UNIGAME: A UNIFIED FOUNDATION MODEL FOR BOARD AND CARD GAMES VIA SHARED–SPECIFIC FUSION OF GAME EXPERTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) for board and card games offer a promising foundation for developing general-purpose agents capable of strategic reasoning, decision-making and interaction across diverse game environments. However, despite recent progress, game-playing LLMs still underperform specialized conventional methods, and a single model is typically optimized for only one game. Here we introduce UniGame, a unified board-and-card-game model that consolidates multiple game experts through shared–specific fusion while preserving their specialized policies. UniGame comprises three complementary designs: Cross-Game Subspace Sharing, which extracts reusable adaptation directions from independently trained LoRA experts; Game-Conditioned Effect Projection, which maps these shared directions to distinct game-specific updates; and Private Residual Adaptation, which preserves specialized knowledge not captured by the shared subspace. This post-hoc decomposition enables cross-game knowledge sharing without forcing heterogeneous games to adopt identical parameter updates, while using game identity to select the appropriate expert components during inference. Experiments across five board and card games demonstrate that UniGame substantially reduces adapter storage while retaining near-specialist gameplay performance and outperforming static merging approaches.

1 INTRODUCTION

Board- and card-game large language models (game LLMs) provide a compelling testbed for general-purpose decision-making. Playing games requires an agent to jointly understand rules, reason about hidden information, model other playersGuo et al. (2024), plan over long horizons, and select actions under uncertainty. Unlike conventional language benchmarks, games offer interactive environments with explicit constraints and objective outcomesGuertler et al. (2025), enabling systematic evaluation of both reasoning and decision quality. A unified model capable of playing multiple games would therefore represent an important step towards general-purpose agents that can acquire and reuse decision-making knowledge across heterogeneous environments.

Existing game-playing systems broadly follow two paradigms. Traditional approaches rely on specialized algorithms, including tree search, reinforcement learning(Zha et al., 2021; Silver et al., 2017), game-theoretic optimization(Zinkevich et al., 2007; Brown & Sandholm, 2019), and domain-specific heuristics (), and have achieved strong performance in individual games. More recently, game LLMs have emerged as a flexible alternative that represents game states and actions through language and leverages the general reasoning capabilities of pretrained models. Despite promising progress, game LLMs still generally lag behind highly optimized traditional systems. Moreover, most existing game LLMs follow a *one-model-per-game* paradigm: each model or adapter is specialized for a single game and cannot be directly reused for others. Low-rank adaptation (LoRA) reduces the cost of training these specialists (Hu et al., 2021), but storing an independent adapter for every game still incurs deployment costs that grow linearly with the number of games. Therefore, building a unified game LLM that supports diverse games within a shared parameterization while preserving game-specific decision-making capabilities is essential.

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



Figure 1: **Overview of UniGame.** UniGame unifies multiple game experts by sharing reusable adaptation directions while preserving game-specific effects through conditioned projections and private residuals.

Training a unified game LLM is non-trivial. Different games have distinct rules, state representations, legal actions, and decision structures, while the same state may admit multiple strategically valid actions. Naively mixing multi-game trajectories can therefore produce conflicting optimization signals and negative transfer. Similarly, collapsing specialist adapters into a single static update forces heterogeneous games to share the same parameter transformation, potentially erasing the specialized policies required by each game. The central challenge is thus to exploit reusable structure across games without conflating their distinct decision mechanisms.

To address this challenge, we introduce **UniGame**, a unified game LLM that consolidates independently trained game experts through a shared–specific LoRA decomposition, as illustrated in Fig. 2. UniGame is motivated by a simple distinction: different games may rely on similar directions in the backbone representation without requiring the same transformation along those directions. Based on this principle, UniGame comprises three complementary designs. First, **Cross-Game Subspace Sharing (CSS)** extracts a common low-rank input basis from multiple specialist adapters and stores it only once, capturing reusable adaptation directions across games. Second, **Game-Conditioned Effect Projection (GEP)** assigns each game an independent output readout over the shared basis, allowing a common direction to induce different game-specific updates. Third, **Private Residual Adaptation (PRA)** introduces a compact private branch for each game to preserve specialist information that cannot be represented by the shared subspace. Together, these components share *where* the model adapts while retaining game-specific control over *how* it adapts. UniGame is constructed post hoc from frozen LoRA specialists under a fixed parameter budget and requires neither mixed-game training, a learned router, nor backbone modification.

We evaluate UniGame by consolidating five LoRA specialists for Dou Dizhu, Limit Hold'em, Leduc Hold'em, Riichi Mahjong, and Guandan. Across all five games, UniGame substantially reduces adapter storage while retaining performance comparable to that of independently trained specialists and consistently outperforming static merging baselines. Further representation analyses confirm that the shared subspace captures transferable cross-game structure, whereas the private components effectively preserve game-specific knowledge. These results demonstrate that UniGame achieves an effective balance between parameter sharing and policy specialization.

Our contributions are threefold:

- We introduce UniGame, a unified game LLM that consolidates independently trained game experts through post-hoc shared–specific LoRA decomposition, avoiding both the optimization conflicts of mixed-game training and the linear storage growth of independent adapters.
- We propose three complementary designs—Cross-Game Subspace Sharing, Game-Conditioned Effect Projection, and Private Residual Adaptation—that separate reusable adaptation directions from game-specific transformations and residual knowledge.

- We evaluate UniGame across five board and card games, demonstrating a 40.0% reduction in adapter parameters with near-specialist gameplay performance, and provide representation-level analyses that quantify the cross-game reuse captured by the shared and private components.

2 Related Work

Traditional Methods for Board and Card Games Classical game-playing systems combine search, game-theoretic optimization, and reinforcement learning according to the information structure of the target game. For perfect-information games, AlphaZero integrates self-play reinforcement learning with Monte Carlo tree search and trains separate agents for chess, shogi, and Go using a common algorithm (Silver et al., 2017). For imperfect-information games, counterfactual regret minimization provides a foundation for strategy computation, while Libratus combines game-theoretic reasoning with subgame solving for heads-up no-limit poker (Zinkevich et al., 2007; Brown & Tuomas Sandholm, 2018). Game-specific deep reinforcement learning systems include DouZero, which employs deep Monte Carlo learning and action encoding for Dou Dizhu, and Suphx, which incorporates global reward prediction and runtime policy adaptation for Mahjong (Zha et al., 2021; Li et al., 2020). Whereas these methods focus on acquiring strong strategies for individual games, UniGame addresses the complementary deployment challenge of unifying already-trained game experts while preserving their specialized policies.

Large Language Models for Board and Card Games Large language models provide a common sequence-modeling interface for representing game observations, action histories, and decisions. Existing studies improve their game-playing capabilities through prompting, domain-specific training, and explicit search. Suspicion-Agent prompts GPT-4 to perform opponent modeling and theory-of-mind reasoning in imperfect-information games (Guo et al., 2024), while PokerBench evaluates poker decision-making and investigates supervised fine-tuning with domain-specific examples (Zhuang et al., 2025). For board games, Schultz et al. combine domain-pretrained language models with external Monte Carlo tree search or internally generated search traces (Schultz et al., 2024). These approaches primarily study how game-playing capabilities are acquired or elicited. In contrast, UniGame starts from independently trained LoRA experts on a shared backbone and consolidates them through shared–specific fusion, enabling a unified board-and-card-game model without retraining a multi-game policy.

Model Merging and Domain-Specific Adaptation Domain-adaptive and task-adaptive pretraining specialize models using relevant corpora, whereas LoRA represents task-specific updates through low-rank factors applied to a frozen backbone (Gururangan et al., 2020; Hu et al., 2021). Although individual LoRA adapters are parameter-efficient, their aggregate storage grows linearly with the number of specialists. Model-merging methods reuse specialization across tasks: Task Arithmetic combines task vectors, TIES resolves parameter-sign conflicts, and KnOTS aligns LoRA update spaces before merging (Ilharco et al., 2022; Yadav et al., 2023; Stoica et al., 2025). No Task Left Behind and Reinforced Agent Merging further distinguish shared and task-specific parameter structure (Marczak et al., 2025; Wu et al., 2026). Closely related approaches include Compress then Serve, which reconstructs LoRA updates using shared input and output factors with adapter-specific cores, and MoDE, which shares a down-projection while routing inputs among rank-one up-projection experts during multi-task training (Brüel-Gabrielsson et al., 2024; Ning et al., 2025). UniGame instead realizes shared–specific fusion through three complementary designs: *Cross-Game Subspace Sharing* extracts reusable adaptation directions, *Game-Conditioned Effect Projection* maps them to distinct game-specific updates, and *Private Residual Adaptation* preserves specialized knowledge outside the shared subspace.

3 Method

Our goal is to construct a unified board-and-card-game LLM that supports multiple games within a single parameter-efficient framework while preserving the strategic capabilities of game-specific experts. Rather than jointly retraining a multi-game model from scratch, we consolidate independently trained LoRA specialists that share the same frozen backbone. The key challenge is to exploit the

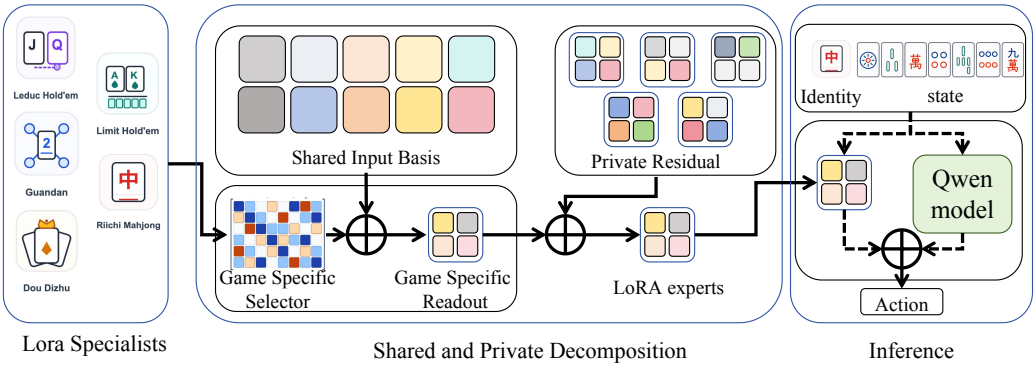


Figure 2: Framework of UniGame. It shares an input basis across games, while preserving game-specific effects through conditioned projections and private residuals selected by game identity.

structural overlap among these specialists without forcing games with distinct rules and strategies to use identical parameter updates.

Formally, let $g \in \{1, \ldots, G\}$ index games and $\ell \in \mathcal{L}$ index adapted linear modules. Each specialist provides a rank-r_0 update $\Delta W_{g,\ell} \in \mathbb{R}^{d_{\text{out},\ell} \times d_{\text{in},\ell}}$ to the frozen backbone weight W_ℓ, with the original LoRA scaling absorbed into $\Delta W_{g,\ell}$. Independently storing all specialists requires

$$P_{\text{sep}} = \sum_{\ell \in \mathcal{L}} G r_0 \left(d_{\text{in},\ell} + d_{\text{out},\ell}\right), \tag{1}$$

which grows linearly with the number of games even when their updates contain substantial redundancy. Conversely, replacing all specialists with a single static update eliminates game-specific storage but forces every game to apply the same transformation. Specifically, with $\bar{\Delta W}_\ell = G^{-1} \sum_g \Delta W_{g,\ell}$, the best unrestricted static approximation satisfies

$$\min_{S_\ell} \sum_{g=1}^{G} \|\Delta W_{g,\ell} - S_\ell\|_F^2 = \sum_{g=1}^{G} \left\|\Delta W_{g,\ell} - \bar{\Delta W}_\ell\right\|_F^2. \tag{2}$$

The remaining error reflects disagreement among the specialists and cannot be eliminated by increasing the rank of S_ℓ. This observation establishes the central requirement for unified game adaptation: parameters should be shared wherever possible, but their effects must remain game-dependent.

To achieve this objective, we introduce **UniGame**, which decomposes each specialist update as

$$\widehat{\Delta W}_{g,\ell} = \underbrace{B_{g,\ell} A_{s,\ell}}_{\text{shared directions, game-specific effects}} + \underbrace{C_{g,\ell} D_{g,\ell}}_{\text{private residual}}. \tag{3}$$

Here, $A_{s,\ell}$ is shared across all games, whereas $B_{g,\ell}$, $C_{g,\ell}$, and $D_{g,\ell}$ are selected according to the known game identity. The decomposition comprises three complementary designs. **Cross-Game Subspace Sharing** identifies reusable adaptation directions across specialists; **Game-Conditioned Effect Projection** allows each game to assign a distinct effect to these directions; and **Private Residual Adaptation** preserves specialized information that cannot be represented within the shared subspace. UniGame is fitted post hoc from existing specialist adapters and requires neither mixed-game retraining nor a learned router. Fig. 2 illustrates the overall framework.

3.1 Cross-Game Subspace Sharing

Although different games require distinct strategies, their specialists may modify overlapping representation directions in the shared backbone. Independently storing each adapter repeatedly encodes these common directions, leading to unnecessary parameter redundancy. We therefore first identify a compact cross-game subspace that captures reusable adaptation structure across specialists.

For example, consider two rank-one updates $\Delta W_{1,\ell} = u_1 a^\top$ and $\Delta W_{2,\ell} = u_2 a^\top$, where $u_1 \neq u_2$. The two specialists act on the same input direction $a^\top$ but produce different output effects. Independent adapters store $a^\top$ twice, whereas a single static update cannot simultaneously represent both transformations. This example suggests that the input direction can be shared even when its downstream effects must remain game-specific.

Accordingly, Cross-Game Subspace Sharing constructs a common input basis

$$A_{s,\ell} \in \mathbb{R}^{r_{s,\ell} \times d_{\text{in},\ell}} \tag{4}$$

for each adapted module. Given an input x, the basis extracts a compact set of shared coordinates $A_{s,\ell}x$ that can be reused across all games. Because $A_{s,\ell}$ is stored only once, this design removes duplicated input-side factors from the original specialist adapters. Importantly, UniGame shares an input subspace rather than a complete update matrix, allowing games to reuse common representation directions without constraining them to produce the same output changes.

The capacity of the shared component is determined by the row space of $A_{s,\ell}$. Let $P_{s,\ell}$ denote the orthogonal projector onto this row space. For a fixed shared basis, the minimum reconstruction error for game g is

$$\min_{B_{g,\ell}} \|\Delta W_{g,\ell} - B_{g,\ell} A_{s,\ell}\|_F^2 = \|\Delta W_{g,\ell}(I - P_{s,\ell})\|_F^2 . \tag{5}$$

Thus, the shared component can recover adaptation effects associated with directions inside the learned subspace, whereas directions outside it cannot be reconstructed by modifying the output readout alone. This formulation explicitly defines the objective of cross-game sharing: to identify a compact input subspace that captures the adaptation directions most reusable across heterogeneous games.

3.2 Game-Conditioned Effect Projection

Sharing adaptation directions does not imply that these directions should have the same effect in every game. Games may process related representations while requiring different actions, values, or strategic responses. A unified model must therefore preserve game-specific transformations over the shared coordinates rather than collapsing all specialists into a common update.

To this end, Game-Conditioned Effect Projection assigns each game a separate readout

$$B_{g,\ell} \in \mathbb{R}^{d_{\text{out},\ell} \times r_{s,\ell}}, \tag{6}$$

which maps the shared coordinates to a game-dependent output update:

$$S_{g,\ell} = B_{g,\ell} A_{s,\ell}. \tag{7}$$

All games use the same basis $A_{s,\ell}$, but their readouts $B_{g,\ell}$ can implement different transformations. In the rank-one example above, setting $A_{s,\ell} = a^\top$ and $B_{g,\ell} = u_g$ exactly reconstructs both specialist updates while storing the common direction $a^\top$ only once. The resulting factorization separates *where* the backbone is adapted, as encoded by the shared basis, from *how* that adaptation affects each game, as determined by the game-specific readout.

At inference time, the environment provides the game identity, which directly selects the corresponding $B_{g,\ell}$. For a module input x, the shared-direction component is computed as

$$B_{g,\ell}(A_{s,\ell}x). \tag{8}$$

This deterministic selection requires neither a learned router nor an input-dependent gate. Once the private branch is included, the complete module computation is

$$y_{g,\ell} = W_\ell x + B_{g,\ell}(A_{s,\ell}x) + C_{g,\ell}(D_{g,\ell}x). \tag{9}$$

Game-Conditioned Effect Projection therefore resolves the conflict identified in Eq. equation 2: UniGame can reuse common adaptation directions while retaining the distinct parameter effects required by each specialist policy.

3.3 PRIVATE RESIDUAL ADAPTATION

A compact shared subspace cannot be expected to capture every adaptation direction required by all games. In particular, heterogeneous rules, action spaces, and strategic structures may induce specialist updates that lie outside the common subspace. As shown in Eq. equation 5, such directions cannot be recovered by the game-specific readout alone. We therefore introduce a private residual branch to preserve game-specific information that is not adequately represented by the shared component.

For each game and adapted module, Private Residual Adaptation introduces the low-rank factors

$$C_{g,\ell} \in \mathbb{R}^{d_{\mathrm{out},\ell} \times r_{p,g,\ell}}, \qquad D_{g,\ell} \in \mathbb{R}^{r_{p,g,\ell} \times d_{\mathrm{in},\ell}}. \tag{10}$$

The resulting term $C_{g,\ell}D_{g,\ell}$ complements the shared component by recovering residual structure specific to each specialist. Here, *private* means that the factors are stored and selected separately for each game; it does not require their semantic content to be unique to that game. We impose no orthogonality constraint between the shared and private branches and instead optimize both components jointly, allowing them to divide the reconstruction burden under a fixed capacity budget.

Without an explicit constraint, increasing the private ranks could trivially reproduce the original collection of independent adapters. We therefore fix the shared and private ranks under a global storage budget. The packed cost of module ℓ is

$$P_\ell = r_{s,\ell} d_{\mathrm{in},\ell} + \sum_{g=1}^{G} \left[r_{s,\ell} d_{\mathrm{out},\ell} + r_{p,g,\ell} \left(d_{\mathrm{in},\ell} + d_{\mathrm{out},\ell} \right) \right], \qquad \sum_{\ell \in \mathcal{L}} P_\ell \le P_{\max}. \tag{11}$$

The first term stores the shared basis once, whereas the remaining terms store the game-specific readouts and private residual factors. Increasing the shared rank is also not free, because every additional shared direction requires a separate output readout for each game. UniGame therefore balances the capacity assigned to reusable cross-game structure against that reserved for game-specific residuals. We use the fixed development-protocol allocation reported in Table 1, repeated across 28 transformer blocks and seven adapted projections per block.

Given this rank allocation, we jointly fit the shared basis, game-specific readouts, and private residual factors by minimizing

$$\mathcal{L}_{\mathrm{mat}} = \sum_{\ell \in \mathcal{L}} \sum_{g=1}^{G} \left\| \Delta W_{g,\ell} - B_{g,\ell} A_{s,\ell} - C_{g,\ell} D_{g,\ell} \right\|_F^2 . \tag{12}$$

The factors within each module are coupled through $A_{s,\ell}$, whereas different modules can be fitted independently. Our main `basis_uniform` construction uses four alternating optimization sweeps over the joint input and output spans of the source updates. This procedure is an approximate post-hoc fit and makes no claim of global optimality. It operates directly on the source adapter matrices and requires neither calibration activations nor gameplay rewards; an activation-weighted variant is evaluated separately in Section 4.3. In our setting, $G = 5$, $|\mathcal{L}| = 196$, and each source specialist has rank $r_0 = 8$. The five independent adapters contain 43,581,440 parameters, whereas the packed UniGame adapter contains 26,148,864 parameters: 1,605,632 for the shared bases, 7,168,000 for the game-specific readouts, and 17,375,232 for the private residual factors. These counts exclude the common frozen backbone, serialization overhead, and runtime memory. The resulting decomposition provides a storage-constrained balance between cross-game parameter reuse and the preservation of specialized game policies.

Table 1: Shared and private ranks for each projection.

Projection	r_s	DDZ	Limit	Leduc	Riichi	Guandan
MLP down	4	3	3	4	3	3
MLP gate/up	2	3	3	4	3	3
Attention key/value	3	3	3	4	4	3
Attention query/output	3	3	3	3	3	3

Table 2: UniGame's win rates against strong baselines across five games.

Method	Params (M)↓	DDZ $W\uparrow$	Limit $R\uparrow$	Leduc $R\uparrow$	Riichi $T_2\uparrow$	Guandan $W\uparrow$
Specialists	43.58	44.50	0.0335	1.0872	38.80	35.00
Mixed SFTWang et al. (2026)	26.15	40.20	−0.0880	1.0450	33.30	31.8
Task ArithmeticIlharco et al. (2022)	34.87	17.80	−0.2361	−0.0009	3.40	6.00
TIESYadav et al. (2023)	34.87	13.00	−0.6832	0.0000	0.20	0.00
Shared S+PLu et al. (2024)	34.87	44.00	0.0340	1.0872	37.60	33.75
Pure-private	26.15	42.50	0.0351	1.0872	**40.40**	34.50
UniGame (ours)	**26.15**	**45.20**	**0.0381**	**1.0872**	38.00	**35.50**

W: win rate (%); R: mean net return per hand; T_2: top-two rate (%).

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We evaluate DDZ against Douzero WP(Zha et al., 2021), Limit Hold'em against an RLCard DQN teacher(Zha et al., 2020; Mnih et al., 2015), Leduc Hold'em against a rule-based agent, Riichi Mahjong against Mortal v4(Equim-chan), and Guandan against Koi(koi). To account for seat or role effects, we evaluate 500 DDZ openings in each role, 5,000 poker seeds in both seats, 125 Riichi seeds across four seats, and 200 Guandan openings in both team positions. We report win rate for DDZ and Guandan, mean net return per hand for the two poker games, and top-two finish rate for Riichi. Results are reported separately for each game because these metrics have different scales.

4.2 MAIN RESULT

We compare UniGame against strong baselines for training and consolidating game-playing language models. As shown in Table 2, UniGame achieves the strongest overall performance across the five games. It consistently outperforms Mixed SFT, with substantial gains on DDZ (45.20% vs. 40.20%), Limit Hold'em (0.0381 vs. -0.0880), Riichi Mahjong (38.00% vs. 33.30%), and Guandan (35.50% vs. 31.80%), while also improving performance on Leduc Hold'em. UniGame also markedly surpasses static merging methods, including Task Arithmetic and TIES, which suffer severe performance degradation when combining heterogeneous game experts. Compared with the stronger Shared S+P and Pure-private baselines, UniGame achieves better performance on most games: it obtains the highest scores on DDZ, Limit Hold'em, and Guandan, matches the best result on Leduc Hold'em, and remains competitive on Riichi Mahjong. Notably, UniGame matches or exceeds the independently stored specialists on four of the five games, demonstrating that shared–specific fusion can effectively preserve specialized game-playing capabilities during consolidation. Moreover, UniGame uses only 26.15M adapter parameters, matching the smallest parameter budget among all compared methods and reducing storage by 40.0% relative to the independent specialists.

We attribute UniGame's superior performance to its shared–specific fusion design, which identifies reusable adaptation directions across games, transforms them into game-specific parameter updates, and retains specialized knowledge through private residuals. Unlike static merging methods that impose a single update across heterogeneous games, UniGame selectively shares transferable structure while preserving the distinct strategic knowledge of each specialist. This decomposition therefore facilitates effective cross-game knowledge sharing without compromising game-specific decision policies.

4.3 ABLATION STUDY

In this part, we analyze whether Cross-Game Subspace Sharing (CSS) provides useful shared capacity, Game-Conditioned Effect Projection (GEP) reconstructs distinct game-specific updates, and Private Residual Adaptation (PRA) preserves information outside the shared subspace. Tables 3, 4, and 5 report gameplay comparisons and a diagnostic of shared-direction participation. All parameter budgets include the adapter factors for all five games but exclude the frozen backbone.

Impact of Cross-Game Subspace Sharing (CSS). To investigate how the source of shared directions influences CSS, we compare four basis constructions in Table 3: a random basis, a basis derived from a single specialist, a leave-one-game-out basis, and the proposed all-five basis. As can be seen, performance generally improves as the basis incorporates information from a broader set of specialists. Most notably, the DDZ win rate increases from 4.20% with the random basis to 41.20% with the all-five basis, which also achieves the best results on Riichi Mahjong (32.60%) and Guandan (27.00%). Although the single-source basis yields the highest returns on the two poker games, these gains do not transfer consistently to the other domains. By contrast, the all-five basis exhibits the strongest overall performance across the heterogeneous game suite, ranking first on three games while remaining competitive on Limit and Leduc Hold'em. Taken together, these findings suggest that incorporating all specialists produces a more broadly transferable shared subspace and improves cross-game generality.

Table 3: Ablation of Cross-Game Subspace Sharing (CSS).

Configuration	DDZ $W\uparrow$	Limit $R\uparrow$	Leduc $R\uparrow$	Riichi $T_2\uparrow$	Guandan $W\uparrow$
Random basis	4.20	-0.0207	0.0143	0.00	0.00
Single-source basis	24.10	**0.0352**	**0.1766**	26.00	9.00
Leave-one-game-out basis	33.50	-0.0207	0.0644	24.80	9.00
All-five basis	**41.20**	-0.0207	0.1111	**32.60**	**27.00**

W: win rate (%); R: mean return; T_2: top-two rate (%).

Impact of Game-Conditioned Effect Projection Module(GEP) To examine whether individual games benefit from distinct readouts over shared directions, we compare a shared-basis representation with a single shared update matrix, each optimized using either uniform or activation-weighted reconstruction (Table 4). As shown, under uniform fitting, the shared-basis representation outperforms the shared matrix on DDZ (45.20% vs. 44.20%) and Guandan (35.50% vs. 34.00%), while matching its Leduc return of 1.0872. Conversely, the shared matrix performs better on Limit and Riichi, achieving 0.0481 and 43.20%, respectively. Within the shared-basis representation, uniform fitting further outperforms activation weighting on DDZ, Riichi, and Guandan, whereas activation weighting yields a higher Limit return and identical performance on Leduc. Overall, the uniform shared-basis configuration ranks first on DDZ and Guandan and ties for first on Leduc, demonstrating that game-conditioned readouts can effectively specialize shared directions across heterogeneous games, although the resulting gains vary by domain.

Table 4: Ablation of Game-Conditioned Effect Projection (GEP).

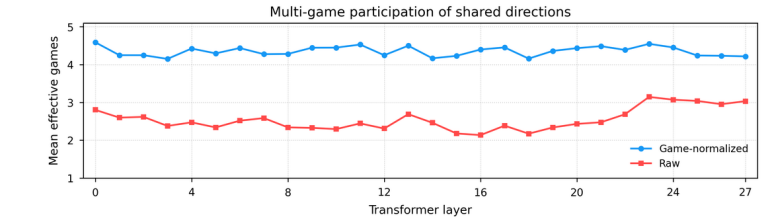
Representation / fit	DDZ $W\uparrow$	Limit $R\uparrow$	Leduc $R\uparrow$	Riichi $T_2\uparrow$	Guandan $W\uparrow$
Shared matrix, uniform	44.20	0.0481	1.0872	**43.20**	34.00
Shared matrix, activation	43.90	-0.0207	0.8136	38.20	34.50
Shared basis, activation	44.10	**0.0436**	**1.0872**	34.50	34.00
Shared basis, uniform (UniGame)	**45.20**	0.0381	**1.0872**	38.00	**35.50**

Impact of Private Residual Adaptation (PRA). To assess whether private residuals preserve game-specific information beyond the shared subspace, we compare a random complement with private features derived from another game, the target game, and the complete UniGame configuration (Table 5). As can be seen, replacing the random complement with target-game features substantially improves performance: the returns on Limit and Leduc increase from -0.0207 to 0.0454 and from 0.7639 to 1.0872, respectively, while the Riichi top-two rate and Guandan win rate rise from 35.20% to 36.80% and from 26.50% to 34.75%. Moreover, features derived from another game generally outperform the random complement, indicating that structured residual directions are more informative than arbitrary ones. Nevertheless, they remain less effective than target-game features in most cases, with DDZ being the only exception (43.80% vs. 43.70%). Building on these observations, the complete UniGame configuration achieves the best results on DDZ, Riichi, and Guandan (45.20%, 38.00%, and 35.50%), ties for the highest Leduc return, and remains competitive on Limit. Taken together, these results show that game-specific residuals effectively complement the shared representation, balancing cross-game knowledge reuse with the preservation of specialist capabilities.

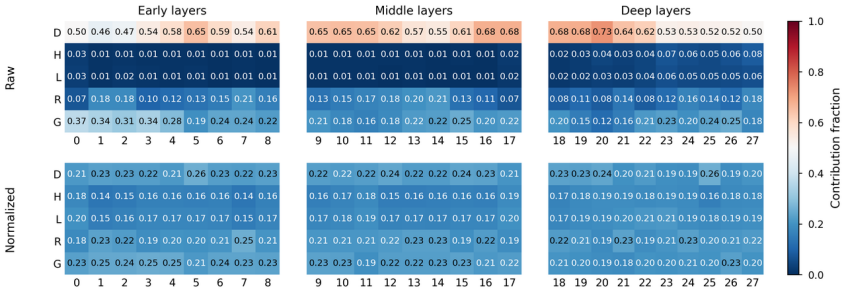
Cross-Game Participation of Shared Directions. To assess whether the learned shared directions integrate contributions from multiple games, we analyze their cross-game participation in Fig. 3. As shown in Fig. 3(a), after accounting for differences in update scale, the mean effective number of

Table 5: Gameplay performance with different private residuals (PRA).

Probe features	DDZ $W\uparrow$	Limit $R\uparrow$	Leduc $R\uparrow$	Riichi $T_2\uparrow$	Guandan $W\uparrow$
Shared features only	—	—	—	—	—
+ Random complement	42.4	−0.0207	0.7639	35.2	26.5
+ Other-game private	43.8	0.0394	0.9888	35.8	27.5
+ Own-game private	43.7	**0.0454**	**1.0872**	36.8	34.75
+ all-game(UniGame)	**45.20**	0.0381	**1.0872**	**38.00**	**35.50**



(a) Layer-wise mean of direction-level participation.



(b) Layer-averaged contribution heatmaps (not training checkpoints).

D=DouDizhu, H=Limit Holdem, L=Leduc Holdem, R=Riichi, G=GuanDan

Figure 3: Cross-game participation of shared directions. (a) Layer-wise mean effective number of contributing games, where a larger value indicates that each shared direction receives substantial contributions from more games. (b) Layer-averaged contribution fractions before and after game normalization; rows denote the five games, columns denote Transformer layers, and each column sums to one. Game normalization removes differences in update scale, revealing whether the shared directions capture balanced cross-game contributions rather than being dominated by a single specialist.

contributing games ranges from 4.15 to 4.59 across all 28 layers, substantially exceeding the range of 2.13–3.15 obtained from the raw contributions. Fig. 3(b) further clarifies this discrepancy: although DDZ dominates the raw contribution fractions, the normalized fractions remain distributed across all five games throughout the network. Overall, 78.6% of the 560 shared directions receive effective contributions from at least four games under the normalized measure. These findings indicate that the shared basis integrates information from multiple specialists rather than being governed by a single dominant update. Because normalization is applied only for this diagnostic, however, the analysis characterizes the composition of the learned basis and should not be interpreted as direct evidence of improved gameplay performance.

5 Conclusion

We have introduced **UniGame**, a unified framework that consolidates independently trained board-and-card-game experts through shared–specific fusion. By combining shared adaptation directions, game-conditioned projections, and private residuals, UniGame enables effective cross-game parameter sharing while preserving game-specific policies. Experiments across five games show that UniGame substantially reduces adapter storage, retains near-specialist performance, and outperforms static merging approaches, providing a practical foundation for building unified strategic agents across heterogeneous game environments.

6 GENERATIVE AI USE DISCLOSURE

In this study, we used generative AI tools to assist with the design and discussion of ablation experiments; the development and revision of training, evaluation, and plotting code; the analysis and presentation of experimental results; and the editing of portions of the manuscript. These tools also assisted in generating figures and tables from experimental results.

The authors reviewed all AI-assisted text, code, figures, tables, and numerical claims against the relevant implementations and recorded experimental results, revising them where necessary. The authors take full responsibility for the final content of this work, including its experimental findings, scientific conclusions, and all artifacts produced with the assistance of generative AI.

REFERENCES

Koi: An internal guandan evaluation agent. Unpublished software. Internally deployed opponent used for GuanDan evaluation.

Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456): 885–890, 2019. doi: 10.1126/science.aay2400. URL https://www.science.org/doi/abs/10.1126/science.aay2400.

Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

Rickard Brüel-Gabrielsson, Jiacheng Zhu, Onkar Bhardwaj, Leshem Choshen, Kristjan Greenewald, Mikhail Yurochkin, and Justin Solomon. Compress then serve: Serving thousands of lora adapters with little overhead. *arXiv preprint arXiv:2407.00066*, 2024.

Equim-chan. Mortal: A riichi mahjong ai. https://github.com/Equim-chan/Mortal. Version 4, accessed September 26, 2026.

Leon Guertler, Bobby Cheng, Simon Yu, Bo Liu, Leshem Choshen, and Cheston Tan. Textarena, 2025. URL https://arxiv.org/abs/2504.11442.

Jiaxian Guo, Bo Yang, Paul Yoo, Bill Yuchen Lin, Yusuke Iwasawa, and Yutaka Matsuo. Suspicion-agent: Playing imperfect information games with theory of mind aware gpt-4, 2024. URL https://arxiv.org/abs/2309.17277.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 8342–8360, 2020.

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, 2021. URL https://arxiv.org/abs/2106.09685.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

Junjie Li, Sotetsu Koyamada, Qiwei Ye, Guoqing Liu, Chao Wang, Ruihan Yang, Li Zhao, Tao Qin, Tie-Yan Liu, and Hsiao-Wuen Hon. Suphx: Mastering mahjong with deep reinforcement learning. *arXiv preprint arXiv:2003.13590*, 2020.

Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. Twin-merging: Dynamic integration of modular expertise in model merging. *Advances in Neural Information Processing Systems*, 37:78905–78935, 2024.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D Bagdanov, and Joost Van De Weijer. No task left behind: Isotropic model merging with common and task-specific subspaces. *arXiv preprint arXiv:2502.04959*, 2025.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Lin Ning, Harsh Lara, Meiqi Guo, and Abhinav Rastogi. Mode: Effective multi-task parameter efficient fine-tuning with a mixture of dyadic experts. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 8248–8261, 2025.

John Schultz, Jakub Adamek, Matej Jusup, Marc Lanctot, Michael Kaisers, Sarah Perrin, Daniel Hennes, Jeremy Shar, Cannada Lewis, Anian Ruoss, et al. Mastering board games by external and internal planning with language models. *arXiv preprint arXiv:2412.12119*, 2024.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 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*, volume 2025, pp. 4501–4519, 2025.

Wei Wang, Fuqing Bie, Junzhe Chen, Dan Zhang, Shiyu Huang, Evgeny Kharlamov, and Jie Tang. Can large language models master complex card games? *Advances in Neural Information Processing Systems*, 38:24786–24821, 2026.

Shawn Wu, Sean Wang, Louie Li, Darcy Chen, Allen Wang, Jiangang Luo, Xudong Zhao, Joseph Shen, Gawain Ma, Jasper Jia, et al. Yuan3. 0 flash: An open multimodal large language model for enterprise applications. *arXiv preprint arXiv:2601.01718*, 2026.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, 36:7093–7115, 2023.

Daochen Zha, Kwei-Herng Lai, Songyi Huang, Yuanpu Cao, Keerthana Reddy, Juan Vargas, Alex Nguyen, Ruzhe Wei, Junyu Guo, and Xia Hu. Rlcard: A platform for reinforcement learning in card games. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 5264–5266. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/764. URL https://doi.org/10.24963/ijcai.2020/764. Demos.

Daochen Zha, Jingru Xie, Wenye Ma, Sheng Zhang, Xiangru Lian, Xia Hu, and Ji Liu. Douzero: Mastering doudizhu with self-play deep reinforcement learning. In *international conference on machine learning*, pp. 12333–12344. PMLR, 2021.

Richard Zhuang, Akshat Gupta, Richard Yang, Aniket Rahane, Zhengyu Li, and Gopala Anumanchipalli. Pokerbench: training large language models to become professional poker players. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i24.34814. URL https://doi.org/10.1609/aaai.v39i24.34814.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In J. Platt, D. Koller, Y. Singer, and S. Roweis (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper_files/paper/2007/file/08d98638c6fcd194a4b1e6992063e944-Paper.pdf.

A Appendix

A.1 Specialist Models and Experimental Artifacts

This appendix details the construction, accounting, and evaluation of UniGame. We first specify the source specialists and their interfaces, then derive the shared–specific factorization and its parameter cost. Finally, we describe the gameplay protocol and report complementary analyses of policy fidelity and local update recoverability. These analyses address different questions: gameplay measures interactive performance, policy fidelity measures agreement with a source specialist, and reconstruction probes measure which linear effects a representation can express.

A.1.1 Backbone and source adapters

All five source specialists share the Qwen3-1.7B backbone. Adaptation is applied to seven linear projections in each of 28 Transformer blocks: query, key, value, attention output, MLP gate, MLP up, and MLP down. Consequently, the decomposition covers 196 modules. Each source adapter has rank $r_0 = 8$. The source LoRA scale is absorbed into the effective update $\Delta W_{g,\ell}$ throughout the derivations below; this convention avoids confusing an adapter's stored factors with the actual transformation added to the backbone.

A fixed source specialist is used for each game across the merging methods. UniGame is constructed from these frozen adapters while preserving the game-specific policy interfaces.

A.1.2 Documented training configuration

The available DDZ configuration specifies supervised LoRA fine-tuning with rank 8, alpha 16, and dropout 0.05, using all linear adaptation targets. It uses the `qwen3_nothink` template, a maximum sequence length of 4096, BF16 arithmetic, fused AdamW, a learning rate of 10^{-4}, cosine decay, and a warmup ratio of 0.03. The configured per-device batch size is 6 with five gradient-accumulation steps. The configuration permits 15,000 optimizer steps and saves every 1,000 steps; the registered source is checkpoint 13,000. These settings document the DDZ source-training configuration and are not assumed to apply to every specialist.

The source registry additionally records 2,000,000 training examples for Riichi and 2,051,401 for Guandan, together with 12,933 held-out Guandan decisions. Training examples, held-out decisions, and interactive games are distinct units and are not combined into a single sample count. UniGame construction starts from the resulting frozen adapters; the uniform construction does not retrain the backbone or optimize a multi-game supervised objective.

A.1.3 Preserving the game interface

Game identity is supplied by the environment and selects the corresponding readout and private factors. It is not inferred by an auxiliary router. Each game retains its own prompt and action serialization, so parameter sharing does not require a common action vocabulary beyond the underlying tokenizer. In the registered interface, the common system message is "You are a multi-game decision model." The observations and available actions remain game-specific. This separation lets the decomposition operate on compatible parameter tensors while preserving the policy interface expected by each evaluator.

A.2 Mathematical Properties of Shared–Specific Fusion

For clarity, we suppress the module index in this section. Let $T_g = \Delta W_g \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote a source update. UniGame represents it as

$$\widehat{T}_g = B_g A_s + C_g D_g, \qquad A_s \in \mathbb{R}^{r_s \times d_{\text{in}}}. \tag{13}$$

The factor A_s is stored once, whereas B_g, C_g, D_g are game-specific. The following derivations characterize the representational constraints and the uniform reconstruction objective; they do not assume that the finite optimization procedure reaches a global optimum.

A.2.1 WHY A STATIC UPDATE RETAINS CROSS-GAME DISAGREEMENT

Let $\overline{T} = G^{-1} \sum_g T_g$. For any common update S, expansion around the mean gives

$$\sum_g \|T_g - S\|_F^2 = \sum_g \|T_g - \overline{T}\|_F^2 + G\|S - \overline{T}\|_F^2. \tag{14}$$

The cross term vanishes because $\sum_g (T_g - \overline{T}) = 0$. Thus, even an unrestricted static update has minimum error $\sum_g \|T_g - \overline{T}\|_F^2$. Increasing its rank cannot remove disagreement among distinct specialists. Game-conditioned readouts relax this constraint: if $T_g = u_g a^\top$, a shared basis $A_s = a^\top$ and readouts $B_g = u_g$ reproduce every update exactly, even when the u_g differ.

A.2.2 WHAT A FIXED SHARED BASIS CAN RECONSTRUCT

Let $A_s^\dagger$ denote the Moore–Penrose pseudoinverse and $P_s = A_s^\dagger A_s$ the orthogonal projector onto the row space of A_s. For a residual target $R_g = T_g - C_g D_g$, one least-squares readout is $B_g^\star = R_g A_s^\dagger$. Consequently,

$$B_g^\star A_s = R_g P_s, \qquad \min_{B_g} \|R_g - B_g A_s\|_F^2 = \|R_g(I - P_s)\|_F^2. \tag{15}$$

To see this, decompose R_g into its row-space and orthogonal components. Every product $B_g A_s$ lies in the first component, while the second component is Frobenius-orthogonal to all such products. This yields a precise interpretation of CSS and GEP: CSS determines which input directions are available, and GEP determines the output transformation along those directions.

A.2.3 CONDITIONAL UPDATES FOR THE UNIFORM OBJECTIVE

For fixed private updates, stack residual targets vertically as $R = [R_1^\top, \ldots, R_G^\top]^\top$. A globally optimal shared row space for this conditional problem is spanned by the leading r_s right singular vectors of R. Taking these vectors as the rows of an orthonormal A_s gives $B_g = R_g A_s^\top$. For fixed shared terms, define $E_g = T_g - B_g A_s$. The best rank-$r_{p,g}$ private update is the truncated singular value decomposition

$$(E_g)_{r_{p,g}} = U_{g,r} \Sigma_{g,r} V_{g,r}^\top, \quad C_g = U_{g,r} \Sigma_{g,r}^{1/2}, \quad D_g = \Sigma_{g,r}^{1/2} V_{g,r}^\top. \tag{16}$$

These conditional solutions explain the alternating fitting strategy. Each exact conditional minimization cannot increase the uniform objective, but the joint problem remains non-convex because shared and private terms interact. The recorded construction uses four alternating sweeps. A fixed sweep budget therefore specifies an approximate factorization, rather than a convergence or global-optimality certificate.

A.2.4 COMPUTATION IN THE JOINT SOURCE SPANS

The factorization can be performed in compact coordinates determined by the source adapters. Let Q_o and Q_i have orthonormal columns spanning the union of the source output and input spaces, respectively. Then

$$T_g = Q_o M_g Q_i^\top, \qquad M_g = Q_o^\top T_g Q_i. \tag{17}$$

Because each source has rank at most 8 and $G = 5$, both joint spans have dimension at most 40. For approximations in these spans,

$$\|T_g - Q_o \widehat{M}_g Q_i^\top\|_F^2 = \|M_g - \widehat{M}_g\|_F^2. \tag{18}$$

This identity follows from orthonormal invariance of the Frobenius norm. It allows the alternating updates to use small core matrices while retaining the original uniform reconstruction metric. The final factors are lifted back to the original module dimensions for inference. In particular, the dimension reduction is a property of the source update spaces; it does not reduce the backbone hidden width.

A.2.5 Activation weighting and the meaning of private factors

The activation-weighted variant replaces an isotropic parameter-space metric with the second moment of specialist inputs. For $\Sigma_g = \mathbb{E}[xx^\top]$,

$$\mathbb{E}\big[\|(T_g - \widehat{T}_g)x\|_2^2\big] = \operatorname{tr}\big((T_g - \widehat{T}_g)\Sigma_g(T_g - \widehat{T}_g)^\top\big). \tag{19}$$

The recorded variant uses per-game projected teacher-input second moments, trace normalization, and identity shrinkage of 10^{-4}. It uses four outer sweeps and 12 inner alternating least-squares iterations for the weighted basis update. These activation statistics are unnecessary for the uniform construction.

The deployed private factors are not constrained to be orthogonal to the shared basis. Moreover, for any invertible H, the substitution $A_s \leftarrow HA_s$ and $B_g \leftarrow B_g H^{-1}$ leaves the shared update unchanged. Individual factor coordinates are therefore not uniquely identifiable. We interpret "private" as game-specific storage and selection, not as proof of semantically exclusive knowledge. The diagnostic orthogonalization in Subsection A.5 is performed after fitting and does not alter the deployed branches.

A.3 Parameter Budget and Inference Representation

A.3.1 Exact accounting

Independent rank-r_0 specialists require $Gr_0(d_{\text{in}} + d_{\text{out}})$ parameters per module. UniGame instead requires

$$P_\ell = r_{s,\ell}d_{\text{in},\ell} + Gr_{s,\ell}d_{\text{out},\ell} + \sum_g r_{p,g,\ell}(d_{\text{in},\ell} + d_{\text{out},\ell}). \tag{20}$$

The first term is counted once, while both the readouts and residuals are counted for every game. Table 6 provides the dimensions and rank allocation, repeated across all 28 blocks. Dimensions are written as input $\rightarrow$ output.

Table 6: Per-projection dimensions and ranks. Private-rank columns follow the game order used throughout the appendix.

Projection	Dimensions	r_s	DDZ	Limit	Leduc	Riichi	Guandan
MLP down	$6144 \rightarrow 2048$	4	3	3	4	3	3
MLP gate/up	$2048 \rightarrow 6144$	2	3	3	4	3	3
Attention key/value	$2048 \rightarrow 1024$	3	3	3	4	4	3
Attention query/output	$2048 \rightarrow 2048$	3	3	3	3	3	3

Summing Eq. equation 20 gives 1,605,632 shared-basis parameters, 7,168,000 game-specific readout parameters, and 17,375,232 private parameters, totaling 26,148,864. The corresponding five independent adapters contain 43,581,440 parameters. The exact reduction is

$$1 - \frac{26{,}148{,}864}{43{,}581{,}440} = 0.40. \tag{21}$$

At two bytes per parameter, the nominal factor payload is 49.875 MiB rather than 83.125 MiB, a reduction of 33.25 MiB. These are tensor-payload calculations, excluding the common backbone, file metadata, temporary buffers, and optimizer state.

A.3.2 Packed storage and active computation

For a known game, the two branches can be expressed as one conventional low-rank update:

$$\widehat{T}_g = [B_g \quad C_g] \begin{bmatrix} A_s \\ D_g \end{bmatrix}. \tag{22}$$

The active inner dimension is $r_s + r_{p,g}$, which ranges from 5 to 8 under Table 6; the algebraic rank can be smaller. Per-token adapter multiplication has nominal cost proportional to

$(r_s+r_{p,g})(d_{\text{in}}+d_{\text{out}})$. This describes adapter arithmetic, not end-to-end latency, which also depends on the backbone, batching, kernels, and serving implementation.

The storage saving assumes that the shared basis is stored once in a packed representation. Exporting five standalone adapters via Eq. equation 22 duplicates this basis and therefore has a different storage cost. In the evaluation pipeline, materialized active adapters serve as per-game execution views of the packed factors. This distinction is necessary when translating a parameter budget into an implementation.

A.3.3 Exact reconstruction as an implementation check

Before evaluating compressed models, the pipeline constructs an exact routed representation of the five source adapters. After correcting an intermediate FP32-to-BF16-to-FP32 rounding path during adapter loading, the archived check reports zero legal-policy Jensen–Shannon divergence, 100% top-action agreement, and 100% generated-token agreement on 100 checked prompts per game. This verifies the tested loading and routing path independently of lossy compression; it is not a claim of exhaustive equivalence on all possible game states.

A.4 Gameplay Protocol and Complementary Opponents

A.4.1 Evaluation units and outcome measures

Table 7 summarizes the frozen gameplay protocol. Role or seat rotations are kept together because outcomes sharing an opening or seed are statistically dependent. DDZ averages success across landlord and farmer roles. The poker metrics are mean net returns per hand in the evaluator's native units. Riichi uses the fraction of games finishing in the top two, and Guandan uses team win rate in the first-deal setting.

Table 7: Gameplay evaluation protocol. Counts apply per evaluated opponent.

Game	Main opponent	Evaluation construction	Outcomes
DDZ	DouZero WP	500 openings, two roles	1,000
Limit Hold'em	DQN teacher	5,000 seeds, two seats	10,000
Leduc Hold'em	Rule policy	5,000 seeds, two seats	10,000
Riichi	Mortal v4	125 seeds, four seat rotations	500
Guandan	Koi	200 openings, two team positions	400

The protocol uses greedy inference with BF16 serving. Poker records retain the initial-deck fingerprint to check paired initialization. In the revised DDZ comparison, per-deal Python, NumPy, and Torch seeds are derived from the physical opening hash and shared across methods. Guandan evaluates both team positions, limits the exposed legal-action list to 256, and restricts Koi's visibility to the acting player's hand. Its history cap is 1,000,000. These controls specify the evaluated interface rather than an unrestricted game-solving setting.

A.4.2 Paired uncertainty estimates

The uniform-versus-activation comparison uses 5,000 paired cluster-bootstrap replicates. A cluster comprises one DDZ or Guandan opening, one poker seed with both seats, or one Riichi seed with all four rotations. For a difference $\widehat{\delta} = \widehat{m}_{\text{activation}} - \widehat{m}_{\text{uniform}}$, each replicate resamples clusters with replacement and preserves the pairing of methods and rotations. Percentile endpoints give the reported 95% interval. These intervals quantify sampling variation over the recorded openings; they do not measure variation across independently trained specialists.

The paired results show game-dependent responses to activation weighting rather than a uniform improvement. For each of the five main-opponent metrics, the interval contains zero. The additional Limit rule-opponent comparison favors uniform fitting in the recorded evaluation, but the intervals are pointwise and do not provide a simultaneous multiple-comparison guarantee. Identical Leduc returns establish equality of the recorded outcome metric, not equality of the full action distributions or strategic strength against other opponents.

Table 8: Archived paired gameplay comparison. Differences are activation minus uniform. Win-rate and top-two-rate entries, including their differences and intervals, are percentages or percentage points; poker entries retain native return units.

Game	Opponent	Uniform	Activation	Difference [95% interval]
DDZ	WP	45.20	44.10	-1.10 $[-2.50, \quad 0.20]$
DDZ	RLCard	87.10	87.00	-0.10 $[-1.20, \quad 1.00]$
Limit	Teacher	0.0381	0.0436	0.0055 $[-0.0055, 0.0182]$
Limit	Rule	0.41555	0.41135	-0.00420 $[-0.00780, -0.00060]$
Leduc	Rule	1.0872	1.0872	0.0000 $[0.0000, 0.0000]$
Leduc	Teacher	0.0000	0.0000	0.0000 $[0.0000, 0.0000]$
Riichi	Mortal v4	38.00	39.40	1.40 $[-2.40, 5.20]$
Riichi	Release4p	43.00	42.80	-0.20 $[-3.80, \quad 3.80]$
Guandan	Koi	35.50	34.00	-1.50 $[-4.25, \quad 1.00]$

The complementary opponents further illustrate why outcome quality cannot be inferred from a single measure of specialist agreement. In Limit Hold'em, activation weighting changes mean return by $+0.0055$ against the teacher but by -0.00420 against the rule policy. Thus, even the direction of the observed return change depends on the opponent. Leduc provides a different case: the two constructions attain identical returns against each recorded opponent despite differences in their offline distributional fidelity. These point estimates motivate a direct examination of fidelity and outcomes, rather than assuming that improved reconstruction necessarily improves gameplay.

Historical openings are reused in these evaluations. Consequently, they should not be described as newly collected confirmation data. Koi is accessed through a remote service whose server-side version cannot be pinned by the local artifacts. Evaluator hashes and execution records preserve the controllable parts of the protocol. Request failures must remain distinguishable from completed losses.

A.5 Policy Fidelity and Representation Diagnostics

A.5.1 Data and metric definitions

The diagnostic protocol uses 128 calibration prompts and 128 validation prompts per game. Teacher-input second moments are computed from at most 64 evenly spaced tokens per prompt. The archived development pipeline also retains 100 prompts per game for implementation checks. These information-set counts are separate from the gameplay counts in Table 7. Available game identifiers support grouped splitting for DDZ and Guandan; the older poker and Riichi summaries guarantee prompt separation rather than complete trajectory-level separation.

For a candidate update $\widehat{T}_{g,\ell}$ and validation input second moment $\Sigma^{\mathrm{val}}_{g,\ell}$, define the relative functional error

$$\varepsilon_g = \frac{\sum_\ell \operatorname{tr}[(T_{g,\ell} - \widehat{T}_{g,\ell})\Sigma^{\mathrm{val}}_{g,\ell}(T_{g,\ell} - \widehat{T}_{g,\ell})^\top]}{\sum_\ell \operatorname{tr}[T_{g,\ell}\Sigma^{\mathrm{val}}_{g,\ell}T_{g,\ell}^\top]}. \tag{23}$$

Summing energies before taking their ratio prevents a low-energy module from receiving the same weight as a high-energy module solely because both are modules. Cross-game summaries then average the five game-level ratios equally. For a refitted feature probe, $100(1-\varepsilon_g)$ is called explained update energy. It is not an action-accuracy percentage or a win rate, and it need not lie between zero and 100 for an arbitrary poorly fitting probe.

Policy fidelity is evaluated separately using legal-action distributions. For specialist and candidate distributions p and q, the Jensen–Shannon divergence is $\frac{1}{2}\mathrm{KL}(p\|m) + \frac{1}{2}\mathrm{KL}(q\|m)$, where $m = (p+q)/2$. Top-action agreement compares their highest-scoring legal actions. Generated-token agreement evaluates the recorded generation comparison and is distinct from both distributional divergence and interactive reward.

A.5.2 REPRESENTATION AND FITTING METRIC

Table 9 reports the archived four-cell diagnostic and its pure-private control at the 26,148,864-parameter budget. Shared-matrix models use a common update together with private residuals, whereas shared-basis models retain game-specific readouts. The control establishes that local fidelity alone does not require sharing.

Table 9: Offline fidelity on the diagnostic validation prompts. JS and relative functional error are equal-game means; agreement is reported in percent. Lower is better for JS and error.

Representation / fit	JS	Top action	Generation	Relative error
Shared matrix / uniform	0.003251	96.88	96.72	0.025139
Shared matrix / activation	0.015103	92.81	91.25	0.033358
Shared basis / uniform	0.001194	98.28	97.66	0.008971
Shared basis / activation	0.001234	98.28	98.12	0.003410
Pure-private control	0.000628	98.91	98.91	0.002883

Under uniform fitting, the shared-basis representation has lower policy divergence and functional error than the shared-matrix representation. Within the shared-basis family, activation weighting reduces functional error while leaving top-action agreement unchanged and slightly increasing mean JS. The pure-private control attains the best fidelity values in this table. These observations show that local update recovery, policy agreement, and gameplay are related but non-interchangeable evaluation targets; they do not establish universal superiority of sharing.

The size of the reconstruction gain makes this distinction consequential. Activation weighting reduces the shared-basis model's mean relative functional error by 61.99%, from 0.008971 to 0.003410, while the mean top-action agreement remains 98.28%. Generated-token agreement increases from 97.66% to 98.12%, yet the gameplay changes include increases, decreases, and ties. Improving a continuous local approximation can therefore leave the selected action unchanged, or modify decisions whose strategic consequences differ substantially. The leading pure-private fidelity scores likewise establish strong specialist reconstruction, not an ordering of final win rates.

A.5.3 DECISION PRESERVATION VERSUS GAMEPLAY OUTCOMES

We examine this relationship using the uniform and activation-weighted shared-basis models, for which the ten per-game adapter hashes match between the offline fidelity records and the paired gameplay protocol. This links the diagnostics to the evaluated policies without pooling unrelated configurations. Table 10 reports within-game changes. Poker returns, win rates, and top-two rates retain their own units; no correlation is computed by mixing their raw levels.

Table 10: Matched changes from uniform to activation-weighted fitting. Agreement changes are percentage points; ΔJS is multiplied by 10^3. Lower JS is better. Outcomes are win rate (DDZ/Guandan), top-two rate (Riichi), or native return (poker); rate changes are percentage points.

Game	Δtop-action agreement	Δgeneration agreement	$10^3\Delta$JS	Δoutcome
DDZ	-0.781	$+1.563$	$+0.134$	-1.10
Limit	0.000	0.000	$+0.163$	$+0.0055$
Leduc	0.000	0.000	-0.057	0.0000
Riichi	-0.781	0.000	$+0.048$	$+1.40$
Guandan	$+1.563$	$+0.781$	-0.089	-1.50

The per-game comparisons do not follow a consistent monotonic relationship. Guandan improves both top-action agreement (97.66% to 99.22%) and distributional fidelity, but its win rate decreases from 35.50% to 34.00%. Riichi exhibits the opposite pattern: top-action agreement decreases from 97.66% to 96.88% and JS increases, while the top-two rate rises from 38.00% to 39.40%. DDZ also distinguishes different notions of preservation: top-action agreement decreases, whereas generated-token agreement increases, even though both compare the same pair of models. Limit maintains 100% top-action agreement on the diagnostic prompts while changing return. These are counterex-

amples to a deterministic monotonic mapping from the measured fidelity scores to gameplay point estimates, rather than individually significant outcome reversals.

To assess directional association without conflating outcome units, we perform an exploratory exact sign-concordance test across games. Let ΔF_g denote the change in a fidelity score oriented so that positive means better preservation: activation minus uniform for agreement, and uniform minus activation for JS and functional error. Let ΔO_g be activation minus uniform for the main-opponent outcome. Among games with both changes nonzero, define

$$K = \sum_g \mathbf{1}\{\Delta F_g \Delta O_g > 0\}, \qquad K \sim \mathrm{Binomial}(n, 1/2) \quad \text{under the reference null.} \tag{24}$$

The reference null assumes independent, equally likely concordant and discordant signs. We enumerate all 2^n sign outcomes and use a two-sided exact test. Tied changes are excluded separately for each diagnostic, so the tested sample size is the number of informative games, not the number of prompts or played hands.

Table 11: Exploratory directional association between fidelity improvements and gameplay improvements. All four diagnostics are reported. The tests use game-level point estimates and are not Pearson or Spearman correlation tests.

Fidelity diagnostic	Non-tied games	Concordant	Discordant	Exact p
Top-action agreement	3	1	2	1.000
Generated-token agreement	2	0	2	0.500
Lower JS	4	1	3	0.625
Lower functional error	4	2	2	1.000

No statistically significant directional association is detected at the 0.05 level for any of the four diagnostics in this matched comparison. However, only two to four games contribute to each test, giving limited power and coarse attainable p-values. The analysis uses noisy point estimates, does not propagate their estimation uncertainty, and treats a deliberately selected game suite under a reference sign model. Moreover, the main-opponent paired outcome intervals all contain zero. The warranted conclusion is therefore that these results do not establish fidelity as a reliable predictor of final gameplay performance; they do not establish zero population correlation or statistical independence.

A.5.4 Why agreement need not track final reward

The distinction has a sequential-decision explanation. Agreement treats each sampled decision equally, whereas return depends on the strategic value of the action changes and on the histories visited after those changes. For a fixed opponent and a finite horizon T, let π_0 denote the specialist, π the compressed policy, and h_t the observable history at the agent's tth decision. Writing $Q_t^{\pi_0}(h_t, a)$ for expected remaining reward after action a followed by π_0, the performance difference can be expressed as

$$J(\pi) - J(\pi_0) = \sum_{t=0}^{T-1} \mathbb{E}_{h_t \sim d_t^{\pi}} \left[\sum_a \big(\pi(a \mid h_t) - \pi_0(a \mid h_t)\big) Q_t^{\pi_0}(h_t, a) \right]. \tag{25}$$

Here, d_t^{π} is the history distribution induced by the evaluated policy and the fixed opponent; terminal histories can be padded with zero reward. The identity follows by defining the specialist advantage as $Q_t^{\pi_0} - V_t^{\pi_0}$ and telescoping the specialist value along trajectories of π. It is a characterization of expected return, not an additional measured quantity in our experiments.

Equation equation 25 exposes three reasons that fidelity alone cannot determine the sign of the outcome change. First, agreement and JS do not weight a disagreement by the difference in continuation value between the affected actions: many low-impact changes can matter less than one pivotal error. Second, the diagnostics use fixed specialist-input prompts, while gameplay evaluates the histories induced by the compressed policy, so early deviations can change later opportunities and risks. Third, an imperfect specialist is a behavioral reference rather than an optimality certificate; departing from its action can either improve or reduce return, depending on the opponent and

history. These mechanisms explain why non-monotonic outcomes are possible without identifying which mechanism caused an individual observed difference.

Accordingly, we use fidelity to characterize preservation and diagnose compression, while evaluating strategic utility through direct gameplay. High preservation remains useful evidence that a compressed policy retains much of the specialist's behavior, but it is insufficient to rank nearby methods by win rate. Conversely, a favorable gameplay point estimate does not imply more accurate matrix reconstruction or more faithful reproduction of the original decisions.

A.5.5 CROSS-GAME SUBSPACE RECOVERABILITY

The CSS probe tests whether directions obtained from other specialists can express a target specialist's local updates. For each target game, a leave-one-game-out basis is refitted from the other four expert matrices with their original private-rank allowances and four uniform alternating sweeps. Comparators include the saved all-five basis, single-source SVD bases, and 20 seeded random bases of the same dimension. A target-game linear readout is fitted on calibration second moments and evaluated using validation second moments.

All probes are linear without an intercept and use trace-scaled ridge regularization of 10^{-6}. Target calibration information is used to fit the readout even in the leave-one-game-out condition. The all-five reference also contains the target expert when constructing its basis. Furthermore, the compact coordinate system is the joint source span. These conditions make the experiment a controlled recoverability diagnostic, rather than a zero-shot transfer test to an unseen game.

Table 12: CSS probe: explained update energy (%). Random entries average 20 seeded replicates; the mean column weights games equally.

Feature construction	DDZ	Limit	Leduc	Riichi	Guandan	Mean
Random basis	57.20	64.03	64.65	68.54	72.73	65.43
Single-source basis (mean)	68.94	84.44	83.93	78.03	81.52	79.37
Leave-one-game-out basis	81.20	83.41	82.35	90.02	89.65	85.32
All-five basis	98.04	87.06	84.74	94.99	97.58	92.48

The leave-one-game-out basis explains 85.32% of update energy on average, exceeding the single-source mean of 79.37% and the random-basis mean of 65.43%. The all-five basis reaches 92.48%. The improvement from multiple source games supports the presence of reusable linear directions, while the gap to the all-five reference identifies target-specific structure that source-only fitting does not fully recover.

The game-level pattern is more informative than the mean alone. Moving from the single-source mean to the leave-one-game-out basis improves explained energy by 12.25 percentage points in DDZ, 11.99 in Riichi, and 8.13 in Guandan, but slightly reduces it in Limit and Leduc. Adding the target source to form the all-five reference gives the largest additional gain in DDZ (16.85 points), compared with 3.66 and 2.39 points in the two poker games. Thus, pooling sources yields useful but heterogeneous representational coverage. These gains quantify what a refitted linear readout can recover; without corresponding deployed-policy interventions, they cannot be converted into win-rate gains or used as independent samples in the fidelity–outcome association test.

A.5.6 PRIVATE-FEATURE COMPLEMENTARITY

The PRA diagnostic asks how much additional target-update energy can be recovered beyond the saved shared row space. For each source game, its saved private update is projected off that row space, and leading right singular directions form a private feature basis. Feature ranks are matched to the minimum effective private rank across games within each module. Each augmented probe concatenates the shared features with one private feature basis and refits a target-game readout; 20 same-rank random complements provide a control. This operation tests information available in the factors and is not an intervention on the deployed policy.

Own-game private features raise mean explained energy from 92.48% to 99.60%. The comparison with other-game and random features in Table 13 distinguishes the value of structured residual directions from the effect of simply increasing feature dimension. Because all readouts are refitted, these

Table 13: PRA probe: explained update energy (%) after refitting a target-game readout. Other-game entries average the four non-target private-feature sources.

Feature construction	DDZ	Limit	Leduc	Riichi	Guandan	Mean
Shared features	98.04	87.06	84.74	94.99	97.58	92.48
+ Random complement	98.51	91.36	90.07	96.83	98.43	95.04
+ Other-game private (mean)	98.83	94.02	93.23	97.13	98.62	96.36
+ Own-game private	99.78	99.32	99.22	99.79	99.89	99.60

values quantify linear recoverability under the specified probe, rather than the fraction of gameplay performance causally attributable to the private branch.

The controls separate a dimensionality effect from target-specific residual structure. A random complement increases mean explained energy to 95.04%, while other-game private features reach 96.36%, showing that structured features provide more recovery than arbitrary additional directions. Own-game features further increase recovery to 99.60%; their gains over shared-only features are especially large in Limit (12.26 points) and Leduc (14.49 points), whose shared-only coverage is lower. Nevertheless, the last increments of recovered energy need not correspond to strategically important action changes. Near-complete linear recovery is therefore evidence for residual complementarity, not a prediction of near-perfect play or a guaranteed advantage over a less faithful policy.

A.5.7 Participation of shared directions

To characterize whether several games contribute to a shared direction, the diagnostic first subtracts each saved private update from its source update and rotates the shared row space into canonical directions using pooled residual energy. For each direction, let e_g be the squared residual response associated with game g. Raw contribution fractions are $p_g = e_g / \sum_h e_h$. A second statistic first normalizes the contribution by each game's residual-update scale and then renormalizes across games. For either set of fractions, effective participation is

$$N_{\text{eff}} = \exp\left(-\sum_{g=1}^{G} p_g \log p_g \right), \qquad 1 \le N_{\text{eff}} \le G, \tag{26}$$

with $0 \log 0 = 0$. Equal contributions attain G, while concentration on one game gives 1.

Across the 560 saved shared directions, mean normalized participation is 4.35, and 78.6% of directions have at least four effective contributing games. Layer-wise normalized means range from 4.15 to 4.59, compared with 2.13 to 3.15 before normalization. The difference shows that raw energy is sensitive to specialist scale. Broad normalized participation supports structural reuse in the fitted subspaces; it does not imply that the games share identical strategies or that individual directions have unique semantic meanings.

Participation and outcome quality answer different questions. The normalized statistic shows that sharing is not restricted to a small subset of games, while its contrast with the raw statistic shows that broad coverage can coexist with unequal update magnitudes. Neither statistic includes the continuation value of actions or the opponent-dependent visitation distribution in Eq. equation 25. A direction can therefore be broadly reused without contributing equally to each game's win rate. Participation supports the design rationale for CSS, while direct gameplay remains necessary to assess the utility of a particular shared–private allocation.

A.6 Scope of the Supplementary Evidence

UniGame addresses consolidation of known specialists sharing a compatible backbone and adapted module layout. Its results do not establish generalization to an unseen game, automatic game identification, or a reduction in the common backbone's storage. The 40% reduction is an exact statement about the packed adapter budget. The gameplay results concern the specified opponents and interfaces, while the representation analyses concern frozen specialist inputs and local linear transformations.

The diagnostic validation prompts have been used during method development, so their results are presented as explanatory evidence. Random-probe replicates measure variability in the sampled control features, not confidence intervals for gameplay. The matched comparison detects no significant directional association between the examined preservation diagnostics and final gameplay changes, within the limited game-level sample. The evidence supports shared input directions as a compact representation of cross-game update structure, but it does not justify using specialist agreement as a substitute for outcome evaluation. Whether a particular allocation improves interactive performance remains an empirical, game-dependent question.