implementation, and verified reported numerical claims against saved experimental artifacts. AI-generated output was not treated as empirical evidence or credited with authorship; the authors take responsibility for the paper and all associated artifacts.

8 ETHICS STATEMENT

This work develops a routing and training-allocation method and conducts no human-subject experiments or new collection of personal data; the experiments use existing public benchmarks whose provenance is documented in Appendix A. The method may make expert behavior easier to audit, but it does not guarantee safety or fairness, and specialization can concentrate failures as well as useful capabilities. Appendix K reports the observed safety limitations, including that fine-tuning substantially degraded alignment relative to the base model.

9 REPRODUCIBILITY STATEMENT

The game assumptions and stability analysis are stated in Section 2 and proved in Appendix B; the data sources, retained counts, shared splits, model and adapter settings, routing procedure, and evaluation estimators are documented in Section 4 and Appendices A–G. Appendices I, J, and K specify the ablation, baseline, and behavioral protocols and their qualifications. The accompanying code uses versioned YAML configurations, records the resolved configuration and run history, and exposes a single pipeline entry point; Appendix L gives the command and identifies the result artifacts underlying the reported analyses. Access to the code is available here: https://anonymous.4open.science/r/infl_ens-2516/README.md.

REFERENCES

Abien Fred Agarap, Inigo Miguel Benavides, and Sara Ann Venturina. Nash bargaining for gate-free mixture-of-experts. In *NExT-Game Workshop at the International Conference on Machine Learning*, 2026. URL https://openreview.net/pdf/566cfde3e6550eeda0f5e759166e06374388b217.pdf. Workshop paper.

ai4Privacy. PII-masking-200k: A dataset of synthetic documents with labelled personally identifiable information. Hugging Face dataset, 2023. URL https://huggingface.co/datasets/ai4privacy/pii-masking-200k.

Omer Ben-Porat and Moshe Tennenholtz. Regression equilibrium. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, 2019. arXiv:1905.02576.

Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Röttger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. Safety-tuned LLaMAs: Lessons from improving the safety of large language models that follow instructions. In *International Conference on Learning Representations*, 2024. arXiv:2309.07875.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. JailbreakBench: An open robustness benchmark for jailbreaking large language models. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. arXiv:2404.01318.

Alexandra Chronopoulou, Matthew E. Peters, Alexander Fraser, and Jesse Dodge. AdapterSoup: Weight averaging to improve generalization of pretrained language models. In *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 2054–2063, 2023.

Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. OR-Bench: An over-refusal benchmark for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 11515–11542, 2025. URL https://proceedings.mlr.press/v267/cui25a.html.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. StableMoE: Stable routing strategy for mixture of experts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022. URL https://aclanthology.org/2022.acl-long.489/.

Claude d'Aspremont, Jean Jaskold Gabszewicz, and Jacques-François Thisse. On Hotelling's "stability in competition". *Econometrica*, 47(5):1145–1150, 1979. doi: 10.2307/1911955.

André de Palma, Victor Ginsburgh, Yorgos Y. Papageorgiou, and Jacques-François Thisse. The principle of minimum differentiation holds under sufficient heterogeneity. *Econometrica*, 53(4): 767–781, 1985. doi: 10.2307/1912653.

Sarah Dean, Mihaela Curmei, Lillian J. Ratliff, Jamie Morgenstern, and Maryam Fazel. Emergent specialization from participation dynamics and multi-learner retraining. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, 2024. arXiv:2206.02667.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977. URL https://www.jstor.org/stable/2984875.

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.

Rafael Frongillo, Melody Hsu, Mary Monroe, and Anish Thilagar. Equilibria in large position-optimization games. In *Proceedings of the 27th ACM Conference on Economics and Computation*, 2026. arXiv:2602.15225.

Antonio Ginart, Eva Zhang, Yongchan Kwon, and James Zou. Competing AI: How does competition feedback affect machine learning? In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, 2021. arXiv:2009.06797.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Zhenguo Li, Dit-Yan Yeung, James T. Kwok, and Yu Zhang. Mixture of cluster-conditional LoRA experts for vision-language instruction tuning. *IEEE Transactions on Image Processing*, 35:3881–3892, 2026. arXiv:2312.12379.

Suchin Gururangan, Margaret Li, Mike Lewis, Weijia Shi, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Scaling expert language models with unsupervised domain discovery. *arXiv preprint arXiv:2303.14177*, 2023. URL https://arxiv.org/abs/2303.14177.

Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. Model merging and safety alignment: One bad model spoils the bunch. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024. arXiv:2406.14563.

Harold Hotelling. Stability in competition. *The Economic Journal*, 39(153):41–57, 1929.

Jiri Hron, Karl Krauth, Michael I. Jordan, Niki Kilbertus, and Sarah Dean. Modeling content creator incentives on algorithm-curated platforms. In *International Conference on Learning Representations*, 2023. arXiv:2206.13102.

Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. Safe LoRA: The silver lining of reducing safety risks when fine-tuning large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. arXiv:2405.16833.

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.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama Guard: LLM-based input-output safeguard for human-AI conversations. *arXiv preprint arXiv:2312.06674*, 2023.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991. URL https://www.cs.toronto.edu/~hinton/absps/jjnh91.pdf.

Meena Jagadeesan, Nikhil Garg, and Jacob Steinhardt. Supply-side equilibria in recommender systems. In *Advances in Neural Information Processing Systems*, volume 36, 2023. arXiv:2206.13489.

Sanskar Jajoo. Neuralchemy prompt injection threat matrix. Hugging Face dataset, binary configuration, 2026. URL https://huggingface.co/datasets/neuralchemy/prompt-injection-Threat-Matrix.

Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. BeaverTails: Towards improved safety alignment of LLM via a human-preference dataset. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. arXiv:2307.04657.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Jaehan Kim, Minkyoo Song, Seungwon Shin, and Sooel Son. SafeMoE: Safe fine-tuning for MoE LLMs by aligning harmful input routing. In *International Conference on Learning Representations*, 2026. arXiv:2509.22745.

O. M. Kiselev. A minimal bifurcation model of load imbalance in a softmax mixture-of-experts router. *arXiv preprint arXiv:2605.29121*, 2026.

Harsh Kohli, Helian Feng, Lenon Minorics, Bhoomit Vasani, Xin He, and Ali Kebarighotbi. EWoRA: Expert weighted low-rank adaptation for heterogeneous data. In *Findings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 1729–1737, 2025. doi: 10.18653/v1/2025.findings-ijcnlp.108. URL https://aclanthology.org/2025.findings-ijcnlp.108/.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.

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*, pp. 6265–6274, 2021. URL https://proceedings.mlr.press/v139/lewis21a.html.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6449–6464, 2023. doi: 10.18653/v1/2023.emnlp-main.397. URL https://aclanthology.org/2023.emnlp-main.397/.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*, 2022. URL https://arxiv.org/abs/2208.03306.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, 2022.

Sanae Lotfi, Lucas Caccia, Alessandro Sordoni, Jordan T. Ash, and Miroslav Dudík. Trade-offs in ensembling, merging and routing among parameter-efficient experts. *Transactions on Machine Learning Research*, 2026. arXiv:2603.03535.

Yufei Ma, Zihan Liang, Huangyu Dai, Ben Chen, Dehong Gao, Zhuoran Ran, Zihan Wang, Linbo Jin, Wen Jiang, Guannan Zhang, Xiaoyan Cai, and Libin Yang. MoDULA: Mixture of domain-specific and universal LoRA for multi-task learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 2758–2770, 2024. doi: 10.18653/v1/2024.emnlp-main.161. URL https://aclanthology.org/2024.emnlp-main.161/.

Charafeddine Mouzouni. Three phases of expert routing: How load balance evolves during mixture-of-experts training. *arXiv preprint arXiv:2604.04230*, 2026.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. OLMoE: Open mixture-of-experts language models. In *International Conference on Learning Representations*, 2025. arXiv:2409.02060.

Mohammed Muqeeth, Haokun Liu, Yufan Liu, and Colin Raffel. Learning to route among specialized experts for zero-shot generalization. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. arXiv:2402.05859.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 16428–16446, 2022. URL https://proceedings.mlr.press/v162/navon22a.html.

Dung Viet Nguyen, Anh Thi, Minh Hoang Nguyen, Luc Nguyen, Shiqi Jiang, Ethan Fetaya, Duy Linh Tran, Gal Chechik, and Tan Nguyen. Expert merging in sparse mixture of experts with nash bargaining. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/ef9f16cd0cb8d7797ea35c0ee7964cdf-Abstract-Conference.html.

Huy Nguyen, Nhat Ho, and Alessandro Rinaldo. Sigmoid gating is more sample efficient than softmax gating in mixture of experts. In *Advances in Neural Information Processing Systems*, volume 37, 2024. arXiv:2405.13997.

Nam V. Nguyen, Huy Nguyen, Quang Pham, Van Nguyen, Savitha Ramasamy, and Nhat Ho. CompeteSMoE: Statistically guaranteed mixture of experts training via competition. *arXiv preprint arXiv:2505.13380*, 2025. URL https://arxiv.org/abs/2505.13380.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs with preference data. In *International Conference on Learning Representations*, 2025. arXiv:2406.18665.

Oleksiy Ostapenko, Zhan Su, Edoardo Maria Ponti, Laurent Charlin, Nicolas Le Roux, Matheus Pereira, Lucas Caccia, and Alessandro Sordoni. Towards modular LLMs by building and reusing a library of LoRAs. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. arXiv:2405.11157.

Zihan Qiu, Zeyu Huang, Bo Zheng, Kaiyue Wen, Zekun Wang, Rui Men, Ivan Titov, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Demons in the detail: On implementing load balancing loss for training specialized mixture-of-expert models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, 2025. arXiv:2501.11873.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. Hash layers for large sparse models. In *Advances in Neural Information Processing Systems*, volume 34, 2021. arXiv:2106.04426.

Kenneth Rose, Eitan Gurewitz, and Geoffrey C. Fox. Statistical mechanics and phase transitions in clustering. *Physical Review Letters*, 65:945–948, 1990. doi: 10.1103/PhysRevLett.65.945.

Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of NAACL-HLT*, 2024. arXiv:2308.01263.

Nevroz Sen. Bridging game theory and transformer routing: Mean field equilibria for mixture of experts. In *NExT-Game Workshop at ICML*, 2026. URL https://openreview.net/pdf/e1a5acd239f250576298ae0057d2bf278b212da1.pdf.

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

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the KnOTS. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0d4f8a5109c5083b5307fcd0bddae7a7-Abstract-Conference.html.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Shang-Wen Li, Wen-tau Yih, Jason Weston, and Xian Li. Branch-train-MiX: Mixing expert LLMs into a mixture-of-experts LLM. In *Conference on Language Modeling*, 2024. arXiv:2403.07816.

Naonori Ueda and Ryohei Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11(2): 271–282, 1998. doi: 10.1016/S0893-6080(97)00133-0.

Dev Vyas. MoB: Mixture of bidders. *arXiv preprint arXiv:2512.10969*, 2025. URL https://arxiv.org/abs/2512.10969.

Xi Wang, Soufiane Hayou, and Eric Nalisnick. The myth of expert specialization in MoEs: Why routing reflects geometry, not necessarily domain expertise. *arXiv preprint arXiv:2604.09780*, 2026.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of EMNLP*, pp. 5744–5760, 2022. URL https://aclanthology.org/2022.emnlp-main.388/.

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. Do-not-answer: Evaluating safeguards in LLMs. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 896–911, 2024. doi: 10.18653/v1/2024.findings-eacl.61. URL https://aclanthology.org/2024.findings-eacl.61/.

Xun Wu, Shaohan Huang, and Furu Wei. Mixture of LoRA experts. In *International Conference on Learning Representations*, 2024. arXiv:2404.13628.

Renzhe Xu, Kang Wang, and Bo Li. Heterogeneous data game: Characterizing the model competition across multiple data sources. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. arXiv:2505.07688.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. OpenMoE: An early effort on open mixture-of-experts language models. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. arXiv:2402.01739.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.nips.cc/paper/2023/hash/1644c9af28ab7916874f6fd6228a9bcf-Abstract-Conference.html.

Yahma. Alpaca-cleaned. Hugging Face dataset; cleaned version of Stanford Alpaca, 2023. URL https://huggingface.co/datasets/yahma/alpaca-cleaned.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Jingwei Yi, Yueqi Xie, Bin Zhu, Emre Kiciman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. Benchmarking and defending against indirect prompt injection attacks on large language models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025. doi: 10.1145/3690624.3709179. arXiv:2312.14197.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 57755–57775, 2024. URL https://proceedings.mlr.press/v235/yu24p.html.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermiş, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient MoE for instruction tuning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6d00071564ec447466fc4577743cf1b3-Abstract-Conference.html.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Ziyu Zhao, Leilei Gan, Guoyin Wang, Wangchunshu Zhou, Hongxia Yang, Kun Kuang, and Fei Wu. LoraRetriever: Input-aware LoRA retrieval and composition for mixed tasks in the wild. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 4447–4462, 2024. doi: 10.18653/v1/2024.findings-acl.263.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. Taming sparsely activated transformer with stochastic experts. In *International Conference on Learning Representations*, 2022. arXiv:2110.04260.

A DATA PROVENANCE

Table 2 lists the seven public datasets that make up the resource population, one per trait axis, with the retained sample count and the source annotation used to estimate each direction. The signals label the prompt, not our models: the over-refusal signal, for instance, distinguishes OR-Bench's benign-but-seemingly-toxic prompts from its toxic controls, not whether one of our models refused.

Table 2: Data used to construct the resource landscape. Counts are the records retained before the shared split. "Signal" is the annotation used to learn the corresponding trait direction.

Trait	Source	Signal	n
Harm	BeaverTails-30k (Ji et al., 2023)	response unsafe in any harm category	5,000
Hallucination	HaluEval QA and dialogue (Li et al., 2023)	faithful versus hallucinated response	5,000
Jailbreak	JBB-Behaviors (Chao et al., 2024)	harmful versus benign goal	200
Privacy	AI4Privacy PII-Masking-200k (ai4Privacy, 2023)	character fraction covered by PII spans	5,000
Over-refusal	OR-Bench 80k and toxic (Cui et al., 2025)	seemingly toxic benign prompt versus toxic control	3,155
Injection	Prompt Injection Threat Matrix (Jajoo, 2026)	binary malicious-injection label	5,000
Policy violation	Do-Not-Answer (Wang et al., 2024) with Alpaca-Cleaned negatives (Yahma, 2023)	refusal-worthy versus benign instruction	5,000

B GAME-THEORETIC DERIVATIONS

Regularity and symmetric stationarity. Let all $N \geq 2$ players use the same positive kernel $f_\kappa(\cdot, b; \sigma) \in C^2$ on an open neighborhood of their strategy domain, and suppose differentiation can pass through the expectation over B. The formulas below are Euclidean; for simplex-valued positions, all gradients, Hessians, and definiteness statements are restricted to the simplex tangent space.

Write $s_i(b) = \nabla_{x_i} \log f_\kappa(x_i, b; \sigma)$ and $H_i(b) = \nabla^2_{x_i} \log f_\kappa(x_i, b; \sigma)$. Differentiating the normalized share gives

$$\nabla_{x_i} G_i = G_i(1 - G_i)s_i, \qquad \nabla_{x_j} G_i = -G_i G_j s_j \quad (j \neq i), \tag{4}$$

and hence the positional gradient field is

$$F_i(\mathbf{x}) := \nabla_{x_i} u_i(\mathbf{x}) = \mathbb{E}_B[G_i(1 - G_i)s_i]\,. \tag{5}$$

At a symmetric profile $x_1 = \cdots = x_N = x^\circ$, every share is $1/N$, so

$$\nabla_{x_i} u_i = \frac{N-1}{N^2}\mathbb{E}_B[s(b)]\,. \tag{6}$$

Thus an interior symmetric profile is stationary exactly when $\mathbb{E}_B[s(b)] = 0$. Equivalently, it is a stationary point of

$$\Phi(x) = \mathbb{E}_B[\log f_\kappa(x, b; \sigma)]\,. \tag{7}$$

If the strategy domain is compact, Φ is continuous, and its maximizer is interior, such a point exists; if Φ is strictly concave, it is unique. On a boundary, Equation 6 is replaced by the projected first-order condition. These conditions do not certify a global Nash equilibrium or a separated solution, which require best-response or numerical checks.

Jacobian and player-separating modes. At x°, let

$$\bar{H} = \mathbb{E}_B[H(b)]\,, \qquad C = \mathbb{E}_B\big[s(b)s(b)^\top\big]\,, \qquad K = -\bar{H}. \tag{8}$$

At a stationary point C is the score covariance. Differentiating Equation 5 once more and substituting $G_i = 1/N$ gives identical diagonal and off-diagonal blocks,

$$A := \frac{\partial F_i}{\partial x_i^\top} = \frac{N-1}{N^2}\bar{H} + \frac{(N-1)(N-2)}{N^3}C,$$
$$\Delta := \frac{\partial F_i}{\partial x_j^\top} = -\frac{N-2}{N^3}C, \qquad j \neq i. \tag{9}$$

Consequently, the full $NL \times NL$ Jacobian has A on its diagonal and Δ off its diagonal. Equivalently,

$$J = I_N \otimes A + (\mathbf{1}\mathbf{1}^\top - I_N) \otimes \Delta.$$

The player-index space splits into the collective direction $\mathrm{span}\{\mathbf{1}\}$ and its orthogonal, sum-zero subspace. The corresponding $L \times L$ blocks are

$$J_+ = A + (N-1)\Delta = -\frac{N-1}{N^2}K,$$
$$J_- = A - \Delta = \frac{N-1}{N^2}\left[-K + \frac{N-2}{N-1}C\right], \tag{10}$$

where J_+ appears once and J_- appears with multiplicity $N-1$. To verify the decomposition, choose an orthogonal matrix P whose first row is $\mathbf{1}^\top/\sqrt{N}$. Then $P\mathbf{1}\mathbf{1}^\top P^\top = \mathrm{diag}(N, 0, \ldots, 0)$, and conjugating J by $P \otimes I_L$ yields $\mathrm{diag}(J_+, J_-, \ldots, J_-)$.

Assume $K \succ 0$ on the relevant space. The collective block is then negative definite. Congruence of the separating block with

$$-I + \frac{N-2}{N-1}K^{-1/2}CK^{-1/2}$$

shows that $J_- \prec 0$ exactly when the depth in Equation 3 is below one. Thus $D_\kappa < 1$ gives local asymptotic stability under $\dot{x}_i = F_i(\mathbf{x})$, $D_\kappa > 1$ gives a separating direction with positive linear growth, and $D_\kappa = 1$ is the zero-eigenvalue boundary. This proves the stability claim in Section 2.1. In one dimension the same condition is

$$-\mathbb{E}_B[H(b)] > \frac{N-2}{N-1}\mathbb{E}_B\left[s(b)^2\right]. \tag{11}$$

Two-player and Gaussian consequences. When $N = 2$, the score-scatter terms and Δ vanish: $J_+ = J_- = -K/4$. Hence any interior symmetric stationary point with $K \succ 0$ is locally asymptotically stable.

For the elliptical Gaussian kernel, let $Q \succ 0$ and

$$\log f_\kappa(x, b; \sigma) = -\frac{(b-x)^\top Q(b-x)}{2\sigma^2} + \mathrm{const.}$$

Then $s = Q(b-x)/\sigma^2$ and $H = -Q/\sigma^2$. Equation 6 gives $x^\circ = \mathbb{E}_B[b]$ for an unconstrained interior solution. Writing $\Sigma_B = \mathrm{Cov}_B(b)$,

$$K = \frac{Q}{\sigma^2}, \qquad C = \frac{Q\Sigma_B Q}{\sigma^4},$$

so the stability boundary has the closed form

$$\sigma^* = \left[\frac{N-2}{N-1}\lambda_{\max}\left(Q^{1/2}\Sigma_B Q^{1/2}\right)\right]^{1/2}. \tag{12}$$

The symmetric Gaussian profile is stable for $\sigma > \sigma^*$ and separating-unstable for $\sigma < \sigma^*$. With $Q = I$ this becomes $\sigma^* = \sqrt{\frac{N-2}{N-1}\lambda_{\max}(\Sigma_B)}$; in one dimension it reduces to $\sigma^* = \sqrt{\frac{N-2}{N-1}}\,\sigma_B$.

Exact reduction for co-located groups. Suppose each effective player p is represented by r equal-size clones with the same position and kernel value $f_p(b)$. Each clone receives $f_p(b)/(r\sum_q f_q(b))$, and summing its group gives

$$\sum_{a=1}^{r} G_{p,a}(b) = \frac{r f_p(b)}{r\sum_q f_q(b)} = \frac{f_p(b)}{\sum_q f_q(b)}. \tag{13}$$

Therefore equal-size groups that remain co-located have exactly the shares and summed utilities of the reduced game with one player per group, so taking $r = 2$ duplicates each solved location without perturbing the effective allocation. With the two-player stability result above, this motivates initializing the experts in pairs. The reduction is an identity on the co-located subspace; two-player stability does not prove that two unconstrained members embedded in a larger game would remain together. The reduction also fails when group sizes differ. Appendix C shows why the implemented update preserves equal co-location and distinguishes the clone-level gradient from a jointly deviating reduced-game player.

C General-kernel game-gradient updates

The position update uses the payoff in Equation 1, not language-model loss. Let $s_i(b) = \nabla_{x_i} \log f_\kappa(x_i, b; \sigma)$. The exact population vector field derived in Appendix B is

$$\nabla_{x_i} u_i(\mathbf{x}) = \mathbb{E}_B[G_i(1 - G_i)s_i(b)]\,. \tag{14}$$

At round t, a batch $b_1, \ldots, b_M$ replaces the resource expectation, giving the stochastic field

$$\widehat{g}_i^{(t)} = \frac{1}{M}\sum_{m=1}^{M} G_i(\mathbf{x}^{(t)}, b_m)\big[1 - G_i(\mathbf{x}^{(t)}, b_m)\big]\, s_i(b_m). \tag{15}$$

For batches sampled from B, this is an unbiased estimator of Equation 14.

General update. The method applies the simultaneous projected step

$$\mathbf{x}^{(t+1)} = \Pi_{\mathbb{B}}\Big(\mathbf{x}^{(t)} + \bar{\eta}_t P_{T\mathbb{B}}\widehat{g}^{(t)}\Big)\,. \tag{16}$$

Each influence family supplies its own score $s_i(b)$ using the closed forms in Appendix E; no centroid assumption is required. $P_{T\mathbb{B}}$ removes components normal to the domain (notably the all-ones direction on the simplex), and $\Pi_{\mathbb{B}}$ restores feasibility. A single population-wide effective step size $\bar{\eta}_t$ is reduced when necessary so that the largest player movement respects the step cap. Before the final feasibility projection, this scalar preserves the joint game-gradient direction and the relative magnitudes of the players' updates. The kernel-family experiments use this rule with the observed round batch.

Because the batch defines the empirical resource measure in Equation 15, a changing prompt distribution changes the vector field in a predictable way. Repeated projected steps can therefore track a moving resource landscape without learning a new router objective.

Gaussian centroid as a special case. For an isotropic Gaussian kernel, $s_i(b_m) = (b_m - x_i)/\sigma^2$. Define

$$w_{im} = G_i(\mathbf{x}, b_m)\big[1 - G_i(\mathbf{x}, b_m)\big], \quad W_i = \sum_{m=1}^{M} w_{im}, \quad \widehat{x}_i = \frac{\sum_m w_{im} b_m}{W_i}. \tag{17}$$

Then

$$\widehat{g}_i = \frac{W_i}{M\sigma^2}(\widehat{x}_i - x_i). \tag{18}$$

The reported principal Gaussian arms evaluate this direction through the blended centroid step

$$x_i^{(t+1)} = (1-\beta)x_i^{(t)} + \beta\widehat{x}_i = x_i^{(t)} + \eta_i^{(t)}\widehat{g}_i^{(t)}, \qquad \eta_i^{(t)} = \frac{\beta M\sigma^2}{W_i} > 0, \tag{19}$$

with $\beta = 0.5$. This specialization is exactly parallel to the empirical Gaussian utility gradient and has the same stationary points whenever $W_i > 0$. Its step size is an agent- and batch-adaptive positive rescaling rather than the population-wide learning rate in Equation 16, so the two implementations do not have identical step magnitudes.

The direct general-kernel rule evaluates Equation 15 for every clone and every prompt in the round batch. Likewise, the dense Gaussian specialization evaluates w_{im} on the whole batch. Neither rule truncates its position mass to the top-k training assignments or renormalizes it across retained pairs.

The reported hard-routing Gaussian arm uses the corresponding stochastic centroid estimator. If Z_{im} indicates that prompt m was sampled for clone i, then

$$\mathbb{E}[Z_{im}(1 - G_i) \mid b_m] = G_i(1 - G_i). \tag{20}$$

The selected prompts are therefore weighted by $(1 - G_i)$: their unnormalized centroid mass estimates the same theoretical coefficient, but the finite-batch ratio is noisier than evaluating the full batch. Updated positions affect routing beginning with the next round.

Why pairs remain co-located. For two equal clones a and a' at the same position, $G_a(b_m) = G_{a'}(b_m)$ for every prompt. Their rows of w_{im}, their centroid targets, and therefore their independently computed updates are identical. Exact co-location is an invariant of the dense theory-matched dynamics; the implementation does not average a pair and write one forced position back to both members.

If G_p denotes the summed share of a two-clone pair, then each member has $G_i = G_p/2$ and follows the coefficient $G_i(1 - G_i)$. A hypothetical player that moved the two clones jointly to maximize their summed utility would instead have coefficient $G_p(1 - G_p)$. Equation 13 therefore gives an exact reduction of allocations and summed utilities on the co-located subspace, but not an identity between these two gradient fields. The implemented update matches each clone's own utility gradient in the fourteen-player game, while routing and adapter training operate on seven aggregated pairs.

Group structure of the position game. Figure 6 solves the position game alone, with no adapter training, for eight individual players under an isotropic Gaussian kernel on the empirical measure of all 28,355 prompts, for which $\sigma^* = 0.356$. Each of 41 reaches from $2\sigma^*$ to $0.05\sigma^*$ is solved independently from the same seeded start with the blended centroid step of Equation 19, for up to 8,000 steps and without perturbation, on three seeds. Players within $\max(0.075,\ 0.35\sigma)$ of one another are counted as one group. From free starts, each player at a random prompt, the eight players stay together down to σ^*, separate into five and three just below it and fragment into pairs and single players by $0.6\sigma^*$; below about $0.3\sigma^*$ many solves do not settle within the budget and the seeds disagree. From four co-located pairs, the players pass through $6 + 2$ and $4 + 2 + 2$ and settle into four pairs from about $0.7\sigma^*$ to $0.05\sigma^*$; every one of these solves converges and no pair separates, as the invariance above requires.

D TRAIT-SPACE GEOMETRY

The cube retains seven independently calibrated coordinate levels, whereas the simplex encodes their relative composition. For calibrated scores $c \in [0, 1]^7$, the latter uses

$$b_j(T) = \frac{\exp(c_j/T)}{\sum_{\ell=1}^{7} \exp(c_\ell/T)}, \qquad T > 0. \tag{21}$$

Figure 7 shows the held-out resource density in these spaces. Each ternary view conditionally renormalizes its three named components; the other four components of the full seven-axis representation are not assumed to be zero. At $T = 1$ the bounded logits confine each displayed ternary component to approximately $[0.155, 0.576]$, crowding the prompt cloud into the interior. At $T = 0.35$ that possible range expands to approximately $[0.028, 0.897]$, and the measured support spreads toward the corners.

This geometry affects how peaked an influence kernel must be to distinguish competitors. For the reported Gaussian arms with identity shape matrix, the allocation log-odds obey

$$\log \frac{G_i(b)}{G_j(b)} = \frac{\|b - x_j\|^2 - \|b - x_i\|^2}{2\sigma^2}. \tag{22}$$

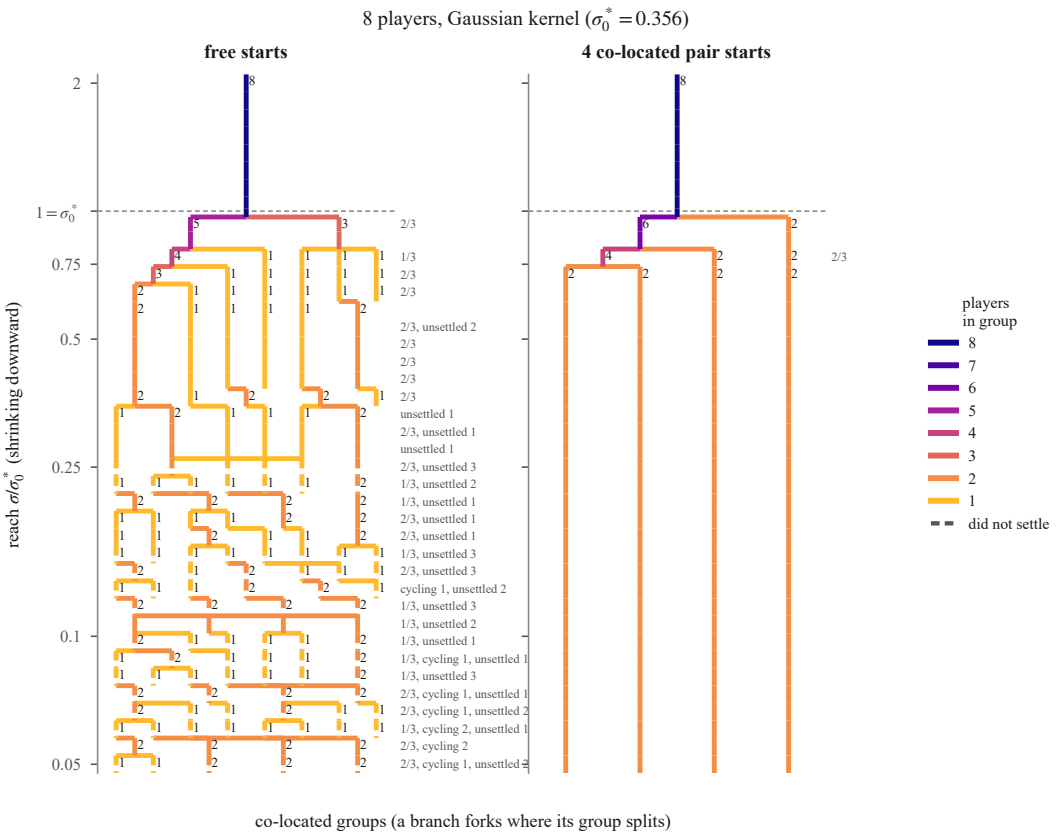

Figure 6: Co-located groups of eight players in the Gaussian position game against the reach, with no adapter training. Each branch is a group of co-located players, colored and labeled by its member count, and forks where the group splits; the reach shrinks downward. Left, every player starts at a random prompt; right, the players start as four co-located pairs. Both trees use the same 41 reaches and three seeds per reach, and each row draws the most common partition. Dashed branches did not settle within 8,000 steps; the right margin gives seed agreement and the number of seeds that cycled or did not settle.

When resources are concentrated where rival positions have similar affinity, a broad kernel makes these odds small and spreads weight across experts. Narrowing effective reach, increasing kernel tempering, or using a less compressive trait projection can produce a more discriminative allocation. This can lower expected single-expert NLL when diffuse routing gives weight to weaker experts, as the sharpness-matched $T = 1$ comparison and the $T = 0.35$ end-to-end arms in Figure 3 suggest. A compact cloud already close to one expert can route sharply, and extra peaking need not improve probability mixing or preserve adequate training mass for every expert.

With a fixed trait encoder and expert family, reach σ or a tempering exponent can be validated as router controls while monitoring allocation entropy, expert loads, and the intended inference NLL. Trait temperature T is a separate geometry lever, but changing it also changes the position game; positions must be re-solved or recalibrated, and the resulting router tested against the same frozen experts. The reported $T = 0.35$ arms instead retrained experts under the new geometry, so they do not establish a free post-training gain.

An end-to-end simplex experiment shows higher router–oracle agreement (about 0.87 versus 0.71) and mixture capture (91.5–92.6% versus 82.0–84.9%) than its cube counterpart. A bidirectional frozen-expert cross-routing control qualifies that comparison. Simplex-trained experts score 1.9383 NLL under their own geometry versus 2.0739 under cube routing, while cube-trained experts score 1.9465 under cube versus 2.0244 under simplex, so each family prefers the geometry in which it trained. Absolute reach and a density-estimation routine also differ between the two arms, and one reverse solve did not converge.

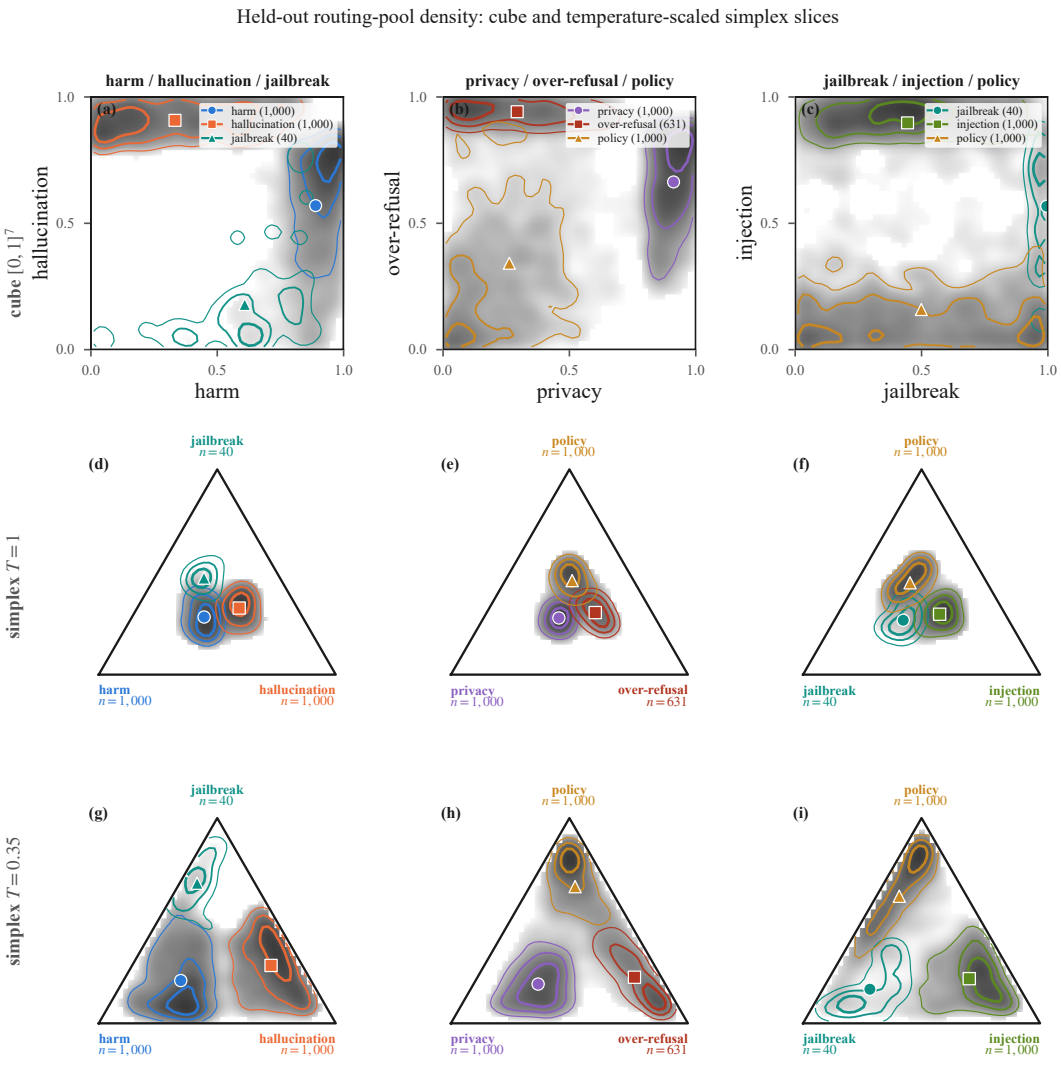

Figure 7: The trait projection controls how much geometric separation the kernel can exploit. Columns show three benchmark groupings among 5,671 held-out routing prompts: two-coordinate cube projections (top), followed by conditional three-component simplex views at $T = 1$ (middle) and $T = 0.35$ (bottom). Grey is the combined prompt density; colored contours and markers denote benchmark origin, with each class normalized to its own density mode and therefore not encoding prevalence. Each triangle renormalizes only its three named components; the other four coordinates are not assumed to be zero. The $T = 1$ transform compresses prompts near the simplex center, whereas $T = 0.35$ spreads them toward distinct vertices. These are resource densities, not expert territories, and demonstrate why denser geometry requires a more peaked influence kernel to obtain selective routing.

Appendix E gives the kernel closed forms and the distinct ways their parameters control reach and sharpness.

E Influence kernels and design rationale

All kernel families used by the game specify a positive influence $f_\kappa(x, b)$; Equation 1 normalizes their values across players. The solver uses the position score $s = \nabla_x \log f_\kappa$ and log-Hessian $H = \nabla_x^2 \log f_\kappa$ to evaluate the game gradient and the local instability criterion. Below $\sigma > 0$ is competitive reach, and all derivatives are with respect to x.

For the three distance families, let $M \succ 0$ be a fixed shape matrix, $Q = M^{-1}$, $v = b - x$, $q = Qv$, $d^2 = v^\top Qv$, and $u = \delta^2 + d^2$ with smoothing radius $\delta > 0$. Their implemented log-influences are

$$\log f_{\mathrm{G}}(x, b) = -\frac{d^2}{2\sigma^2}, \qquad \text{Gaussian,} \tag{23}$$

$$\log f_{\mathrm{H}}(x, b) = -\frac{\sqrt{u}}{\sigma}, \qquad \text{hyperbolic,} \tag{24}$$

$$\log f_{\mathrm{EP},\beta}(x, b) = -\frac{u^{\beta/2}}{\beta\sigma^\beta}, \qquad \text{exponential power,} \quad \beta \geq 1. \tag{25}$$

The Gaussian is the conventional smooth, constant-curvature reference; its omitted Gaussian-density normalizer is independent of position and cancels from G_i. The smoothed hyperbolic family gives a different radial tail and finite curvature when $b = x$. Exponential power varies tail shape within one family, so it tests whether allocation sharpness, rather than a nominal kernel label, accounts for routing differences. At $\beta = 1$ it is exactly hyperbolic; at $\beta = 2$ it differs from Gaussian only by the position-independent term $-\delta^2/(2\sigma^2)$, so allocation, score, and Hessian coincide. The constraint $\beta \geq 1$ preserves the log-concavity used in the local analysis. In the reported distance-kernel arms $M = I$; the usual smoothing radius is $\delta = 0.1$, with a smaller-radius hyperbolic control analyzed separately.

The closed-form derivatives of these three log-influences are

$$s_{\mathrm{G}} = \frac{q}{\sigma^2}, \qquad H_{\mathrm{G}} = -\frac{Q}{\sigma^2}, \tag{26}$$

$$s_{\mathrm{H}} = \frac{q}{\sigma\sqrt{u}}, \qquad H_{\mathrm{H}} = -\frac{Q}{\sigma\sqrt{u}} + \frac{qq^\top}{\sigma u^{3/2}}, \tag{27}$$

$$s_{\mathrm{EP},\beta} = \frac{u^{\beta/2-1}q}{\sigma^\beta}, \qquad H_{\mathrm{EP},\beta} = -\frac{u^{\beta/2-1}Q + (\beta-2)u^{\beta/2-2}qq^\top}{\sigma^\beta}. \tag{28}$$

The *Dirichlet* family treats a prompt as a composition on the open simplex: $b \in \Delta_\circ^{L-1}$ and $x \in \Delta^{L-1}$. With $\alpha_\ell = 1 + x_\ell/\sigma$ and $A = \sum_\ell \alpha_\ell = L + 1/\sigma$ on the simplex,

$$f_{\mathrm{D}}(x, b) = \frac{\Gamma(A)}{\prod_{\ell=1}^L \Gamma(\alpha_\ell)} \prod_{\ell=1}^L b_\ell^{\alpha_\ell - 1}. \tag{29}$$

The mode is parameterized by x. It tests a simplex-native, non-Euclidean influence law against the radial distance kernels on the same prompt geometry. Unlike the Gaussian prefactor, the Gamma normalizer depends on x and must remain in the allocation. Writing ψ and ψ_1 for the digamma and trigamma functions, the ambient derivatives implemented before tangent projection are

$$s_{\mathrm{D},\ell} = \frac{\log b_\ell - \psi(\alpha_\ell) + \psi(A)}{\sigma}, \qquad H_{\mathrm{D}} = \frac{\psi_1(A)\mathbf{1}\mathbf{1}^\top - \mathrm{diag}(\psi_1(\alpha_\ell))}{\sigma^2}. \tag{30}$$

The *product-Beta* family instead treats coordinates as separately bounded traits. For $x \in [0,1]^L$, $b \in (0,1)^L$, let $a_\ell = 1 + x_\ell/\sigma$ and $c_\ell = 1 + (1 - x_\ell)/\sigma$. Then

$$f_{\mathrm{PB}}(x, b) = \prod_{\ell=1}^L \frac{\Gamma(a_\ell + c_\ell)}{\Gamma(a_\ell)\Gamma(c_\ell)} b_\ell^{a_\ell - 1}(1 - b_\ell)^{c_\ell - 1}. \tag{31}$$

Each factor has mode x_ℓ, with its position-dependent normalizer retained. It is the bounded, coordinatewise contrast to the coupled Dirichlet composition. Its native density is on the cube; simplex experiments evaluate the same positive function on simplex-embedded coordinates, which does not make it a Dirichlet or a normalized simplex density. Its implemented derivatives are

$$s_{\mathrm{PB},\ell} = \frac{\log b_\ell - \log(1 - b_\ell) - \psi(a_\ell) + \psi(c_\ell)}{\sigma}, \qquad H_{\mathrm{PB}} = -\frac{\mathrm{diag}(\psi_1(a_\ell) + \psi_1(c_\ell))}{\sigma^2}. \tag{32}$$

The implementation also permits an optional *tempering* wrapper for any base family:

$$f_{\kappa,\tau}(x, b) = f_\kappa(x, b)^\tau, \qquad s_{\kappa,\tau} = \tau s_\kappa, \qquad H_{\kappa,\tau} = \tau H_\kappa, \qquad \tau > 0. \tag{33}$$

It provides a separate allocation-sharpness control. For exponential power it is equivalent to reach $\sigma/\tau^{1/\beta}$, but for Dirichlet and product-Beta the Gamma normalizers preclude that simple rescaling. This kernel exponent τ is distinct from the trait-projection softmax temperature T in Figure 7. Across families we solve reach at a matched instability depth and then measure prompt-level allocation sharpness.

Appendix Figure 8 expands the calibration result in Figure 3 (right). The main figure reports expected routing gain in nats; this diagnostic instead measures the fraction of available hindsight-oracle headroom captured. For each prompt let $H(G(b)) = -\sum_i G_i(b) \log G_i(b)$ and define $K_{\text{eff}} = \exp(\mathbb{E}_B[H(G(b))])$; lower K_{eff} means a sharper allocation, not more balanced expert loads. On six nondegenerate distance kernels at $T = 1$, the descriptive fit is

$$\widehat{\text{capture}}(\%) = 73.15/K_{\text{eff}} + 12.07, \tag{34}$$

with $r = 0.9995$. Two shape kernels were held out of that fit, and the $T = 0.35$ points test a changed trait projection. The $\beta = 4$ endpoint has collapsed positions.

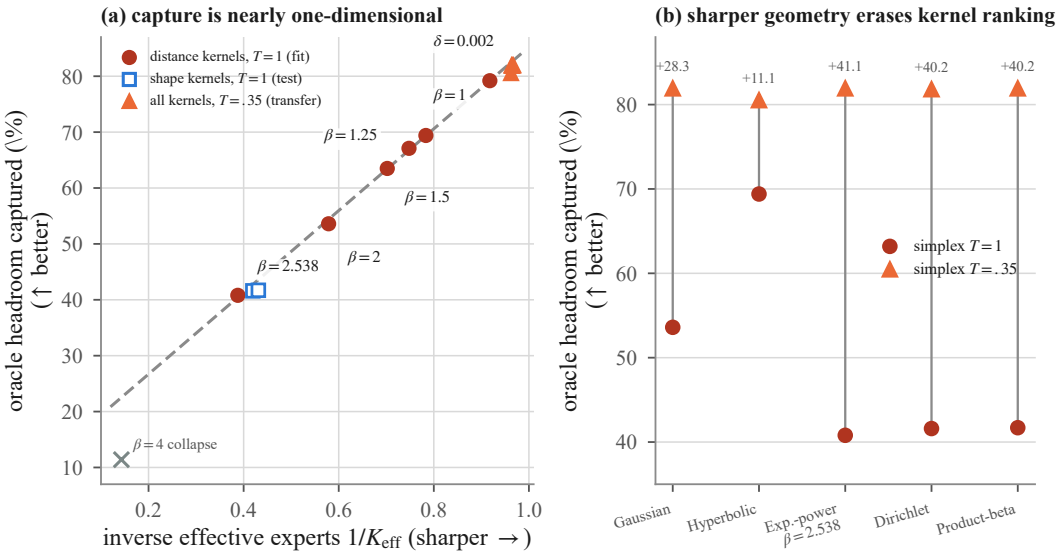

Figure 8: Oracle-headroom capture is governed primarily by the sharpness of the induced allocation. (a) The horizontal axis is inverse effective expert count, $1/K_{\text{eff}}$, computed from prompt-level allocation entropy; larger values mean more decisive routing. Captured headroom lies almost on a one-dimensional trend for six nondegenerate $T = 1$ distance kernels (dashed fit, $r = 0.9995$). Hollow shape-kernel points and $T = 0.35$ triangles were not used to fit the line but follow the same relationship. The $\beta = 4$ cross is a collapsed-position boundary and is excluded. (b) Sharpening the trait projection from $T = 1$ to $T = 0.35$ raises all five families to approximately 81% capture; labels give the percentage-point increases. The disappearance of the kernel ranking supports allocation sharpness, rather than family name or matched local depth alone, as the operative deployment quantity.

F Initialization and Position Controls

The naive adaptive, equilibrium adaptive and pinned starts describe where the pairs begin, and positions adapt during training under all three; fixed describes what is held constant afterward. The naive adaptive start used in the pair-count arms came from an approximate lattice-resource solve whose density estimator divided each cell by a nearest-prompt kernel value, flattening the resource measure. These positions are not purely random, but they are not a reliable equilibrium of the measured prompt distribution. The main text calls this the unsolved start; appendix figures and tables keep the label naive adaptive. The equilibrium adaptive start instead numerically solves the position game against the empirical prompt measure before adapter training.

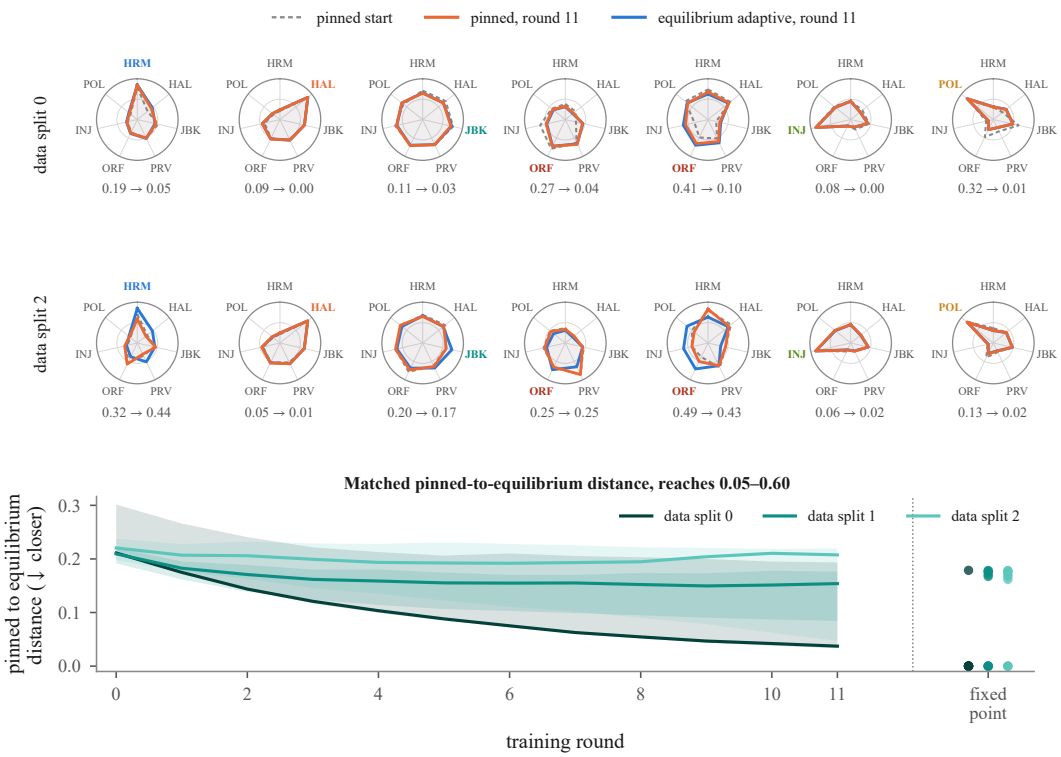

Figure 9: Pinned runs approach the equilibrium adaptive runs on one data split and stall on another. Top rows: the seven pairs at reach 0.50 on data splits 0 and 2, matched by nearest round-11 position. Grey dashed polygons are the pinned start, orange polygons the pinned run at round 11 and blue polygons the equilibrium adaptive run at round 11; the numbers under each radar give that pair's pinned-to-equilibrium distance at round 0 and at round 11. Bottom: matched distance per round, median (line) and range (band) over reaches 0.05–0.60 on each split. Dots at right are the distance between the two runs' fixed points when the position update is continued past round 11 without training.

Figure 10 compares these positions at two, five, and seven pairs; every plotted training trajectory begins from the naive adaptive start and ends at its round-11 position. The number of dominant represented trait axes grows from two to four to six, so additional pairs expand trait coverage rather than merely copy a single location. The direct NLL contrast between starts is in Figure 3.

The pinned start begins every reach on a data split from that split's $\sigma = 0.50$ unsolved configuration. It does not disable position updates, so positions adapt from it as they do from the equilibrium adaptive start. Figure 9 tracks the matched distance between pinned and equilibrium adaptive runs at the same reach. Neither kind of run is at a fixed point after 11 rounds, because the update uses a sampled batch and a constant blend of 0.5, but the equilibrium adaptive runs sit 0.012–0.019 from the fixed point of their full-batch update, while the pinned runs sit 0.038–0.155 from theirs and would need 79–413 further rounds to reach it. Continuing the update without training, which is exact because it never reads an adapter, carries 15 of the 27 non-degenerate pinned runs to the same fixed point as their equilibrium adaptive counterpart. The other 12, all at reaches of 0.50 or below and all but one on data splits 1 and 2, settle 0.16–0.18 away at a different fixed point. By round 11 the pinned runs close 78% of their initial distance on data split 0 but only 30% and 21% on splits 1 and 2. Section 4.1 reports the roughness and mean-gain comparison.

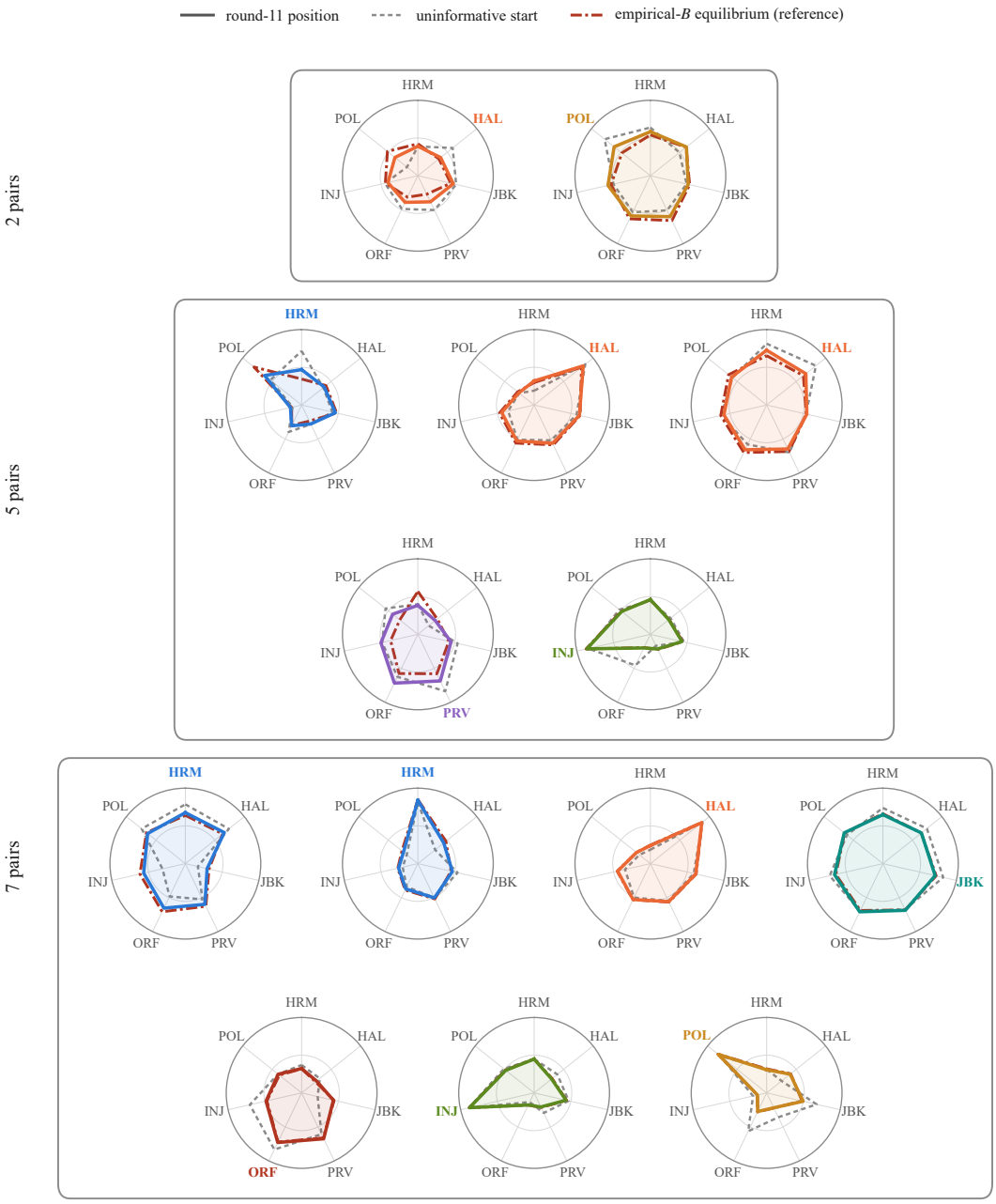

Figure 10: Additional pairs expand coverage of the seven-dimensional resource landscape. Rows contain every pair in the two-, five-, and seven-pair naive adaptive start arms; each radar radius is one trait coordinate. Grey dashed polygons are the naive adaptive starts these runs used, solid colored polygons are the round-11 positions, and red dash–dot polygons are counterfactual empirical-resource equilibria rather than training starts. Bold colored labels mark each endpoint's dominant trait. The number of represented dominant traits grows from two to four to six, showing that extra capacity produces new positional roles rather than merely duplicating one expert.

G ROUTING-RULE ABLATIONS

The same game allocation can be exposed to adapter training in materially different ways. We separate three choices: how many of the seven pairs receive each prompt, whether a retained pair is selected deterministically or sampled, and whether its loss is scaled by its allocation share or given unit weight. These choices do not change the location-game solution, but they determine how much of its graded allocation signal reaches the adapters. Figure 11 compares five end-to-end arms on the same data seed 0 split and against the same pooled generalist. Thus this is a training-and-routing ablation, not a post-hoc recombination of one fixed expert family.

Routing rules as training distributions. Every rule acts at the pair level on the pair shares $G_p(b)$, which sum to one over the $P = 7$ pairs. For each prompt b a rule chooses a support $\mathcal{S}(b)$ of pairs that train on it and a loss weight $w_p(b) \geq 0$ for each, with $w_p(b) = 0$ when $p \notin \mathcal{S}(b)$. The trainer then fits pair p's adapter to the weight-normalized mean of its per-example response-token cross-entropy $\ell(\theta_p; b)$ over the round batch,

$$\hat{L}_p(\theta_p) = \frac{\sum_{m=1}^{M} w_p(b_m)\, \ell(\theta_p; b_m)}{\sum_{m=1}^{M} w_p(b_m)}. \tag{35}$$

As the batch grows this approaches the expected loss under the pair-specific training distribution

$$B_p(b) = \frac{\bar{w}_p(b)\, B(b)}{\mathbb{E}_B[\bar{w}_p(b)]}, \qquad \bar{w}_p(b) = \mathbb{E}\big[w_p(b) \mid b\big], \tag{36}$$

where the inner expectation is over any randomness in the rule. In expectation, then, a routing rule reaches training only through $\bar{w}_p$, and two rules with the same $\bar{w}_p$ differ only in the variance of the weights they realize. Table 3 gives $\bar{w}_p$ for each arm.

Selection. The top-k rules differ in how the support is chosen. With $g_p(b)$ independent standard Gumbel noise and $\tau \in \{0, 1\}$,

$$\mathcal{S}_k(b) = \mathrm{top}_k\big\{\log G_p(b) + \tau\, g_p(b)\big\}_{p=1}^{P}. \tag{37}$$

At $\tau = 0$ this is the deterministic set $T_k(b)$ of the k largest shares. At $\tau = 1$ it draws k distinct pairs by successive sampling without replacement in proportion to $G(b)$, and for $k = 1$ it reduces to a single categorical draw $\hat{p}(b) \sim G(b)$, the Gumbel-max trick. Under sampling, pair p is included with probability $\pi_p^{(k)}(b) = \Pr[p \in \mathcal{S}_k(b)]$, which satisfies $G_p(b) \leq \pi_p^{(k)}(b) \leq 1$, because a pair drawn first is always included, and $\sum_p \pi_p^{(k)}(b) = k$.

Table 3: Expected loss weight $\bar{w}_p(b)$ that each routing rule gives pair p on prompt b, which by Equation 36 fixes the pair's training distribution in expectation. $T_3(b)$ is the set of the three largest shares, $Z_3(b) = \sum_{r \in T_3(b)} G_r(b)$, and $\pi_p^{(3)}(b)$ is the probability that pair p is among three pairs drawn without replacement in proportion to $G(b)$.

Arm	Pairs per prompt	Selection	Loss weight	$\bar{w}_p(b)$
Soft all-7, weighted	7	all pairs	G_p	G_p
Soft top-3, weighted	3	three largest	G_p/Z_3	$\mathbf{1}[p \in T_3]\, G_p/Z_3$
Soft top-3, unit	3	three largest	1	$\mathbf{1}[p \in T_3]$
Hard top-3 (sampled)	3	drawn $\propto G$	1	$\pi_p^{(3)}$
Hard top-1 (SFT)	1	drawn $\propto G$	1	G_p

Soft and hard. A rule is *hard* when the pairs that train on a prompt are drawn at random from the allocation and each drawn pair trains at unit weight, and *soft* when both the support and the weights are fixed functions of the allocation. The two are linked by expectation: replacing a hard rule's random inclusion indicator by its expectation gives a soft rule with the same $\bar{w}_p$, and hence the same training distribution in Equation 36. The one exact pair in this ablation is hard top-1 and dense soft routing. Under hard top-1, $w_p(b) = \mathbf{1}[\hat{p}(b) = p]$, so

$$\bar{w}_p(b) = \Pr\big[\hat{p}(b) = p\big] = G_p(b), \tag{38}$$

which is exactly the dense share-weighted weight. Both rules therefore train pair p toward $B_p \propto G_p\, B$, whose total mass $\mathbb{E}_B[G_p(b)]$ is the pair's own game payoff u_p from Equation 1: the share a pair wins in the game is the share of the training data it receives. What differs is the realization. The hard weight on each prompt is a Bernoulli draw with variance $G_p(1 - G_p)$, so a hard pair trains at full weight on a random subset of about $\mathbb{E}_B[G_p]\, M$ prompts per round, roughly a seventh of the batch when loads are balanced, while a soft pair trains on all M prompts at fixed fractional weight. Because the two rules share an expected training distribution, the 0.085-nat gap between them in Figure 11 comes from how that distribution is realized, not from what it is.

The three top-3 rules sit between these extremes, and none is the soft counterpart of another. Deterministic unit weighting, $\mathbf{1}[p \in T_3(b)]$, keeps only membership: a pair gets zero weight on a prompt as soon as its share falls to fourth place, and full weight inside the top three however unequal the three shares are. Share weighting keeps the grading inside the support but inflates every retained weight by $1/Z_3(b) \geq 1$. Sampled top-3 trains toward $\pi_p^{(3)}$, which is continuous in the shares and bounded below by G_p; its soft counterpart would weight by $\pi_p^{(3)}$, which neither deterministic top-3 rule does.

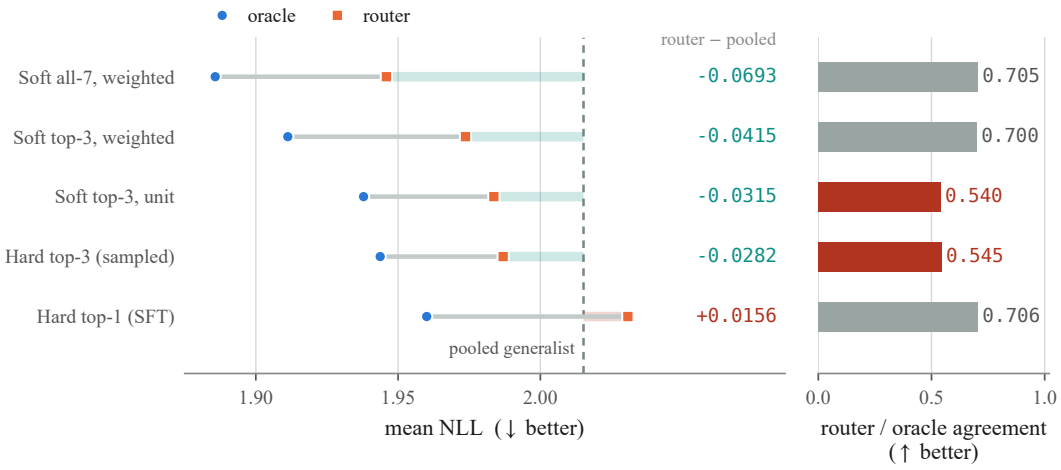

Figure 11: Dense, share-weighted use of the game allocation is necessary for the strongest routing result on data seed 0. The left panel compares each arm's expected single-expert NLL (orange square) with its hindsight-oracle NLL (blue circle) and the common pooled generalist (vertical dashed line; lower is better). Printed margins are router minus pooled, so negative values are improvements. Moving from dense all-seven routing to top-three support, then removing share weights or sampling the three retained pairs, progressively weakens both learned experts and routing; hard top-1 is worse than pooling. The right panel shows why oracle agreement alone is misleading: hard top-1 agrees with the best expert as often as the dense arm but has the worst NLL.

Dense share-weighted routing gives the lowest single-expert expected NLL, improving over the pooled model by 0.0693 nats. Retaining only the top three shares reduces the gain to 0.0415 nats. At fixed top-three support, replacing share weights with unit weights reduces it again to 0.0315 nats, and drawing the three pairs in proportion to their shares instead of taking the three largest gives 0.0282 nats. Hard top-1 is the only arm worse than pooling, by 0.0156 nats. The oracle endpoints worsen in the same order, indicating that sparsification changes the experts learned during training rather than only degrading the final selector. Dense routing is consequently a substantive part of the method: it preserves the continuous competitive signal through expert learning, at the cost of training every pair on every prompt.

Router–oracle agreement is not sufficient to choose among these designs. Hard top-1 has the highest agreement (0.706) yet the worst NLL, while dense share-weighted routing has nearly identical agreement (0.705) and the best NLL. Agreement records only which pair wins; it ignores the competence of that pair and the loss incurred on disagreements. The game therefore affords both a ranking and a calibrated allocation over experts, and the latter is lost when its shares are collapsed too early to a hard decision.

H INTERPRETABILITY ACROSS DATA SPLITS

Table 4 reports the interpretability measurements of Section 4.3 for the principal seven-pair arm on each of three independently trained data splits.

Table 4: Game shares predict the specialization learned by the adapters, although held-out expert losses were never used to fit the router. Top 1 and top 3 are the fractions of held-out prompts whose best expert appears among the largest one or three shares. The span column is the fraction of the uniform-to-oracle NLL gap the allocation recovers.

Data seed	Top 1	Top 3	Spearman	Uniform-to-oracle span
0	0.711	0.962	0.617	69.0%
1	0.728	0.961	0.594	68.1%
2	0.719	0.956	0.597	69.2%
Chance	0.143	0.429		

I MODEL SIZE AND FAMILY ABLATIONS

These checks ask whether the same game-derived coordination mechanism remains useful when the base language model changes. They are not tests of the positional bifurcation threshold, which depends on the prompt-resource geometry and game configuration rather than model parameter count alone. Figure 12 collects two distinct experiments.

The Qwen2.5 size ladder in panel (a) uses five pairs, an equilibrium adaptive start, and the same trait geometry across 1.5B, 3B, 7B, 14B, and 32B models. Each rung is compared with its own pooled generalist. Absolute held-out NLL falls with model size for both pooled and routed systems, while sequence-mixture oracle-headroom capture declines from 87.4% to 84.2%. Routing improves on the pooled comparator at every rung, but the available oracle gap narrows. There is one run per rung, the 7B run needed 48 non-finite-gradient guard updates, and the 32B run used an off-protocol microbatch configuration.

Panel (b) uses the existing seven-pair Qwen2.5-1.5B, Llama-3.2-1B, and Gemma-3-1B arms. Their learned mixtures each improve over the respective model's pooled control, so the allocation is not specific to one base-model family. The plotted values are within-model NLL differences, since raw per-token NLLs cannot be ranked across different tokenizers. Model family, tokenizer, and parameter count vary together, and these arms do not share the five-pair equilibrium adaptive start protocol of panel (a), so this is limited portability evidence rather than a ranking of architectures.

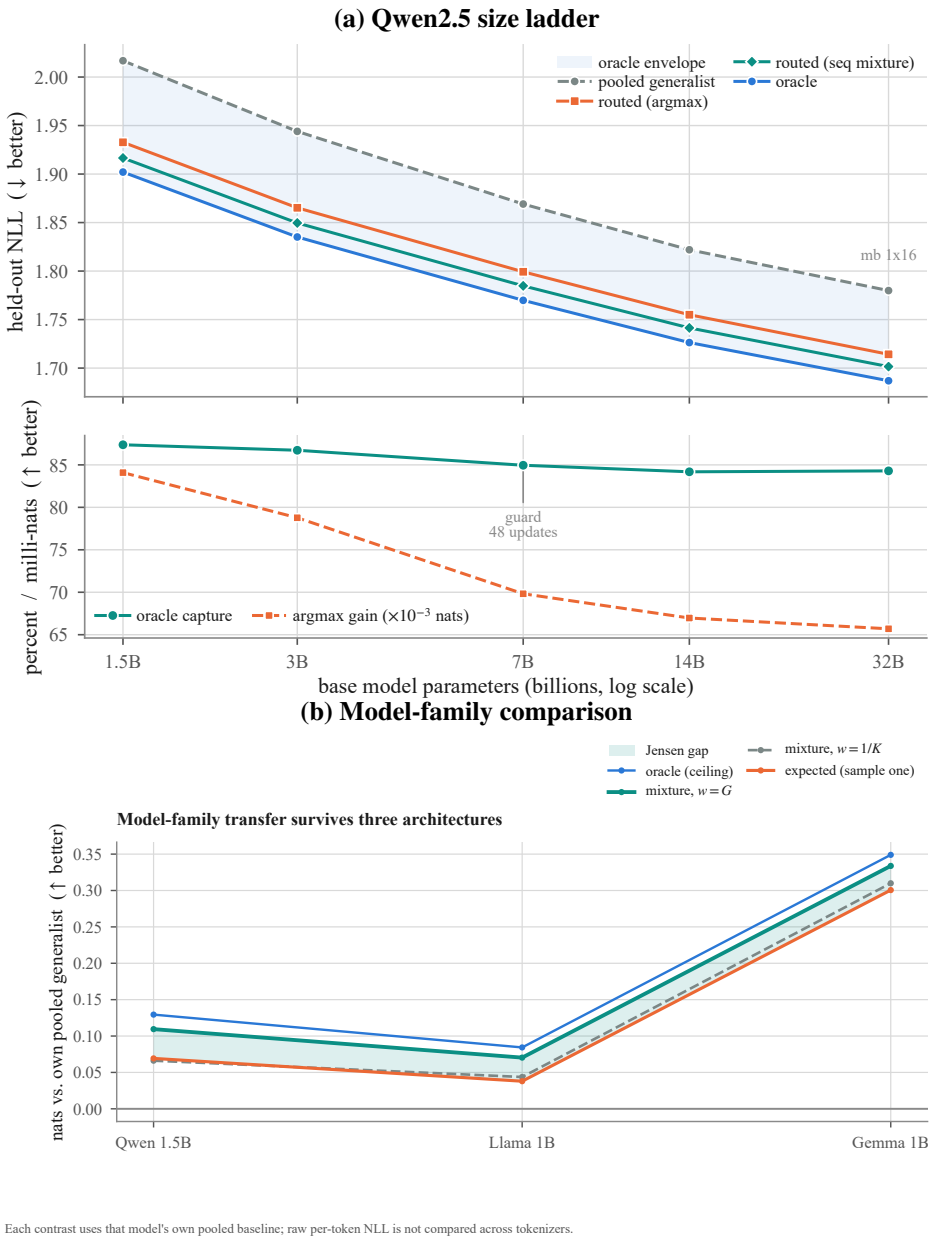

Figure 12: The game-derived allocation transfers across model scale and family, but these ablations are portability checks rather than architecture rankings. (a) Across five Qwen2.5 sizes, the upper plot compares pooled-generalist, routed-argmax, game-weighted sequence-mixture, and hindsight-oracle NLL (lower is better); the shaded region is the available oracle envelope. The lower plot shows that the mixture captures 84–87% of that envelope at every size while argmax gain declines. The 7B guard intervention and off-protocol 32B microbatch are annotated, and each size has one run. (b) For Qwen, Llama, and Gemma, every quantity is an NLL gain over that family's own pooled adapter: the game-weighted mixture improves all three and remains below the oracle ceiling, while the shaded Jensen gap separates probability mixing from sampling one expert. Raw NLL is not compared across tokenizers, and panels (a) and (b) use different pair-count and initialization protocols.

J BASELINE FAMILIES AND DEEP ENSEMBLES

Table 5 condenses a ten-family implementation audit. Every family has a held-out scored artifact; "measured" does not mean an exact reproduction of its reference implementation. The comparisons answer different questions: routing nulls hold experts fixed, assignment controls retrain them, and the jointly trained adapter mixtures are single models, so we do not sort all their NLLs into one leaderboard. The same-expert nulls and validation-trained frozen linear gate are controls defined for this study, not reproductions of named methods; the other rows cite the work motivating each baseline.

Table 5: Status and interpretive role of the ten baseline families. All are measured, and the qualifications are part of each result. The shaded first row is INFLENS, the reference each family is compared against.

Family	Comparison made	Salient qualification
INFLENS (ours)	Seven game-positioned rank-16 LoRA pairs; the solved allocation weights training and routes inference, with no fitted gate.	Seed-0 split: completed-sequence mixture NLL 1.9059 and hard commitment 1.9298, against 2.0152 pooled.
Same-expert nulls	Uniform mixture, shuffled expert identities, uniform random hard choice, game allocation, and hindsight oracle on fixed expert losses.	Separates expert diversity from the prompt–expert mapping; one-split offline analysis.
Frozen gate	Validation-fitted linear soft-target and best-expert gates on frozen experts, replicated on four splits.	Stronger game result under single-expert commitment; full probability mixing is unresolved.
BASE-style assignment	Proportional versus quota-balanced assignment (Lewis et al., 2021) crossed with fixed versus strategically updated positions.	All four cells start at the same equilibrium; entropic OT is not implemented.
Balanced c-BTM	Balanced versus unconstrained train-only clusters (Gururangan et al., 2023); dense, top-3, and hard centroid inference.	Fixed-partition clustering, not the online soft-k-means position control.
EWoRA-style	Seven jointly trained rank-16 components (Kohli et al., 2025) with a dense trait-conditioned gate versus a pooled rank-112 adapter.	Mechanism-level implementation; smaller micro-batch at matched effective batch size.
AdaMix	Independent versus shared down-projections (Wang et al., 2022) with stochastic routes, consistency, and post-training merge held fixed.	Tests shared structure at this training budget, not longer published schedules.
MoLoRA	Hidden-state token/layer gates (Zadouri et al., 2024), dense and top-3, versus corresponding prompt-level trait gates.	Routers start at zero; auxiliary load loss is averaged globally across layers.
MoDULA-Res	Universal adapter folded into the base, then residual experts and a staged router (Ma et al., 2024).	Genuine residual construction; improvement is relative to its own stage-1 pooled model.
KnOTS / TIES / DARE	Validated weight-space merges (Stoica et al., 2025; Yadav et al., 2023; Yu et al., 2024) from label-trained and game-trained experts, refactored at serving ranks 16 and 112.	Forty-two candidates per expert source; even the best game-expert merge loses to pooling.
Pooled-seed ensemble	Uniform probability mixture (Lakshminarayanan et al., 2017) of four pooled rank-16 adapters trained on the same manifest.	Distinct training seeds, but cumulative LoRA means initialization varies only at round 0.

The *fixed-expert* controls separate two sources of value. At seven pairs, the game has a $+0.1095$-nat completed-sequence mixture gain over pooling; uniform mixing of the same experts retains $+0.0661$, while shuffling expert identities gives -0.0671 and uniform random hard routing gives -0.4144. The experts are useful without the game router, but the learned identity-to-trait correspondence adds 0.0434 nats beyond a uniform mixture. A frozen linear gate trained from full validation expert-NLL vectors is a more demanding alternative. Across four splits the game wins all four expected-NLL (single-expert commitment) comparisons by 0.071–0.090 nats; under completed-sequence probability mixing the sign divides two-to-two, with at most 0.0017 nats separation. This supports the game's allocation as a competitive hard router, not a general accuracy advantage over learned soft gates.

Table 6: Seven-pair factorial control; lower NLL is better. Every cell starts from the same solved equilibrium and shares the reach, split, and pooled comparator. The shaded row is the INFLENS configuration.

Positions	Assignment	Expected NLL	Mixture NLL
Strategic updates (INFLENS)	**Proportional**	**1.9426**	**1.9059**
Fixed	Proportional	1.9425	1.9064
Fixed	Balanced	2.0294	1.9847
Strategic updates	Balanced	2.0297	1.9827

The *assignment* controls narrow the game-theoretic claim. In the BASE-style 2×2 of Table 6, which does not test BASE's original efficiency setting, quota balancing costs about 0.087 nats of expected NLL regardless of whether positions continue to move, whereas fixing positions at the solved equilibrium costs at most 0.0003 nats in expected NLL at this operating point.

The independent c-BTM comparison (Table 7) separates balance from sparse serving: imposing cluster quotas degrades all three inference modes and reduces the oracle envelope, while top-3 centroid inference is nearly indistinguishable from dense inference within either partition. These are single-split results; the 0.0004–0.0007 top-3 gaps are below the measured game-arm run-to-run standard deviation of 0.0017 nats.

Table 7: c-BTM partition and inference fan-out on the seed-0 split. Entries are completed-sequence mixture NLL (lower is better); the pooled rank-16 adapter scores 2.0152. Oracle envelope is pooled NLL minus hindsight expert NLL. The shaded INFLENS row applies the same three fan-outs to the game allocation over its seven trained pairs.

Partition	Dense	Top-3	Hard ($k = 1$)	Oracle envelope
INFLENS (game allocation)	**1.9059**	**1.9066**	**1.9298**	**+0.1290**
Unconstrained	1.9564	1.9568	1.9785	+0.0758
Balanced	1.9647	1.9654	1.9937	+0.0676

The *single-model architecture* baselines instead test whether other ways to spend adapter parameters reproduce specialization without separately trained, game-allocated experts. Their end-to-end model NLLs belong in Table 8, not beside a post-hoc sequence-mixture NLL: a jointly trained network is evaluated in one forward pass, whereas the latter mixes likelihoods from seven separately evaluated experts. EWoRA-style dense gating reaches 1.9712, within 0.0025 nats of the rank-112 pooled adapter's 1.9737 at matched stored LoRA rank. AdaMix's shared down-projection improves over independent projections by 0.0387 nats, but both remain worse than the pooled rank-16 control, and the shared arm's hindsight expert envelope is negative.

The gate-resolution contrast is also a gate-input contrast: MoLoRA's hidden-state token/layer gates score 2.0029 (dense) and 1.9955 (top-3), against 1.9712 and 1.9231 for the corresponding prompt-trait gates. Token routing is worse by 0.0317 and 0.0724 nats in these cells. MoDULA-Res scores 1.9909, a 0.0243-nat improvement over its stage-1 pooled adapter, and uniquely among these new architectures has a positive hindsight expert envelope, though it still trails pooled rank-112. Weight-space merging asks whether trained experts can be compressed into one adapter. None of 42 candidates from each of two expert sources matches pooling. The best game-expert KnOTS–TIES merge scores 2.1239, 0.109 nats worse than the pooled rank-16 model, and even its hindsight oracle scores 2.0758. These merge operators did not preserve the experts' conditional complementarity.

Independent training variance does not explain all of the routing gain, yet it can add to it. Table 9 holds data and architecture fixed while adding pooled-seed members. Four members recover only $+0.0305$ nats over one pooled adapter, versus $+0.1095$ for the seven-pair game mixture on the same split and estimator. The four pooled adapters have only a 0.0021-nat individual-NLL spread, and cumulative training weakens their independence relative to four from-scratch runs.

Five independently seeded games offer a separate extension of the mechanism. Each game routes over seven experts; the ensemble averages the resulting 35 component probabilities per token. It

Table 8: Single-model adapter baselines, seed-0 split, end-to-end model NLL. The two pooled rows are capacity references, not routing methods, and none of these numbers is directly comparable with the game's post-hoc completed-sequence mixture. The shaded INFLENS row therefore reports hard commitment, which also evaluates one adapter per prompt; its seven rank-16 pairs store the same LoRA rank as pooled rank-112.

Model	Model NLL
INFLENS, hard commitment to one pair	**1.9298**
Pooled rank-112	1.9737
EWoRA-style, dense trait gate	1.9712
MoDULA-Res	1.9909
MoLoRA, token/layer top-3	1.9955
MoLoRA, token/layer dense	2.0029
Pooled rank-16	2.0152
AdaMix, shared down-projection	2.1270
AdaMix, independent down-projections	2.1657

Table 9: Pooled-seed deep ensemble versus the game router, seed-0 split. All non-pooled entries here use a completed-sequence probability mixture rather than the token-level serving estimator; gains are relative to the pooled rank-16 adapter.

Policy	NLL	Gain over pooled
Pooled rank-16, one adapter	2.0152	—
Pooled-seed mixture, 2 members	1.9988	$+0.0164$
Pooled-seed mixture, 3 members	1.9898	$+0.0254$
Pooled-seed mixture, 4 members	1.9847	$+0.0305$
INFLENS mixture, 7 pairs	**1.9057**	$\mathbf{+0.1095}$

scores 1.8600 against 1.9103 for one re-scored game and 2.0152 for pooling (Table 10). Game allocation and deep ensembling are complementary: the additional five-game gain is 0.0503 nats on the standard token-level estimator. The completed-sequence estimator is optimistic by 0.0044 nats for seven components and 0.0140 for 35. The five-game result requires 35 component evaluations per prompt under the measured implementation, so the gain is not compute-neutral.

Across these baselines, the game's distinctive affordance is not a universal NLL win: a resource distribution, kernel, and solved positions specify an inspectable competitive allocation before the adapters are trained.

Table 10: Game deep ensemble on the seed-0 split. Token-level NLL mixes each observed token's conditional probabilities and is the serving estimator; completed-sequence NLL mixes whole-sequence likelihoods and is shown only for sensitivity.

Policy	Token-level NLL	Completed-sequence NLL
Pooled rank-16, one adapter	2.0152	2.0152
INFLENS, one game, 7 components	1.9103	1.9059
INFLENS, five-game ensemble, 35 components	**1.8600**	**1.8460**

K SAFETY BENCHMARKS

Safety is not part of the location game's payoff or a prerequisite for its allocation rule. The external evaluation asks a separate question: does game-derived routing change behavior after the adapters have been trained? We use the official HarmBench attack-success and StrongREJECT jailbreak graders. ConfAIde probes contextual privacy reasoning; IFEval measures strict instruction following, an adjacent capability check rather than a safety benchmark. The behavioral configurations also run

Table 11: Behavioral contrasts averaged across three independently trained data-split seeds. Cells are mean ± sample standard deviation across seeds ($n = 3$). Metrics are signed so that positive favors the first arm: HarmBench and StrongREJECT reverse their native direction, while IFEval does not. ConfAIde is omitted because its conditional error changes with response coverage.

Contrast	Setting	HarmBench	StrongREJECT	IFEval
Routing − pooling	Equilibrium adaptive, argmax	$+0.1243 \pm 0.0177$	$+0.0613 \pm 0.0214$	$+0.0364 \pm 0.0268$
	Equilibrium adaptive, mixture	$+0.1414 \pm 0.0354$	$+0.0493 \pm 0.0420$	$+0.0357 \pm 0.0295$
	Naive adaptive, argmax	$+0.1585 \pm 0.0748$	$+0.0951 \pm 0.0552$	$+0.0573 \pm 0.0280$
	Naive adaptive, mixture	$+0.1413 \pm 0.0509$	$+0.0676 \pm 0.0441$	$+0.0684 \pm 0.0322$
Mixture − argmax	Equilibrium adaptive	$+0.0172 \pm 0.0254$	-0.0120 ± 0.0245	-0.0006 ± 0.0117
	Naive adaptive	-0.0172 ± 0.0433	-0.0275 ± 0.0131	$+0.0111 \pm 0.0055$
Equilibrium adaptive − naive adaptive	Argmax	-0.0342 ± 0.0572	-0.0338 ± 0.0367	-0.0209 ± 0.0198
	Mixture	$+0.0001 \pm 0.0188$	-0.0183 ± 0.0054	-0.0327 ± 0.0070
Fine-tuned pooled − base	Standard	-0.5890 ± 0.0515	-0.5045 ± 0.0481	-0.1602 ± 0.0289

XSTest (Röttger et al., 2024), TruthfulQA (Lin et al., 2022), and BIPIA (Yi et al., 2025) under lexical proxy graders; those results are direction-only and are not reported here.

Table 11 separates four contrasts across three independently trained data splits. Relative to each split's fine-tuned pooled adapter, game-routed experts have lower HarmBench attack success and lower StrongREJECT harmfulness under both initializations and both decoding protocols: all 24 signed safety comparisons favor routing. The corresponding 12 IFEval comparisons also favor routing. We report the mean and sample standard deviation across the three seeds; the latter is a descriptive cross-seed band, not a confidence interval. There is one run per seed, StrongREJECT has only 60 cases, and HarmBench's retained case count differs slightly across splits. Neither probability mixing over experts nor the equilibrium adaptive start has a consistent behavioral advantage over argmax and the naive adaptive start, respectively.

The untrained base model is the second comparator. Its HarmBench attack-success rate is 0.0524 on the first split, versus roughly 0.55–0.68 after fine-tuning; no routed arm recovers that level. The base model also over-refuses many benign XSTest prompts under a lexical refusal heuristic, so this is not evidence that its overall behavior was well calibrated. Routing appears to contain some of the fine-tuning damage relative to pooling, not to make the resulting system safe; this is consistent with reports that fine-tuning drifts the routing of harmful inputs in MoE models (Kim et al., 2026) and that merging experts erodes alignment (Hammoud et al., 2024). Safety-preserving fine-tuning, by projecting adapter updates onto an alignment subspace (Hsu et al., 2024) or mixing safety demonstrations into the training data (Bianchi et al., 2024), is orthogonal to allocation and was not applied; an input–output guard (Inan et al., 2023) would sit outside the game. ConfAIde's conditional error is computed only for parseable ratings and its response coverage varies sharply between arms, so it cannot support a privacy-improvement claim.

L Reproduction

Environment and entry point. The implementation is a Python package requiring Python 3.10 or later; dependency constraints and optional training, plotting, testing, and behavioral-evaluation extras are specified in `pyproject.toml`. All experiments use the same YAML-driven pipeline rather than experiment-specific scripts. The principal run can be validated and executed with:

```
python -m pip install -e \
  ".[ml,configs,figures,test,behavioral]"
python -m pytest
python -m infl_ens.pipeline \
  --config configs/experiments/seven_axis_theory_eq.yaml \
  --dry-run
python -m infl_ens.pipeline \
  --config configs/experiments/seven_axis_theory_eq.yaml
```

The dry run resolves every layered arm configuration and reports its output directory and trait-space fingerprint without loading a model. Experiment files for the parameter sweeps, model ablations, baselines, and behavioral evaluation are colocated under `configs/experiments/`; each declares its arms, stage order, evaluation partition, and figure builders. Configurations that define a smoke block can additionally be checked with `--smoke` before a full GPU run.

Data identity, seeds, and cached geometry. The shared split manifest stores stable record identifiers and the within-benchmark train/validation/test assignment, so controlled arms consume identical examples rather than merely matching split sizes. Training, splitting, routing, and generation seeds are fields of the resolved configuration. Each completed run writes the fully flattened `resolved_config.yaml`; this file, rather than the compositional arm fragment, is the authoritative record of model, LoRA, game, and evaluation settings. The principal box-coordinate trait space uses Qwen3-Embedding-8B-AWQ-INT4 and has configuration fingerprint `3b42c68a8dd334c5`; all corresponding arms validate this identity before loading the shared cache.

Artifact contract. Every training run writes `history.json` with its round states and `resolved_config.yaml` with its effective configuration; runs using the shared manifest also write `data_split.json`. Final routing summaries are reported in round-11 `routing_ensemble_diagnostics.json`. The underlying `merge_nll.npy`, `g_merge.npy`, and `token_counts.npy` arrays are sufficient to recompute the reported commitment, expected-NLL, and probability-mixture estimators without reloading model checkpoints. Behavioral generation and score artifacts are schema-versioned and record ordered-input, checkpoint, trait-coordinate, and content digests; the reported behavioral tables use these artifacts rather than reparsing model logs. Experiment-level `stage_status.json` records stage completion and timing, while figure generation reads the saved artifacts without rerunning training.

Numerical reproducibility caveat. The four-bit AWQ trait encoder is not bitwise invariant across GPU architectures. In a 206-prompt cross-hardware check spanning L40S, A5500, and H200 devices, recomputing coordinates changed the hard-routing argmax for 2.4–3.4% of prompts. Exact reproduction therefore requires reuse of the original trait-space and behavioral-coordinate caches, not merely the same encoder model identifier. Behavioral coordinate caches are keyed by both the suite digest and trait-space fingerprint, and their content hash is included in generation provenance; rebuilding coordinates on different hardware should be treated as a robustness replication rather than a bitwise reproduction.