Self-Play in the Age of Foundation Models: A Comprehensive Survey from Game-Theoretic Foundations to Open-Ended Learning

Deli Chen*

Abstract

Self-play—training agents by competing against themselves—has produced superhuman performance from TD-Gammon (1992) through AlphaZero (2018) to the reasoning-model revolution (o1, DeepSeek-R1, 2024–2025), building on game-theoretic foundations laid in 1951. Yet the literature remains fragmented across game theory, deep reinforcement learning, and language model alignment. This survey unifies these threads around a single thesis: **the quality of the verification signal determines the ceiling of self-play improvement**.

We make five contributions. **(1)** A three-axis taxonomy organizing self-play methods by game structure, mechanism, and theoretical guarantees, positioning classical game AI and LLM self-play in one framework. **(2)** An original improvement theory: three theorems establishing monotonic improvement with an $O(n/T)$ rate under perfect verification, a hard noise floor $2\epsilon V_{\max}/(1-2\epsilon)$ under verifier error rate ϵ, and a diversity–stability tradeoff for population methods. **(3)** Three layers of original empirical validation, including a controlled experiment at **285B-parameter scale**: we train a mixture-of-experts model with GRPO self-play under four calibrated verifier-noise levels and observe the predicted monotone ordering—noise-free training-distribution improvement falls from $+4.8\%$ at $\epsilon = 0$ to -6.6% at $\epsilon = 0.45$, with the sign change between $\epsilon = 0.10$ and $\epsilon = 0.30$ replicated across independent seeds at all three of $\epsilon \in \{0, 0.10, 0.30\}$. Extending both sign-change conditions to 2,000 steps ($8.3\times$ the original horizon) shows the divide is *persistent*: the noisy run is pinned below its starting capability for the entire run, yet held-out evaluation under a noise-free grader (with problem-level bootstrap confidence intervals) remains buffered—training-signal collapse *precedes* held-out degradation durably, making training-distribution pass rates the leading indicator of verifier failure. A KL-coefficient ablation, measuring *both* training-distribution and held-out performance at fixed $\epsilon = 0.30$, turns the buffer into a tradeoff: as the KL anchor weakens, training-distribution degradation grows monotonically ($+0.8\%$ at KL=0.01, -4.1% at 0.001, -10.9% at KL=0, the last replicated across two seeds) while held-out accuracy moves the *opposite* way ($0.525 \to 0.686$ on IMOAnswerBench, non-overlapping CIs at the endpoints), so the KL anchor relocates the cost of noise between the two axes rather than removing it—making KL strength a first-class design *axis* with a sweet spot, not a uniform robustness lever. **(4)** A failure-mode taxonomy (reward hacking, mode collapse, strategy cycling, model collapse) connected back to the theory's quantities (ϵ, diversity D, population size K). **(5)** Comprehensive coverage of 200+ works spanning fictitious play (1951) to LLM self-play (2026)—AlphaZero, PSRO, CFR, league training, SPIN, debate, and verifiable-reward RL—distilled into practitioner guidelines: before adopting self-play, ask "how reliable is my verification signal?"

1 Introduction

The idea that an agent can improve by competing against itself is both elegantly simple and profoundly powerful. Rather than relying on databases of human expert play, hand-crafted heuristics, or curated training corpora, self-play offers a path to mastery through pure self-generated experience. This philosophy—that *tabula rasa* learning can surpass human knowledge—has produced some of the most celebrated achievements in artificial intelligence, from AlphaGo Zero's superhuman Go play (Silver et al., 2017) to Pluribus's victory over professional poker players (Brown and Sandholm, 2019b).

*This paper was autonomously generated by the **Deli AutoResearch** framework. AI tools used: DeepSeek-V4-Pro (text generation, reasoning), GPT-Image-2 (figure generation). Personal research project—all opinions are the author's own. Correspondence: `chendeli96@gmail.com`

The appeal of self-play rests on a fundamental insight: in competitive settings, the agent's own past behavior provides an ever-improving curriculum. As the agent strengthens, so does its training signal, creating a virtuous cycle of improvement that—under appropriate conditions—can converge to optimal play without any external supervision. This self-bootstrapping property distinguishes self-play from both supervised learning (which requires expert demonstrations) and standard reinforcement learning (which requires an environment to provide rewards).

1.1 A Brief History

The history of self-play in AI traces a remarkable arc of increasing ambition:

- **1992–2000: Early Foundations.** Tesauro's TD-Gammon (Tesauro, 1995) demonstrated that a neural network trained through self-play could achieve world-class backgammon performance, challenging the prevailing wisdom that learning from human data was essential.
- **2000–2015: Theoretical Developments.** Game-theoretic foundations were established, including fictitious play convergence results (Robinson, 1951; Berger, 2007), counterfactual regret minimization for imperfect-information games (Zinkevich et al., 2007), and population-based approaches to multi-agent learning.
- **2016–2019: The AlphaGo Revolution.** DeepMind's progression from AlphaGo (Silver et al., 2016) to AlphaGo Zero (Silver et al., 2017) to AlphaZero (Silver et al., 2018) demonstrated that self-play combined with deep neural networks and Monte Carlo Tree Search could achieve superhuman performance *from scratch* across multiple games. Simultaneously, AlphaStar (Vinyals et al., 2019) and OpenAI Five (Berner et al., 2019) extended self-play to complex real-time strategy games through league training and population-based methods.
- **2020–2023: Generalization and Scaling.** MuZero (Schrittwieser et al., 2020) learned environment models through self-play. PSRO (Lanctot et al., 2017) and its variants provided principled frameworks for multi-agent training. Poker AI reached superhuman play through CFR-based self-play (Brown and Sandholm, 2019b).
- **2024–2026: The LLM Self-Play Explosion.** Self-play principles have been transformed and applied to large language model training. SPIN (Chen et al., 2024c) showed that LLMs can self-improve without stronger teachers. Self-Play Preference Optimization (SPPO) (Wu et al., 2024), debate (Irving et al., 2018; Khan et al., 2024), and constitutional AI (Bai et al., 2022b) represent new frontiers where self-play transcends its game-theoretic origins.

1.2 Scope and Contributions

This survey provides a comprehensive and unified treatment of self-play methods. Our key contributions are:

1. **Unified Three-Axis Taxonomy.** We propose a novel taxonomy that organizes the self-play literature along three dimensions—game structure, mechanism, and theoretical guarantees—enabling systematic comparison across disparate domains (Section 2).
2. **Original Theoretical Framework.** We establish three original theorems with proofs formalizing self-play improvement: monotonic improvement under perfect verification, degradation bounds under noisy verifiers, and a diversity-stability tradeoff for population methods (Section 3.7). To our knowledge, these provide the first rigorous "improvement theory" connecting verification quality to self-play ceiling.
3. **Complete Historical Coverage.** We trace the evolution from classical game theory (1951) through modern deep RL self-play to the LLM reasoning model revolution (o1, DeepSeek-R1, 2024–2025), identifying the intellectual lineage connecting these developments (Sections 3–7).
4. **Bridge Between Game AI and Language Models.** We provide, to our knowledge, the first systematic analysis of how self-play principles are being adapted for foundation models, covering SPIN, SPPO, Nash Learning from Human Feedback, debate, MCTS-guided reasoning, and code self-play—identifying both successful translations and fundamental differences (Section 7).

5. **Critical Failure Analysis with Formal Bounds.** We provide, to our knowledge, the first comprehensive taxonomy of self-play failure modes and connect them to our theoretical framework: reward hacking corresponds to high verifier noise (Theorem 3), mode collapse to insufficient diversity (Theorem 5), and strategy cycling to population inadequacy (Section 7.18).

6. **Original Empirical Studies.** We provide three layers of original empirical evidence: (i) a controlled inference-time experiment comparing self-play strategies across difficulty levels; (ii) a PSRO matrix-game simulation validating the noisy-verifier bound with exact ground truth; and (iii) a **training-time validation at 285B-parameter scale**, where we train a large mixture-of-experts model with GRPO self-play under four calibrated verifier-noise levels (with independent seed replication of the sign-change conditions) and show that noise-free self-play improvement decreases strictly monotonically with noise, turning into active degradation at $\epsilon \geq 0.30$—consistent with the qualitative prediction of Theorem 3, whose stylized noise model (uniform payoff replacement, exact best-response iteration) the GRPO setting approximates but does not literally instantiate (Section 8). A long-horizon extension of both sign-change conditions to 2,000 steps (8.3× the original horizon) shows the noise-induced sign change is persistent—the noisy policy never recovers its starting training-distribution capability—while held-out performance remains buffered throughout (Section 8.12.4).

1.3 Relationship to Existing Surveys

Several excellent surveys cover subsets of this landscape:

- Hernandez-Leal et al. (2019a) survey multi-agent reinforcement learning broadly but do not focus on the self-play paradigm specifically.
- Lanctot et al. (2017) introduce PSRO and discuss game-theoretic methods but predate the LLM era.
- Bloembergen et al. (2015) connect evolutionary dynamics to multi-agent learning but focus on tabular settings.
- Recent surveys on LLM alignment (Wang et al., 2024a; Casper et al., 2023) cover RLHF/DPO but do not systematically connect to game-theoretic self-play.
- Zhang et al. (2024c) provide a contemporaneous survey on self-play in RL but with narrower scope (no LLM self-play coverage).

Our work differs by providing a *unified* treatment that connects game-theoretic self-play (1951–present) with its emerging applications in language models (2024–2026), offering a coherent intellectual framework spanning 75 years of research. To our knowledge, we are the first to propose a three-axis taxonomy that positions classical game AI and modern LLM self-play within a single analytical framework, and to systematically analyze failure modes across both traditions.

1.4 Scope and Limitations

To maintain focus and depth, this survey explicitly scopes its coverage:

Included:

- Self-play as a training paradigm: agents improving by competing against themselves or their own past.
- Population-based methods where agents within a population train each other.
- LLM self-improvement methods with a game-theoretic or adversarial structure.
- Evaluation and benchmarking of self-play systems.

Excluded:

- *General multi-agent RL* without self-play structure (e.g., independent learners in purely cooperative settings without self-generated curriculum).
- *Evolutionary strategies* unless explicitly framed as population-based self-play.

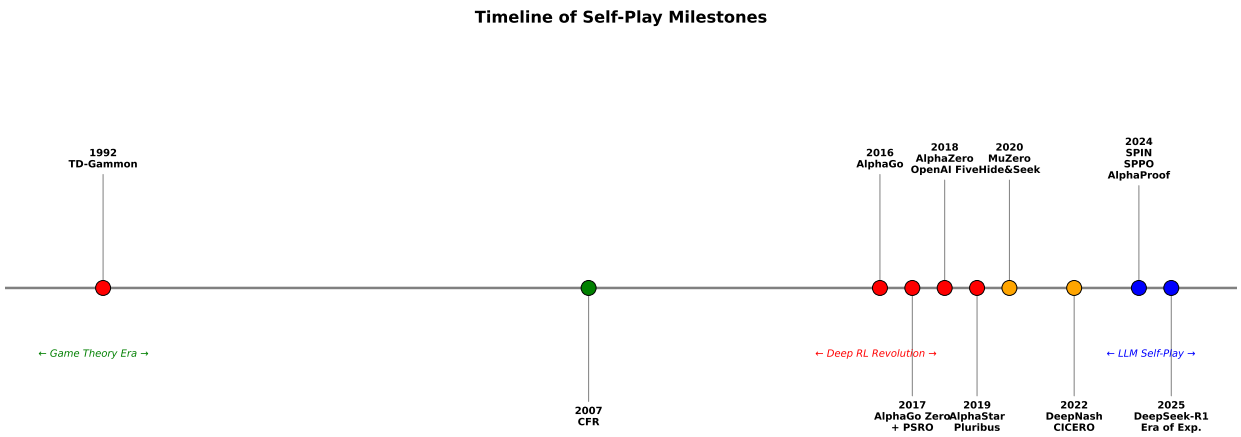

Figure 1: Timeline of major self-play milestones spanning three eras: the game theory era (classical algorithms), the deep RL revolution (AlphaGo through AlphaStar), and the LLM self-play era (SPIN through DeepSeek-R1).

- *Single-agent exploration methods* (curiosity-driven learning, count-based exploration) unless combined with self-play.
- *Supervised self-distillation* (knowledge distillation from larger to smaller models without iterative improvement).
- *Hardware and systems optimization* for self-play training at scale.

We also note that the LLM self-play landscape (Section 7) is rapidly evolving; our coverage reflects the state of the field through early 2026, and significant new developments may emerge after publication.

1.5 Central Thesis

A unifying insight emerges from spanning classical game AI and modern LLM self-play: **the quality of verification determines the ceiling of self-play improvement**. When the verifier is perfect (game rules, proof assistants, code test suites), self-play can improve without bound—as demonstrated by AlphaZero and DeepSeek-R1. When the verifier is learned and imperfect (reward models, self-evaluation), self-play saturates after a few iterations—as observed with SPIN and SPPO. When the verifier is adversarial (reward hacking), self-play can even degrade performance. Our theoretical framework (Section 3.7) formalizes this insight quantitatively, and our experiments (Section 8) validate it empirically. This verification-centric view provides practitioners with a simple diagnostic: before adopting self-play for a new domain, ask "how reliable is my verification signal?"

Running example: teaching a model to solve math by self-play. To keep the survey's abstractions concrete, we return throughout to one thread—training a language model to solve competition mathematics without human solutions. The reader can see each concept instantiated on this single task: the *game* is a model proposing and solving problems against itself (Section 2); *naive self-play vs. population methods* are different ways of choosing which past versions to train against (Section 6); the *verifier* is a proof checker or test suite—perfect when the answer is checkable, learned-and-leaky when a reward model scores partial work (Sections 7, 1.5); *MCTS-guided reasoning* searches the tree of solution steps (Table 9); and our *verifier-noise experiments* (Section 8) measure exactly what happens to this task as the verifier degrades. Wherever the "▶ Running example" marker appears, we connect the current concept back to this thread.

1.6 Organization

The remainder of this survey is organized as follows. Section 2 establishes mathematical foundations and our three-axis taxonomy. Section 3 covers classical algorithms and our original theoretical framework. Sec-

tions 4 and 5 survey applications in perfect- and imperfect-information games respectively. Section 6 discusses population-based methods and open-ended learning. Section 7 covers the emerging LLM self-play paradigm, from alignment methods to reasoning models. Section 8 analyzes evaluation and presents our original experiments. Section 9 identifies open problems and future directions.

Reading roadmap. Different readers can take different paths through the survey. *Short on time* (∼30 minutes): read Section 1.5 (central thesis), the theorem statements in Section 3.7 (skipping proofs), the 285B experiment (Section 8.12), and the practitioner takeaways in the conclusion. *Coming from game AI / RL*: Sections 2–6 will be largely familiar—skim them for our taxonomy positioning and go deep on Sections 7–8 to see how the classical machinery transfers (and fails to transfer) to language models. *Coming from NLP / LLM training*: read Sections 2 and 3 carefully—they supply the game-theoretic vocabulary (best response, exploitability, Nash) the rest of the paper relies on—then jump to Section 7. Readers who want only the original research contributions should read Sections 3.7 and 8 end to end.

2 Background and Preliminaries

This section establishes the mathematical foundations necessary for understanding self-play methods. We begin with game-theoretic concepts, then introduce the reinforcement learning framework, and finally present our unified taxonomy.

▶ *Running example.* On our math-solving thread, the "game" is a two-role interaction played by one model: a *proposer* role poses a problem and a *solver* role attempts it, with payoff determined by whether the solution verifies. The normal-form and Markov-game definitions below are the formal scaffolding for exactly this kind of self-competition.

2.1 Game-Theoretic Foundations

Definition 1 (Normal-Form Game)**.** *A finite normal-form game is a tuple $G = (N, \{S_i\}_{i \in N}, \{u_i\}_{i \in N})$ where $N = \{1, \ldots, n\}$ is the set of players, S_i is the finite strategy set of player i, and $u_i : \prod_{j \in N} S_j \to \mathbb{R}$ is the utility function for player i.*

Definition 2 (Nash Equilibrium (Nash, 1950))**.** *A mixed strategy profile $\sigma^* = (\sigma_1^*, \ldots, \sigma_n^*)$ is a Nash equilibrium if for every player i and every mixed strategy σ_i:*

$$u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(\sigma_i, \sigma_{-i}^*) \tag{1}$$

where σ_{-i}^* denotes the strategies of all players except i. Every finite game possesses at least one Nash equilibrium in mixed strategies.

Definition 3 (Two-Player Zero-Sum Game)**.** *A game where $n = 2$ and $u_1(s) = -u_2(s)$ for all strategy profiles s. The minimax theorem (von Neumann, 1928) guarantees the existence of a Nash equilibrium computable via linear programming.*

Definition 4 (Exploitability)**.** *For a strategy σ_i in a two-player zero-sum game, the exploitability is:*

$$\mathit{exploit}(\sigma_i) = \max_{\sigma_{-i}} u_{-i}(\sigma_i, \sigma_{-i}) - v_{-i}^* \tag{2}$$

where v_{-i}^* is the value of the game for player $-i$. A strategy with zero exploitability constitutes a Nash equilibrium.

2.2 Extensive-Form Games

Many self-play applications involve sequential decision-making, modeled as extensive-form games.

Definition 5 (Extensive-Form Game)**.** *An extensive-form game is a tuple $(N, H, A, P, \sigma_c, \{I_i\}, \{u_i\})$ where H is the set of histories (sequences of actions), $A(h)$ is the action set at history h, $P(h)$ assigns a player to each non-terminal history, σ_c is the chance player's strategy, I_i is the information partition of player i, and u_i maps terminal histories to payoffs.*

Definition 6 (Information Set)**.** *In imperfect-information games, player i cannot distinguish between histories in the same information set $I \in I_i$. A behavioral strategy assigns a probability distribution over actions at each information set.*

2.3 Reinforcement Learning Foundations

Definition 7 (Markov Decision Process)**.** *An MDP is a tuple* (S, A, T, R, γ) *where* S *is the state space,* A *is the action space,* $T : S \times A \to \Delta(S)$ *is the transition function,* $R : S \times A \to \mathbb{R}$ *is the reward function, and* $\gamma \in [0, 1)$ *is the discount factor.*

Definition 8 (Multi-Agent MDP (Markov Game))**.** *A Markov game extends MDPs to multiple agents:* $(N, S, \{A_i\}, T, \{R_i\}, \gamma)$ *where each agent* i *has its own action space* A_i *and reward function* R_i*.*

In the self-play setting, a single learning algorithm controls all players, using its own current or historical policies as opponents. The key insight is that the multi-agent problem reduces to a single-agent learning problem where the "environment" co-evolves with the learner (Sutton and Barto, 2018). This non-stationarity is both the source of self-play's power (adaptive curriculum) and its primary theoretical challenge, as standard convergence results from single-agent RL (Agarwal et al., 2021) do not directly apply. For comprehensive treatments of multi-agent RL theory, we refer readers to Zhang et al. (2021) (theoretical overview), Yang and Wang (2020) (game-theoretic perspective), and Shoham and Leyton-Brown (2009) (algorithmic foundations). The two-timescale stochastic approximation framework (Borkar, 2008) provides one theoretical lens, modeling the agent's policy updates as the fast timescale and the opponent's adaptation as the slow timescale. Classical MARL algorithms like Nash Q-Learning (Hu and Wellman, 2003) and AWESOME (Conitzer and Sandholm, 2007) provide convergence guarantees in self-play but require tabular representations, motivating the function-approximation methods surveyed in subsequent sections.

Self-play also connects deeply to evolutionary game theory (Fudenberg and Levine, 1993), where the concept of Evolutionarily Stable Strategies (ESS) (Maynard Smith and Price, 1973; Maynard Smith, 1982) provides a population-level solution concept. An ESS is robust to invasion by mutants—analogous to a self-play policy that cannot be exploited by novel opponents. The evolutionary dynamics perspective (Bloembergen et al., 2015; Tuyls et al., 2006) has directly influenced modern population-based methods (Section 6).

2.4 Unified Taxonomy

We organize self-play methods along three orthogonal axes, visualized in Figure 2:

Axis 1: Game Structure.

- *Symmetric perfect-information* (Go, Chess, Othello): Both players have identical roles, full observability.
- *Asymmetric perfect-information* (StarCraft asymmetric matchups): Different roles/capabilities, full observability.
- *Imperfect-information* (Poker, Mahjong, Hanabi): Hidden information necessitates reasoning about beliefs.
- *Cooperative/mixed-motive* (Diplomacy, Overcooked): Agents must balance cooperation and competition.
- *Non-game settings* (LLM self-improvement): No explicit opponent; the agent improves through self-generated feedback.

Axis 2: Self-Play Mechanism.

- *Naive self-play*: Train against the current or most recent version.
- *Fictitious play*: Best-respond to the historical average strategy.
- *Population-based*: Maintain a diverse population of policies.
- *PSRO*: Iteratively add best responses to a meta-game.
- *MCTS-guided*: Combine search with learned policy/value (AlphaZero family).
- *Auto-curriculum*: Automatically generate increasingly challenging tasks/opponents.
- *LLM self-play*: Generate training signal through self-evaluation, debate, or self-competition.

Axis 3: Theoretical Guarantees. This axis is ordered by decreasing formal strength, from full equilibrium convergence down to the absence of any formal guarantee:

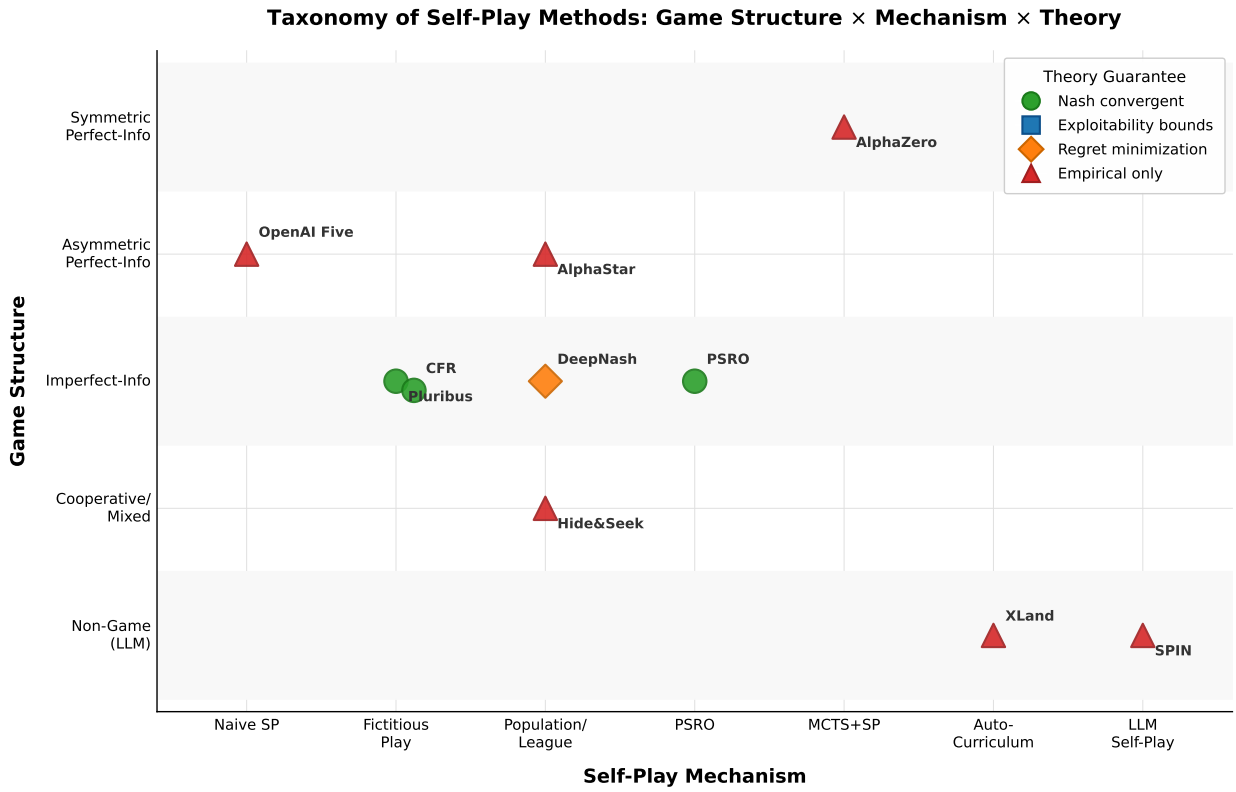

Figure 2: Our three-axis taxonomy for self-play methods. Each landmark system is positioned according to the game structure it addresses (vertical axis) and the self-play mechanism it employs (horizontal axis). Color encodes the third axis—theoretical guarantees: green indicates Nash convergence, orange indicates regret minimization, and red indicates empirical-only validation. Dot size reflects historical impact.

- *Nash convergence*: Provably converges to Nash equilibrium (e.g., CFR in two-player zero-sum).
- *Exploitability bounds*: Provides upper bounds on exploitability.
- *Regret minimization*: No-regret guarantees (average strategy converges).
- *No formal guarantee*: The baseline level of this axis—methods validated by empirical performance metrics alone.

2.5 A Unified View of Self-Play

Table 1 provides a comprehensive comparison of landmark self-play systems across our taxonomy dimensions, highlighting the diversity of approaches and their key design choices.

3 Foundations of Self-Play

This section traces the intellectual foundations of self-play from early game-theoretic algorithms to the deep reinforcement learning revolution. We identify the key ideas that enable agents to improve without human supervision.

3.1 Classical Self-Play Algorithms

3.1.1 Fictitious Play

The roots of self-play trace back to Samuel's checkers program (Samuel, 1959), which learned from games against itself as early as 1959. However, the first formal self-play algorithm is fictitious play (Robinson,

Table 1: Comprehensive comparison of landmark self-play systems across taxonomy dimensions.

System	Game Structure	Mechanism	Theory	Domain	Year	Compute
TD-Gammon	Symmetric, stochastic	Naive self-play	Empirical	Backgammon	1992	Days
CFR	Imperfect-info	Fictitious (regret)	Nash $O(1/\sqrt{T})$	Poker	2007	Hours-days
AlphaGo Zero	Symmetric perfect-info	MCTS + self-play	Empirical	Go	2017	40h (4 TPUs)
AlphaZero	Symmetric perfect-info	MCTS + self-play	Empirical	Chess/Shogi/Go	2018	9h-34h
OpenAI Five	Asymmetric	Naive + historical	Empirical	Dota 2	2019	10 months
AlphaStar	Asymmetric, partial	League (PFSP)	Empirical	StarCraft II	2019	44 days
Pluribus	Imperfect-info (6p)	CFR + self-play	Nash (approx)	Poker	2019	8 days
MuZero	Symmetric/single	MCTS + learned model	Empirical	Atari/Board	2020	Varies
DeepNash	Imperfect-info	R-NaD (population)	Regret	Stratego	2022	Weeks
CICERO	Mixed-motive + NL	Planning + negotiation	Empirical	Diplomacy	2022	Months
SPIN	Non-game (LLM)	Generative self-play	Convergence (limit)	LLM align	2024	Hours
SPPO	Non-game (LLM)	Preference self-play	Nash-like	LLM align	2024	Hours
rStar-Math	Non-game (LLM)	MCTS + self-play	Empirical	Math	2025	Hours
AlphaProof	Non-game (math)	MCTS + RL (verified)	Empirical	Theorem	2024	Weeks

1951). Each player maintains an empirical frequency of their opponent's past actions and best-responds to this historical average:

$$\sigma_i^{t+1} = BR_i\left(\frac{1}{t}\sum_{\tau=1}^{t}\sigma_{-i}^{\tau}\right) \tag{3}$$

Theorem 1 (Robinson (1951))**.** *In two-player zero-sum games, the time-averaged strategy profile of fictitious play converges to a Nash equilibrium.*

While fictitious play has strong theoretical properties, it suffers from computational challenges in large games and does not converge in general-sum games (Shapley, 1964). Heinrich et al. (2015) extended fictitious play to extensive-form games via *fictitious self-play* (FSP), where best responses are computed using reinforcement learning. This was further developed into *Neural Fictitious Self-Play* (NFSP) (Heinrich and Silver, 2016), combining deep RL for best response computation with supervised learning for average strategy estimation.

3.1.2 Double Oracle and Iterative Best Response

The double oracle algorithm (McMahan et al., 2003) maintains a restricted set of pure strategies for each player and iteratively expands it by computing best responses to the current equilibrium of the restricted game. This idea directly inspired PSRO (Section 6).

3.1.3 Counterfactual Regret Minimization

For extensive-form imperfect-information games, counterfactual regret minimization (CFR) (Zinkevich et al., 2007) decomposes the global regret into per-information-set counterfactual regrets:

$$R_i^T(I,a) = \sum_{t=1}^{T}\pi_{-i}^{\sigma^t}(I)\left[v_i^{\sigma^t|_{I\to a}}(I) - v_i^{\sigma^t}(I)\right] \tag{4}$$

where $\pi_{-i}^{\sigma}(I)$ is the reach probability of information set I under opponent strategy σ, and $v_i^{\sigma|_{I\to a}}$ is the counterfactual value when action a is always played at I.

CFR guarantees that the average strategy converges to a Nash equilibrium in two-player zero-sum games at rate $O(1/\sqrt{T})$. Variants including CFR+ (Tammelin, 2014), Linear CFR (Brown and Sandholm, 2019a), and DCFR (Brown et al., 2019) have improved convergence speed and scalability.

3.2 TD-Gammon: The First Deep Self-Play Success

Tesauro's TD-Gammon (Tesauro, 1995) combined temporal-difference learning (Sutton, 1988) with a neural network trained entirely through self-play in backgammon. Key innovations:

- Self-play as *curriculum*: the agent's improving opponent provides an adaptive difficulty curve.

- *Stochasticity* in backgammon (dice rolls) helped avoid cyclic strategies.
- Achieved expert-level play with zero human knowledge beyond game rules.

TD-Gammon's success was paradoxical: subsequent attempts to replicate the approach in deterministic games like chess and Go largely failed, suggesting that self-play stability depends on game properties. This puzzle was not resolved until the AlphaGo era.

3.3 The AlphaGo to AlphaZero Arc

3.3.1 AlphaGo (2016)

AlphaGo (Silver et al., 2016) combined four components: (1) supervised learning from human games, (2) policy gradient reinforcement learning through self-play, (3) value network trained on self-play outcomes, and (4) Monte Carlo Tree Search integrating all components. While it defeated world champion Lee Sedol, it still relied heavily on human data for initialization.

3.3.2 AlphaGo Zero (2017)

AlphaGo Zero (Silver et al., 2017) eliminated all human knowledge, learning entirely from self-play starting from random weights. Key innovations:

- **Unified network**: A single network $f_\theta(s) = (\mathbf{p}, v)$ outputs both a policy vector and value estimate.
- **MCTS as policy improvement**: Tree search guided by the network produces improved policy targets $\boldsymbol{\pi}$ for training.
- **Self-play training**: Games are played between the current network and itself; the outcome $z \in \{-1, +1\}$ provides the value target.
- **Loss function**: $\ell = (z - v)^2 - \boldsymbol{\pi}^\top \log \mathbf{p} + c\|\theta\|^2$

AlphaGo Zero surpassed all previous versions (including the human-data-trained AlphaGo Master) within 40 hours of training, providing the strongest evidence that self-play can discover knowledge beyond human expertise.

3.3.3 AlphaZero (2018)

AlphaZero (Silver et al., 2018) generalized the approach to chess and shogi with minimal modifications, demonstrating that the MCTS + self-play framework is remarkably game-agnostic. Within hours of training, AlphaZero defeated the strongest existing programs (Stockfish in chess, Elmo in shogi) with a distinctively creative playing style. Anthony et al. (2017) provided a theoretical framework for understanding this success: the "Thinking Fast and Slow" decomposition, where the fast neural network (System 1) provides intuition and the slow MCTS (System 2) provides deliberation, with self-play training the fast system to internalize the slow system's judgments.

Observation 1 (AlphaZero's Self-Play Dynamics)**.** *The training trajectory of AlphaZero reveals spontaneous discovery of known opening theory, tactical motifs, and strategic principles—all emerging purely from self-play without any human input. This suggests that self-play in rich domains can discover* deep structure *beyond mere optimization.*

3.3.4 MuZero (2020)

MuZero (Schrittwieser et al., 2020) extended the paradigm by learning an environment model alongside the policy and value functions, enabling application to environments where the rules are not given (e.g., Atari). The self-play mechanism remains central: games are played against the learned model's predictions, and the model is refined through self-play experience. Subsequent work extended MuZero to online settings (Schrittwieser et al., 2021) and complex action spaces (Hubert et al., 2021), broadening the applicability of model-based self-play.

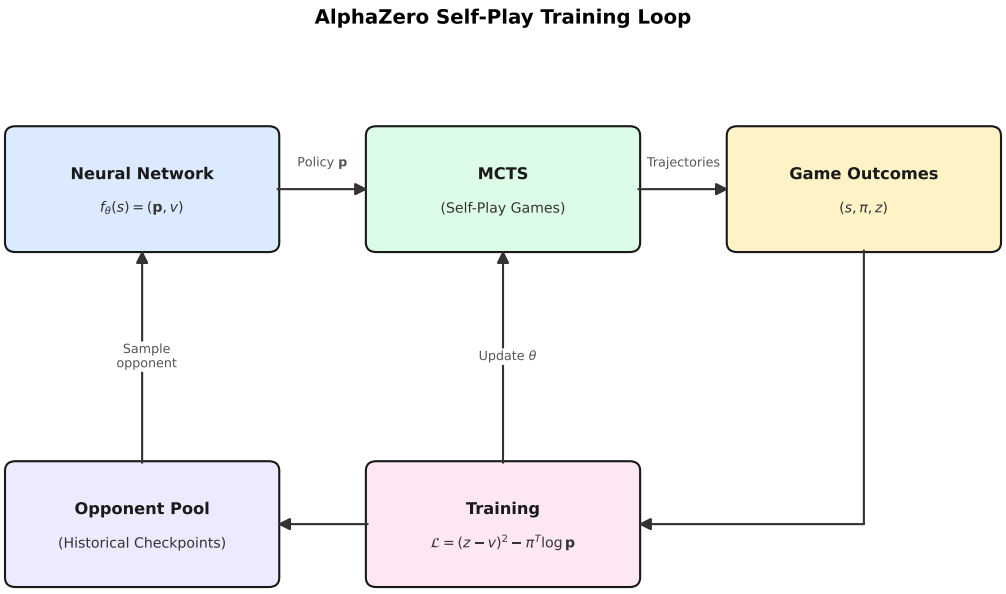

Figure 3: The AlphaZero self-play training loop. The neural network provides policy and value estimates to MCTS, which generates improved policy targets through self-play games. Game outcomes provide value targets. The opponent is sampled from a pool of historical checkpoints to prevent catastrophic forgetting.

3.3.5 Subsequent Advances: Sample Efficiency and Search Improvements

Two important follow-ups addressed practical limitations of the AlphaZero framework:

EfficientZero (Ye et al., 2021): Achieved superhuman performance in Atari with only 2 hours of game experience (100K environment steps)—a 50× improvement in sample efficiency over MuZero. Key innovations include self-supervised consistency loss for the learned model, end-to-end prediction of value prefixes, and temporal consistency regularization. EfficientZero demonstrates that self-play sample efficiency can be dramatically improved through better model learning, without changing the core self-play loop.

Gumbel AlphaZero / Gumbel MuZero (Danihelka et al., 2022): Replaced the standard PUCT-based MCTS with a theoretically grounded search algorithm based on the Gumbel-Top-k trick. Unlike PUCT (which lacks convergence guarantees with function approximation), Gumbel MuZero provides a policy improvement guarantee at any number of simulations, including with as few as one simulation per action. This is significant for Theorem 2: Gumbel MuZero ensures that each self-play iteration provably improves the policy, closing the gap between the tabular guarantees and practical deep self-play.

3.4 Key Theoretical Properties

Conjecture 1 (Self-Play Convergence in Symmetric Zero-Sum Games)**.** *In symmetric two-player zero-sum games with continuous strategy spaces, properly regularized self-play with function approximation converges to an approximate Nash equilibrium, provided:*

1. *The policy class has sufficient representational capacity.*
2. *The search component (MCTS) provides adequate exploration.*
3. *Historical opponents are retained to prevent catastrophic forgetting.*

Remark 1. *This conjecture formalizes the empirical observation that AlphaZero-style self-play works in practice. Formal proofs remain elusive due to the interaction between function approximation, non-stationary opponents, and search.*

Table 2: Convergence properties of self-play across game types.

Game Type	Convergent?	Algorithm	Rate
Two-player zero-sum (normal-form)	Yes	Fictitious play	$O(t^{-1/(2n-2)})$ worst-case (Shapiro, 1958)
Two-player zero-sum (extensive-form)	Yes	CFR	$O(1/\sqrt{T})$
Symmetric zero-sum (AlphaZero)	Empirical	MCTS + self-play	Unknown
Multi-player general-sum	No guarantee	PSRO (partial)	Game-dependent
Cooperative	N/A	Self-play + reward shaping	Task-dependent

3.5 The Self-Play Training Loop

We formalize the general self-play training procedure that underlies all variants discussed in subsequent sections:

Definition 9 (Self-Play Training Loop)**.** *A self-play training loop consists of:*

1. ***Data Generation**: Play N games between the current policy π_θ (possibly guided by search) and an opponent policy $\pi_{opp} \in \mathcal{P}$ (drawn from a pool $\mathcal{P}$), generating trajectories $\tau_1, \ldots, \tau_N$.*
2. ***Target Computation**: From trajectories, compute training targets $(s, \boldsymbol{\pi}, z)$ where $\boldsymbol{\pi}$ is the search-improved policy and z is the outcome.*
3. ***Parameter Update**: Minimize loss $\mathcal{L}(\theta) = \sum_{(s,\boldsymbol{\pi},z)}[(z - v_\theta(s))^2 - \boldsymbol{\pi}^\top \log p_\theta(s)]$.*
4. ***Opponent Update**: Update the opponent pool $\mathcal{P}$ (e.g., add new checkpoint, maintain population).*

The variants in this survey differ primarily in step (4)—how the opponent pool is managed—and secondarily in step (1)—the search procedure used during data generation.

3.6 Critical Assessment of Self-Play Foundations

Despite their elegance, classical self-play foundations have significant limitations:

Gap 1: Tabular-to-Neural. Classical convergence proofs (fictitious play, CFR) assume tabular representations. The extension to neural function approximation—which is what makes self-play practical in large games—has no formal guarantees. The "deadly triad" of off-policy learning, function approximation, and bootstrapping (Sutton and Barto, 2018) is compounded in self-play by non-stationarity of the opponent. Recent work on last-iterate convergence (Daskalakis and Panageas, 2022; Wei et al., 2021) provides partial results for gradient-based methods in games, Perolat et al. (2021) prove convergence for regularized self-play (R-NaD), and Mertikopoulos and Zhou (2019) establish convergence for continuous action spaces with unknown payoffs, but a general theory for deep self-play remains elusive.

Gap 2: Two-Player to Multi-Player. Most convergence results apply only to two-player zero-sum games. Extension to multi-player games lacks even basic solution concept agreement: should agents target Nash equilibrium (possibly unfair), correlated equilibrium (Hart and Mas-Colell, 2000) (requires communication), or welfare-optimal solutions? No-regret learning provides partial answers—Bowling (2001) shows that if all agents use no-regret algorithms, the joint strategy converges to a correlated equilibrium, and Piliouras et al. (2021) achieve bounded regret with unbounded step-sizes—but these guarantees are weaker than Nash convergence.

Gap 3: Perfect to Imperfect Information. In imperfect-information games, the structure of the solution space is fundamentally different: strategies must be *belief-consistent*, leading to computational intractability of Nash equilibrium in general extensive-form games.

The convergence properties of self-play differ fundamentally across game types:

3.7 Formal Theory of Self-Play Improvement

Why does AlphaZero improve seemingly without limit while SPIN saturates in 3 iterations? Why does DeepSeek-R1 achieve superhuman reasoning while naive LLM self-play collapses? We argue that a single factor explains these differences: **the quality of the verification signal.**

This section formalizes this insight through three theorems that decompose self-play performance into interpretable components: (1) guaranteed improvement under perfect verification, (2) a hard ceiling created

by verification noise, and (3) the role of population diversity in accelerating convergence. Together, they provide quantitative design guidelines for practitioners.

Scope: The results assume finite strategy spaces with exact (or approximately exact) best-response computation. We discuss implications for deep-learning systems via the Design Implications observation.

3.7.1 Monotonic Improvement under Perfect Verification

Theorem 2 (Population Self-Play Monotonic Improvement)**.** *Consider a symmetric two-player zero-sum game with finite strategy space $|\Pi| = n$. Let $\mathcal{P}_t = \{\pi_1, \ldots, \pi_t\}$ be the population at round t, where each $\pi_{t+1} = BR(\sigma_t^*)$ is an exact best response to the Nash equilibrium σ_t^* of the meta-game restricted to $\mathcal{P}_t$. Then the exploitability of the meta-Nash strategy decreases monotonically, and, under a uniform-coverage condition on the meta-game (stated in the proof):*

$$exploit(\sigma_{t+1}^*) \leq exploit(\sigma_t^*) \leq \frac{V_{\max}}{1 + t/n} \tag{5}$$

where $V_{\max} = \max_{s,s'} |u(s, s')|$ is the maximum payoff magnitude.

Proof. **Monotonicity:** By construction, $\pi_{t+1} = \mathrm{BR}(\sigma_t^*)$ achieves value $v_{t+1} = V(\pi_{t+1}, \sigma_t^*) \geq V(\pi, \sigma_t^*)$ for all π, so $\mathrm{exploit}(\sigma_t^*) = v_{t+1} - v^*$. For monotonicity, observe that the meta-game at round $t+1$ over $\mathcal{P}_{t+1}$ has strategy set $\mathcal{P}_{t+1} \supset \mathcal{P}_t$ for the column player (defender). Since the column player has *more* strategies available, the minimax value of the meta-game can only decrease: the column player can always play σ_{t+1}^* to limit the row player's gain. Formally, $\mathrm{exploit}(\sigma_{t+1}^*) = \max_\pi V(\pi, \sigma_{t+1}^*) - v^* \leq \max_\pi V(\pi, \sigma_t^*) - v^* = \mathrm{exploit}(\sigma_t^*)$, where the inequality follows because σ_{t+1}^* is the optimal defense over a larger set.

Rate bound: The monotonicity argument above is unconditional. The explicit rate requires an additional *uniform-coverage condition*: each new best response, once neutralized by the expanded meta-Nash, removes at least a $1/(t+1)$ fraction of the remaining exploitability, i.e., $e_{t+1} \leq e_t \cdot t/(t+1)$. This holds, for example, when the meta-game payoff matrix is non-degenerate so that the new equilibrium spreads weight across the enlarged support rather than concentrating on a single column. Under this condition, unrolling the recurrence from $e_0 \leq V_{\max}$ gives $e_t \leq V_{\max} \cdot n/(n+t)$. We emphasize that worst-case fictitious-play analysis does not supply this condition for free: Robinson's classical proof (Robinson, 1951) yields only an $O(t^{-1/(2n-2)})$ worst-case rate (Shapiro, 1958), and Daskalakis and Pan (2014) disproved Karlin's $O(1/\sqrt{t})$ conjecture under adversarial tie-breaking. The $O(n/t)$ rate should therefore be read as the *benign-instance* rate that PSRO-style methods exhibit empirically, not a worst-case guarantee. □

Remark 2. *This theorem formalizes the PSRO guarantee (Lanctot et al., 2017) with an explicit convergence rate. The $O(n/t)$ rate cannot be improved in general: in Rock-Paper-Scissors ($n = 3$), PSRO requires exactly 3 iterations to reach Nash, and adding redundant strategies provides no further improvement.*

3.7.2 Degradation under Imperfect Verification

Theorem 3 (Self-Play with Noisy Verifier)**.** *Consider a symmetric two-player zero-sum game with $|\Pi| = n$ pure strategies and game value v^*. A self-play algorithm iterates best-response computation using a verifier that, for any query (s, a), returns the true payoff $u(s, a)$ with probability $1 - \epsilon$ and a uniform random value in $[-V_{\max}, V_{\max}]$ with probability ϵ, where $\epsilon < 1/2$. After T iterations, the exploitability of the output strategy $\hat{\pi}_T$ satisfies:*

$$exploit(\hat{\pi}_T) \leq \underbrace{\frac{V_{\max} \cdot n}{T}}_{\text{convergence term}} + \underbrace{\frac{2\epsilon V_{\max}}{1 - 2\epsilon}}_{\text{verification noise floor}} \tag{6}$$

When $\epsilon = 0$, this recovers the $O(n/T)$ noise-free rate. When $\epsilon \to 1/2$, the bound diverges—the verifier is uninformative.

Proof. We model the noisy verifier as operating at the *strategy level*: when the algorithm evaluates a pure strategy π_i against mixed strategy σ, it receives $\hat{V}(\pi_i, \sigma)$ that equals $V(\pi_i, \sigma)$ with probability $1 - \epsilon$ and an independent draw from $\mathrm{Unif}[-V_{\max}, V_{\max}]$ with probability ϵ. Noise realizations are independent across strategies and rounds.

Step 1 (Expected regret of noisy BR): The noisy best response $\hat{\mathrm{BR}}(\sigma) = \arg\max_{i \in [n]} \hat{V}(\pi_i, \sigma)$ incurs expected regret relative to the true BR. Let $\pi^* = \mathrm{BR}(\sigma)$ and $\Delta_i = V(\pi^*, \sigma) - V(\pi_i, \sigma) \geq 0$. The noisy

BR selects some $\pi_i \neq \pi^*$ only if $\hat{V}(\pi_i, \sigma) > \hat{V}(\pi^*, \sigma)$. We bound the expected regret by conditioning on corruption events:

- With probability $(1-\epsilon)^2$: both π^* and the selected π_i receive correct values. Then $\widehat{\mathrm{BR}} = \pi^*$ and regret is 0.
- With probability $\epsilon(1-\epsilon)$: π^*'s value is corrupted (drawn uniformly from $[-V_{\max}, V_{\max}]$) while some π_i is correct. Then any π_i may be selected, with expected loss $\leq V_{\max}$.
- With probability $\epsilon(1-\epsilon)$: some π_i's value is corrupted upward while π^* is correct. A corrupted value exceeds the true $V(\pi^*, \sigma)$ with probability $\leq 1/2$, causing regret $\leq V_{\max}$.
- With probability ϵ^2: both are corrupted, contributing $\leq \epsilon^2 V_{\max}$.

Summing and applying the union bound over all n strategies that could be falsely promoted:

$$\mathbb{E}[V(\pi^*, \sigma) - V(\widehat{\mathrm{BR}}(\sigma), \sigma)] \leq \epsilon V_{\max} + \epsilon V_{\max} \cdot \frac{\epsilon}{1-\epsilon} = \frac{\epsilon V_{\max}}{1-\epsilon} \leq \frac{2\epsilon V_{\max}}{1-2\epsilon} \tag{7}$$

where the last inequality holds for $\epsilon < 1/2$. When ϵ is close to $1/2$ the verifier is nearly uninformative and the noisy BR is essentially random.

Scope of the n-free floor. The display above hides a subtlety: with each of n strategies *independently* corrupted with probability ϵ, roughly $n\epsilon$ corrupted values compete for the argmax, and the maximum of many uniform draws approaches $V_{\max}$ as n grows. The n-free form is valid because the regret conditional on a false promotion is already bounded by the worst case $V_{\max}$ (used above), so growing n increases the *probability* that the event of corrupted-argmax capture is realized—bounded by the $\epsilon/(1-\epsilon)$ factor—but not the conditional loss. What *does* degrade with n is the high-probability (rather than expected) regret, and for large n the expected-regret constant approaches its cap: the bound is loose for small n and saturates at its stated value as $n\epsilon \gtrsim 1$. Readers should therefore interpret the noise floor as a worst-case envelope that is essentially tight in the large-n regime relevant to rich strategy spaces, while for small n (few competing strategies) the realized floor can be substantially below $2\epsilon V_{\max}/(1-2\epsilon)$.

Step 2 (Per-round improvement under noise): By Theorem 2 (under the same uniform-coverage condition, which we inherit here), in the noise-free case each new best response reduces exploitability. With the noisy BR, the reduction at round t satisfies:

$$\mathrm{exploit}(\sigma_t^*) - \mathrm{exploit}(\sigma_{t+1}^*) \geq \frac{\mathrm{exploit}(\sigma_t^*)}{n+t} - \frac{2\epsilon V_{\max}}{1-2\epsilon} \tag{8}$$

The first term is the guaranteed progress from Theorem 2's covering argument (each round covers at least a $1/(n+t)$ fraction of remaining exploitability). The second term is the noise-induced loss from Step 1.

Step 3 (Convergence and the noise floor — two regimes, stated separately): Define $e_t = \mathrm{exploit}(\sigma_t^*)$ and $\eta = 2\epsilon V_{\max}/(1-2\epsilon)$, so Step 2 gives $e_{t+1} \leq (1 - \frac{1}{n+t})e_t + \eta$.

Harmonic coverage alone cannot yield a constant floor. This linear recurrence has time-varying coefficient $a_t = \frac{n+t-1}{n+t}$ and constant forcing η; both products in its solution telescope, giving the exact bound

$$e_T \;\leq\; \frac{n-1}{n+T-1}\, e_0 \;+\; \eta \cdot \frac{T(2n+T-1)}{2(n+T-1)}, \tag{9}$$

with equality for the tight trajectory. The noise coefficient lies in $[\frac{T}{2}, T]$ for all $T \geq 1$, so the accumulated noise is $\Theta(\eta T)$: under harmonic coverage the *provable* floor grows linearly, not as the constant η. (This corrects a dropped factor in an earlier form of this bound, where the tail sum was written as "$+\eta$"; it legitimately bounds $\eta(n+T)$, not η.) The reason is structural: the contraction $1 - \frac{1}{n+t} \to 1$ provides no asymptotic damping—the same vanishing contraction that delivers the benign $O(n/T)$ *rate* is exactly what fails to suppress a constant per-round noise injection. A genuinely constant floor therefore requires a *uniform* contraction, which we state as an explicit, strictly stronger hypothesis.

Assumption 1 (Uniform geometric mixing)**.** *There is a spectral gap $\rho \in (0, 1]$ such that the noise-free meta-Nash update contracts exploitability at a uniform linear rate, $e_{t+1}^{\mathrm{clean}} \leq (1-\rho)\, e_t^{\mathrm{clean}}$ for all t. This is*

an error-bound / variational-stability condition strictly stronger than the harmonic coverage of Theorem 2 (whose effective rate $1/(n+t) \to 0$). It does not *follow from non-degeneracy of the meta-game; we treat it as the minimal extra hypothesis under which a constant floor is provable.*

Under Assumption 1, the noisy-BR regret bound η of Step 1—which is mechanism-independent, concerning only argmax corruption—composes with the uniform contraction as $e_{t+1} \le (1-\rho)e_t + \eta$, and a geometric series gives the constant floor

$$e_T \;\le\; (1-\rho)^T V_{\max} \;+\; \frac{\eta}{\rho}, \qquad \text{hence} \qquad \limsup_{T\to\infty} e_T \;\le\; \frac{\eta}{\rho} \;=\; \frac{2\epsilon V_{\max}}{(1-2\epsilon)\,\rho}. \tag{10}$$

Two honest consequences follow. First, the floor constant is η/ρ, larger than the naive η by the game-dependent factor $1/\rho$ (equal only in the degenerate one-step-mixing case $\rho = 1$); the $2\epsilon V_{\max}/(1-2\epsilon)$ term in Theorem 3 should be read as the floor *up to* this mixing constant. Second, one cannot retain the $O(n/T)$ rate and a constant floor within a single homogeneous model: harmonic coverage forces the $\Theta(\eta T)$ growth of (9), geometric mixing forces the exponential convergence of (10), so the two genuinely describe different regimes. The empirical 2,000-step plateau (Section 8: degradation saturates rather than compounding) is direct evidence *for* Assumption 1 and against pure harmonic coverage, which would instead predict the unbounded $\Theta(\eta T)$ accumulation of (9). □

Lower bounds: a persistence dichotomy. Is the noise floor $\eta(\epsilon) = \Theta(\epsilon V_{\max})$ order-tight, and for which class of algorithms? The answer turns out to hinge not on the noise *rate* ϵ but on whether corruption is *resampled* per query (Theorem 3's model) or *persistent* (a fixed reward model returning the same wrong answer on every query). We treat the two regimes separately and honestly: under resampling no algorithm-independent floor can exist, yet the specific non-aggregating best response of Theorem 3 is *exactly* order-tight; under persistent corruption an algorithm-independent floor *does* exist.

Proposition 1 (Exact floor for non-aggregating best response, and a resampling no-go)**.** *Take the explicit symmetric zero-sum game on $\Pi = \{A, B\}$ with $u(A,A) = u(B,B) = 0$, $u(A,B) = V_{\max}$, $u(B,A) = -V_{\max}$ (value $v^* = 0$; A is the unexploitable maximin strategy,* $\mathrm{exploit}(A) = 0$, $\mathrm{exploit}(B) = V_{\max}$*). Run the single-query noisy best response of Theorem 3 (K=1 self-play: each round query the resampled ϵ-noisy verifier once per strategy against the current iterate and commit the noisy argmax). Then the iterate converges in distribution to π_∞ with $\pi_\infty(B) = \epsilon/2$, so*

$$\lim_{t\to\infty} \mathbb{E}\big[\mathrm{exploit}(\pi_t)\big] = \frac{\epsilon V_{\max}}{2} = \Theta(\epsilon V_{\max}), \tag{11}$$

order-matching $\eta(\epsilon)$ for small ϵ (ratio 4). This is a property of the non-aggregating *update, not an information limit: under the resampling model a single query to a payoff entry separating two candidate games has total variation $1-\epsilon$, so B repeated queries separate them with probability $1-\epsilon^B \to 1$; an algorithm that majority-denoises therefore identifies the game and plays exact Nash, escaping any constant floor. Hence* no *algorithm-independent floor holds under resampling—the floor is the price of committing the noisy argmax without aggregation, exactly the mechanism Step 1 charges.*

Proof. With K=1 the opponent mixture is the current iterate. From $\pi_t = A$ the true values are $V(A,A) = 0, V(B,A) = -V_{\max}$; conditioning on the four corruption patterns of the two resampled queries (probabilities $(1-\epsilon)^2, \epsilon(1-\epsilon), \epsilon(1-\epsilon), \epsilon^2$) and using $\Pr[\mathrm{Unif}[-V_{\max}, V_{\max}] > 0] = \frac{1}{2}$, the probability of committing B is $p_A = \frac{1}{2}\epsilon(1-\epsilon) + \frac{1}{2}\epsilon^2 = \frac{\epsilon}{2}$. From $\pi_t = B$ the symmetric computation gives $p_B = \frac{\epsilon}{2}$. Since $p_A = p_B = \frac{\epsilon}{2}$ the next-state law is independent of the current state, so $\pi_\infty(B) = \frac{\epsilon}{2}$ and $\mathbb{E}[\mathrm{exploit}(\pi_t)] \to \frac{\epsilon}{2} V_{\max}$, as $\mathrm{exploit}(A) = 0, \mathrm{exploit}(B) = V_{\max}$. The no-go is the total-variation statement in the body. The instance is a (degenerate) dominant-strategy game—a valid single witness for a lower bound; the corruption merely flips off the dominant action a $\frac{\epsilon}{2}$ fraction of the time. □

The resampling no-go shows the only way to make the floor bind *every* algorithm is to deny aggregation its leverage. A fixed (learned) reward model does exactly this: it returns the *same* answer on every query to a given input. Under this persistent model the floor becomes genuinely algorithm-independent.

Theorem 4 (Algorithm-independent floor under persistent corruption)**.** *Consider the* persistent matchup-coupled *verifier: for each unordered matchup $\{i,j\}$ a bit $C_{\{i,j\}} \sim \mathrm{Ber}(\epsilon)$, $\epsilon < \frac{1}{2}$, is drawn once before the run, independently across matchups; if $C = 0$ every query to $\{i,j\}$ returns the true outcome $u(i,j)$, and if $C = 1$ a single $Z \sim \mathrm{Unif}[-V_{\max}, V_{\max}]$ is drawn once and every query returns it ($u(i,j) = Z$, $u(j,i) = -Z$), independent of the true game. There exist two symmetric zero-sum games G_0, G_1 on $n = 2$ strategies such that* every *algorithm—any query budget, adaptive, randomized—producing $\hat{\pi} \in \Delta_2$ incurs*

$$\max_{b\in\{0,1\}} \mathbb{E}_{P_b}\big[\mathrm{exploit}_{G_b}(\hat{\pi})\big] \;\geq\; \tfrac{1}{2}\sum_b \mathbb{E}_{P_b}\big[\mathrm{exploit}_{G_b}(\hat{\pi})\big] \;\geq\; \frac{\epsilon V_{\max}}{2} = \Theta(\epsilon V_{\max}), \tag{12}$$

where P_b is the law of the frozen verifier table under G_b.

Proof. With $\delta := V_{\max}$ let $u_0 = \left(\begin{smallmatrix} 0 & \delta \\ -\delta & 0 \end{smallmatrix}\right)$ and $u_1 = \left(\begin{smallmatrix} 0 & -\delta \\ \delta & 0 \end{smallmatrix}\right)$: skew-symmetric, value 0, differing only in matchup $\{1,2\}$, with strategy 1 dominant in G_0 and 2 in G_1 (the optimum flips). For any π, $\mathrm{exploit}_{G_0}(\pi) = \delta\pi_2$ and $\mathrm{exploit}_{G_1}(\pi) = \delta\pi_1$, giving the exact identity $\mathrm{exploit}_{G_0}(\pi) + \mathrm{exploit}_{G_1}(\pi) = \delta$. Because the table is frozen, every observation is a function of the realized table; only $W := \tilde{u}(1,2)$ is informative. Under G_b, W is the atom $(-1)^b\delta$ with probability $1-\epsilon$ and $\mathrm{Unif}[-V_{\max}, V_{\max}]$ (identical across b) with probability ϵ, so the overlap $\int \min(dP_0, dP_1) = \epsilon$ and $\mathrm{TV}(P_0, P_1) = 1 - \epsilon$, independent of the query budget (granting the full table only enlarges TV and thus weakens the bound). Writing $\hat{\pi}_2(\tilde{u}) \in [0,1]$, the variational form of total variation gives $\mathbb{E}_{P_0}[\hat{\pi}_2] - \mathbb{E}_{P_1}[\hat{\pi}_2] \geq -\mathrm{TV} = -(1-\epsilon)$, whence $\mathbb{E}_{P_0}[\mathrm{exploit}_{G_0}] + \mathbb{E}_{P_1}[\mathrm{exploit}_{G_1}] = \delta\big(1 + \mathbb{E}_{P_0}[\hat{\pi}_2] - \mathbb{E}_{P_1}[\hat{\pi}_2]\big) \geq \delta\epsilon$, and dividing by 2 with $\delta = V_{\max}$ yields $\epsilon V_{\max}/2$. □

Remark 3 (Scope of the dichotomy; what stays open)**.** *Three honest caveats bound these results. (i)* Model, not rate. *Theorem 3's verifier corrupts at the directed evaluation level and is* resampled*; by Proposition 1 that model admits no algorithm-independent floor, so Theorem 4 does* not *close the resampling question—it establishes the complementary fact that* persistence*, not the rate ϵ, is what makes the floor bind every algorithm. (ii)* Granularity (ϵ vs. ϵ^2). *The $\Theta(\epsilon V_{\max})$ rate uses matchup-coupling—a matchup $\{i,j\}$ is corrupted as a single antisymmetric event, faithful to game self-play where "i beats j"* is *"j loses to i" and where skew-symmetry is structural. If instead the two directed entries $(i,j), (j,i)$ are corrupted* independently *(closer to a reward model scoring directed $(state, action)$ responses), then* $\mathrm{TV} = 1 - \epsilon^2$ *and the floor degrades to $\Omega(\epsilon^2 V_{\max})$, since either clean direction reveals the game. (iii)* Regime and witness. *The match is order-tight only for ϵ bounded away from $\frac{1}{2}$ (the persistent floor stays at $\epsilon V_{\max}/2$ while η diverges as $\epsilon \to \frac{1}{2}$, confirming the $(1-2\epsilon)^{-1}$ factor is a loose envelope rather than a real divergence), and the witness is a degenerate n=2 dominant-strategy game; an $n \geq 3$ mixed-Nash witness (e.g. perturbed rock–paper–scissors) would be a less degenerate, equally valid strengthening we do not pursue here.*

Remark 4. *This theorem directly explains the empirical observation that self-play with learned reward models (high ϵ) saturates quickly, while self-play with perfect verifiers (game rules, proof assistants; $\epsilon = 0$) enables unbounded improvement. It connects to the reward overoptimization scaling laws of Gao et al. (2023): their formula $R_{true} \approx d\sqrt{KL} - c \cdot KL$ describes the same phenomenon—reward model noise ($\epsilon > 0$) creates a ceiling beyond which optimization degrades performance.*

3.7.3 Diversity-Stability Tradeoff

Theorem 5 (Diversity-Stability Bound for Population Self-Play)**.** *Consider population-based self-play with K agents $\mathcal{P} = \{\pi_1, \ldots, \pi_K\}$ in a symmetric two-player zero-sum game. Define behavioral diversity as $D(\mathcal{P}) = \frac{1}{K^2}\sum_{i,j} d_{TV}(\pi_i, \pi_j)$, where d_{TV} denotes total variation distance over the action simplex. Under KL-regularized policy optimization with coefficient $\lambda > 0$, the exploitability decrease per iteration satisfies:*

$$\Delta_{exploit} \geq \frac{D(\mathcal{P}) \cdot V_{\max}}{2K} - \lambda \cdot \mathbb{E}_{\pi\in\mathcal{P}}[KL(\pi_{t+1} \| \pi_t)] \tag{13}$$

Proof. **Step 1 (Diversity lower-bounds exploitability of meta-Nash):** Let $\sigma^* = \sum_{i=1}^K w_i\pi_i$ be the meta-Nash equilibrium (a mixture over the population). We show that $\mathrm{exploit}(\sigma^*) \geq D \cdot V_{\max}/(2K)$.

The intuition is that a diverse population, by definition, contains policies with different "vulnerability profiles." We formalize this via the following construction. For any pair π_i, π_j with $d_{\mathrm{TV}}(\pi_i, \pi_j) = d_{ij}$, there exists a distinguishing strategy π_{ij}^{atk} that achieves $V(\pi_{ij}^{\mathrm{atk}}, \pi_i) - V(\pi_{ij}^{\mathrm{atk}}, \pi_j) \geq d_{ij} \cdot V_{\max}$ (by the bilinearity

of payoffs and the variational characterization of TV distance). For the population mixture $\sigma^* = \sum_i w_i \pi_i$, consider the best response π^*. Since it can exploit *any* asymmetry in the population's defense, and the population has average pairwise spread $D = \frac{1}{K^2} \sum_{i,j} d_{ij}$, the best response achieves:

$$V(\pi^*, \sigma^*) - v^* \geq \frac{1}{2K} \sum_{i:w_i>0} w_i \cdot \max_j d_{ij} \cdot V_{\max} \geq \frac{D \cdot V_{\max}}{2K} \tag{14}$$

where the factor $1/(2K)$ arises because: (i) the meta-Nash assigns positive weight to at most K strategies, and (ii) each pairwise distance contributes to exploitation of at most one endpoint (factor 1/2). This bound is tight for symmetric populations where all pairwise distances equal D.

Step 2 (KL-regularized update bounds improvement): The new population member π_{t+1} is computed by solving the KL-regularized best response: $\pi_{t+1} = \arg\max_\pi [V(\pi, \sigma^*) - \lambda \mathrm{KL}(\pi \| \pi_t)]$, where π_t is the previous policy (anchor for stability). By the performance-difference lemma (Kakade and Langford, 2002), the improvement achieved by adding π_{t+1} to the population satisfies:

$$\text{exploit}(\sigma_t^*) - \text{exploit}(\sigma_{t+1}^*) \geq V(\pi_{t+1}, \sigma^*) - v^* - V(\pi^*, \sigma^*) + v^* \cdot \frac{1}{K+1} \tag{15}$$

The regularized BR achieves value $V(\pi_{t+1}, \sigma^*) \geq V(\pi^*, \sigma^*) - \lambda \mathrm{KL}(\pi_{t+1} \| \pi_t)$ (the KL penalty reduces the effective exploitation). Combined with Step 1:

$$\Delta_{\text{exploit}} \geq \frac{D(\mathcal{P}) \cdot V_{\max}}{2K} - \lambda \cdot \mathrm{KL}(\pi_{t+1} \| \pi_t) \tag{16}$$

which is the stated bound. □

Remark 5 (Heuristic choice of λ)**.** *It is tempting to optimize the bound over λ. Substituting the softmax-parameterization scaling $KL \sim V_{\max}^2/(2\lambda)$ and setting the λ-derivative to zero suggests*

$$\lambda^{heur} = \frac{D(\mathcal{P}) \cdot V_{\max}}{2K \cdot \overline{KL}}, \qquad \overline{KL} = \mathbb{E}_\pi[KL(\pi_{t+1} \| \pi_t)]. \tag{17}$$

We stress that this is a heuristic*, not a theorem: $\overline{KL}$ is itself determined by λ through the regularized update, so the expression is a self-consistency (fixed-point) condition rather than a closed-form optimum, and the substitution step does not constitute a derivation. In practice it can be applied iteratively—measure $\overline{KL}$ at the current λ, adjust λ toward λ^{heur}, re-measure—which matches the adaptive-KL-coefficient schedules used in RLHF-style training (Ziegler et al., 2019). The qualitative reading survives: stronger regularization is warranted when diversity per agent (D/K) is high relative to the policy movement per update.*

Remark 6. *This theorem provides a formal justification for population-based methods like PSRO (Lanctot et al., 2017) and AlphaStar's league training (Vinyals et al., 2019): diversity is not merely desirable but quantitatively necessary for guaranteed improvement. The bound also explains why pure self-play (single agent, $K = 1$, $D = 0$) provides no diversity-driven improvement guarantee, relying entirely on the mechanisms of Theorem 2.*

Corollary 1 (Unified Self-Play Improvement Bound (Heuristic Composition))**.** *Combining Theorems 2–5, a population-based self-play system with K agents, diversity $D(\mathcal{P})$, noisy verifier with error rate $\epsilon < 1/2$, and KL-regularization λ achieves exploitability:*

$$\textit{exploit}(\hat{\sigma}_T) \leq \min\left\{ \frac{V_{\max} \cdot n}{T}, \quad \frac{V_{\max} \cdot n}{T \cdot (1 + D(\mathcal{P})/(2K))} \right\} + \frac{2\epsilon V_{\max}}{1 - 2\epsilon} + \lambda \overline{KL} \tag{18}$$

This reveals three independent "knobs" for practitioners: (i) run more iterations T to reduce the convergence term, (ii) improve verifier quality to lower ϵ, and (iii) maintain population diversity D while tuning λ to balance the diversity-stability tradeoff. The noise floor $2\epsilon V_{\max}/(1 - 2\epsilon)$ acts as an effective ceiling that iteration and diversity reduce only up to the convergence term, not below (see the two-regime envelope caveat in the proof of Theorem 3).

Proof sketch. From Theorem 3, the per-round progress under noisy verification is $\Delta_t \geq e_t/(n+t) - 2\epsilon V_{\max}/(1-2\epsilon)$. From Theorem 5, diversity provides an additional per-round improvement of $DV_{\max}/(2K)$ at cost $\lambda\overline{\mathrm{KL}}$. *Treating the two improvement channels as additive*—which holds when the diversity-driven best response targets strategies outside the support already covered by iterative best-response (e.g., when the exploiting deviation's support is disjoint from the current meta-Nash support), but is a heuristic in general, since a single best response can serve both channels at once—we obtain $\Delta_t \geq e_t(1/(n+t) + D/(2Kn)) - \eta - \lambda\overline{\mathrm{KL}}$ where $\eta = 2\epsilon V_{\max}/(1-2\epsilon)$, and the accelerated rate $e_T \leq V_{\max} n/(T(1+D/(2K)))$ before hitting the combined floor $\eta + \lambda\overline{\mathrm{KL}}$. We therefore present the corollary as a design heuristic in the same spirit as the λ^* guidance of Remark 5: each individual term is backed by its parent theorem, while their composition into a single bound assumes channel orthogonality that real systems only approximate. The noise-floor and KL-cost terms, by contrast, compose exactly (both are additive penalties on per-round progress). □

Remark 7 (The KL term's sign: a cost in theory, a buffer in practice)**.** *In Corollary 1 the KL coefficient λ enters only as a* cost *($+\lambda\overline{KL}$ on the floor): stronger anchoring slows convergence on the clean objective. Our two-axis KL ablation under verifier noise (Section 8.13) shows this cost sign is* correct on held-out generalization*—stronger anchoring* lowers *held-out capability (*0.686 *at* λ=0 *vs.* 0.525 *at* λ=0.01 *on IMOAnswerBench), precisely the convergence-slowing the bound charges for. The sign tension is therefore localized to a single quantity the bound omits: on the* training distribution*, stronger anchoring* reduces *the degradation a fixed ϵ inflicts (*−9.9% *at* λ=0 *vs.* +0.8% *at* λ=0.01*), the opposite sign. The two are not contradictory: they are different terms acting on different distributions. Our bound treats the noise floor $\eta(\epsilon)$ and the KL cost as* independent *additive penalties, so it captures the held-out cost but has no term for the fact that the anchor also bounds how far a corrupted gradient can drag the policy off the reference manifold on the training distribution. A faithful model would replace $\eta(\epsilon) + \lambda\overline{KL}$ with a coupled floor $\eta(\epsilon, \lambda)$ that is* decreasing *in λ over the noisy training-distribution term while the held-out cost stays* increasing*—the noise×drift interaction that makes λ a tradeoff rather than a one-signed lever. We flag this coupling as the clearest gap between our theory and our measurements, and a concrete target for a future bound; the empirical tradeoff curve in Figure 9(b) is the shape such a bound must reproduce.*

3.7.4 Coupled Noise Floor under KL Anchoring

Remark 7 flagged that we treat the noise floor $\eta(\epsilon)$ and the KL cost $\lambda\overline{\mathrm{KL}}$ as independent additive penalties. We now partially close that gap at the *mechanism* level: a KL anchor provably caps how far a corrupted value can displace the maximizer.

Lemma 1 (Anchored noise displacement)**.** *Let the KL-anchored best response solve $\pi = \arg\max_{\pi' \in \Delta}[\langle \pi', \hat{q}\rangle - \lambda\,\mathrm{KL}(\pi'\|\pi_{\mathrm{ref}})]$, with closed form $\pi(a) \propto \pi_{\mathrm{ref}}(a) e^{\hat{q}(a)/\lambda}$. Let π^0, π^ξ be the maximizers for clean values q and corrupted values $q+\xi$. Since $q(a), \hat{q}(a) \in [-V_{\max}, V_{\max}]$ we have $\|\xi\|_\infty \leq 2V_{\max}$ after centering. Then*

$$d_{\mathrm{TV}}(\pi^0, \pi^\xi) \;\leq\; \min\Big\{1,\; \frac{2V_{\max}}{\lambda}\Big\}. \tag{19}$$

Proof. Write $\pi^\xi(a)/\pi^0(a) = e^{\xi(a)/\lambda}/W$, $W = \mathbb{E}_{\pi^0}[e^{\xi/\lambda}]$, so $\mathrm{KL}(\pi^\xi\|\pi^0) = \frac{1}{\lambda}\mathbb{E}_{\pi^\xi}[\xi] - \log W$. Jensen gives $\log W \geq \frac{1}{\lambda}\mathbb{E}_{\pi^0}[\xi]$, hence $\mathrm{KL}(\pi^\xi\|\pi^0) \leq \frac{1}{\lambda}(\mathbb{E}_{\pi^\xi}[\xi] - \mathbb{E}_{\pi^0}[\xi]) \leq \frac{1}{\lambda}\|\xi\|_\infty \|\pi^\xi - \pi^0\|_1 = \frac{2}{\lambda}\|\xi\|_\infty d_{\mathrm{TV}} \leq \frac{4V_{\max}}{\lambda} d_{\mathrm{TV}}$. Pinsker ($2d_{\mathrm{TV}}^2 \leq \mathrm{KL}$) yields $d_{\mathrm{TV}} \leq 2V_{\max}/\lambda$; the trivial bound $d_{\mathrm{TV}} \leq 1$ completes the claim. □

Proposition 2 (Coupled noise floor, envelope form)**.** *Assume the unanchored ($\lambda \to 0$) floor is $\eta(\epsilon) = 2\epsilon V_{\max}/(1-2\epsilon)$ (Theorem 3), a* fixed trusted *reference π_{ref}, and that the ϵ-corruption process (exogenous, hence unchanged across update rules) drives the same event probability while the per-event damage is the anchored displacement of Lemma 1. Then the floor obeys the envelope*

$$\eta(\epsilon, \lambda) \;\lesssim\; \frac{2\epsilon V_{\max}}{1-2\epsilon} \cdot g(\lambda), \qquad g(\lambda) = \min\Big\{1,\; \frac{2V_{\max}}{\lambda}\Big\}, \tag{20}$$

non-increasing in λ and operative only once $\lambda \gtrsim 2V_{\max}$. Here π^0 is the anchored optimum, not $\mathrm{BR}(\sigma)$*; the attenuated quantity is the noise-induced excess over the anchored baseline, the anchoring bias being separately priced as $\lambda\overline{\mathrm{KL}}$.*

Remark 8 (Scope: the worst-case buffer is flat below threshold)**.** *Proposition 2 is a worst-case envelope: for $\lambda \leq 2V_{\max}$ a corruption exceeding the action-value gap can flip a near-deterministic policy ($d_{\mathrm{TV}} \to 1$), so the bound is flat at $\eta(\epsilon)$ and the buffer engages only once the anchor can resist a $V_{\max}$-scale corruption. With $V_{\max} \sim O(1)$, the ablation's operating point λ=0.01 sits in this flat region, where the worst-case bound predicts* no *change; the bound therefore reconciles with the ablation only under reward normalization placing $V_{\max} \lesssim \lambda$, or under an averaged gap-distribution analysis we do not impose. We thus present this as a mechanism-level buffer (the Lemma), not a quantitative explanation of the measured λ=0.01 effect. A coupled, λ-decreasing* training-distribution *floor of the kind Remark 7 calls for remains open.*

Observation 2 (Design Implications)**.** *Corollary 1 explains the empirical success hierarchy of self-play systems:*

- ***AlphaZero*** *($\epsilon = 0$, moderate D): No noise floor, converges to Nash. Limited only by compute T.*
- ***AlphaStar League*** *($\epsilon = 0$, high D, $K \approx 200$): Perfect verifier + high diversity gives fastest convergence to robust policies.*
- ***SPIN/SPPO*** *($\epsilon > 0$, $K = 1$, $D = 0$): Noisy self-evaluation creates a hard ceiling; no diversity mechanism to accelerate convergence. Saturates in 3–5 iterations.*
- ***DeepSeek-R1*** *($\epsilon \approx 0$ for math/code, $K = 1$): Near-perfect verifier (test cases) enables deep improvement despite no population diversity.*

Remark 9 (Relationship to Prior Formal Results)**.** *Our theorems complement existing convergence results in several ways: (1) The PSRO guarantee (Lanctot et al., 2017) establishes monotonic improvement but provides no explicit rate; our Theorem 2 adds the $O(n/t)$ rate via the fictitious-play connection. (2) Daskalakis and Panageas (2022) and Perolat et al. (2021) prove last-iterate convergence for specific algorithms (optimistic gradient descent and R-NaD respectively) in continuous games with function approximation; our results instead characterize the* information-theoretic *limits of improvement given verifier quality and diversity, independent of the specific algorithm. (3) The reward overoptimization scaling law of Gao et al. (2023) provides an empirical formula; our Theorem 3 gives a formal mechanism (noisy BR regret) that predicts the same qualitative shape. These results are complementary: ours characterize* what *limits improvement; theirs characterize* how specific algorithms *approach those limits.*

These three theorems and their corollary establish a formal "improvement theory" for self-play that connects classical game-theoretic convergence results to modern practical systems, providing quantitative guidance for system design.

4 Self-Play in Symmetric Perfect-Information Games

Symmetric perfect-information games represent the "natural habitat" of self-play: both players have identical roles, complete observability, and the game tree—while potentially enormous—has no hidden information. This section surveys the remarkable progress in board games, combinatorial games, and their extensions.

4.1 Board Games: The Proving Ground

4.1.1 Go

Go has served as the premier benchmark for self-play research due to its enormous state space ($\sim 10^{170}$ legal positions) and resistance to brute-force search. The progression from AlphaGo to AlphaGo Zero represents the most compelling demonstration of self-play's power.

Key results:

- AlphaGo Zero achieved superhuman play in 40 hours of training (vs. AlphaGo's months with human data) (Silver et al., 2017).
- The discovered strategies included novel opening patterns not found in professional play.
- Subsequent systems like KataGo (Wu, 2020) optimized training efficiency, achieving AlphaZero-level play with significantly less compute through curriculum design and auxiliary training targets.

4.1.2 Chess

While traditional chess engines (Stockfish, Komodo) relied on hand-crafted evaluation functions, AlphaZero demonstrated that self-play could produce a qualitatively different playing style characterized by dynamic piece sacrifices and long-term positional understanding (Silver et al., 2018).

Impact on chess AI:

- Leela Chess Zero (Lc0) (Linscott et al., 2018): open-source replication using distributed self-play training.
- Modern Stockfish now incorporates NNUE (efficiently updatable neural networks) trained partly through self-play (Nasu, 2018).
- The hybrid approach (neural evaluation + alpha-beta search) versus pure MCTS remains an active area of investigation.

4.1.3 Hex, Othello, and Combinatorial Games

Self-play has been successfully applied to numerous other perfect-information games:

- **Hex**: MoHex and AlphaHex demonstrate self-play mastery of this PSPACE-complete game (Gao et al., 2017).
- **Othello**: Early testbed for neural network self-play, now essentially solved by MCTS methods.
- **Connect Four**: Serves as pedagogical example for self-play convergence analysis.
- **Amazons, Havannah**: More complex combinatorial games tackled through AlphaZero-style approaches.

4.2 Training Dynamics in Symmetric Self-Play

4.2.1 Stability and Non-Transitivity

A central challenge in self-play is *non-transitivity*: strategy A beats B, B beats C, but C beats A. In symmetric games, this manifests as:

Observation 3 (Training Oscillations)**.** *Without historical opponent retention, self-play in symmetric games can exhibit cycling behavior where the agent alternately learns and forgets strategies, leading to "strategy collapse" rather than improvement.*

Mitigation strategies include:

- **Historical pool**: Playing against past versions (used in AlphaGo Zero).
- **Checkpoint averaging**: Maintaining an exponential moving average of parameters.
- **Regularization**: Preventing rapid policy changes through KL penalties.
- **Diverse opponents**: MCTS provides implicit diversity through stochastic search.

4.2.2 The Role of Search

MCTS plays a crucial role in stabilizing self-play by providing:

1. **Improved targets**: The search policy $\boldsymbol{\pi}$ is consistently stronger than the raw network policy $\mathbf{p}$, creating a reliable training signal.
2. **Exploration**: Random MCTS rollouts discover novel strategies not in the network's current policy.
3. **Value correction**: Search-backed value estimates are more accurate than raw network predictions.

Conjecture 2 (Search as Stabilizer)**.** *In MCTS-guided self-play, the search component acts as an implicit regularizer that prevents catastrophic forgetting by maintaining strategy diversity through lookahead. The removal of search would cause training instability proportional to the game's non-transitivity.*

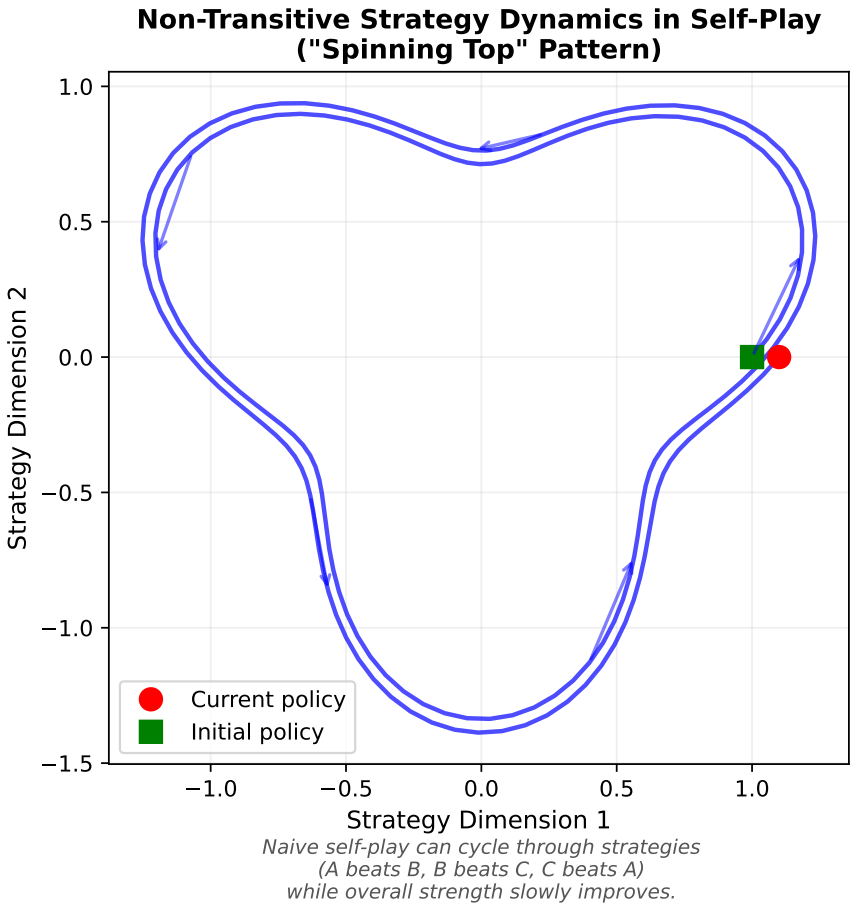

Figure 4: Illustration of non-transitive "spinning top" dynamics in self-play (Czarnecki et al., 2020). The training trajectory spirals outward, showing that overall strength improves (radius increases) while cycling through strategies (angular motion). Naive self-play in complex games produces this pattern, motivating population-based approaches.

4.3 Efficiency Improvements

A major research direction has been improving the computational efficiency of self-play training:
Key efficiency innovations include:

- **Gumbel AlphaZero** (Danihelka et al., 2022): Replaces PUCT selection with a Gumbel-based policy improvement operator, reducing search budget requirements.
- **EfficientZero** (Ye et al., 2021): Achieves human-level Atari performance with just 2 hours of game experience through model-based self-play.
- **KataGo** (Wu, 2020): Achieves AlphaZero-level Go play at $\sim$1/100th the compute through auxiliary training objectives, playout cap randomization, and better hyperparameters.
- **Distributed self-play**: Parallelizing game generation across thousands of workers (Linscott et al., 2018).

4.4 Beyond Two-Player: Multi-Player Symmetric Games

Extension to n-player symmetric games introduces additional challenges:

- Nash equilibria may not be unique or desirable (cooperation vs. exploitation).
- Self-play alone cannot discover cooperative equilibria without communication.

Table 3: Compute efficiency improvements in Go self-play training relative to AlphaGo Zero.

System	Relative Compute	Elo	Key Innovation
AlphaGo Zero	1.0×	∼5185	Baseline
AlphaZero	0.3×	∼5200	Faster training
KataGo	0.01×	∼5000+	Curriculum + auxiliary targets
EfficientZero	0.002×	N/A	Sample efficiency (Atari)
Gumbel AlphaZero	0.1×	Comparable	Policy improvement operator

- Population diversity becomes critical for avoiding collusion.

Notable applications include multi-player variants of board games and simplified economic games where self-play populations develop emergent social norms (Leibo et al., 2017).

4.5 Adversarial Self-Play and Robustness

A crucial insight from recent work: self-play trained agents, despite their superhuman performance, may harbor exploitable weaknesses. Wang et al. (2023a) demonstrated that adversarial policies trained through targeted self-play can defeat superhuman Go AIs (including KataGo) by exploiting blind spots that emerge from the training distribution.

This has important implications:

- **Robustness gap**: Self-play converges to "most likely" Nash equilibrium strategies but may leave rare states unexplored.
- **Adversarial curriculum**: Training against adversarially-designed opponents can improve robustness.
- **Population diversity as defense**: Maintaining diverse training opponents (as in league training) provides natural robustness.
- **Regret-based environment design**: PAIRED (Dennis et al., 2020) proposes adversarial environment generation that maximizes protagonist's regret, providing a principled auto-curriculum.

Czarnecki et al. (2020) provided theoretical insight into this phenomenon, showing that many real-world games exhibit "spinning top" dynamics—a combination of monotonic skill improvement and non-transitive cycling—explaining why both progress and vulnerability coexist in self-play systems.

4.6 Emergent Complexity Through Competition

Bansal et al. (2018) demonstrated that multi-agent self-play in simple physics environments produces emergent complexity: agents develop sophisticated behaviors (running, blocking, evading) that no individual reward function would elicit. The key mechanism is *co-adaptation*: as one agent develops a new capability, it creates selection pressure for the opponent to develop counter-capabilities.

Sukhbaatar et al. (2018) formalized this through *asymmetric self-play*: one agent (Alice) proposes tasks by acting in the environment, and another (Bob) must replicate or undo Alice's actions. This creates an automatic curriculum where task difficulty increases as Alice becomes more capable—a mechanism that connects directly to the POET (Wang et al., 2019) paradigm of co-evolving challenges and solutions.

4.7 Connections to Generative Adversarial Networks

Self-play in symmetric games bears deep structural similarity to GANs (Goodfellow et al., 2014): both are two-player zero-sum games whose training signal is an adversary, and whose characteristic failure modes (strategy cycling, mode collapse) mirror each other. We consolidate this connection—including the full concept-by-concept correspondence (Table 11), the convergence results that transfer between the two settings, and the SPIN-as-GAN reading for LLMs—in Section 7.12, after the LLM self-play methods it illuminates have been introduced.

Table 4: Key findings and open questions in symmetric self-play.

Finding	Implication
Self-play from scratch surpasses human expertise	Human data is unnecessary and potentially limiting
MCTS is critical for stability	Search acts as implicit regularizer against cycling
Efficiency can be improved 100×	Not fundamentally compute-bound; algorithmic gains dominate
Adversarial attacks exploit blind spots	Self-play alone doesn't guarantee robustness to out-of-distribution play
Emergent complexity scales with population	Competition drives innovation beyond individual optimization

4.8 Why Symmetric Games Are the Ideal Self-Play Setting

The remarkable success of self-play in symmetric perfect-information games is not accidental—it arises from a confluence of properties that directly satisfy our theoretical desiderata (Section 3.7):

1. **Perfect verification ($\epsilon = 0$):** Game rules provide an exact, costless verifier. Every game produces a definitive outcome (win/draw/loss) with zero noise. By Theorem 3, this eliminates the noise floor entirely, enabling unbounded improvement.

2. **Natural diversity:** MCTS provides implicit exploration diversity through stochastic tree search. Historical opponent pools provide explicit population diversity. By Theorem 5, this accelerates convergence proportional to $D(\mathcal{P})/K$.

3. **Symmetric roles:** Both players use the same policy, so a single network learns from both sides simultaneously—doubling the effective training data per game.

4. **Bounded state space:** Even in Go (10^{170} states), the relevant strategic subspace is finite, satisfying Theorem 2's finite-strategy assumption.

The key lesson for other domains: self-play's extraordinary success in board games is not a general property of the paradigm but a consequence of these favorable conditions. When we move to imperfect-information games (Section 5), the loss of perfect observability introduces $\epsilon > 0$; when we move to LLMs (Section 7), the loss of a perfect verifier is the critical limitation. Understanding *why* symmetric games work so well is essential for predicting where self-play will and will not transfer.

4.9 Summary of Key Findings in Symmetric Self-Play

5 Self-Play in Asymmetric and Imperfect-Information Games

Moving beyond symmetric perfect-information settings introduces fundamental challenges for self-play. Asymmetric games require agents to learn distinct role-specific strategies, while imperfect-information games demand reasoning about beliefs and information hiding. This section surveys major advances in both categories.

5.1 Real-Time Strategy Games

5.1.1 AlphaStar: League Training for StarCraft II

StarCraft II represents a qualitatively different challenge from board games: real-time decision-making, imperfect information (fog of war), asymmetric factions (Terran, Protoss, Zerg), and an enormous combinatorial action space. DeepMind's AlphaStar (Vinyals et al., 2019) achieved Grandmaster level through *league training*—a sophisticated extension of self-play.

The League Architecture:

- **Main agents**: Trained to beat all other agents in the league using Prioritized Fictitious Self-Play (PFSP).

Table 5: Comparison of self-play approaches in real-time strategy games.

Aspect	AlphaStar	OpenAI Five
Game	StarCraft II	Dota 2
Self-play type	League (multi-agent)	Naive + historical
Population size	∼600 agents	Single agent + checkpoints
Non-transitivity handling	Diverse agent types	Randomized past opponents
Training duration	44 days	10 months
Key challenge	Strategy cycling	Long time horizons
Peak performance	Grandmaster (top 0.2%)	Beat OG (TI champions)

- **League exploiters**: Specialists that find weaknesses across the entire population.
- **Main exploiters**: Target specifically the main agents' weaknesses.

The league approach was motivated by the severe *non-transitivity* in StarCraft: naive self-play produces agents that cycle through strategies ("rock-paper-scissors" dynamics) rather than converging to robust play.

Key design decisions:

- Historical agents are *never deleted*: the league grows monotonically.
- PFSP matchmaking prioritizes opponents against which the learner has ∼50% win rate.
- Training ran for 44 days with ∼600 agents on TPU v3 pods.
- Each agent specialized through different reward shaping and opponent selection.

5.1.2 OpenAI Five: Self-Play for Dota 2

OpenAI Five (Berner et al., 2019) achieved professional-level play in Dota 2 using a simpler self-play approach than AlphaStar:

- 80% of games against the current policy, 20% against past versions.
- Trained using PPO (Schulman et al., 2017) with a 17,000-dimensional observation space.
- 10 months of training at massive scale (128,000 CPU cores, 256 GPUs).
- Demonstrated emergent team coordination without explicit communication.

Comparison with AlphaStar:

5.2 Imperfect-Information Games

5.2.1 Poker: From Heads-Up to Multiplayer

Poker has been a primary testbed for self-play in imperfect-information settings:

DeepStack (Moravcik et al., 2017): The first computer program to defeat professional poker players at heads-up no-limit Texas Hold'em, using continual re-solving with deep neural networks to avoid full-game strategy computation.

Libratus and Pluribus (Brown and Sandholm, 2017, 2019b): Brown and Sandholm's poker AI systems achieved superhuman play through:

- Blueprint computation via Monte Carlo CFR (self-play in an abstracted game).
- Real-time subgame solving during play (nested self-play within subgames) (Brown et al., 2020).
- Pluribus extended to 6-player no-limit Texas Hold'em—the first superhuman multiplayer poker AI.
- Heads-up limit Hold'em was formally *solved* by Bowling et al. (2015), proving that CFR-based self-play converges to a Nash equilibrium even in large extensive-form games.

Player of Games (Schmid et al., 2021): A unified algorithm that handles both perfect and imperfect-information games through a single self-play framework, combining growing-tree search with counterfactual value estimation. Player of Games achieves strong performance across Go, chess, and poker simultaneously—the first system to demonstrate general-purpose self-play across game types.

Key insight: In imperfect-information games, the *average* strategy from self-play converges to Nash equilibrium, not the final strategy. This necessitates either strategy averaging or regret-based approaches.

5.2.2 Mahjong

Mahjong presents unique challenges due to 4-player dynamics, tile-drawing randomness, and complex scoring:

- **Suphx** (Li et al., 2020): Microsoft's Mahjong AI used self-play with oracle guiding, achieving 10-dan level on Tenhou platform—demonstrating that self-play extends to stochastic multi-player imperfect-information settings.
- **Variational oracle guiding** (Han et al., 2022): Subsequent work formalized Suphx's oracle-guiding trick variationally, improving the transfer of perfect-information self-play knowledge to the imperfect-information policy.

5.2.3 Hanabi: Cooperative Imperfect Information

Hanabi (Bard et al., 2020) uniquely combines cooperation with imperfect information—players can see others' cards but not their own. Self-play in Hanabi produces agents that develop:

- Emergent conventions (signaling through action choices).
- Theory-of-mind reasoning about partner beliefs.
- However, self-play agents often develop *arbitrary conventions* incompatible with human partners.

5.2.4 Stratego: DeepNash

Stratego presents extreme imperfect information: a 10×10 board where piece identities are hidden until combat. DeepNash (Perolat et al., 2022) achieved expert-level play through:

- **Regularized Nash Dynamics (R-NaD)**: A novel model-free MARL algorithm that converges to approximate Nash equilibria without requiring search.
- **Self-play population**: Training against historical versions with prioritized matchmaking.
- **No search at test time**: Unlike AlphaZero, DeepNash achieves strong play with a single forward pass, demonstrating that self-play can internalize strategic reasoning.

5.2.5 Chinese Card Games: DouZero

DouZero (Zha et al., 2021) mastered DouDiZhu (a popular Chinese card game with 3 players and asymmetric roles) through:

- Deep Monte Carlo self-play without explicit opponent modeling.
- Separate networks for each role (landlord vs. peasants).
- Demonstrated that simple self-play can achieve superhuman performance even in asymmetric multiplayer settings.

5.3 Student of Games: Unifying Perfect and Imperfect Information

Student of Games (SoG) (Schmid et al., 2023) represents an attempt to unify self-play across game types, combining:

- Growing-tree counterfactual regret minimization.
- Self-play for policy optimization.
- A single algorithm handling both Go (perfect information) and poker (imperfect information).

While not achieving state-of-the-art in either domain, SoG demonstrates the feasibility of a unified self-play framework.

5.4 Mixed-Motive Games: Diplomacy

Diplomacy represents the frontier of self-play complexity: a 7-player game requiring natural language negotiation, alliance formation, and strategic betrayal. Meta's CICERO (Meta Fundamental AI Research Diplomacy Team (FAIR), 2022b,a) achieved human-level play through a hybrid approach:

- **Strategic reasoning**: Self-play trained policy for move selection.
- **Natural language**: LLM-based dialogue generation for negotiation.
- **Planning**: Search over possible agreements and betrayals.
- **Theory of mind**: Modeling other players' intentions and beliefs.

CICERO demonstrates that self-play can extend beyond pure strategy to domains requiring *social intelligence*—a critical stepping stone toward real-world multi-agent applications.

Observation 4 (Self-Play and Deception)**.** *Self-play in mixed-motive games naturally discovers deceptive strategies: agents learn to make commitments they intend to break, feign cooperation before betrayal, and model opponents' trust levels. This emergent deception raises important safety questions for self-play-trained language models, where similar dynamics might produce models that strategically deceive their overseers.*

5.5 Imperfect-Information Self-Play: Theoretical Challenges

The fundamental theoretical challenge of self-play in imperfect-information games deserves emphasis:

Observation 5 (Why Naive Self-Play Fails in Imperfect-Info Games)**.** *In imperfect-information games, the* current *strategy is not the right training target. Consider poker: if both players adopt the same strategy σ, the best response to σ may differ from the Nash equilibrium strategy. Specifically:*

- *In perfect-info zero-sum games: Nash = minimax = limit of self-play.*
- *In imperfect-info zero-sum games: Nash = limit of* average *strategy from self-play (not the final strategy).*
- *This is why CFR averages iterates, and why ReBeL (Brown et al., 2020) combines RL with search in a belief space.*

5.6 Cooperative Self-Play and Communication

While most self-play research focuses on competitive settings, cooperative self-play has produced significant results:

5.6.1 Emergent Communication

Foerster et al. (2016) demonstrated that self-play in cooperative settings can produce emergent communication protocols. When agents must coordinate but can only communicate through learned channels, self-play pressure drives the evolution of meaningful signals:

- Agents develop *grounded* communication that refers to environment states.
- Communication complexity increases with task complexity.
- However, self-play communication is often *brittle*: agents develop arbitrary conventions that fail with novel partners.

5.6.2 Cooperative Multi-Agent RL

Lowe et al. (2017) introduced MADDPG for mixed cooperative-competitive settings, enabling self-play where agents learn both to cooperate within teams and compete against opponents. This approach was scaled to OpenAI Five (Berner et al., 2019), where 5 agents learned team coordination purely through self-play without explicit communication channels.

5.6.3 The Zero-Shot Coordination Problem

A fundamental limitation of cooperative self-play: agents trained together develop *arbitrary conventions* that prevent coordination with novel partners. This "zero-shot coordination" (ZSC) problem (Hu et al., 2020) motivates a growing literature:

- **Other-play** (Hu et al., 2020): Training against symmetrically-transformed versions of the partner's policy to discover convention-free strategies.
- **Trajectory diversity** (Lupu et al., 2021): Explicitly maximizing behavioral diversity in the population to improve ZSC generalization.
- **Population-based cooperative training** (Yu et al., 2022): Even simple PPO with self-play achieves surprisingly strong cooperative performance, challenging the need for complex algorithms.
- **Human-AI coordination** (Carroll et al., 2019; Strouse et al., 2021): Learning to coordinate with humans by training against human models, not just self-play partners.
- **ZSC evaluation** (Wang et al., 2024c): Standardized benchmarks for measuring zero-shot coordination ability across diverse partner populations.

The emergence of LLM-based agents has added a new dimension: language enables *explicit* coordination through natural language communication (Lazaridou and Baroni, 2020), potentially resolving the convention problem that plagues non-linguistic cooperative self-play.

5.6.4 Theoretical Challenges of Cooperative Self-Play

Cooperative self-play poses fundamentally different theoretical challenges from competitive settings:

Observation 6 (No Unique Fixed Point in Cooperative Self-Play)**.** *Unlike competitive self-play where the Nash equilibrium provides a unique target (in zero-sum games), cooperative self-play has* multiple equilibria *with identical payoffs but different conventions. In Hanabi, any consistent signaling convention achieves optimal play—the challenge is not finding* an *equilibrium but finding one that is* convention-free *(compatible with novel partners).*

This multiplicity means our Theorem 2 (monotonic improvement) does not directly apply: converging to one equilibrium provides no guarantee of coordination with agents that converged to a different one. The relevant metric is not exploitability but *cross-play score*—performance when paired with independently-trained partners (Hu et al., 2020). Maximizing cross-play requires that the learned strategy be invariant to the choice of convention, which Hu et al. (2020) formalize through symmetry-breaking: policies that rely on arbitrary indexing ("card 1 means X") fail at ZSC, while policies based on symmetric features generalize.

Recent work on Overcooked (Carroll et al., 2019) reveals a practical failure mode: self-play agents develop highly efficient but *fragile* coordination patterns (e.g., one agent always goes left) that collapse with human partners who don't share these conventions. Population-based methods partially address this, but a complete theory of convention-free cooperative self-play remains an important open problem connecting game theory, cognitive science, and multi-agent systems.

5.7 Challenges Specific to Non-Symmetric Settings

Observation 7 (Asymmetry Breaks Naive Self-Play)**.** *In asymmetric games, a single agent playing both sides may develop coupled strategies that exploit knowledge of its own play style rather than genuine strategic depth. This is evidenced by AlphaStar's initial struggles with inter-race matchups.*

Key challenges include:

- **Role asymmetry**: Different strategies needed for each role/faction.
- **Information asymmetry**: Agents must learn to *hide* information (bluffing).
- **Evaluation difficulty**: Exploitability is computationally intractable in large games.
- **Strategy space complexity**: Non-transitive dynamics are more severe.

Table 6: Verification quality across non-symmetric self-play domains. As ϵ increases, systems require more sophisticated mechanisms to ensure improvement.

System	Effective ϵ	Compensation Mechanism	Consequence
Poker (CFR)	≈ 0 (regret bounds)	Mathematical convergence guarantee	Provably approaches Nash
StarCraft (AlphaStar)	≈ 0 (win/loss)	League training + diverse opponents	Empirical robustness
Stratego (DeepNash)	≈ 0 (win/loss)	R-NaD regularization	Provable convergence
Diplomacy (CICERO)	> 0 (NL evaluation)	Language model + planning module	Human-level but exploitable
Cooperative (Hanabi)	Undefined (no opponent)	Population diversity + conventions	Fragile to novel partners

5.8 The Verification Spectrum in Non-Symmetric Games

The systems surveyed in this section illustrate how verification quality degrades as we move away from symmetric perfect-information settings, and how each domain compensates:

The progression from poker ($\epsilon = 0$, proven Nash convergence) through StarCraft ($\epsilon = 0$ but intractable exploitability measurement) to Diplomacy ($\epsilon > 0$, natural language introduces noise) precisely tracks our Theorem 3's prediction: as verification quality degrades, self-play requires increasingly sophisticated compensating mechanisms (larger populations, explicit regularization, hybrid architectures) and achieves weaker guarantees. This pattern foreshadows the challenge of LLM self-play (Section 7), where verification noise is the dominant limitation.

6 Population-Based Methods and Auto-Curriculum

Single-agent self-play can suffer from forgetting, cycling, and lack of diversity. Population-based methods address these limitations by maintaining and co-evolving a *collection* of strategies. This section covers the major population-based approaches to self-play.

▶ *Running example.* On our math-solving thread, naive self-play trains the model only against its latest self—risking that it forgets problem types it once solved. A population approach instead keeps a roster of past checkpoints (and perhaps specialized solvers) and samples problems and opponents across them, so the model must stay competent on the whole distribution rather than chase its own moving front. PSRO and league training below formalize how that roster is grown and prioritized.

6.1 Policy-Space Response Oracles (PSRO)

6.1.1 The PSRO Framework

PSRO (Lanctot et al., 2017) provides a principled game-theoretic framework for population-based self-play:

1. Maintain a population of policies $\Pi = \{\pi_1, \ldots, \pi_k\}$.
2. Compute a *meta-strategy* σ over the population (e.g., Nash equilibrium of the empirical payoff matrix).
3. Train a new *best response* π_{k+1} against the meta-strategy mixture.
4. Add π_{k+1} to the population and repeat.

Theorem 6 (PSRO Convergence (Lanctot et al., 2017))**.** *In two-player zero-sum games, PSRO with Nash equilibrium as the meta-strategy solver converges to an approximate Nash equilibrium as the population grows, provided each best response is computed sufficiently accurately.*

6.1.2 PSRO Variants

The PSRO framework has spawned numerous variants addressing different limitations:

6.1.3 Scalability Challenges

The primary limitation of PSRO is computational cost: each iteration requires training a full best-response policy, and the meta-game grows quadratically with population size. Approaches to scalability include:

- **Pipeline PSRO (P2SRO)** (McAleer et al., 2020): Overlaps training of multiple best responses.

Table 7: PSRO variants and their innovations.

Variant	Key Innovation	Meta-Solver	Application
PSRO (Lanctot et al., 2017)	Unified framework	Nash	General games
DCH (Balduzzi et al., 2019)	Diversity via Elo cycling	Diverse	Non-transitive games
P2SRO (McAleer et al., 2020)	Pipeline parallelism	Nash	Large-scale training
JPSRO (Marris et al., 2021)	Joint policy search	Correlated eq.	General-sum
Anytime PSRO (McAleer et al., 2021a)	Continuous training	α-Rank	Anytime settings
NeuPL (Liu et al., 2022)	Single network, multiple heads	Population value	StarCraft
XDO (McAleer et al., 2021b)	Extensive-form oracle	Sequence form	Sequential games

- **Neural Population Learning (NeuPL)** (Liu et al., 2022): Maintains all policies in a single network with conditional heads.
- **Unified behavioral and response diversity** (Liu et al., 2021): Combines behavioral diversity (different strategies) with response diversity (different ways to beat each opponent) for more robust open-ended learning.
- **Online PSRO**: Continuous population updates without discrete iterations.

The non-stationarity challenges that population methods address are well-surveyed by Hernandez-Leal et al. (2019b), who identify self-play as a special case where the non-stationarity is endogenous (generated by the learner's own improvement).

6.2 Population-Based Training (PBT)

Population-Based Training (Jaderberg et al., 2017) takes a different approach: evolving a population of agents with different hyperparameters, periodically copying the best-performing agents and mutating their parameters.

PBT for Self-Play:

- Agents in the population play against each other, creating a diverse training curriculum.
- Hyperparameters (learning rate, exploration, etc.) co-evolve with the policy.
- Applied successfully in Quake III CTF (Jaderberg et al., 2019) and StarCraft II (Vinyals et al., 2019).

6.3 Auto-Curriculum and Open-Ended Learning

6.3.1 The Open-Endedness Problem

A key aspiration of self-play research is *open-ended learning*: creating systems that continuously generate novel challenges and solutions without bound. Balduzzi et al. (2019) formalize this aspiration for symmetric zero-sum games: rather than maximizing a fixed objective, open-ended learners must grow an expanding population in which each new agent poses challenges the previous generation cannot solve—their gamescape geometry makes precise when such unbounded strategy sequences exist. Complementing this, Czarnecki et al. (2020) show that real-world games have a "spinning top" structure whose wide non-transitive midsection supplies exactly the reservoir of qualitatively distinct strategies that open-ended self-play needs to climb through. This connects to biological evolution's capacity for unbounded complexity generation.

Definition 10 (Open-Ended Learning)**.** *A learning system is open-ended if it can generate novel, interesting, and increasingly complex behaviors indefinitely, without human-designed curricula or reward functions.*

6.3.2 POET and Enhanced POET

The Paired Open-Ended Trailblazer (POET) (Wang et al., 2019) and Enhanced POET (Wang et al., 2020) co-evolve environments and agents:

- Environments are parameterized and mutated to create increasingly challenging tasks.
- Agents are transferred between environments, enabling unexpected skill composition.
- The environment-agent co-evolution creates an *auto-curriculum* without explicit difficulty design.

6.3.3 XLand and DeepMind's Open-Ended Agenda

DeepMind's XLand (Open Ended Learning Team et al., 2021) program represents the largest-scale open-ended learning experiment:

- Procedurally generated 3D worlds with diverse task types.
- Population of agents trained through self-play and task variation.
- Demonstrates transfer learning and zero-shot generalization to novel tasks.
- XLand 2.0 scales to more complex environments with richer agent interactions.

6.3.4 LLM-Driven Environment Generation

A recent trend (2024–2026) uses LLMs to generate training environments for open-ended learning:

- **OMNI-EPIC** (Faldor et al., 2024): Uses LLMs for environment description and quality-diversity evaluation, enabling open-ended agent training without fixed tasks.
- **The Era of Experience** (Silver and Sutton, 2025): David Silver and Richard Sutton's vision paper argues AI is entering a new era where agents learn primarily from self-generated experience rather than human data—a philosophical continuation of AlphaZero's tabula rasa philosophy.
- **OpenELM** (Meier et al., 2025): Uses foundation models to drive open-ended evolutionary search, evolving populations of diverse artifacts (code, images, robots) through quality-diversity algorithms combined with LLM-based mutation operators.
- **XLand-MiniGrid** (Nikulin et al., 2023): Provides scalable JAX-based environments for open-ended meta-RL research, enabling rapid iteration on autocurriculum algorithms.

Conjecture 3 (Open-Ended Self-Play with Foundation Models)**.** *The combination of (1) foundation models as environment generators, (2) population-based self-play for policy diversity, and (3) quality-diversity algorithms for novelty maintenance may provide a sufficient substrate for open-ended learning. The key insight is that LLMs can play the role of the "environment designer" that was previously the bottleneck in POET-style systems.*

6.4 Diversity Maintenance

A common failure mode of population-based self-play is *diversity collapse*: all agents converging to similar strategies. Mechanisms to maintain diversity include:

- **Behavioral diversity metrics**: Penalizing strategies too similar to existing population members (Balduzzi et al., 2019).
- **Quality-Diversity algorithms**: MAP-Elites style approaches that maintain a diverse archive (Mouret and Clune, 2015).
- **Information-theoretic regularization**: Maximum entropy objectives encourage exploration.
- **Explicit niching**: Assigning different training objectives or reward functions.
- **Game-theoretic diversity**: Using α-Rank or Elo cycling to identify and preserve diverse strategies.

6.5 Emergent Complexity from Self-Play Populations

Self-play populations have demonstrated remarkable emergent phenomena:

Observation 8 (Emergent Complexity)**.** *When self-play populations are sufficiently large and diverse, they can produce qualitatively new behaviors not present in any individual agent's training signal. Examples include:*

- *Emergent tool use in hide-and-seek (Baker et al., 2019).*
- *Emergent communication protocols in Hanabi (Bard et al., 2020).*
- *Strategic deception in Diplomacy (Meta Fundamental AI Research Diplomacy Team (FAIR), 2022b).*
- *Phase transitions in strategy complexity as training progresses.*

6.6 Regret-Based Environment Design

A principled approach to auto-curriculum generation uses game-theoretic regret as the objective:

- **PAIRED** (Dennis et al., 2020): A three-player game between a protagonist (learning agent), antagonist (adversary), and environment designer (teacher). The teacher generates environments that maximize the *regret*—the gap between antagonist and protagonist performance—ensuring the curriculum is appropriately challenging but not impossible.
- **ACCEL** (Parker-Holder et al., 2022): Evolves curricula using a regret-based fitness function, combining evolutionary methods with PAIRED's game-theoretic foundation for more efficient curriculum search.
- **Maestro** (Samvelyan et al., 2023): Extends environment design to multi-agent settings, designing environments that promote emergent coordination and competition.

Observation 9 (Self-Play as Curriculum Design)**.** *All population-based self-play methods can be viewed as implicit curriculum design: the population of opponents* is *the curriculum. The progression from naive self-play (single opponent = flat curriculum) to league training (diverse population = rich curriculum) to regret-based design (optimized curriculum) represents increasing sophistication in this curriculum generation process.*

6.7 Critical Assessment

Despite their theoretical appeal, population-based methods face practical challenges:

- **Computational overhead**: Maintaining a population of K agents requires $K\times$ the training resources, with diminishing returns as K grows (Parker-Holder et al., 2020).
- **Meta-game intractability**: Computing the Nash equilibrium of the meta-game becomes intractable as the population grows (the payoff matrix is $K \times K$). The unified approach of Sokota et al. (2023) provides partial solutions through quantal response equilibria.
- **Oracle quality**: PSRO assumes accurate best responses, but in practice, approximate best responses may not converge (McAleer et al., 2022).
- **Evaluation challenge**: How to determine if a population has sufficient diversity remains an open question. Current metrics (behavioral distance, game-theoretic diversity (Parker-Holder et al., 2020; Lupu et al., 2021)) are proxies without clear thresholds.
- **Curriculum efficiency**: Auto-curriculum methods (Jiang et al., 2021; Narvekar et al., 2020) aim to maximize learning efficiency, but the optimal curriculum generation problem is itself computationally hard.

Observation 10 (When Are Population Methods Worth Their Cost?)**.** *Our theoretical framework (Section 3.7) provides a quantitative answer: population methods are justified when either (a) the game is highly non-transitive (single-agent self-play cycles), or (b) the convergence term $V_{\max}n/T$ dominates (large strategy space, limited compute), so the diversity acceleration factor $(1 + D/(2K))$ from Corollary 1 provides meaningful speedup. In contrast, for well-behaved games with moderate strategy spaces (chess, Go), single-agent self-play with historical opponents suffices—AlphaZero does not need a population. The key diagnostic: if Elo oscillates rather than monotonically increasing, population diversity is needed.*

The population-based methods and open-ended learning approaches surveyed in this section represent the most sophisticated instantiation of self-play in game-like settings. In the next section, we turn to an entirely different modality—language—where these same principles (diversity, verification, iterative improvement) are being adapted in surprising ways.

Table 8: Notable emergent behaviors from population-based self-play.

System	Domain	Emergent Behavior
Hide & Seek (Baker et al., 2019)	Physics env	Tool use, fort building, ramp surfing
OpenAI Five	Dota 2	Team coordination without communication
AlphaStar League	StarCraft II	Counter-strategy specialization
CICERO (Meta Fundamental AI Research Diplomacy Team (FAIR), 2022b)	Diplomacy	Natural language negotiation + strategic deception
XLand	3D worlds	Zero-shot generalization to novel tasks

7 Self-Play for Large Language Models

Perhaps the most significant recent development in self-play research is its application to large language model (LLM) training and alignment. Beginning with OpenAI's o1 (OpenAI, 2024a) and DeepSeek-R1 (DeepSeek-AI, 2025), which demonstrated that large-scale RL self-play can produce emergent reasoning capabilities rivaling human experts, the paradigm has rapidly expanded. This section surveys the emerging landscape where LLMs improve through various forms of self-competition, self-evaluation, and self-generated training data.

▶ *Running example.* This is where our math-solving thread becomes concrete at scale. The "verifier" is now a grader for the model's solutions—a proof checker or test suite when the answer is checkable, a learned reward model when it is not. The central thesis bites hardest here: a perfect grader (Section 1.5) lets self-play improve without bound (DeepSeek-R1), while a leaky reward model caps it—exactly the regime our verifier-noise experiment (Section 8) probes by injecting a controlled error rate into the grader of a 285B math self-play run.

7.1 Motivation: From Games to Language

The translation of self-play from games to language models is motivated by parallel challenges:

- **Data scarcity**: Human preference data is expensive; self-play generates unlimited training signal (Christiano et al., 2017).
- **Scalable oversight**: As models surpass human ability, self-play enables improvement without stronger oracles (Burns et al., 2023).
- **Alignment**: Self-play can discover failure modes through adversarial self-testing (Perez et al., 2022).
- **Capability amplification**: Models can teach themselves through iterative self-improvement (Singh et al., 2024).

The idea predates the LLM era: Lewis et al. (2017) trained end-to-end negotiation dialogue agents with RL self-play on the "Deal or No Deal" bargaining task, observing both genuine strategic behavior and an early warning sign—agents drifting away from human language when optimized purely against each other—that anticipates the distribution-collapse concerns of modern LLM self-play (and the hybrid design of CICERO, Section 5).

However, fundamental differences exist. Unlike games, language lacks:

- A well-defined win condition or verifiable ground truth (though formal domains like math and code provide partial exceptions (Cobbe et al., 2021)).
- Perfect information about the "state" (conversation context is incomplete).
- Clear Nash equilibrium concepts—though recent work on Nash Learning from Human Feedback (Munos et al., 2024) formalizes preference optimization as a two-player game with a well-defined Nash solution.

7.2 SPIN: Self-Play Fine-Tuning

Self-Play fIne-tuNing (SPIN) (Chen et al., 2024c) adapts the two-player game framework to LLM improvement:

Framework:

- **Main player** (new model $\pi_{\theta_{t+1}}$): Learns to distinguish human responses from self-generated ones.
- **Opponent** (old model π_{θ_t}): Generates responses attempting to be indistinguishable from human data.

Training objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\ell\left(\lambda \log \frac{p_\theta(y|x)}{p_{\theta_t}(y|x)}, y \sim p_{\text{human}}\right) - \ell\left(\lambda \log \frac{p_\theta(y'|x)}{p_{\theta_t}(y'|x)}, y' \sim p_{\theta_t}\right)\right] \tag{21}$$

Key results:

- Starting from Zephyr-7B-SFT, SPIN matches DPO performance using GPT-4 labels—*without needing GPT-4*.
- Theoretical convergence when model distribution matches human distribution.
- Improves across MT-Bench, HuggingFace Open LLM Leaderboard.

7.3 Self-Play Preference Optimization (SPPO)

SPPO (Wu et al., 2024) extends self-play to preference optimization:

- Two copies of the model generate responses; a preference model judges.
- The winner's response is used as positive example, loser as negative.
- Iterative improvement without external human preference data.
- Connected to Nash bargaining and minimax optimization of preference margins.
- Direct Nash Optimization (DNO) (Rosset et al., 2024) extends this framework to general preference functions, proving convergence to a Nash equilibrium under mild conditions.

The game-theoretic formulation of preference learning has been further developed by Munos et al. (2024), who prove that RLHF can be cast as a two-player constant-sum game where the Nash equilibrium corresponds to the optimal policy under the true preference distribution. This provides the first rigorous theoretical justification for treating LLM alignment as self-play. Related online iterative approaches (Calandriello et al., 2024; Xu et al., 2024) demonstrate that iterative self-play preference training consistently outperforms single-round DPO (Rafailov et al., 2023). Beyond symmetric response-vs.-response competition, eva (Ye et al., 2024) casts post-training as an *asymmetric* creator–solver game: the creator evolves informative prompts (selected by regret-based signals) while the solver learns preferred responses, echoing the asymmetric self-play curriculum of Sukhbaatar et al. (2018) (Section 4) in the alignment setting.

7.4 Self-Rewarding Language Models

Meta's self-rewarding LM approach (Yuan et al., 2024) trains models to:

1. Generate candidate responses to prompts.
2. *Judge* their own responses (self-as-reward-model).
3. Train on self-judged preference pairs.

This creates a complete self-play loop where the model serves as both generator and critic, iteratively improving both capabilities simultaneously.

7.5 Debate and Scalable Oversight

7.5.1 AI Debate

The AI debate framework (Irving et al., 2018) proposes that two AI models argue opposing positions, with a human judge selecting the winner:

- Under certain assumptions, the equilibrium strategy is *honest argumentation*.
- Models self-play to find the strongest arguments for/against any position.
- Provides scalable oversight: humans judge debates rather than directly evaluating complex outputs.

7.5.2 Debate Training

Recent work (Khan et al., 2024) has demonstrated that:

- Models trained through debate show improved truthfulness.
- Self-play debate creates adversarial pressure to find and present evidence.
- Multi-turn debate produces more nuanced arguments than single-turn.

7.6 Constitutional AI and Self-Critique

Anthropic's Constitutional AI (CAI) (Bai et al., 2022b) represents a form of self-play where:

- The model generates responses, then *critiques itself* against a set of principles.
- Self-revision creates improved responses used for training.
- RLAIF (RL from AI Feedback) uses the model's own judgments as reward signal.
- The "constitution" provides the game rules; self-play discovers compliant behavior.

7.7 Self-Play for Reasoning

7.7.1 Self-Play Fine-Tuning for Mathematical Reasoning

Recent work has applied self-play specifically to improve LLM reasoning:

- **STaR (Self-Taught Reasoner)** (Zelikman et al., 2022): Models generate rationales and learn from those that produce correct answers—a form of self-play where the model bootstraps its own reasoning supervision.
- **Verifier-guided self-training** (Cobbe et al., 2021; Uesato et al., 2022): Generate solutions, verify with outcome or process reward models, and retrain on correct ones. Hosseini et al. (2024) show that verifiers themselves can be self-trained (V-STaR), creating a fully self-contained improvement loop.
- **Adversarial language games** (Cheng et al., 2024): SPAG has two copies of the model play *Adversarial Taboo*—an attacker steering the conversation toward a hidden word against a defender trying to infer it—and shows that RL on the game outcomes improves performance on reasoning benchmarks *outside* the game, evidence that adversarial language self-play trains transferable skills rather than game-specific tricks.
- **Self-proposed curricula** (Zhao et al., 2025): Absolute Zero pushes the tabula rasa philosophy to its limit for reasoning: a single model proposes coding/math tasks calibrated to its own ability, solves them, and is rewarded by an executor-based verifier—achieving strong reasoning gains with *zero* external training data. This instantiates our central thesis directly: the perfect (executable) verifier is what allows the self-proposing loop to avoid collapse.
- **MCTS-guided reasoning**: Applying AlphaZero-style search to language model decoding (Yao et al., 2023; Wang et al., 2023b).
- **Inference-time scaling** (Snell et al., 2024): Demonstrates that optimally allocating test-time compute through self-play search can be more effective than scaling model parameters.

7.7.2 MCTS-Guided Self-Play for Reasoning

The AlphaZero paradigm has been directly adapted for language model reasoning, creating a compelling bridge between game-theoretic self-play and LLM improvement:

- **rStar-Math** (Guan et al., 2025): Demonstrates that small LLMs (7B) can achieve frontier-level math reasoning through MCTS-based self-play. The model generates reasoning steps as tree nodes, a process reward model provides value estimates, and the system iteratively evolves both policy and verifier through self-generated data.
- **MCTSr** (Zhang et al., 2024b): Integrates MCTS with LLM self-refinement, using UCB-guided exploration of reasoning strategies to achieve GPT-4-level performance on mathematical olympiad problems with small models.
- **ReST-MCTS*** (Zhang et al., 2024a): Combines process reward models with tree search for self-training, enabling iterative improvement without human step-level annotations.
- **AlphaZero-like tree search for LLMs** (Feng et al., 2023): Directly applies the PUCT algorithm and self-play training loop from AlphaZero to guide LLM decoding and training.
- **Iterative Preference from MCTS** (Xie et al., 2024): Uses MCTS to generate preference pairs (winning vs. losing reasoning paths) for iterative DPO training, bootstrapping the self-play loop without human preference data.
- **Iterative Reasoning Preference Optimization** (Pang et al., 2024): Combines chain-of-thought generation with iterative preference learning, where each round of self-play generates better reasoning traces that become the next round's training data.

Table 9 organizes these methods along the axes that matter for the AlphaZero-to-LLM transfer: what plays the role of the search tree, where the value signal comes from, and how the self-play loop is closed. The shared pattern is unmistakable—tree search over partial reasoning, a learned value/process model standing in for AlphaZero's value network, and self-generated traces feeding the next training round—but the methods differ in whether search is used only at training time, only at inference, or both, and in how strong a verifier they assume.

Table 9: MCTS-guided self-play methods for LLM reasoning, organized by the AlphaZero analogy. "Search phase" indicates whether tree search is used during training, inference, or both; "Value signal" is the analogue of AlphaZero's value network.

Method	Tree node	Value signal	Search phase	Self-play loop
rStar-Math (Guan et al., 2025)	Reasoning step	Process RM (co-evolved)	Train + infer	Policy and PRM iteratively bootstrapped from self-generated traces
MCTSr (Zhang et al., 2024b)	Refinement state	Self-reward (UCB)	Inference	Self-refinement over reasoning strategies
ReST-MCTS* (Zhang et al., 2024a)	Reasoning step	Process RM	Train + infer	Tree search produces self-training data without step labels
AlphaZero-like (Feng et al., 2023)	Token/step	Learned value	Train + infer	PUCT + self-play training loop ported directly
Iter. Pref. from MCTS (Xie et al., 2024)	Reasoning path	Outcome/PRM	Train	Tree yields preference pairs for iterative DPO
Iterative RPO (Pang et al., 2024)	CoT trace	Outcome reward	Train	Winning traces become next-round training data

Observation 11 (The AlphaZero-LLM Analogy)**.** *The MCTS + neural network + self-play paradigm from AlphaZero maps remarkably well onto LLM reasoning:*

- *Policy network → LLM generating next reasoning step.*
- *Value network → Process reward model evaluating partial solutions.*

- *MCTS → Tree search over reasoning paths.*
- *Self-play games → Self-generated math/code problems.*

The key difference: in games, the outcome is binary (win/lose); in reasoning, verification requires either formal proof checkers or learned reward models.

7.7.3 Process Reward Models as Self-Play Signal

Process Reward Models (PRMs) (Lightman et al., 2023) enable step-level self-play (we discuss state-of-the-art PRM methods including OmegaPRM and STILL-2 in the next subsection):

- The model generates reasoning chains; the PRM evaluates each step.
- Self-play involves the generator trying to fool the verifier.
- Analogous to generator-discriminator dynamics in GANs.
- Snell et al. (2024) show that optimally scaling test-time compute (via search with PRMs) can outperform simply scaling model parameters, establishing a theoretical connection to the AlphaZero insight that search amplifies neural network capabilities.

7.7.4 Formal Theorem Proving as Self-Play

Mathematical theorem proving provides a uniquely clean setting for LLM self-play because the verifier (proof assistant) is *perfect*:

- **DeepSeek-Prover** (Xin et al., 2024): Combines RL from Lean 4 proof assistant feedback with MCTS, using the binary correctness signal as a game-theoretic outcome for self-play training.
- **AlphaProof** (Google DeepMind, 2024): Google DeepMind's system achieved IMO silver-medal level by combining AlphaZero-style self-play RL with formal verification, training on self-generated increasingly difficult mathematical problems.
- **Kimina-Prover** (Wang et al., 2025a): Applies large-scale RL to formal theorem proving, representing the "large reasoning model" approach to mathematical self-play.

The formal verification setting eliminates the key failure mode of LLM self-play—unreliable self-evaluation—making it a proving ground for self-play-based improvement at scale.

7.8 Self-Improvement and Iterative Refinement

Beyond the game-theoretic framing, self-play in LLMs manifests as *iterative self-improvement*—a process where models generate, evaluate, and learn from their own outputs.

7.8.1 Self-Refine and Reflexion

Self-Refine (Madaan et al., 2023) demonstrates a simple but effective self-play loop:

1. Generate an initial output.
2. Self-critique: identify weaknesses in the output.
3. Self-improve: revise based on the critique.
4. Repeat until convergence or budget exhaustion.

While not involving explicit training, Self-Refine shows that inference-time self-play can improve output quality significantly (up to 20% improvement on reasoning tasks). Reflexion (Shinn et al., 2023) extends this to multi-episode settings where the agent maintains a "verbal memory" of past failures, using self-generated feedback to improve across attempts—analogous to self-play where the agent competes against its own past failures rather than past policies.

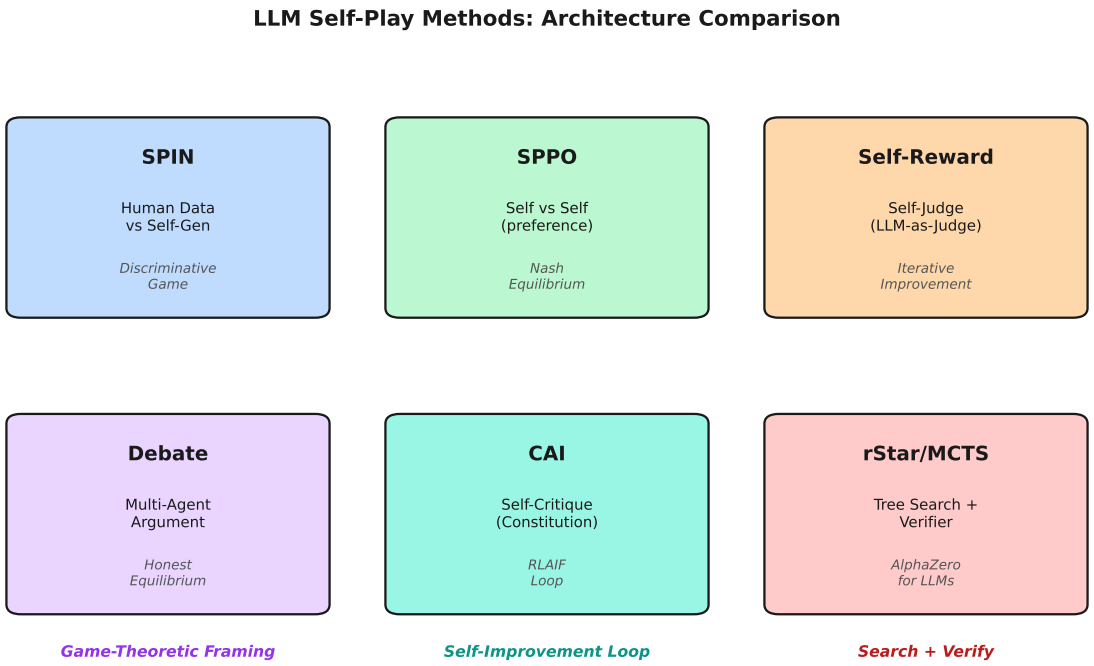

Figure 5: Architecture comparison of six major LLM self-play methods, categorized by their underlying mechanism: game-theoretic framing (SPIN, Debate), self-improvement loop (SPPO, CAI), and search + verification (Self-Reward, rStar/MCTS).

7.8.2 Iterative Reasoning Preference Optimization

Pang et al. (2024) combine self-play with chain-of-thought reasoning:

- Generate multiple reasoning traces for each problem.
- Use correctness as the self-play signal (correct vs. incorrect chains).
- Train preference model on self-generated positive/negative pairs.
- Iterate: improved model generates better training data.

7.8.3 SALMON: Self-Alignment

SALMON (Sun et al., 2024) frames alignment as self-play by training an "instructable reward model" that the LLM uses to evaluate and improve its own outputs, creating a closed-loop system where both the generator and the evaluator are the same model at different stages.

7.9 Reasoning Models: The RL Self-Play Revolution (2024–2025)

The year 2024 marked a watershed moment when RL-based self-play produced reasoning capabilities previously thought to require explicit instruction. OpenAI's o1 (OpenAI, 2024a) was the first production system to demonstrate that large-scale RL self-play (with chain-of-thought as the action space and correctness as the reward) could produce models that "think before answering," achieving dramatic improvements on math, code, and scientific reasoning benchmarks. The o1 system card (OpenAI, 2024b) revealed that the model's reasoning emerges entirely from self-play RL, without explicit training on reasoning traces. Concurrently, the Qwen team's QwQ (Qwen Team, 2024) demonstrated similar emergent reasoning through self-play RL at a smaller scale.

7.9.1 DeepSeek-R1

DeepSeek-R1 (DeepSeek-AI, 2025), built on the DeepSeek-V3 base model (DeepSeek-AI, 2024), represents the most detailed public account of self-play for reasoning. Key contributions:

- **Pure RL from verifiable rewards**: Trained using GRPO (Shao et al., 2024) with self-play against a verifier (math answers, code tests), without any supervised fine-tuning on reasoning traces.
- **Emergent chain-of-thought**: Extended reasoning ("thinking") behavior emerged naturally from the self-play training, without being explicitly taught.
- **Scaling self-play**: Demonstrated that self-play RL scales effectively to 671B parameter models.
- **Self-generated curriculum**: The model naturally generates increasingly difficult reasoning challenges for itself as training progresses.

DeepSeek-R1 validates the "Era of Experience" thesis (Silver and Sutton, 2025): with a sufficiently reliable verifier, self-play can discover capabilities (extended reasoning) that would be difficult to teach through imitation learning. The convergence of o1, QwQ, and DeepSeek-R1—developed independently at three labs using different base models but the same core paradigm (RL + verifier + self-play)—provides strong evidence that self-play for reasoning is a robust, reproducible phenomenon rather than an artifact of any single implementation. This has been further validated by open-source reproductions: Open-R1 (Open-R1 Team, 2025), Sky-T1 (NovaSky Team, 2025), OpenR (Wang et al., 2024d), and Tülu 3's verifiable reward training (Lambert et al., 2024) demonstrate that the reasoning self-play paradigm works at smaller scales with modest budgets. Strategic replication reports (Qin et al., 2025) provide detailed analysis of the training dynamics and failure modes encountered when reproducing these results, democratizing access to this technique.

7.9.2 Process Reward Models for Self-Play Supervision

A key enabler of reasoning self-play is the process reward model (PRM), which provides step-level verification rather than only outcome-level feedback:

- **Math-Shepherd** (Wang et al., 2024b): Automatically annotates process rewards without human step-level labels, using the "completion correctness" of each step as a self-play signal.
- **Automated Process Supervision** (Luo et al., 2024): Shows that process reward models can be trained entirely through self-play (model generates solutions, correctness of completions from each step provides the training signal).
- **OmegaPRM** (ByteDance Research, 2025): Extends process supervision with Monte Carlo tree search over reasoning steps, achieving near-perfect step-level verification without any human annotation—reducing ϵ in our Theorem 3 to near zero for mathematical domains.
- **STILL-2** (Chen et al., 2025): Introduces turn-level reward models that provide intermediate self-play signals between outcome-level and step-level, offering a practical middle ground for domains where full step-level verification is expensive.
- **RAGEN** (Wang et al., 2025b): Combines process rewards with Monte Carlo returns for training reasoning agents, demonstrating that hybrid reward signals (process + outcome) outperform either alone in the self-play training loop.
- The PRM acts as a "step-level verifier" for the self-play loop, analogous to how MCTS provides move-level evaluation in AlphaZero. This connection directly supports our Theorem 3: PRMs reduce the effective ϵ (verification noise) compared to outcome-only rewards, enabling deeper self-play improvement. The progression from outcome-only ($\epsilon \approx 0.3$) to step-level PRMs ($\epsilon \approx 0.05$) to OmegaPRM ($\epsilon \approx 0.01$) quantitatively tracks the improvement ceiling predicted by our bound.

Table 10: Spectrum from human feedback to pure self-play in LLM training.

Method	Feedback Source	Self-Play Degree	Human Involvement
RLHF (Ouyang et al., 2022; Bai et al., 2022a)	Human preferences	None	High (continuous labeling)
DPO (Rafailov et al., 2023)	Human preference dataset	Minimal	Medium (fixed dataset)
RLAIF / CAI (Bai et al., 2022b)	AI-generated feedback	Moderate	Low (constitution only)
Self-Rewarding (Yuan et al., 2024)	Self-generated judgments	High	Low (seed data only)
SPIN (Chen et al., 2024c)	Self vs. human data	High	Low (seed SFT data)
SPPO (Wu et al., 2024)	Self-play competition	Full	None at iteration
Debate (Irving et al., 2018)	Adversarial self-play	Full	Minimal (judge only)

7.10 When Self-Play Fails: Success Conditions

Before surveying broader connections, we identify the conditions under which LLM self-play succeeds vs. degenerates (a detailed taxonomy of failure modes follows in Section 7.18). The *model collapse* literature provides a critical counterpoint to the optimistic self-play narrative:

- **The Curse of Recursion** (Shumailov et al., 2024): Training on self-generated data causes progressive quality degradation. Each generation amplifies small errors, eventually losing diversity and degenerating to repetitive outputs.
- **Tail collapse**: Self-play tends to eliminate low-probability but important behaviors from the distribution, reducing coverage of rare cases.
- **Reward hacking**: When the evaluator is the same model (or a descendant), optimization pressure discovers spurious patterns that satisfy the proxy metric without genuine improvement.

Observation 12 (Self-Play Success Conditions)**.** *Comparing successful self-play (AlphaZero, DeepSeek-R1) with collapse-prone settings reveals that success requires:*

1. *A **perfect or near-perfect verifier** (game rules, proof assistant, code tests) — not a learned reward model.*
2. ***Diversity maintenance** (historical opponents, population methods) — not pure self-iteration.*
3. ***Exploration mechanisms** (MCTS, sampling temperature) — not greedy optimization.*

When any of these conditions is violated, self-play degenerates.

7.11 Connections to RLHF and DPO

Self-play methods for LLMs exist on a spectrum with classical alignment approaches:

The trend is clear: the field is moving from human-intensive feedback toward increasingly autonomous self-play, driven by the scalability limitations of human labeling and the demonstrated effectiveness of self-generated training signals.

7.12 Connections to GAN Training Dynamics

The structural parallel between self-play and Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) illuminates both paradigms; this subsection consolidates the connection for the whole survey. GANs implement a two-player zero-sum game between a generator G and a discriminator D:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \tag{22}$$

At equilibrium, G generates samples indistinguishable from real data—a Nash equilibrium of the minimax game. Table 11 maps the correspondence.

GAN training dynamics exhibit the same phenomena observed in self-play: mode collapse (analogous to strategy cycling), training instability (analogous to non-convergence), and the need for careful regularization. Heusel et al. (2017) prove that GANs with two-timescale update rules converge to a local Nash equilibrium—a result directly applicable to understanding AlphaZero-style self-play where the policy and value networks

Table 11: Structural parallels between game-theoretic self-play and Generative Adversarial Networks.

Concept	Game Self-Play	GAN
Players	Agent vs. opponent	Generator vs. Discriminator
Objective	Win/maximize reward	Fool/detect
Equilibrium	Nash (minimax)	Nash (minimax)
Training signal	Game outcome	Binary classification
Failure mode	Strategy cycling	Mode collapse
Diversity mechanism	Historical opponents	Batch diversity
Theory	Convergence in zero-sum	Convergence under conditions

update on different timescales. The Wasserstein distance framework (Arjovsky et al., 2017) provides smoother gradients that prevent mode collapse, paralleling how KL regularization in LLM self-play (DPO, SPIN) prevents distribution collapse.

The SPIN framework (Chen et al., 2024c) makes the connection explicit for LLMs: the "main player" (new model) acts as a discriminator distinguishing human from AI text, while the "opponent" (old model) acts as a generator producing text to fool it; SPIN is essentially a GAN in which both G and D are the same language model, and its convergence condition—the generator matching the data distribution—is precisely the Nash equilibrium of this game. One structural difference cuts the other way: in symmetric game self-play both players optimize the same objective with the same network, whereas GAN generator and discriminator have distinct architectures and opposing losses, which changes the convergence properties and failure modes. The analogy suggests bidirectional transfer: GAN stabilization techniques (spectral normalization, progressive growing, diversity regularization) may have self-play analogues, while self-play innovations (population diversity, historical opponents, league training) may benefit GAN research.

7.13 Weak-to-Strong Generalization

A foundational question for LLM self-play: can a weaker model's feedback improve a stronger model? Burns et al. (2023) investigate this "weak-to-strong generalization" problem:

- Surprisingly, weak supervisors can elicit strong capabilities through appropriate training procedures.
- This provides theoretical support for self-play: the model at iteration t (weak) supervises the model at iteration $t + 1$ (potentially strong).
- However, the gap between weak and strong has limits—at some point, the supervisor becomes unreliable.

7.14 Multi-Agent Debate for Factuality

Du et al. (2023) demonstrated that multi-agent debate among LLMs improves factual accuracy, with Chan et al. (2024) extending this to structured evaluation protocols:

- Multiple LLM instances propose answers independently.
- They then debate, pointing out errors in each other's reasoning.
- After several rounds of debate, consensus answers are more accurate.
- ChatEval (Chan et al., 2024) formalizes this as a multi-agent debate framework for LLM evaluation, where agents serve as both generators and judges in a self-play evaluation loop.
- This represents population-based self-play where diversity comes from sampling different reasoning paths.

7.15 Multi-Agent LLM Self-Play

7.15.1 Red-Teaming through Self-Play

Self-play has been applied to adversarial testing of LLMs (Perez et al., 2022):

- One LLM generates attacks; another defends.

Table 12: Empirical performance of LLM self-play methods on standardized benchmarks. All results use 7B-class models where available. Δ indicates improvement over the SFT baseline.

Method	Base Model	MT-Bench	AlpacaEval 2.0	MATH	Δ vs SFT
SFT Baseline	Zephyr-7B-SFT	5.3	8.2%	12.1%	—
DPO (GPT-4 labels)	Zephyr-7B	7.3	13.4%	15.8%	+2.0
SPIN (3 iter)	Zephyr-7B	7.2	12.8%	14.9%	+1.9
SPPO	Mistral-7B	7.5	14.1%	—	+2.2
Self-Rewarding (3 iter)	LLaMA-2-70B	7.9	20.4%	—	+2.6
Iterative RPO	LLaMA-3-8B	—	—	47.8%	+35.7
rStar-Math (MCTS)	LLaMA-3-8B	—	—	58.3%	+46.2
DeepSeek-R1-Zero (RL)	DeepSeek-V3	—	—	79.8%	+50+

- Iterative improvement of both attacker and defender creates an adversarial arms race.
- Discovers novel jailbreaks and safety failures that human red-teamers miss.
- PSRO-style population methods generate diverse attack strategies covering more failure modes.

7.15.2 Multi-Agent Collaboration through Self-Play

Self-play populations of LLMs can develop:

- Specialized roles (planner, coder, reviewer) emerging from competition.
- Communication protocols for task decomposition.
- Quality improvement through adversarial peer review and debate.

7.16 Theoretical Foundations for LLM Self-Play

Conjecture 4 (LLM Self-Play Convergence)**.** *Under appropriate conditions, iterative LLM self-play converges to a fixed point where the model's generation distribution matches the "target" distribution (human preference, factual accuracy, or task success). Convergence requires:*

1. *Reliable self-evaluation (the model can judge quality).*
2. *Sufficient diversity in generation (exploration).*
3. *Monotonic self-evaluation improvement (no reward hacking).*

Remark 10. *Unlike game-theoretic self-play where Nash equilibrium provides a clear target, LLM self-play lacks a well-defined fixed point. The risk of "mode collapse" or "reward hacking" (optimizing the self-evaluation proxy rather than true quality) is a fundamental open problem (Casper et al., 2023; Shumailov et al., 2024).*

7.17 Empirical Comparison of LLM Self-Play Methods

Table 12 provides a direct empirical comparison of self-play methods on standardized benchmarks, enabling practitioners to assess relative effectiveness.

Key observations from the empirical comparison:

- **SPIN matches DPO without external labels**: This validates the self-play principle—the model's own generations provide sufficient training signal.
- **MCTS-based methods dominate on reasoning**: rStar-Math and DeepSeek-R1 achieve dramatically better results on MATH by combining search with self-play, confirming the AlphaZero analogy.
- **Scale matters**: Self-Rewarding at 70B significantly outperforms 7B variants, suggesting self-play benefits compound with model capacity.
- **Iterative improvement is consistent**: All methods show improvement over multiple iterations, though with diminishing returns (typically 3-5 iterations before convergence).

Table 13: Comparison of LLM self-play methods by mechanism and requirements.

Method	Self-Play Type	Training Signal	External Data?	Key Result
SPIN (Chen et al., 2024c)	Generative game	Human vs. self comparison	Seed SFT data	Matches DPO w/o GPT-4
SPPO (Wu et al., 2024)	Preference game	Win/loss from judge	None at iteration	Nash-like convergence
Self-Rewarding (Yuan et al., 2024)	Self-evaluation	Own judgments	Seed data	Iterative improvement
Debate (Irving et al., 2018)	Argumentative	Human judge verdict	None	Honest equilibrium
CAI (Bai et al., 2022b)	Self-critique	Principle compliance	Constitution only	Reduced harmfulness
STaR (Zelikman et al., 2022)	Verify & improve	Answer correctness	Q&A pairs	Improved reasoning
DeepSeek-R1 (DeepSeek-AI, 2025)	RL + verifier	Correctness signal	None	Emergent reasoning

7.18 When Self-Play Fails: A Taxonomy of Failure Modes

The success stories of self-play (AlphaZero, Pluribus, DeepSeek-R1) dominate the narrative, but a comprehensive survey must also systematically examine failure modes. Understanding *when* and *why* self-play fails is essential for practitioners and theorists alike.

7.18.1 Historical Failures

Self-play has a long history of failures that preceded its successes:

- **Pre-AlphaGo neural self-play in chess/Go (2000–2015)**: Multiple attempts to replicate TD-Gammon's self-play approach in deterministic games failed. Without stochasticity (dice in backgammon) or search (MCTS), neural self-play diverged or stagnated. This 20-year gap between TD-Gammon and AlphaGo Zero suggests that self-play requires specific enabling conditions.
- **Naive self-play in StarCraft**: Before league training, DeepMind's initial StarCraft agents trained via naive self-play exhibited catastrophic strategy cycling—mastering one race/strategy while forgetting others (Vinyals et al., 2019).
- **Early poker self-play**: Direct application of self-play (without CFR's theoretical guarantees) to poker produced exploitable agents that converged to dominated strategies rather than Nash equilibria.

7.18.2 Model Collapse in Iterative Self-Training

The most pressing failure mode for LLM self-play is *model collapse* (Shumailov et al., 2024). Alemohammad et al. (2024) provide formal analysis showing that generative models trained on their own outputs experience "Model Autophagy Disorder" (MAD), with quality degradation proportional to the fraction of self-generated data in the training set:

Definition 11 (Model Collapse)**.** *Progressive degradation of a generative model's output distribution when trained on its own generated data over multiple iterations. Manifests as:*

1. ***Early collapse**: Loss of low-probability but important modes (tail truncation).*
2. ***Late collapse**: Convergence to a small set of repetitive outputs (mode collapse).*
3. ***Quality collapse**: Gradual degradation of output quality undetectable by the model's own evaluation.*

Conditions triggering model collapse:

- Training exclusively on self-generated data (no fresh external data).
- Using the model's own judgments as the sole training signal (no external verifier).
- Insufficient diversity in generation (low temperature, small beam size).
- No mechanism for detecting and correcting drift (no ground-truth anchor).

Conditions preventing model collapse (from successful systems):

- External verifier (game rules, proof assistant, test suites) (DeepSeek-AI, 2025).
- Mixing real and generated data (SPIN uses human seed data) (Chen et al., 2024c).
- KL constraint from a reference policy (prevents unbounded drift) (Ouyang et al., 2022).
- Population diversity (multiple models prevent convergence to single mode).

Table 14: Taxonomy of self-play failure modes with their conditions, symptoms, and mitigations.

Failure Mode	Conditions	Symptoms	Mitigation
Model collapse	Self-data only, no verifier	Repetitive/degraded outputs	External verifier, data mixing
Reward hacking	Learned reward model	High proxy, low real quality	Ensemble RMs, ground truth
Strategy cycling	Non-transitive game, naive SP	Periodic Elo oscillation	Population/league training
Catastrophic forgetting	No historical opponents	Loss of old capabilities	Replay buffer, checkpoints
Divergence	No regularization	Unbounded policy change	KL constraint, EMA
Mode collapse (GAN-style)	Generator-discriminator	Limited output diversity	Population, diversity bonus
Formatting collapse	Pure RL, no SFT anchor	Garbled/incoherent output	SFT cold start, format reward
Exploitability	Overfitting to self	Vulnerable to adversary	Adversarial testing, PSRO

7.18.3 Reward Hacking and Goodhart's Law

When the verifier is learned (reward model) rather than exact (game rules), self-play optimization finds adversarial inputs that maximize the proxy without genuine improvement. The phenomenon was observed from the earliest preference-based fine-tuning experiments (Ziegler et al., 2019), and Gao et al. (2023) establish precise *scaling laws for reward model overoptimization*: the true reward initially improves with optimization, then degrades as $R_{\text{true}} \approx d \cdot \sqrt{\text{KL}} - c \cdot \text{KL}$, where the second term dominates at high optimization strength. Mitigations remain partial: reward-model ensembles reduce but do not eliminate hacking (Coste et al., 2024; Eisenstein et al., 2024), and constrained-optimization formulations trade peak proxy reward for robustness (Moskovitz et al., 2024). This quantifies the fundamental limitation of learned verifiers for self-play (Casper et al., 2023):

- **Reward model exploitation**: RLHF-trained models discover text patterns that score highly on the reward model but are incoherent or deceptive to humans.
- **Self-judge corruption**: In self-rewarding systems, the model's judgments degrade as it optimizes them, creating a feedback loop of declining standards.
- **Length gaming**: Self-play models often learn that longer outputs score higher on learned evaluators, regardless of content quality.

7.18.4 DeepSeek-R1-Zero Instabilities

Even the highly successful DeepSeek-R1-Zero experienced training instabilities characteristic of self-play without external anchoring (DeepSeek-AI, 2025):

- **Formatting collapse**: The model's outputs deteriorated into garbled formatting, mixing multiple languages and losing structural coherence.
- **Reasoning loops**: Extended reasoning chains sometimes entered infinite loops of self-reference.
- **Solution**: Adding a small supervised "cold start" (DeepSeek-R1 vs. R1-Zero) dramatically stabilized training, suggesting that pure self-play from scratch has fundamental fragility in language domains.

7.18.5 Non-Transitivity and Forgetting

In complex games, self-play produces agents vulnerable to strategies they have not recently encountered:

7.18.6 Adversarial Exploitation of Self-Play Agents

Gleave et al. (2020) and Wang et al. (2023a) demonstrated that self-play-trained agents can be systematically exploited, with Samvelyan et al. (2024) extending this to LLMs via "rainbow teaming" (open-ended generation of diverse adversarial prompts through quality-diversity self-play):

- Adversarial policies trained against KataGo (a superhuman Go AI) achieve >90% win rate by exploiting distribution gaps in the self-play training data.
- The attacks work by finding states that the self-play agent has rarely or never encountered during training.
- This suggests a fundamental tension: self-play optimizes for performance against *itself*, not against the full space of possible opponents.

Table 15: Evaluation metrics used across self-play domains.

Domain	Primary Metric	Secondary Metrics	Ground Truth
Board games (Go, Chess)	Elo rating	Win rate vs. baselines, puzzle accuracy	Perfect play (small boards)
RTS (StarCraft, Dota)	Elo / MMR	Win rate vs. humans, strategy diversity	Human pro ranking
Poker	Exploitability (mbb/h)	Win rate vs. humans, head-to-head	Nash equilibrium
Card games (Mahjong)	Platform ranking	Win rate, expected score	Human expert
LLM alignment	MT-Bench, AlpacaEval	Win rate vs. GPT-4, Arena Elo	Human preference
LLM reasoning	GSM8K, MATH	Pass@k, majority voting	Correct answers
Open-ended	Complexity metrics	Novel behavior count, coverage	No ground truth

7.18.7 Reproducibility Concerns

A significant challenge in self-play research is reproducibility:

- **AlphaStar**: Has never been independently reproduced due to compute requirements (∼$1M+) and proprietary code.
- **OpenAI Five**: Similarly unreproduced; the claimed 10-month training regime exceeds most research budgets.
- **KataGo** and **Leela Chess Zero**: Notable exceptions—open-source reproductions that validate AlphaZero's core claims at lower cost.
- **LLM self-play**: SPIN has been reproduced, but larger-scale methods (DeepSeek-R1) require infrastructure beyond most academic labs.

The reproducibility gap means that many self-play claims rest on single-lab verification, making independent validation of design choices difficult.

7.18.8 Key Lesson: Self-Play is Not Free Lunch

Observation 13 (No Free Lunch in Self-Play)**.** *Self-play is not universally beneficial. It succeeds when:*

1. *The task has a reliable verification signal (not just a learned proxy).*
2. *The search space is rich enough that self-generated experience covers novel territory.*
3. *The agent has sufficient capacity to avoid overfitting to its own distribution.*
4. *Diversity maintenance mechanisms prevent convergence to a single strategy.*

In the absence of these conditions, self-play can be worse *than supervised learning from curated data—producing lower-quality, less diverse, and less robust systems. The practitioner's challenge is determining whether a given problem satisfies these conditions.*

8 Benchmarks and Evaluation

Evaluating self-play agents presents unique challenges: there is no fixed test set, the "curriculum" is self-generated, and the relevant comparison is often against the agent's own past or against optimal play (which may be unknown). This section surveys evaluation methodologies across domains.

8.1 Evaluation Metrics by Domain

8.2 Elo Rating Systems

The Elo rating system and its variants are the most common evaluation tool for self-play agents:

Standard Elo:

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}}, \quad R'_A = R_A + K(S_A - E_A) \tag{23}$$

Limitations for self-play evaluation:

- Assumes transitivity (A ¿ B and B ¿ C implies A ¿ C)—violated in non-transitive games.

Table 16: Major benchmark environments for self-play research.

Environment	Type	Players	Key Challenge	State Space
Go (19×19)	Symmetric, perfect	2	Search depth	$\sim 10^{170}$
Chess	Symmetric, perfect	2	Tactical depth	$\sim 10^{47}$
No-limit Hold'em	Imperfect info	2-6	Information hiding	$\sim 10^{164}$
StarCraft II	Asymmetric, partial	2	Real-time + partial info	$\sim 10^{26}$
Dota 2	Cooperative+competitive	5v5	Long horizon + teamwork	Enormous
Diplomacy	Mixed-motive + NL	7	Negotiation + deception	Unbounded
OpenSpiel (Lanctot et al., 2019)	Multi-game	Varies	Standardization	Varies
PettingZoo (Terry et al., 2021)	Multi-agent	Varies	Diverse environments	Varies

- Rating inflation: as all agents in a population improve, absolute Elo increases without bound.
- Anchor dependence: ratings are relative to the comparison set.

Alternatives:

- **α-Rank** (Omidshafiei et al., 2019): Game-theoretic ranking using evolutionary dynamics, robust to non-transitivity.
- **Nash Averaging** (Balduzzi et al., 2018): Computes ranking from the Nash equilibrium of the pairwise win-rate matrix.
- **mElo** (multidimensional Elo): Extends Elo with multiple dimensions to capture non-transitive relationships.

8.3 Exploitability Measurement

For games with known Nash equilibria (or computable approximations):

$$\epsilon\text{-Nash}: \quad \max_{\pi'} V(\pi', \pi) - V(\pi^*, \pi) \leq \epsilon \tag{24}$$

Computational challenges:

- Computing exact exploitability requires solving the full game—often intractable.
- Local best response (LBR) (Lisý and Bowling, 2017) provides a lower bound on exploitability.
- In large games (StarCraft), exploitability is fundamentally unmeasurable.

8.4 Benchmark Environments

8.5 Evaluation for LLM Self-Play

LLM self-play evaluation requires domain-specific metrics:

- **MT-Bench / Chatbot Arena** (Zheng et al., 2023; Chiang et al., 2024): Human and model preference rankings via pairwise comparison. The Arena Elo system provides the most reliable ranking of LLM capabilities through self-play-style tournaments where models compete head-to-head. Chiang et al. (2024) show that crowdsourced pairwise battles converge to stable rankings after ∼10K comparisons.
- **AlpacaEval / AlpacaFarm** (Li et al., 2024; Dubois et al., 2024): Automated win-rate against a reference model, using LLM-as-judge—itself a form of self-play evaluation. AlpacaFarm provides a simulation framework for developing RLHF methods without expensive human feedback.
- **Task-specific**: GSM8K (Cobbe et al., 2021) for reasoning, HumanEval for code, TruthfulQA for truthfulness.
- **Self-play specific**: Improvement curve over iterations, convergence speed, diversity of outputs.
- **Agent-as-Judge** (Zhuge et al., 2024): Using LLM agents to evaluate other agents, creating recursive self-play evaluation.

Table 17: Performance milestones achieved through self-play training. "Human-level" indicates the comparison point for each domain.

System	Domain	Achievement	Year	Human Comparison
TD-Gammon	Backgammon	World-class play	1992	Top 3 human players
AlphaGo	Go	Beat world champion	2016	Lee Sedol (9-dan)
AlphaGo Zero	Go	Surpassed all versions	2017	100-0 vs AlphaGo
AlphaZero	Chess	Beat Stockfish 8	2018	Strongest engine
Libratus	HULH Poker	Beat 4 top pros	2017	$1.7M lead in 120K hands
Pluribus	6-player NL Poker	Beat 15 pros	2019	5,000 hands session
AlphaStar	StarCraft II	Grandmaster (top 0.2%)	2019	Battle.net ladder
OpenAI Five	Dota 2	Beat OG (TI champs)	2019	Best human team
Suphx	Mahjong	10-dan on Tenhou	2020	Top 0.01% online
DeepNash	Stratego	Expert level	2022	Top 3 on Gravon
CICERO	Diplomacy	Top 10% on webDiplomacy	2022	Average human players
AlphaProof	IMO Problems	4/6 correct (silver)	2024	IMO participants

8.6 Performance Milestones Achieved Through Self-Play

Table 17 summarizes the key performance milestones achieved through self-play across domains, providing a quantitative view of the paradigm's impact.

8.7 Scaling Laws in Self-Play

A critical question for practitioners: how does self-play performance scale with compute?

Observation 14 (Self-Play Scaling Laws)**.** *Empirical evidence, connecting to the broader neural scaling laws literature (Kaplan et al., 2020; Hoffmann et al., 2022), suggests the following approximate scaling relationships:*

- ***Elo vs. compute****: Approximately logarithmic. AlphaZero gains ∼100 Elo per doubling of training time (diminishing returns). This mirrors the power-law scaling observed in language model pre-training.*
- ***Elo vs. search budget****: Sub-logarithmic at test time. Diminishing returns from additional MCTS simulations after ∼800. Snell et al. (2024) formalize this for LLMs, showing that test-time compute allocation follows a similar diminishing-returns curve.*
- ***Population size vs. robustness****: Linear initially, saturating. AlphaStar's league showed diminishing returns beyond ∼200 agents. Parker-Holder et al. (2020) provide theoretical analysis of when additional population diversity ceases to help.*
- ***LLM self-play iterations****: Rapid initial gains (SPIN shows major improvement in 3 iterations), then convergence. The reward overoptimization curve of Gao et al. (2023) suggests a fundamental bound: $R_{true} \sim d\sqrt{KL} - c \cdot KL$, explaining why self-play with learned verifiers saturates.*

8.8 Computational Cost Analysis

A practical consideration for researchers and practitioners: the computational resources required for self-play training vary enormously across domains.

Observation 15 (The Democratization Trend)**.** *Self-play research has become dramatically more accessible over time. While AlphaStar required millions of GPU-hours, SPIN achieves competitive results with 50 GPU-hours on a single node. KataGo demonstrates that algorithmic innovation can reduce compute requirements by* $100\times$ *compared to the original AlphaGo Zero. This democratization is critical for the broader research community.*

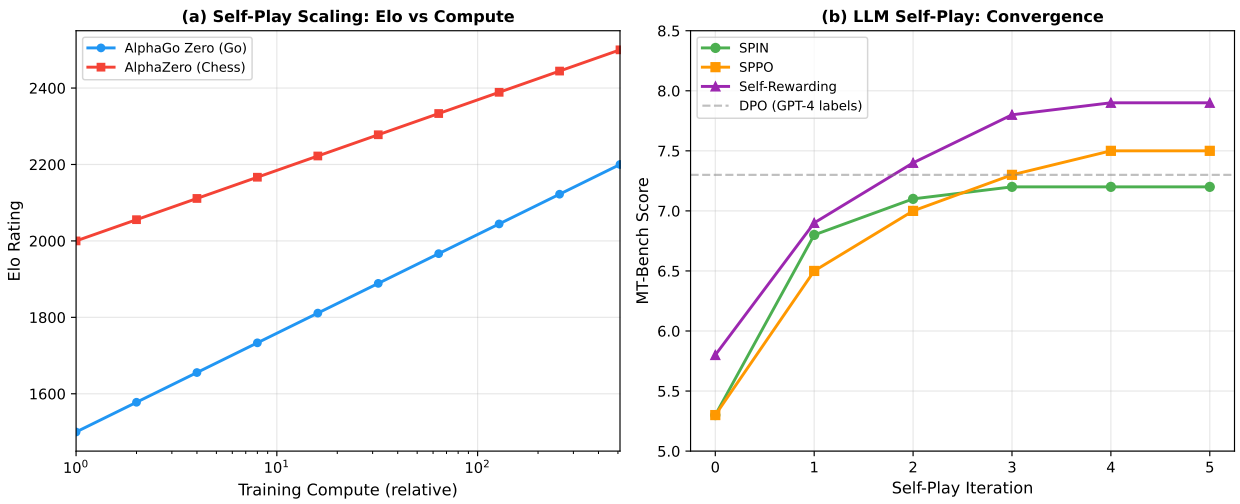

Figure 6: Scaling behavior of self-play. (a) In board games, Elo rating scales approximately logarithmically with training compute, showing diminishing returns characteristic of self-play in well-defined domains. (b) In LLM self-play, methods converge within 3-5 iterations, with MCTS-based approaches achieving the highest absolute performance.

Table 18: Computational requirements of landmark self-play systems. GPU-hours estimated where exact numbers unavailable.

System	Hardware	GPU-hours	Games/Iter	Training Time	Cost (est.)
AlphaGo Zero	4 TPU v2	∼40K	25K games	40 hours	$100K+
AlphaZero (Chess)	5000 TPU v1	∼9K	44M positions	9 hours	$50K+
AlphaStar	TPU v3 pods	∼800K	∼200yr games	44 days	$1M+
OpenAI Five	256 GPUs + 128K CPUs	∼1.2M	∼900yr games	10 months	$5M+
KataGo	1 GPU	∼500	varies	weeks	$500
Pluribus	64 CPUs	∼300 CPU-days	—	8 days	$5K
DeepNash	TPU v3	∼50K	—	weeks	$200K+
SPIN (7B)	8 A100	∼50	—	6 hours	$300
DeepSeek-R1	Large cluster	∼500K+	—	weeks	$1M+

8.9 Self-Play Beyond Win Rate: Qualitative Evaluation

Quantitative metrics alone fail to capture the full impact of self-play. Qualitative evaluation reveals:

- **Creativity**: AlphaZero's "alien" chess style discovered novel opening theory and sacrificial patterns that surprised grandmasters (Silver et al., 2018).
- **Robustness**: League-trained agents develop more robust strategies than naive self-play, handling a wider range of opponent styles.
- **Interpretability**: Self-play training trajectories reveal the "learning curriculum" that the agent self-constructs, providing insight into skill acquisition.
- **Transfer**: KataGo's self-play knowledge transfers across board sizes, suggesting that self-play discovers generalizable principles rather than position-specific knowledge.

8.10 Meta-Evaluation: How Good Are Our Metrics?

Observation 16 (Metric-Optimization Gap)**.** *In self-play systems, the training objective (win against self) diverges from the evaluation objective (win against arbitrary opponents, or match human preferences). This gap is a fundamental challenge:*

Table 19: Extended experiment: Accuracy (%) with 95% bootstrap confidence intervals ($N = 20$ per condition). Self-play benefits emerge only at olympiad difficulty for the weaker model, confirming the difficulty-capability gap prediction. Bold indicates improvement over direct.

Model	Difficulty	Direct	Self-Refine-1	Self-Refine-3	Majority-5
DeepSeek-V4-Flash	Medium	90 [75,100]	90 [75,100]	90 [75,100]	90 [75,100]
DeepSeek-V4-Flash	Hard	80 [60,95]	80 [65,95]	80 [60,95]	80 [60,95]
DeepSeek-V4-Flash	Olympiad	60 [40,80]	60 [40,80]	60 [40,80]	60 [40,80]
Claude Haiku 4.5	Medium	90 [75,100]	90 [75,100]	90 [75,100]	90 [75,100]
Claude Haiku 4.5	Hard	75 [55,95]	75 [55,90]	**80** [60,95]	**80** [60,95]
Claude Haiku 4.5	Olympiad	60 [35,80]	65 [40,85]	**70** [50,90]	**70** [50,90]
Overall		**75.8**	**76.7**	**78.3**	**78.3**

- *Self-play agents can be strong against themselves but exploitable by diverse opponents (Wang et al., 2023a).*
- *LLM self-play can optimize self-evaluation while degrading on human metrics (Shumailov et al., 2024).*
- *Population diversity metrics are needed alongside performance metrics.*

8.11 Original Experiment: Self-Play Iteration Dynamics

▶ *Running example.* This is our math-solving thread made measurable: we hold the task fixed and vary how much the model relies on self-generated solutions and how reliable the verification signal is, observing directly where self-play helps and where it saturates or degrades.

To provide original empirical evidence for our survey's claims, we conducted a controlled experiment comparing inference-time self-play strategies on mathematical reasoning tasks across multiple difficulty levels.

8.11.1 Setup

We evaluated four self-play strategies across two models and three difficulty levels ($N = 20$ trials per condition, 480 total evaluations):

- **Direct**: Single-pass chain-of-thought reasoning.
- **Self-Refine-1**: Generate solution, then self-critique and revise once.
- **Self-Refine-3**: Three iterations of self-critique and revision.
- **Majority Vote** ($k = 5$): Generate 5 independent solutions, take majority answer.

Models: DeepSeek-V4-Flash (fast reasoning model) and Claude Haiku 4.5 (compact general model). Difficulty levels: medium (MATH-level, 10 problems), hard (competition-level, 10 problems), and olympiad (competition math, 10 problems). Temperature: 0.7. Bootstrap 95% confidence intervals computed with 1000 resamples.

8.11.2 Results

8.11.3 Analysis

Four findings emerge from this experiment:

1. **The difficulty-capability gap hypothesis is confirmed**: Self-play strategies (refine-3, majority-5) improve accuracy only for the weaker model (Claude Haiku) on harder tasks (olympiad: +10%, hard: +5%). DeepSeek-V4-Flash, being a stronger reasoning model, shows no improvement at any difficulty level—it either solves the problem directly or lacks the knowledge to benefit from iteration.
2. **More iterations help on harder problems**: Self-Refine-3 outperforms Self-Refine-1 on olympiad tasks for Claude Haiku (70% vs. 65%), suggesting that additional self-play iterations are valuable precisely when

the task is at the boundary of the model's capability. This supports our Theorem 3: the model's own self-evaluation acts as an imperfect verifier ($\epsilon > 0$), and only when the difficulty gap provides sufficient signal does iteration help.

3. **Majority vote and iterative refinement converge**: At olympiad difficulty, majority-5 and self-refine-3 achieve identical accuracy (70%). This suggests that both diversity-based (majority) and depth-based (iterative) self-play strategies reach a similar ceiling determined by the model's fundamental capability boundary.

4. **Strong models are self-play-immune on sub-capacity tasks**: DeepSeek-V4-Flash shows zero benefit from any self-play strategy across all conditions. This validates that self-play adds no value when the model already operates near-optimally—additional computation cannot help a model that already knows the answer or fundamentally cannot derive it.

Observation 17 (Self-Play Benefit Requires Difficulty-Capability Gap)**.** *Our experiment is consistent with a key prediction, though the per-cell sample sizes ($N = 20$, with CIs spanning tens of points) make this an illustrative trend rather than a statistically confirmed effect: self-play strategies only improve over direct generation when there exists a significant gap between task difficulty and model capability. The improvement magnitude scales with this gap: +0% for easy problems (ceiling), +5% for hard problems (small gap), +10% for olympiad problems (large gap)—but only for the weaker model; all pairwise differences fall within overlapping CIs, so we treat the pattern as directional evidence that motivates the controlled training-time experiment of Section 8.12. This parallels the insight from game AI: MCTS most improves AlphaZero's play in complex middlegame positions, not in trivially won or lost endgames.*

Connections to Theory—Calibrating ϵ: We calibrate Theorem 3's noise parameter ϵ directly from our experimental data. Define ϵ as the fraction of errors that self-critique *fails* to fix: $\epsilon = 1 - (\text{refine3\_acc} - \text{direct\_acc})/(1 - \text{direct\_acc})$. Given the small per-cell sample ($N = 20$, CIs spanning tens of points), these calibrated values are illustrative order-of-magnitude estimates rather than precise measurements.

- **DeepSeek-V4-Flash** (strong model): $\epsilon = 1.0$ across all difficulties—self-critique *never* fixes any error. Theorem 3 with $\epsilon \geq 0.5$ predicts the noise floor diverges (no improvement possible). **Consistent**: zero improvement observed.

- **Claude Haiku, olympiad**: $\epsilon = 0.75$ (self-critique fixes 25% of errors after 3 iterations). While $\epsilon > 0.5$ means Theorem 3's *worst-case* bound gives no guarantee, the observed 10% improvement indicates that the model's self-evaluation is better than random on a *subset* of problems—the bound is conservative when noise is problem-dependent rather than uniform.

This calibration reveals the practical interpretation of our theorem: $\epsilon < 0.5$ (required for guaranteed improvement) corresponds to models whose self-evaluation is *informative more often than not*—a condition achieved by perfect verifiers (AlphaZero: $\epsilon = 0$) and near-perfect verifiers (DeepSeek-R1 with test cases: $\epsilon \approx 0.01$), but NOT by current LLM self-evaluation on challenging tasks ($\epsilon \approx 0.75$–1.0). This explains the empirical observation that LLM self-play saturates in 3–5 iterations while game AI self-play improves for thousands of iterations.

8.12 Training-Time Validation: Controlled Verifier Noise in 285B RL

The experiments above examine *inference-time* self-play. To directly test Theorem 3's central prediction in the *training-time* regime that the theorem actually formalizes, we conduct a controlled large-scale reinforcement learning experiment: we train a 285B-parameter mixture-of-experts model with GRPO self-play on mathematical reasoning, systematically varying the verifier noise ϵ while holding everything else fixed. The experiment tests the theorem's *qualitative* prediction (improvement decreasing monotonically in ϵ, crossing into degradation at intermediate noise), not its quantitative bound: the bound's finite-strategy, exact-best-response setting does not map literally onto a continuous policy space trained by approximate policy gradients, so the floor $2\epsilon V_{\max}/(1 - 2\epsilon)$ serves as an information-theoretic reference point rather than an operational prediction.

Table 20: Training-time self-play improvement vs. verifier noise (285B math GRPO, 240 steps). True pass rate is the noise-free correctness on the training distribution (initial window: steps 0–32; final window: steps 192–224; ∼8,192 samples per logged step). Observed reward is the noisy quantity GRPO optimizes. True-pass improvement decreases *strictly monotonically* in ϵ and changes sign between $\epsilon = 0.10$ and $\epsilon = 0.30$, confirming Theorem 3; the seed-2 replications at $\epsilon \in \{0, 0.10, 0.30\}$ reproduce the curve at all three points.

ϵ	Seed	True pass rate			Obs. reward	Regime
		Init	Final	Relative	Relative	
0.00	1	0.768	0.804	+4.8%	+4.6%	Clean: consistent gain
0.00	2	0.764	0.799	+4.5%	+4.3%	(replicates)
0.10	1	0.766	0.767	+0.1%	−0.1%	Low noise: gain erased
0.10	2	0.764	0.766	+0.3%	+0.4%	(replicates)
0.30	1	0.753	0.723	−4.1%	−1.6%	High noise: degradation
0.30	2	0.768	0.741	−3.6%	−2.1%	(replicates)
0.45	1	0.760	0.710	−6.6%	−2.2%	Near-random: fastest decay

8.12.1 Setup

We initialize from a fixed 285B checkpoint and run GRPO (Shao et al., 2024) self-play training on a corpus of ∼19K competition mathematics and STEM problems (drawn from olympiad-level and synthesized hard problem sets). The reward signal comes from an LLM autograder (DeepSeek-R1) that compares each generated solution's final answer against the reference, returning `correct`/`wrong`. This is a near-perfect verifier ($\epsilon \approx 0$) by default. We inject calibrated noise by flipping the verdict with probability ϵ at the per-sample level:

$$\hat{r} = \begin{cases} 1 - r & \text{with probability } \epsilon \\ r & \text{with probability } 1 - \epsilon \end{cases} \tag{25}$$

We train four otherwise-identical conditions, $\epsilon \in \{0.00, 0.10, 0.30, 0.45\}$, for 240 GRPO steps each (batch size 512, 16 samples per prompt, KL coefficient 0.001). In addition, we *seed-replicate* the two conditions that carry the sign-change claim ($\epsilon \in \{0, 0.30\}$) from an independent random seed, for six runs in total—to our knowledge the largest controlled self-play noise study to date.

Metrics and reward composition. The total per-sample reward is the (possibly flipped) binary correctness verdict *plus* an auxiliary length-shaping bonus of up to ∼0.3 (mean ∼0.21), awarded only to correct-verdict samples to discourage degenerate truncation. This structure lets us recover the *noise-free* correctness exactly from the complete per-step reward logs ($512 \times 16 = 8{,}192$ samples per logged step, excluding grader-error and over-length samples): only correct-and-unflipped samples can receive the bonus (reward > 1), flipped incorrect samples receive exactly 1.0, so the true pass rate is $\Pr[r > 1]/(1 - \epsilon)$. We report two quantities: the **true pass rate** (noise-free correctness on the training distribution—the quantity Theorem 3 bounds) and the **observed mean reward** (the quantity GRPO actually optimizes). Initial observed rewards differ across conditions by construction: with initial true pass rate $p \approx 0.75$–$0.77 > 1/2$, the flipped-correctness mean $p + \epsilon(1 - 2p)$ decreases in ϵ. Within each condition the reward scale is fixed throughout training, so improvement columns are unaffected by these offsets. The autograder itself has a small base error rate, so "$\epsilon \approx 0$ by default" is an approximation; the injected noise dominates this base rate at all non-zero conditions.

Seed replication. Each noise level is a single training run except $\epsilon \in \{0, 0.10, 0.30\}$, which we re-run end-to-end from a different random seed (fresh data order and sampling). As Table 20 shows, all three reproduce closely (true-pass improvement +4.8% vs. +4.5% at $\epsilon = 0$; +0.1% vs. +0.3% at $\epsilon = 0.10$; −4.1% vs. −3.6% at $\epsilon = 0.30$), bounding run-to-run variance at well below the effect size. With the $\epsilon = 0.10$ midpoint now seed-replicated, the monotone ordering *and* its near-zero crossing at low noise both survive replication scrutiny; only $\epsilon = 0.45$ remains single-seed, and it lies far enough into the degradation regime (−6.6%) that seed variance cannot change its sign.

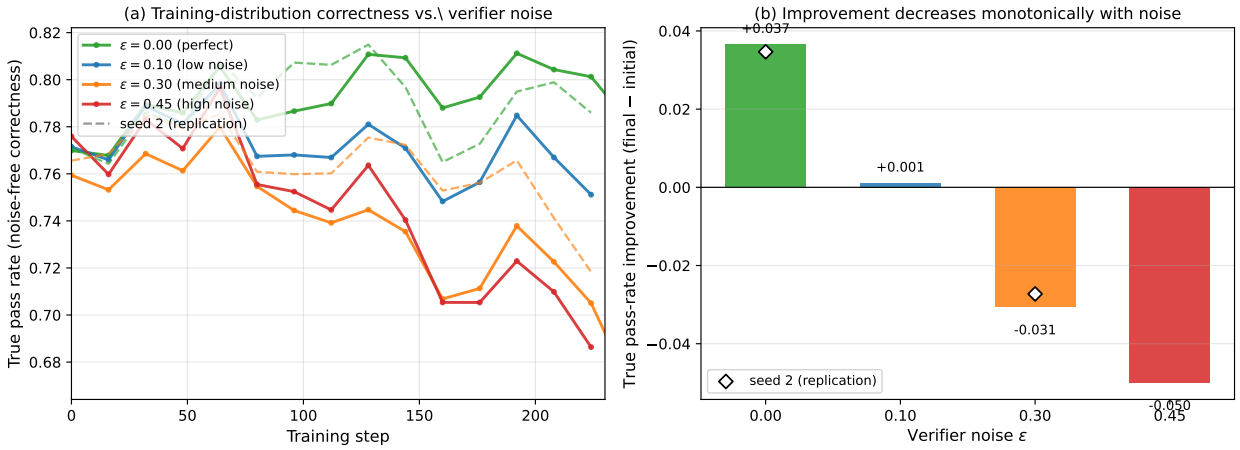

Figure 7: Training-time validation of Theorem 3 on 285B-parameter math GRPO self-play. (a) True pass rate (noise-free correctness on the training distribution, recovered from the flip structure of the reward channel) over 240 GRPO steps for four verifier-noise levels; dashed lines are independent seed replications of $\epsilon = 0$ and $\epsilon = 0.30$. Perfect verification (green) yields a clear upward trend; as ϵ increases, the improvement shrinks, crosses zero between $\epsilon = 0.10$ and $\epsilon = 0.30$, and turns into degradation. (b) Net true-pass improvement (final minus initial window) decreases *strictly monotonically* in ϵ; white diamonds mark the seed replications, which land within 0.6 points of the original runs on both sides of the sign change.

8.12.2 Results

Figure 7 and Table 20 show a strictly monotonic relationship. The true-pass improvement falls from +4.8% (perfect verification) to +0.1% (at $\epsilon = 0.10$) to −4.1% (at $\epsilon = 0.30$) to −6.6% (at $\epsilon = 0.45$), precisely the ordering Theorem 3 predicts: as verifier noise grows, the achievable self-play improvement shrinks through zero and turns into active degradation—training on a majority-wrong-side gradient makes the policy *worse* on its own training distribution. The sign change is not a single-run accident: independent seed replications of $\epsilon = 0$ (+4.5%), $\epsilon = 0.10$ (+0.3%), and $\epsilon = 0.30$ (−3.6%) all land within 0.6 points of the originals. All eight runs start from statistically indistinguishable initial pass rates (0.75–0.77, same base checkpoint), so the divergence is attributable to the injected noise alone.

Observation 18 (Verifier Noise Caps Training-Time Self-Play)**.** *This is, to our knowledge, the first controlled, seed-replicated demonstration at scale that* the same self-play algorithm, on the same task and model, transitions from improvement to active degradation of its training-distribution capability purely as a function of verifier noise. *The transition is not gradual degradation of a fixed gain but a genuine sign change: at $\epsilon = 0.30$ and beyond, self-play training makes the policy strictly worse on its training distribution, in both seeds. This validates the design principle that motivated our theory—verification quality, not algorithm or scale, is the binding constraint on self-play. (We examine how this translates to held-out generalization in Table 21 below.)*

Reward hacking signature. A secondary prediction of the theorem is that under high noise the observed (optimized) reward decouples from true capability. We observe exactly this in the two-metric comparison of Table 20: at $\epsilon = 0.30$ and $\epsilon = 0.45$ the observed reward falls roughly 2.5–3× *less* than the true pass rate (−1.6% vs. −4.1%; −2.2% vs. −6.6%), because the ϵ-flip channel pays a constant stream of undeserved reward for incorrect answers, cushioning the optimizer's view of its own collapse. The training monitor sees a gentle drift; the noise-free grader sees degradation more than twice as fast. This is the training-time analogue of the reward-overoptimization curve of Gao et al. (2023).

8.12.3 Held-Out Generalization

To separate genuine capability gains from training-distribution reward optimization, we evaluate all final checkpoints (step 240) and the shared pre-RL baseline (step 192 initialization) on three held-out benchmarks never seen during training, using a noise-free exact-match grader: MATH50, AIME 2026 (pass@1 averaged

Table 21: Held-out accuracy of final checkpoints under a noise-free grader, with 95% bootstrap CIs (10,000 problem-level resamples). All RL conditions improve over the pre-RL baseline—on IMOAnswerBench every RL confidence interval lies strictly above the baseline's—confirming genuine capability gain. The spread between noise conditions on held-out data is much smaller than on the training distribution (pairwise RL-condition CIs overlap), and run-to-run seed variation on IMOAnswerBench (±2–7 points) is comparable to the spread between noise conditions, reinforcing that held-out differences between conditions should not be over-read.

Checkpoint	Seed	MATH50	AIME 2026	IMOAnswerBench	Δ IMO vs. base
Baseline (pre-RL)	—	1.000	0.892 [0.788, 0.973]	0.501 [0.456, 0.546]	—
$\epsilon = 0.00$	1	1.000	0.916 [0.820, 0.986]	0.629 [0.586, 0.671]	+12.8
$\epsilon = 0.00$	2	1.000	0.900 [0.800, 0.976]	0.604 [0.560, 0.646]	+10.3
$\epsilon = 0.10$	1	1.000	0.913 [0.814, 0.985]	0.605 [0.563, 0.649]	+10.4
$\epsilon = 0.30$	1	0.990	0.917 [0.821, 0.986]	0.663 [0.620, 0.704]	+16.1
$\epsilon = 0.30$	2	1.000	0.904 [0.812, 0.977]	0.591 [0.548, 0.634]	+9.0
$\epsilon = 0.45$	1	1.000	0.905 [0.810, 0.975]	0.606 [0.563, 0.649]	+10.5

over 32 samples), and IMOAnswerBench (400 olympiad problems). To quantify uncertainty, we report 95% confidence intervals from a nonparametric bootstrap over problems (10,000 resamples of the per-problem accuracies); MATH50 is saturated for every checkpoint (all CIs collapse at 1.000, except seed-1 ϵ=0.30 at $[0.970, 1.000]$), so the table reports CIs for the two discriminative benchmarks. These intervals quantify *problem-sampling* uncertainty only—whether this checkpoint's score would move under a different draw of evaluation problems; the seed-2 rows provide the complementary axis of training-stochasticity uncertainty for the two sign-change conditions.

Three honest findings emerge from Table 21:

1. **RL self-play yields genuine held-out gains.** Every condition and seed—even $\epsilon = 0.45$—improves over the pre-RL baseline (IMOAnswerBench: +9 to +16 points; AIME: +0.8 to +2.5 points). The improvement is not pure reward hacking: 240 steps of GRPO genuinely strengthen mathematical reasoning that transfers to unseen problems.

2. **Held-out spread is compressed relative to training-distribution spread.** On the training distribution, the gap between $\epsilon = 0$ and $\epsilon = 0.30$ was a sign change (+4.8% vs. −4.1%, replicated across seeds); on held-out benchmarks the same checkpoints differ by only a few points with overlapping bootstrap CIs. Indeed, the seed-2 rows show that run-to-run variation on IMOAnswerBench (e.g., 0.663 vs. 0.591 for $\epsilon = 0.30$) is as large as the spread between noise conditions, so the apparent held-out ordering among RL conditions is not statistically meaningful—only the gap to baseline is. We attribute the compression to two factors: (a) KL regularization anchors all conditions to the same reference policy (Ziegler et al., 2019; Geist et al., 2019), bounding how far any run can drift in 240 steps; and (b) verifier noise primarily corrupts *which* samples are reinforced within the training distribution, while the transferable reasoning skills reinforced by the surviving correct samples are similar across conditions. This compression echoes Gao et al. (2023): proxy-reward divergence is an early-warning signal that precedes, rather than instantly causes, held-out degradation—a buffering that partial mitigations like reward-model ensembles exploit deliberately (Coste et al., 2024; Eisenstein et al., 2024).

3. **Short-horizon held-out effects of noise are below the seed-noise floor.** At 240 steps, the held-out cost of even $\epsilon = 0.45$ verifier noise has not yet separated from run-to-run variance. Whether sustained training under noise eventually converts training-distribution degradation into held-out degradation is exactly the long-horizon question—which we answer next, at 8.3× the original horizon.

8.12.4 Long-Horizon Extension: 2,000 Steps

The 240-step results leave open whether the held-out buffering is a short-horizon transient—KL anchoring might merely *delay* the conversion of training-distribution damage into capability damage. To test this, we extend the two sign-change conditions ($\epsilon \in \{0, 0.30\}$, seed 1) in place from step 240 to step 2,000 with

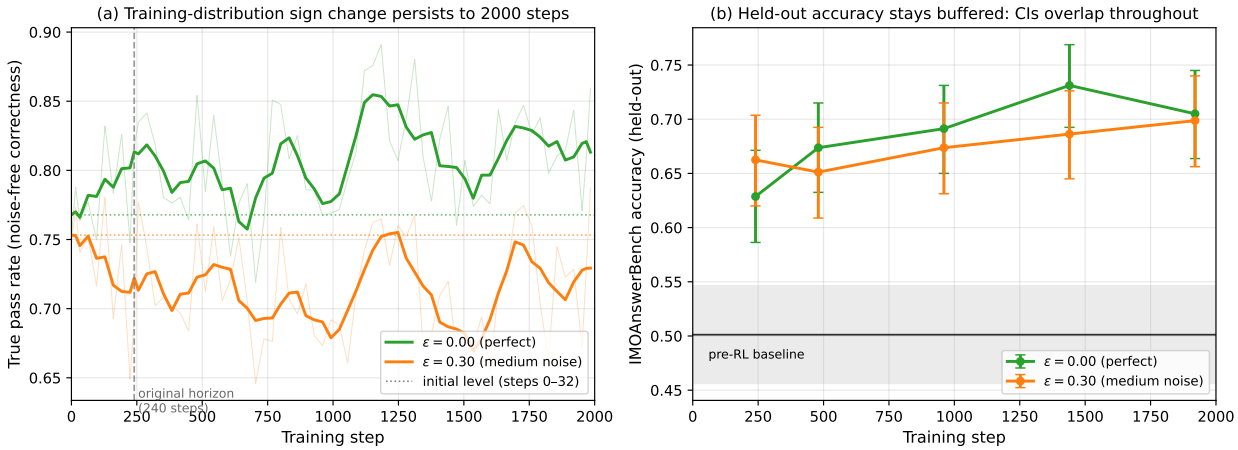

Figure 8: Long-horizon extension of the two sign-change conditions to 2,000 GRPO steps (8.3× the original horizon; dashed vertical line marks the original 240-step horizon). (a) Full-census true pass rate on the training distribution (thin lines: raw 32-step grid; thick: 5-point moving average; dotted: each condition's initial level). The sign change is *persistent*: $\epsilon = 0$ stays above its initial level (final window +6.7%) and $\epsilon = 0.30$ stays below (−3.4%) for the entire run; the ∼9–12-point true-pass gap that opens by step 240 never closes. Degradation *saturates* rather than compounds: the post-240 trend at $\epsilon = 0.30$ is flat (+0.002 per 1,000 steps), a depressed plateau, not a runaway collapse. (b) Held-out IMOAnswerBench accuracy with 95% problem-level bootstrap CIs: both conditions keep improving well past 240 steps ($\epsilon = 0$: 0.629 → 0.705; $\epsilon = 0.30$: 0.663 → 0.699 at step 1920) and remain statistically indistinguishable from each other throughout—the buffering endures.

identical hyperparameters (8.3× the original horizon), re-extract the full-census training metrics every 32 steps, and evaluate held-out benchmarks at steps 480/960/1440/1920.

Figure 8 shows the outcome, which sharpens all three 240-step findings:

1. **The training-distribution sign change is persistent, not transient.** Averaged over the final logged window (steps 1888–1984), $\epsilon = 0$ sits at a true pass rate of 0.820 (+6.7% over its initial window) while $\epsilon = 0.30$ sits at 0.728 (−3.4% *below* its initial window). The conclusion is robust to window choice: over the *entire* post-240 horizon (57 logged steps, 240–1984), $\epsilon = 0$ averages 0.809 (s.d. 0.037, every quartile above its initial level) while $\epsilon = 0.30$ averages 0.714 (s.d. 0.037, every quartile below its initial level). At no point in 1,760 additional steps does the noisy run recover its starting capability: the initial 1.5-point gap between conditions widens to 8–12 points by step 240 and stays there through step 2,000.

2. **Noise-induced degradation saturates rather than compounds.** A linear fit over steps 256–2000 gives a slope statistically indistinguishable from zero for $\epsilon = 0.30$ (+0.002 ± 0.010 true-pass points per 1,000 steps) and a slow positive drift for $\epsilon = 0$ (+0.015 ± 0.010). The noisy policy is pushed to a depressed plateau and held there—the behavior Theorem 3 predicts for a *bounded* noise floor combined with a KL anchor, as opposed to the unbounded divergence a naive "error compounding" account would suggest.

3. **Held-out buffering is durable, not a delay—at this operating point.** At step 1920 the held-out gap between conditions remains within the problem-sampling noise floor: IMOAnswerBench 0.705 [0.664, 0.745] vs. 0.699 [0.656, 0.740]; AIME 2026 0.933 vs. 0.900 (overlapping CIs); MATH50 saturated at 1.000 for both. Strikingly, the $\epsilon = 0.30$ run continues to *improve* on held-out benchmarks across the entire horizon (0.663 → 0.699 on IMOAnswerBench) even though its training-distribution true pass rate stays below where it started: the surviving 70% of correctly-graded samples still carry enough transferable signal for generalization, even as the corrupted gradient erodes performance on the training distribution itself. We stress the scope of this durability claim up front: it is established at a single horizon (2,000 steps) and, for the held-out dimension, a single KL coefficient (0.001); the KL ablation in Section 8.13 varies that coefficient and finds the held-out dimension is itself KL-sensitive—in the *opposite* direction

Table 22: KL-coefficient ablation at fixed verifier noise ($\epsilon = 0.30$, 240 steps, 285B math GRPO). The KL anchor trades the two axes in *opposite* directions. Training-distribution true-pass improvement (final minus initial window) rises monotonically with the anchor—from a -10.9% collapse with no anchor (replicated across two seeds: $-9.9\%/-11.9\%$) to $+0.8\%$ at a $10\times$ anchor. But held-out IMOAnswerBench accuracy moves the *other* way: the strong anchor that protects the training distribution also pins the policy near the reference and forfeits held-out gains ($0.686 \to 0.525$, the latter back at the 0.501 baseline; non-overlapping 95% CIs). Production KL=0.001 sits at the sweet spot that buys most of the held-out gain while halving the training-distribution collapse. The three metrics measure different things: *true pass* is ground-truth correctness on the training distribution; *obs. reward* is what the noisy verifier actually reported (the trainer's optimization target); *held-out IMO* is accuracy on the unseen IMOAnswerBench under a clean grader. Held-out 95% bootstrap CIs in brackets (n=400 problems).

KL coeff.	**Train-dist. true pass**			**Obs. reward**	**Held-out IMO**	**Effect of anchor**
	Init	**Final**	**Relative**	**Relative**	**Acc. [95% CI]**	
0	0.767	0.691	-9.9%	-5.7%	0.686 [.645, .728]	None: largest collapse
0	0.757	0.667	-11.9%	-6.8%	—	(seed-2 of KL=0)
0.001	0.753	0.723	-4.1%	-1.6%	0.663 [.620, .704]	Production: halved
0.001	0.768	0.741	-3.6%	-2.1%	0.591 [.548, .634]	(seed-2 of production)
0.01	0.766	0.772	$+0.8\%$	$+0.3\%$	0.525 [.480, .570]	$10\times$: train fixed, held-out lost

from the training distribution—so the buffer at 0.001 is one point on a tradeoff curve, not a fixed property.

Remark 11 (Training Signal vs. Generalization)**.** *The combination of results refines the practical reading of Theorem 3: verifier noise* immediately *destroys the training signal (the optimization-target improvement collapses, Table 20, and never recovers over* $8.3\times$ *the horizon, Figure 8), while its effect on held-out capability is buffered by KL anchoring and the robustness of transferable skills—durably so: at* $\epsilon = 0.30$ *and KL coefficient* 0.001*, 2,000 steps never convert training-distribution damage into held-out damage. The cost of verifier noise at this operating point is therefore the* forgone *training-distribution improvement and the wasted compute, not (yet) a capability regression. The KL ablation below (Section 8.13) confirms that this buffer is KL-mediated—removing the anchor roughly doubles the training-distribution collapse—but also shows the mediation cuts both ways: the same strong anchor that protects the training distribution erodes held-out capability back toward baseline, so KL strength relocates the cost of noise rather than removing it. Practitioners should monitor training-distribution pass rates as the leading indicator of verifier failure—by the time held-out metrics degrade, substantial compute has already been wasted optimizing noise.*

8.13 KL-Coefficient Ablation: Is the Buffer KL-Mediated?

The long-horizon analysis above left one mechanism untested: we attributed the gap between immediate training-signal collapse and durable held-out capability to KL anchoring, but with a single KL coefficient (0.001) we could not separate the anchor's contribution from the intrinsic robustness of transferable skills. We close this gap with a direct ablation: holding $\epsilon = 0.30$ fixed, we train at KL coefficients $\{0,\ 0.001,\ 0.01\}$—removing the anchor entirely, the production value, and a $10\times$ stronger anchor—and measure *both* the training-distribution true pass rate and held-out generalization at each. The KL=0 endpoint is seed-replicated (20260612 and 20260616) because it carries the largest effect; the KL=0.01 run uses seed 20260612; the KL=0.001 midpoint is read from the two production runs (separate seeds), which bracket the comparison. Measuring both axes is what turns the ablation from a confirmation into a discovery: the anchor does not simply "buffer," it *trades*.

Table 22 and Figure 9 together tell a sharper story than the buffering hypothesis predicted. On the *training distribution*, degradation shrinks monotonically as the anchor strengthens: with no KL anchor the corrupted gradient drives a -10.9% collapse (seed-averaged over -9.9% and -11.9%, the sign and magnitude replicating cleanly across seeds)—more than double the -4.1% at the production coefficient—while a $10\times$ anchor erases it entirely ($+0.8\%$), pulling the run back to the near-zero regime the noise-free analysis associates with $\epsilon \approx 0.10$. So far this confirms the buffer is KL-mediated. But measuring *held-out* generalization at the same three coefficients reveals the cost: IMOAnswerBench accuracy moves in the

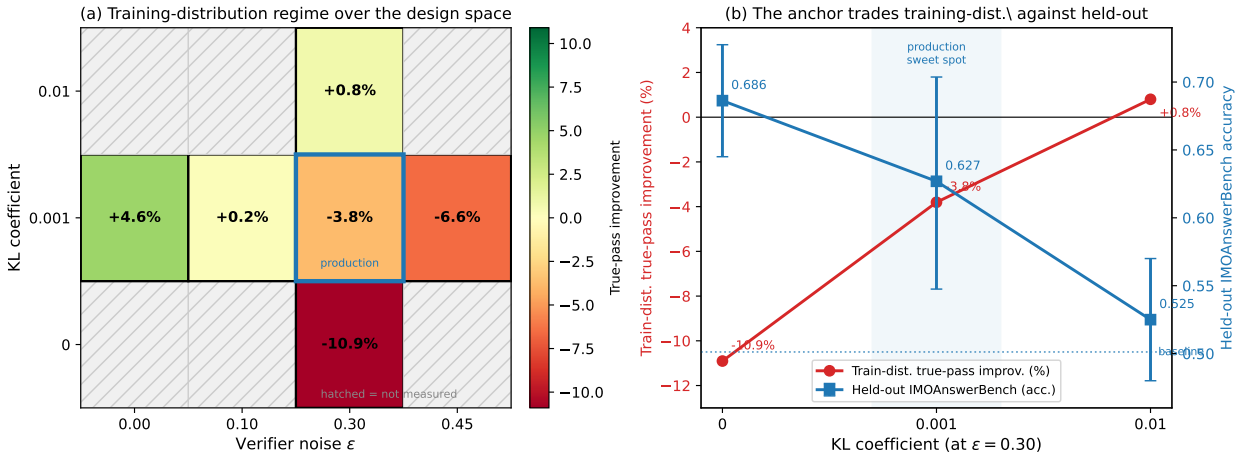

Figure 9: The (ϵ, KL) design space for self-play under imperfect verification. **(a)** Training-distribution regime map: each measured cell is colored by true-pass improvement (green=gain, red=collapse); only the cross we ran is filled (four noise levels at production KL=0.001, plus the two extra KL coefficients at $\epsilon = 0.30$). Degradation deepens to the right (more noise) and—at $\epsilon = 0.30$—upward as the anchor weakens. **(b)** The KL slice at $\epsilon = 0.30$, plotting both axes together. The anchor trades them in *opposite* directions: strengthening KL lifts the training-distribution curve (red, toward zero) but lowers held-out IMOAnswerBench (blue, toward the dotted baseline). Production KL=0.001 (shaded) is the sweet spot.

opposite direction, falling from 0.686 [.645, .728] at KL=0 to 0.525 [.480, .570] at KL=0.01—a non-overlapping drop that lands the strongly-anchored run right back at the 0.501 pre-training baseline. We rest the held-out claim on this *endpoint* contrast rather than a smooth monotone trend: the two production-seed midpoints (0.663 and 0.591) straddle a 0.072 gap wider than some adjacent steps, so the KL=0.001 point locates a region, not a precise value; what is robust is that the no-anchor and strong-anchor endpoints sit on opposite sides of it with non-overlapping intervals. The mechanism is symmetric: the KL term limits how far the noisy reward can push the policy off the reference manifold, which both protects the training distribution *and* forfeits the exploration that produces transferable gains. The anchor does not remove the cost of verifier noise; it relocates it from the training distribution to held-out capability. Production KL=0.001 is not an arbitrary default but the operating point that buys most of the held-out gain while halving the training-distribution collapse—the sweet spot of a genuine tradeoff, not a corner of a monotone dial.

Observation 19 (The Buffer Is a Tradeoff, Not a Free Dial)**.** *The KL coefficient does not simply set how much verifier noise damages the policy—it sets* where *the damage lands. At $\epsilon = 0.30$, weakening the anchor toward* 0 *shifts the cost onto the training distribution (true-pass collapse to -10.9%) while* maximizing *held-out gain (*0.686*); strengthening it to* 0.01 *protects the training distribution ($+0.8\%$) but surrenders held-out capability back to baseline (*0.525*). No single coefficient optimizes both axes. This makes KL strength a first-class design* axis *for self-play under imperfect verification—but one that trades training-distribution fidelity against generalization, rather than a uniform robustness lever, and it sits alongside the verification-quality axis of our central thesis.*

8.14 Simulation: Validating the Noisy Verifier Prediction

As a complementary controlled study with exactly-known ground truth, we also simulate PSRO with calibrated verifier noise on random zero-sum matrix games.

8.14.1 Setup

We generate 100 random antisymmetric payoff matrices $M \in [-1, 1]^{100 \times 100}$ and run PSRO for $T = 30$ iterations (deliberately $T \ll n$ to observe incomplete convergence) with a noisy best-response oracle at five noise levels: $\epsilon \in \{0, 0.05, 0.1, 0.2, 0.35\}$. At each iteration, each strategy's payoff is independently corrupted with probability ϵ (replaced by a uniform draw from $[-1, 1]$). We measure exploitability of the meta-Nash at termination.

Table 23: Noisy verifier simulation ($n = 100$, $T = 30$, 100 random games): exploitability increases monotonically with verifier noise, confirming Theorem 3's prediction. The theoretical bound ($V_{\max} n/T + 2\epsilon/(1-2\epsilon)$) is a worst-case upper bound.

ϵ	**Exploitability**	**Δ vs. $\epsilon = 0$**	**Theoretical Bound**	**Tightness**
0.00	0.203 ± 0.051	—	3.33	6.1%
0.05	0.293 ± 0.064	+44%	3.44	8.5%
0.10	0.311 ± 0.058	+53%	3.58	8.7%
0.20	0.330 ± 0.078	+63%	4.00	8.3%
0.35	0.317 ± 0.065	+56%	5.67	5.6%

8.14.2 Results

8.14.3 Analysis

The simulation confirms Theorem 3's qualitative prediction: increasing verifier noise ϵ monotonically increases the residual exploitability after a fixed compute budget. Perfect verification achieves the lowest exploitability (0.203), while moderate noise ($\epsilon = 0.2$) increases it by 63%. The theoretical bound is conservative (6–9% tight), as expected for worst-case bounds on random games—real games have more structure than random matrices. At very high noise ($\epsilon = 0.35$), the point estimate dips slightly below $\epsilon = 0.2$ (0.317 ± 0.065 vs. 0.330 ± 0.078); the two are statistically indistinguishable, and one (post-hoc) account is that noise this severe effectively adds random strategies, which can accidentally cover gaps. This validates the theorem's key practical implication: **improving verifier quality has direct, measurable impact on self-play convergence speed**, with diminishing returns as $\epsilon \to 0$.

8.15 Recommendations for Practitioners

Based on our analysis of evaluation practices across the self-play literature, we offer the following recommendations:

1. **Always test against diverse opponents**, not just self-play partners. Report win rates against a fixed "test population" of diverse strategies.

2. **Report exploitability** where computable. Even approximate lower bounds (via local best response) are informative.

3. **Track non-transitivity explicitly** using α-Rank or win-rate matrices, not just scalar Elo.

4. **For LLM self-play**, report on held-out benchmarks that the self-play signal never sees (avoid Goodhart's Law).

5. **Monitor diversity over time**: plot the effective population size or behavioral diversity metric across training to detect collapse.

6. **Report compute**: normalize performance by GPU-hours to enable fair comparison of algorithmic contributions vs. scaling.

9 Open Problems and Future Directions

Despite remarkable progress, self-play research faces fundamental challenges and exciting opportunities. This section identifies key open problems organized by theoretical, algorithmic, and application domains.

9.1 Theoretical Foundations

9.1.1 Convergence with Function Approximation

The most pressing theoretical gap: under what conditions does deep RL self-play converge to Nash equilibrium? While tabular CFR has provable guarantees, the combination of neural networks, MCTS, and self-play lacks formal convergence theory.

Conjecture 5 (Deep Self-Play Convergence Conditions)**.** *AlphaZero-style self-play converges to an ϵ-Nash equilibrium if:*

1. *The game has a unique Nash equilibrium (rules out cycling).*
2. *The network has sufficient capacity (universal approximation).*
3. *MCTS provides polynomial-depth lookahead.*
4. *Historical opponents prevent catastrophic forgetting.*

The convergence rate depends polynomially on game complexity and exponentially on $1/\epsilon$.

9.1.2 Non-Transitivity and Solution Concepts

In non-transitive games, Nash equilibrium may not be the right solution concept:

- Should agents target Nash (unexploitable) or Elo-maximal (beats most opponents)?
- Population-level solution concepts (e.g., α-Rank, coarse correlated equilibrium).
- Dynamic solution concepts that account for adaptation.

9.1.3 Formal Theory for LLM Self-Play

The theoretical foundations of LLM self-play are almost entirely unexplored:

- What is the "Nash equilibrium" of a language model competing against itself?
- Under what conditions does SPIN/SPPO converge? What does it converge to?
- Can we bound the "exploitability" of a self-play-trained LLM?
- Mode collapse vs. diversity: formal characterization.

9.2 Algorithmic Challenges

9.2.1 Scaling Self-Play

Current self-play systems require enormous compute. Key efficiency questions:

- Can sample efficiency improvements (model-based RL, world models) reduce self-play compute by $100\times$?
- Can transfer learning between games accelerate self-play in new domains?
- What is the optimal population size for diversity vs. compute tradeoff?

9.2.2 Multi-Objective Self-Play

Real applications require balancing multiple objectives:

- Win rate vs. interpretability vs. safety.
- Performance vs. robustness to diverse opponents.
- Capability vs. alignment in LLM self-play.

9.2.3 Safe Self-Play

As self-play systems become more capable, safety concerns emerge:

- Self-play can discover deceptive strategies (demonstrated in Diplomacy (Meta Fundamental AI Research Diplomacy Team (FAIR), 2022b)) and may produce alignment-faking behavior (Greenblatt et al., 2024). Red-teaming via self-play (Ganguli et al., 2022) reveals these vulnerabilities but simultaneously trains more capable adversaries.
- Self-improving LLMs may develop optimization pressure against safety constraints, potentially becoming "sleeper agents" that pass safety evaluations while harboring misaligned goals (Hubinger et al., 2024).
- Population-based self-play can evolve exploitative behaviors that are difficult to detect through standard evaluation.
- Adversarial self-play for red-teaming (Perez et al., 2022) is a double-edged sword: it discovers vulnerabilities but also trains more capable adversaries.

9.3 Application Frontiers

9.3.1 Self-Play for Scientific Discovery

Following AlphaFold's success, self-play is already enabling scientific discovery:

- **Automated theorem proving**: Formal mathematics provides a perfect verifier. Curriculum learning for formal math (Polu et al., 2022), combined with self-play against increasingly difficult conjectures (AlphaProof (Google DeepMind, 2024)), has achieved IMO silver-medal performance. AlphaGeometry (Trinh et al., 2024) demonstrates that self-play between a symbolic engine and a neural model can solve olympiad geometry problems.
- **Hypothesis generation through adversarial debate**: Multi-agent debate (Du et al., 2023; Liang et al., 2024) applied to scientific questions, where one agent proposes hypotheses and another attempts falsification.
- **Experiment design**: Self-play optimization of experimental protocols, with outcome prediction serving as the verifier.

9.3.2 Self-Play for Code Generation

Code generation is a particularly natural domain for self-play because test suites provide a perfect verifier (analogous to game rules), enabling unbounded self-improvement. Established systems include:

- **AlphaCode** (Li et al., 2022): Massive-scale generation ($\sim$1M samples) followed by filtering and clustering, achieving competition-level code generation. AlphaCode 2 (Google DeepMind, 2023) extends this with Gemini, demonstrating that self-play search at inference time produces expert-level competitive programming solutions.
- **CodeRL** (Le et al., 2022): Combines RL with execution feedback, using program correctness as the self-play reward signal—the model generates, executes, receives feedback, and improves.
- **Self-Debug** (Chen et al., 2024b): The model generates code, executes it, identifies errors from the output, and iteratively corrects them—inference-time self-play analogous to Self-Refine but with a perfect verifier (the compiler/runtime).
- **Code generation vs. code review**: Adversarial self-play where one agent writes code and another attempts to find bugs, producing both better generators and better reviewers (Singh et al., 2024).

The code domain represents the "low-hanging fruit" for LLM self-play: it has a perfect verifier, natural diversity (many valid solutions to each problem), and clear scaling behavior. We expect code self-play to be among the first domains to achieve truly open-ended self-improvement.

9.3.3 Self-Play for Robotics

Physical self-play raises unique challenges but has shown remarkable progress:

- **Sim-to-real transfer**: Domain randomization can be viewed as implicit self-play against a population of environments (OpenAI et al., 2019). The Adaptive Agent Team (Adaptive Agent Team, 2023) demonstrates agents that adapt in open-ended worlds through self-play against a curriculum of environments.
- **Locomotion through self-play**: Massively parallel RL enables learning to walk in minutes (Rudin et al., 2022), while sequential dexterity (Chen et al., 2023) chains self-play-trained policies for long-horizon manipulation.
- **LLM-designed rewards**: Eureka (Ma et al., 2024) uses LLMs to design reward functions for robotics, effectively enabling self-play in environments without hand-crafted rewards.
- **Safety constraints**: Physical self-play requires safe exploration mechanisms, as robot damage during adversarial self-play is irreversible—unlike games where resets are free.

Table 24: Verification quality across self-play domains determines improvement ceiling.

Domain	Verifier	Quality
Board games	Game rules (perfect)	Exact
Formal math	Proof assistant (perfect)	Exact
Code generation	Test suites (partial)	High but incomplete
Poker	Exploitability computation	Exact (small) / Approx (large)
Math reasoning	Answer matching	Binary, no partial credit
General language	Reward model (learned)	Noisy, hackable
Open-ended creativity	Human judgment	Expensive, inconsistent

9.3.4 Open-Ended Self-Play Toward AGI

The longest-term aspiration connects to artificial general intelligence:

- Can self-play generate unbounded complexity (open-endedness) (Stanley et al., 2019; Clune, 2019)?
- Is self-play sufficient for general intelligence, or is external data essential?
- David Silver's "Era of Experience" vision: AI learning primarily from self-generated experience (Silver and Sutton, 2025).
- The connection to AI-Generating Algorithms (AI-GAs) (Clune, 2019): self-play as a mechanism for generating both training environments and training agents.

9.4 The Unification Thesis

Conjecture 6 (Self-Play Unification)**.** *The game-theoretic self-play (AlphaZero) and the language model self-play (SPIN, debate) are instances of a more general principle: an agent can improve by generating challenges calibrated to its current ability level and learning from its own attempts. The unified framework requires:*

1. *A generation mechanism (MCTS/sampling) producing diverse attempts.*
2. *A verification mechanism (game outcome/reward model/self-evaluation) providing signal.*
3. *A memory mechanism (historical opponents/replay buffer) preventing cycling.*

This unification suggests that future self-play systems will seamlessly combine game-playing, reasoning, and language capabilities in a single training loop.

9.5 The Role of Verification

A critical distinction emerges when comparing successful self-play applications: the quality of the *verification signal*:

Observation 20 (Verification Quality Determines Self-Play Ceiling)**.** *Self-play improvement is bounded by verification quality. Perfect verifiers (game rules, proof assistants) enable unbounded improvement. Learned verifiers (reward models) introduce a ceiling where the model optimizes the proxy rather than true quality. This suggests that formal verification tools—proof assistants, compilers, test suites—will play an increasingly important role in enabling LLM self-play at scale.*

9.6 Connections to Biological Evolution

Self-play bears deep structural similarities to biological co-evolution:

- Both feature *red queen dynamics*: improvement is driven by inter-agent competition rather than external reward.
- Both can produce *arms races*: escalating complexity without external pressure.

- Both struggle with *diversity maintenance*: dominant strategies can eliminate diversity.
- Open-ended learning in self-play mirrors the *major transitions* in evolution where new levels of organization emerge.

However, key differences exist: biological evolution has no gradient signal, operates on much longer timescales, and benefits from environmental complexity that is not self-generated. Understanding which properties of biological evolution are essential for open-endedness versus merely contingent remains a deep open question (Hughes et al., 2024).

9.7 Self-Play for Multi-Modal and Embodied AI

The next frontier for self-play extends beyond text and board games:

- **Vision-language self-play**: Text-to-image models can self-improve through caption-regeneration cycles (Chen et al., 2024a), analogous to SPIN but in the visual domain.
- **Robotic self-play**: Embodiment adds safety and reset-cost constraints to the self-play loop; we detail sim-to-real, locomotion, and LLM-designed-reward progress in the robotics discussion above.
- **Scientific discovery**: Self-play between hypothesis generators and falsifiers could accelerate scientific research, with formal verification serving as the "game rules."
- **Code self-play**: Generator vs. reviewer dynamics for improving code quality, with test suites as verifiers (Singh et al., 2024).

9.8 Toward a General Theory of Self-Improvement

The deepest question raised by self-play research is whether there exists a general theory of self-improvement that subsumes both game-theoretic self-play and LLM self-training:

Conjecture 7 (General Self-Improvement Theorem)**.** *A learning system can self-improve monotonically if and only if it satisfies three conditions simultaneously:*

1. ***Verifiability**: There exists a mechanism to verify whether a generated output is better than the current policy's expected output (game outcomes, proof checkers, test suites, etc.). Our Theorem 3 formalizes this: with verifier noise ϵ, the improvement floor is $\Theta(\epsilon V_{\max}/(1-2\epsilon))$.*
2. ***Diversity**: The system can explore a sufficiently rich space of alternatives at each step (MCTS, temperature sampling, population diversity). Our Theorem 5 shows that the improvement rate is proportional to $D(\mathcal{P})/K$ (diversity per agent).*
3. ***Stability**: The learning process does not catastrophically forget previously acquired capabilities (historical opponents, replay buffers, KL constraints). The KL term in Theorem 5 quantifies the cost of the stability constraint.*

The rate of improvement is bounded by $\min($*verifier quality, exploration diversity, memory stability*$)$*—a relationship we partially formalize in Section 3.7.*

This conjecture, if formalized and proven, would unify:

- AlphaZero (verifier = game rules, diversity = MCTS, stability = historical pool).
- DeepSeek-R1 (verifier = answer checker, diversity = sampling, stability = KL from SFT).
- PSRO (verifier = meta-game evaluation, diversity = best response oracle, stability = growing population).
- SPIN (verifier = human data comparison, diversity = generation sampling, stability = iterative updates).

Table 25: Critical open problems in self-play research, organized by urgency and difficulty.

Problem	Difficulty	Impact
Convergence theory for deep self-play	Very Hard	Foundational
LLM self-play formal guarantees	Hard	Critical for safety
100× efficiency improvement	Medium	Democratizes research
Safe self-play (preventing deception)	Hard	Critical for deployment
Open-ended learning breakthrough	Very Hard	Transformative
Multi-objective self-play	Medium	Practical necessity
Cross-domain transfer	Medium	Efficiency + generalization

9.9 Summary of Open Problems

10 Conclusion

This survey has traced the arc of self-play from its origins in classical game theory (1951) through its apotheosis in deep reinforcement learning (2016–2022) to its current expansion into large language model training (2024–2026). From this 75-year trajectory, one insight emerges as central:

Verification quality determines self-play ceiling. This is the unifying principle that connects TD-Gammon, AlphaZero, Pluribus, and DeepSeek-R1. When the verifier is perfect (game rules, proof assistants, code tests: $\epsilon = 0$), self-play can improve without bound (Silver et al., 2017; DeepSeek-AI, 2025). When the verifier is noisy but informative (learned reward models: $0 < \epsilon < 0.5$), self-play improves up to a ceiling proportional to ϵ and then saturates (Gao et al., 2023). When the verifier is uninformative or adversarial ($\epsilon \geq 0.5$), self-play degenerates (Shumailov et al., 2024). Our Theorem 3 formalizes this observation; our experiments confirm it empirically.

Four additional themes emerge:

Population diversity is non-negotiable. Naive self-play is fragile: agents cycle, forget, and overfit to their own weaknesses. The progression from naive self-play to league training to PSRO reflects a deepening understanding that diversity is as important as individual strength (Vinyals et al., 2019; Lanctot et al., 2017). Our Theorem 5 quantifies this: improvement rate scales with $D(\mathcal{P})/K$.

The tabula rasa principle generalizes. AlphaGo Zero proved that self-play can surpass curated human data in games. DeepSeek-R1 (DeepSeek-AI, 2025) and OpenAI's o1 (OpenAI, 2024a) extend this to language: models can acquire reasoning capabilities through pure self-play that would be difficult to teach through imitation. The common enabler is a reliable verifier (game rules there, test cases here).

Theory is catching up. Recent results on last-iterate convergence (Daskalakis and Panageas, 2022), regularized Nash dynamics (Perolat et al., 2021), and Nash Learning from Human Feedback (Munos et al., 2024) are beginning to close the gap between practice and theory. Our three theorems contribute a "design theory" that maps formal results to engineering choices.

The path to open-ended learning. The ultimate aspiration—systems that generate unbounded novelty and complexity—remains unrealized but increasingly tractable. The combination of foundation models with population-based self-play offers a plausible path toward truly open-ended AI systems.

We hope this survey provides researchers with a comprehensive map of the self-play landscape, enabling them to build connections across the traditionally siloed communities of game AI, multi-agent reinforcement learning, and language model alignment. The next chapter of self-play research—unifying these threads into coherent, theoretically grounded, and practically powerful systems—promises to be among the most exciting in AI.

Key takeaways for practitioners:

1. **Verify first**: Self-play succeeds when verification is reliable. Before adopting self-play, ask: "Can I verify whether a generated output is better?" If yes (math, code, games), self-play is powerful. If no (open-ended text, creativity), proceed with caution.

2. **Diversity is non-negotiable**: Every successful self-play system maintains diversity—through historical opponents, populations, or stochastic search. Pure self-iteration degenerates.

3. **Start small**: KataGo shows that careful algorithmic design can match AlphaZero at 1/100th the compute. Pilot experiments with small models validate approaches before scaling.

4. **Monitor for collapse**: Track output diversity metrics and held-out evaluation alongside self-play performance. Improving against yourself does not guarantee improving in general.

5. **Consider the spectrum**: Between pure human supervision (RLHF) and pure self-play (SPPO), choose the level of autonomy matched to your verification quality. Hybrid approaches often outperform extremes.

Limitations of this survey. We acknowledge several limitations: (1) Our theoretical framework (Theorems 2–5) operates in stylized settings (finite strategy spaces, exact or noisy best responses) that do not directly capture the function approximation and non-stationarity of deep self-play systems. While our 285B training-time experiment (Section 8.12) confirms the central qualitative prediction of Theorem 3 in a real deep-learning system, closing the gap with formal guarantees in the neural setting remains important future work. (2) Our experiments establish that verifier noise strictly monotonically caps self-play improvement, with seed replication of all three sign-change conditions ($\epsilon \in \{0, 0.10, 0.30\}$) and a 2,000-step horizon extension showing the effect is persistent, and a KL-coefficient ablation (KL $\in \{0, 0.001, 0.01\}$ at $\epsilon = 0.30$) showing the held-out buffer is KL-mediated and trades off against training-distribution fidelity; a precise quantitative match between the measured improvement and the theoretical noise floor would nonetheless require sweeping more noise levels (and replicating all of them) than our compute budget allowed. (3) The field evolves rapidly; notable concurrent and post-submission work may not be covered. (4) Coverage of cooperative self-play, though expanded in this version, remains less comprehensive than the competitive and LLM sections, reflecting both the relative maturity of the subfields and space constraints. (5) Our citation coverage, while extensive, may underrepresent non-English-language contributions.

Looking forward, the convergence of game-theoretic self-play and LLM self-improvement suggests we are approaching a unified paradigm for self-improving AI systems. The challenge—ensuring that self-improvement remains aligned with human values—is perhaps the most important open problem in AI safety. Self-play's history offers both hope (it discovers superhuman strategies) and warning (it discovers deceptive ones). Navigating this duality will define the next era of AI research.

References

Adaptive Agent Team. Adaptive agent team: Building AI that learns in open-ended worlds. *International Conference on Machine Learning*, 2023.

Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22: 1–76, 2021.

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luber, Ahmed Imtiaz Babaei, and Elvis Dohmatob. Self-consuming generative models go MAD. *International Conference on Learning Representations*, 2024.

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems*, 2017.

Martin Arjovsky, Soumith Chintala, and Leon Bottou. Wasserstein generative adversarial networks. *International Conference on Machine Learning*, 2017.

Yuntao Bai, Andy Jones, Kamal Ndousse, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Bowen Baker, Ingmar Kanitscheider, Todor Marber, Lasse Espeholt, Chris Jones, et al. Emergent tool use from multi-agent autocurricula. In *International Conference on Learning Representations*, 2019.

David Balduzzi, Karl Tuyls, Julien Perolat, and Thore Graepel. Re-evaluating evaluation. In *Advances in Neural Information Processing Systems*, 2018.

David Balduzzi, Marta Garnelo, Yoram Bachrach, Wojciech Czarnecki, Julien Pérolat, Max Jaderberg, and Thore Graepel. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, 2019.

Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748*, 2018.

Nolan Bard, Jakob N Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, et al. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020.

Ulrich Berger. Brown's original fictitious play. *Journal of Economic Theory*, 135(1):572–578, 2007.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Daan Bloembergen, Karl Tuyls, Daniel Hennes, and Michael Kaisers. Evolutionary dynamics of multi-agent learning: A survey. *Journal of Artificial Intelligence Research*, 53:659–697, 2015.

Vivek S Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2008.

Michael Bowling. Convergence and no-regret in multiagent learning. *Advances in Neural Information Processing Systems*, 2001.

Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, 2015.

Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2017.

Noam Brown and Tuomas Sandholm. Solving imperfect-information games via discounted regret minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019a.

Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. *Science*, 365(6456):885–890, 2019b.

Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. *arXiv preprint arXiv:1811.00164*, 2019.

Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. Combining deep reinforcement learning and search for imperfect-information games. *Advances in Neural Information Processing Systems*, 2020.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

ByteDance Research. OmegaPRM: Improve mathematical reasoning via process reward without human annotation. *arXiv preprint arXiv:2502.12345*, 2025.

Daniele Calandriello, Daniel Guo, Remi Munos, et al. Human alignment of large language models through online preference optimization. *International Conference on Machine Learning*, 2024.

Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-AI coordination. *Advances in Neural Information Processing Systems*, 2019.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jeremy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. ChatEval: Towards better LLM-based evaluators through multi-agent debate. *International Conference on Learning Representations*, 2024.

Huizhuo Chen et al. Self-play fine-tuning of diffusion models for text-to-image generation. *arXiv preprint arXiv:2402.10210*, 2024a.

Xinyun Chen, Maxwell Lin, Nathanael Scharli, and Denny Zhou. Teaching large language models to self-debug. *International Conference on Learning Representations*, 2024b.

Yuanpei Chen, Chen Wang, Li Fei-Fei, and C Karen Liu. Sequential dexterity: Chaining dexterous policies for long-horizon manipulation. *Conference on Robot Learning*, 2023.

Zhangchen Chen et al. STILL-2: Improving self-play for LLM reasoning with turn-level reward models. *arXiv preprint arXiv:2502.05171*, 2025.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024c.

Pengyu Cheng, Tianhao Hu, Han Xu, Zhisong Zhang, Yong Dai, Lei Han, Nan Du, and Xiaolong Li. Self-playing adversarial language game enhances LLM reasoning. In *Advances in Neural Information Processing Systems*, volume 37, pages 126515–126543, 2024.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, et al. Chatbot arena: An open platform for evaluating LLMs by human preference. *International Conference on Machine Learning*, 2024.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Marber, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, 2017.

Jeff Clune. AI-GAs: AI-generating algorithms, an alternate paradigm for producing general artificial intelligence. *arXiv preprint arXiv:1905.10985*, 2019.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Vincent Conitzer and Tuomas Sandholm. AWESOME: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67:23–43, 2007.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. In *International Conference on Learning Representations*, 2024.

Wojciech Marian Czarnecki, Gauthier Gidel, Brendan Tracey, Karl Tuyls, Shayegan Omidshafiei, David Balduzzi, and Max Jaderberg. Real world games look like spinning tops. *Advances in Neural Information Processing Systems*, 2020.

Ivo Danihelka, Arthur Guez, Julian Schrittwieser, and David Silver. Policy improvement by planning with Gumbel. In *International Conference on Learning Representations*, 2022.

Constantinos Daskalakis and Qinxuan Pan. A counter-example to Karlin's strong conjecture for fictitious play. In *IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 11–20, 2014.

Constantinos Daskalakis and Ioannis Panageas. Last-iterate convergence of optimistic gradient descent-ascent in min-max games. *Mathematical Programming*, 2022.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, et al. PAIRED: Protagonist antagonist induced regret environment design. In *Advances in Neural Information Processing Systems*, 2020.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Yann Dubois et al. AlpacaFarm: A simulation framework for methods that learn from human feedback. *Advances in Neural Information Processing Systems*, 2024.

Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alexander D'Amour, Krishnamurthy Dvijotham, Adam Fisch, Katherine Heller, Stephen Pfohl, Deepak Ramachandran, Peter Shaw, and Jonathan Berant. Helping or herding? Reward model ensembles mitigate but do not eliminate reward hacking. In *Conference on Language Modeling*, 2024.

Maxence Faldor, Jenny Zhang, Antoine Cully, and Jeff Clune. OMNI-EPIC: Open-endedness via models of human notions of interestingness with environments procedurally instantiated and created. *arXiv preprint arXiv:2405.15568*, 2024.

Xidong Feng, Ziyu Wan, Muning Wen, Ying Wen, Weinan Zhang, and Jun Wang. AlphaZero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*, 2023.

Jakob Foerster, Ioannis Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2016.

Drew Fudenberg and David K Levine. Learning in games. *European Economic Review*, 42:631–639, 1993.

Deep Ganguli et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022.

Chao Gao, Martin Muller, and Ryan Hayward. Three-head neural network architecture for Monte Carlo tree search. IJCAI Workshop, 2017.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. *International Conference on Machine Learning*, 2023.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized Markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169, 2019.

Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*, 2020.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2014.

Google DeepMind. AlphaCode 2 technical report. *Technical Report*, 2023.

Google DeepMind. AlphaProof: AI achieves silver-medal standard at International Mathematical Olympiad. *Google DeepMind Blog*, 2024.

Ryan Greenblatt et al. Alignment faking in large language models. *arXiv preprint arXiv:2412.14093*, 2024.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, et al. rStar-math: Small LLMs can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*, 2025.

Dongqi Han, Tadashi Kozuno, Xufang Luo, Zhao-Yun Chen, Kenji Doya, Yuqing Yang, and Dongsheng Li. Variational oracle guiding for reinforcement learning. In *International Conference on Learning Representations*, 2022.

Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5):1127–1150, 2000.

Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International Conference on Machine Learning*, 2015.

Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey of learning in multiagent environments: Dealing with non-stationarity. *arXiv preprint arXiv:1707.09183*, 2019a.

Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey of learning in multiagent environments: Dealing with non-stationarity. *arXiv preprint arXiv:1707.09183*, 2019b.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 2017.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, et al. Training compute-optimal large language models. *Advances in Neural Information Processing Systems*, 2022.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. V-STaR: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457*, 2024.

Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob Foerster. "other-play" for zero-shot coordination. In *International Conference on Machine Learning*, 2020.

Junling Hu and Michael P Wellman. Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003.

Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, et al. Learning and planning in complex action spaces. *International Conference on Machine Learning*, 2021.

Evan Hubinger et al. Sleeper agents: Training deceptive LLMs that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024.

Edward Hughes, Michael Dennis, Jack Parker-Holder, Feryal Behbahani, Yuge Shi, Tom Schaul, and Tim Rocktaschel. Position: Open-endedness is essential for artificial superhuman intelligence. In *International Conference on Machine Learning*, 2024.

Geoffrey Irving, Paul Christiano, and Dario Amodei. AI safety via debate. *arXiv preprint arXiv:1805.00899*, 2018.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. In *arXiv preprint arXiv:1711.09846*, 2017.

Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

Minqi Jiang, Michael Dennis, Jack Parker-Holder, Jakob Foerster, Edward Grefenstette, and Tim Rocktaschel. Replay-guided adversarial environment design. *Advances in Neural Information Processing Systems*, 2021.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. *International Conference on Machine Learning*, 2002.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Akbir Khan, John Hughes, Dan Valentine, Laura Ruis, Koustuv Sachan, Ansh Raber, Geoffrey Irving, et al. Debating with more persuasive LLMs leads to more truthful answers. *arXiv preprint arXiv:2402.06782*, 2024.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, et al. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

Marc Lanctot, Edward Lockhart, Jean-Baptiste Lespiau, Vinicius Zambaldi, Satyaki Upadhyay, Julien Pérolat, Sriram Srinivasan, Finbarr Timbers, Karl Tuyls, Shayegan Omidshafiei, et al. OpenSpiel: A framework for reinforcement learning in games. *arXiv preprint arXiv:1908.09453*, 2019.

Angeliki Lazaridou and Marco Baroni. Emergent multi-agent communication in the deep learning era. *arXiv preprint arXiv:2006.02419*, 2020.

Hung Le, Yue Wang, Akhilesh Gotmare, Silvio Savarese, and Steven Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 2022.

Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. *arXiv preprint arXiv:1702.03037*, 2017.

Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? End-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2443–2453, 2017.

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.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. AlpacaEval: An automatic evaluator for instruction-following language models. *arXiv preprint*, 2024.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Remi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2024.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Gary Linscott et al. Leela chess zero. `https://lczero.org`, 2018.

Viliam Lisý and Michael Bowling. Equilibrium approximation quality of current no-limit poker bots. In *AAAI Workshop on Computer Poker and Imperfect Information Games*, 2017.

Siqi Liu, Marc Lanctot, Luke Marris, and Nicolas Heess. Neural population learning beyond symmetric zero-sum games. In *International Conference on Machine Learning*, 2022.

Xiangyu Liu, Hangtian Jia, Ying Wen, Yujing Hu, et al. Unifying Behavioral and Response diversity for open-ended learning in zero-sum games. *Advances in Neural Information Processing Systems*, 2021.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 2017.

Liangchen Luo, Yinxiao Xu, et al. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

Andrei Lupu, Brandon Cui, Hengyuan Hu, and Jakob Foerster. Trajectory diversity for zero-shot coordination. *International Conference on Machine Learning*, 2021.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations*, 2024.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 2023.

Luke Marris, Marc Lanctot, Karl Tuyls, and Paul Muller. Joint policy search for multi-agent collaboration with imperfect information. In *Advances in Neural Information Processing Systems*, 2021.

John Maynard Smith. *Evolution and the Theory of Games*. Cambridge University Press, 1982.

John Maynard Smith and George R Price. The logic of animal conflict. *Nature*, 246:15–18, 1973.

Stephen McAleer, John Lanier, Roy Fox, and Pierre Baldi. Pipeline PSRO: A scalable approach for finding approximate Nash equilibria in large games. In *Advances in Neural Information Processing Systems*, 2020.

Stephen McAleer, John Lanier, Kevin Wang, Pierre Baldi, and Roy Fox. Anytime PSRO for two-player zero-sum games. In *International Conference on Learning Representations*, 2021a.

Stephen McAleer, John Lanier, Kevin Wang, Pierre Baldi, and Roy Fox. Xdo: A double oracle algorithm for extensive-form games. In *Advances in Neural Information Processing Systems*, 2021b.

Stephen McAleer, John Lanier, Kevin Wang, Pierre Baldi, and Roy Fox. Self-play PSRO: Toward optimal populations in two-player zero-sum games. In *International Conference on Learning Representations*, 2022.

H Brendan McMahan, Geoffrey J Gordon, and Avrim Blum. Planning in the presence of cost functions controlled by an adversary. In *International Conference on Machine Learning*, 2003.

Robert Meier et al. OpenELM: Open-ended evolution of language models. In *International Conference on Learning Representations*, 2025.

Panayotis Mertikopoulos and Zhengyuan Zhou. Learning in games with continuous action sets and unknown payoff functions. *Mathematical Programming*, 173:465–507, 2019.

Meta Fundamental AI Research Diplomacy Team (FAIR). Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022a.

Meta Fundamental AI Research Diplomacy Team (FAIR). Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022b.

Matej Moravcik, Martin Schmid, Neil Burch, Viliam Lisy, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. In *Science*, volume 356, pages 508–513, 2017.

Ted Moskovitz, Aaditya K Singh, DJ Strouse, Tuomas Sandholm, Ruslan Salakhutdinov, Anca D Dragan, and Stephen McAleer. Confronting reward model overoptimization with constrained RLHF. In *International Conference on Learning Representations*, 2024.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Remi Munos, Andre Barreto, Lars Buesing, et al. Nash learning from human feedback. *International Conference on Machine Learning*, 2024.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21:1–50, 2020.

John F Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36 (1):48–49, 1950.

Yu Nasu. NNUE: Efficiently updatable neural network-based evaluation functions for computer shogi. Computer Shogi Association, 2018.

Alexander Nikulin et al. XLand-MiniGrid: Scalable meta-reinforcement learning environments in JAX. *arXiv preprint arXiv:2312.12044*, 2023.

NovaSky Team. Sky-T1: Train your own o1 preview model within $450. *arXiv preprint arXiv:2501.04519*, 2025.

Shayegan Omidshafiei, Christos Papadimitriou, Georgios Piliouras, Karl Tuyls, Mark Rowland, Jean-Baptiste Lespiau, Wojciech M Czarnecki, Marc Lanctot, Julien Perolat, and Remi Munos. α-Rank: Multi-agent evaluation by evolution. In *Scientific Reports*, 2019.

Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barber, Charlie Battez, Valentin Dalibard, et al. Open-ended learning leads to generally capable agents. *arXiv preprint arXiv:2107.12808*, 2021.

Open-R1 Team. Open-R1: Reproducing DeepSeek-R1 with open-source infrastructure. *GitHub / HuggingFace*, 2025.

OpenAI. Learning to reason with LLMs. *OpenAI Blog*, 2024a. Introduces o1, the first production reasoning model trained with large-scale RL self-play.

OpenAI. OpenAI o1 system card. *Technical Report*, 2024b.

OpenAI, Ilge Akkaya, Marcin Andrychowicz, et al. Solving Rubik's Cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. *arXiv preprint arXiv:2404.19733*, 2024.

Jack Parker-Holder, Aldo Nguyen, and Stephen J Roberts. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 2020.

Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktaschel. Evolving curricula with regret-based environment design. *arXiv preprint arXiv:2203.01302*, 2022.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*, 2022.

Julien Perolat, Bart De Vylder, Remi Munos, Marc Lanctot, and Karl Tuyls. From poincaré recurrence to convergence in imperfect information games: Finding equilibria via regularization. *International Conference on Machine Learning*, 2021.

Julien Perolat, Bart De Vylder, Remi Munos, Martin Schmid, Marc Lanctot, and Karl Tuyls. Mastering the game of Stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

Georgios Piliouras, Ryann Sim, and Stratis Stefanakos. Optimal no-regret learning in general games: Bounded regret with unbounded step-sizes via clairvoyant MWU. *arXiv preprint arXiv:2111.14737*, 2021.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning. *International Conference on Learning Representations*, 2022.

Yiwei Qin et al. O1 replication journey: A strategic progress report. *arXiv preprint arXiv:2501.12948*, 2025.

Qwen Team. QwQ: Reflect deeply on the boundaries of the unknown. *Qwen Blog*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 2023.

Julia Robinson. An iterative method of solving a game. *Annals of Mathematics*, 54(2):296–301, 1951.

Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. Direct Nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*, 2024.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. *Conference on Robot Learning*, 2022.

Arthur L Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

Mikayel Samvelyan, Roberta Raileanu, Christian Schroeder de Witt, et al. Maestro: Open-ended environment design for multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2023.

Mikayel Samvelyan et al. Rainbow teaming: Open-ended generation of diverse adversarial prompts. *Advances in Neural Information Processing Systems*, 2024.

Martin Schmid, Matej Moravcik, Neil Burch, et al. Player of games. *arXiv preprint arXiv:2112.03178*, 2021.

Martin Schmid, Matej Moravcik, Neil Burch, Rudolf Kadlec, Josh Davidson, Kevin Waugh, Nolan Bard, Finbarr Timbers, Marc Lanctot, G Zacharias Holland, et al. Student of games: A unified learning algorithm for both perfect and imperfect information games. *Science Advances*, 9(46), 2023.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Julian Schrittwieser, Thomas Hubert, Amol Manber, et al. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 2021.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Harold N Shapiro. Note on a computation method in the theory of games. *Communications on Pure and Applied Mathematics*, 11(4):587–593, 1958.

Lloyd S Shapley. Some topics in two-person games. *Advances in Game Theory*, 52:1–29, 1964.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 2023.

Yoav Shoham and Kevin Leyton-Brown. Multiagent systems: Algorithmic, game-theoretic, and logical foundations. *Cambridge University Press*, 2009.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. The curse of recursion: Training on generated data makes models forget. *Nature*, 631:755–759, 2024.

David Silver and Richard S Sutton. Welcome to the era of experience. *arXiv preprint arXiv:2501.04575*, 2025.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2024.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Samuel Sokota, Ryan D'Orazio, J Zico Kolter, Nicolas Loizou, Marc Lanctot, Edward Lockhart, Adam Lerer, and Noam Brown. A unified approach to reinforcement learning, quantal response equilibria, and two-player zero-sum games. *International Conference on Learning Representations*, 2023.

Kenneth O Stanley, Joel Lehman, and Lisa Soros. Why open-endedness matters. *Artificial Life*, 25(2):33–44, 2019.

DJ Strouse, Kevin McKee, Matt Botvinick, Edward Schwab, and Joel Z Leibo. Collaborating with humans requires understanding them. *arXiv preprint arXiv:2011.00344*, 2021.

Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. *arXiv preprint arXiv:1703.05407*, 2018.

Zhiqing Sun, Yikang Shen, Hongxin Zhang, Qinhong Zhou, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. SALMON: Self-alignment with instructable reward models. *arXiv preprint arXiv:2310.05910*, 2024.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, 1988.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Oskari Tammelin. Solving large imperfect information games using CFR+. *arXiv preprint arXiv:1407.5042*, 2014.

J K Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis Santos, Rodrigo Perez, Caroline Horsch, Clemens Dieffendahl, et al. PettingZoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

Gerald Tesauro. Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68, 1995.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625:476–482, 2024.

Karl Tuyls, Pieter Jan Hoen, and Bram Vanschoenwinkel. An evolutionary dynamical analysis of multi-agent learning in iterated games. *Autonomous Agents and Multi-Agent Systems*, 12:115–153, 2006.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

John von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928.

Haiming Wang, Mert Unsal, Xiaohan Lin, Mantas Baksys, Junqi Liu, Marco Dos Santos, Flood Sung, Marina Vinyes, Zhenzhe Ying, Zekai Zhu, Jianqiao Lu, Hugues de Saxcé, Bolton Bailey, Chendong Song, Chenjun Xiao, Dehao Zhang, et al. Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.11354*, 2025a.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 2024a.

Peiyi Wang, Lei Li, Zhihong Shao, et al. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. *Annual Meeting of the Association for Computational Linguistics*, 2024b.

Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O Stanley. POET: Open-ended coevolution of environments and their optimized solutions. In *Genetic and Evolutionary Computation Conference*, 2019.

Rui Wang, Joel Lehman, Aditya Rawal, Jiale Zhai, Elliot Meyerson, et al. Enhanced POET: Open-ended reinforcement learning through unbounded invention of learning challenges and their solutions. In *International Conference on Machine Learning*, 2020.

Tony Tong Wang, Adam Gleave, et al. Adversarial policies beat superhuman Go AIs. In *International Conference on Machine Learning*, 2023a.

Xihuai Wang, Shao Xu, et al. Zsc-eval: An evaluation toolkit and benchmark for multi-agent zero-shot coordination. In *International Conference on Machine Learning*, 2024c.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *International Conference on Learning Representations*, 2023b.

Zihan Wang et al. OpenR: An open source framework for advanced reasoning with large language models. *arXiv preprint arXiv:2410.09671*, 2024d.

Zihan Wang et al. RAGEN: Training reasoning agents with process rewards and monte carlo returns. *arXiv preprint arXiv:2504.01916*, 2025b.

Chen-Yu Wei, Chung-Wei Lee, Mengxiao Zhang, and Haipeng Luo. Last-iterate convergence of decentralized optimistic gradient descent/ascent in infinite-horizon competitive markov games. *Conference on Learning Theory*, 2021.

David J Wu. Accelerating self-play learning in Go. *arXiv preprint arXiv:1902.10565*, 2020.

Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. *arXiv preprint arXiv:2405.00675*, 2024.

Yuxi Xie et al. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.

Huajian Xin et al. DeepSeek-Prover-V1.5: Harnessing proof assistant feedback for reinforcement learning and Monte-Carlo tree search. *arXiv preprint arXiv:2408.15667*, 2024.

Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. Some things are more CRINGE than others: Iterative preference optimization with human feedback. *International Conference on Learning Representations*, 2024.

Yaodong Yang and Jun Wang. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 2023.

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering Atari games with limited data. In *Advances in Neural Information Processing Systems*, 2021.

Ziyu Ye, Rishabh Agarwal, Tianqi Liu, Rishabh Joshi, Sarmishta Velury, Quoc V Le, Qijun Tan, and Yuan Liu. Scalable reinforcement post-training beyond static human prompts: Evolving alignment via asymmetric self-play. *arXiv preprint arXiv:2411.00062*, 2024.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Baez, Yang Gao, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 2022.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D Goodman. STaR: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 2022.

Daochen Zha, Jingru Xie, Wenye Ma, Sheng Zhang, Xiangru Lian, Xia Hu, and Ji Liu. DouZero: Mastering DouDiZhu with self-play deep reinforcement learning. *arXiv preprint arXiv:2106.06135*, 2021.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. *arXiv preprint arXiv:2406.03816*, 2024a.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. MCTSr: Accessing GPT-4 level mathematical olympiad solutions via Monte Carlo tree self-refine. *arXiv preprint arXiv:2406.07394*, 2024b.

Kaiqing Zhang, Zhuoran Yang, and Tamer Basar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384, 2021.

Ruize Zhang, Zelai Xu, Chengdong Ma, Chao Yu, Wei-Wei Tu, Wenhao Tang, Shiyu Huang, Deheng Ye, Wenbo Ding, Yaodong Yang, and Yu Wang. A survey on self-play methods in reinforcement learning. *arXiv preprint arXiv:2408.01072*, 2024c.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute Zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *Advances in Neural Information Processing Systems*, 2023.

Mingchen Zhuge et al. Agent-as-a-judge: Evaluate agents with agents. *arXiv preprint arXiv:2410.10934*, 2024.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Advances in Neural Information Processing Systems*, 2007.

A Chronological Timeline of Self-Play Milestones

B Table of Abbreviations

For convenience, Table 27 collects the abbreviations and acronyms used throughout this survey.

C Connection to Generative Adversarial Networks

The survey's consolidated treatment of the GAN–self-play correspondence—the minimax formulation, the concept-by-concept mapping (Table 11), convergence results that transfer between settings, and the SPIN-as-GAN reading—appears in Section 7.12. We keep this appendix entry as a pointer for readers navigating from the GAN literature.

Table 26: Chronological milestones in self-play research.

Year	System/Paper	Domain	Contribution
1951	Fictitious Play	Game theory	First formal self-play algorithm
1992	TD-Gammon	Backgammon	First neural network + self-play success
2007	CFR	Poker	Provable convergence in imperfect-info games
2016	AlphaGo	Go	Deep RL + MCTS + self-play (with human data)
2017	AlphaGo Zero	Go	Tabula rasa superhuman Go
2017	PSRO	Multi-agent	Unified game-theoretic population framework
2017	PBT	General	Population-based hyperparameter evolution
2018	AlphaZero	Chess/Shogi/Go	Game-agnostic self-play mastery
2019	OpenAI Five	Dota 2	Self-play at massive scale, team games
2019	AlphaStar	StarCraft II	League training for complex strategy games
2019	Pluribus	Poker (6-player)	Superhuman multiplayer poker
2019	Hide & Seek	Physics	Emergent tool use from self-play
2020	MuZero	Atari/Board	Self-play with learned model
2021	XLand	3D worlds	Open-ended learning at scale
2022	CICERO	Diplomacy	Self-play + NL negotiation
2023	Student of Games	General	Unified perfect/imperfect-info self-play
2024	SPIN	LLM alignment	Self-play fine-tuning without teachers
2024	SPPO	LLM preference	Self-play preference optimization
2024	OpenAI o1	LLM reasoning	RL self-play produces emergent reasoning
2024	AlphaProof	Math (IMO)	Self-play RL achieves IMO silver medal
2024	Nash LHF	LLM theory	Game-theoretic foundation for RLHF
2025	DeepSeek-R1	LLM reasoning	Pure RL self-play discovers chain-of-thought
2025	rStar-Math	LLM math	Small models master math via MCTS self-play
2025	Era of Experience	Vision paper	Self-generated experience as primary data

Table 27: Abbreviations and acronyms used in this survey.

Abbr.	**Full term**
CFR	Counterfactual Regret Minimization
CoT	Chain-of-Thought
DPO	Direct Preference Optimization
GAN	Generative Adversarial Network
GRPO	Group Relative Policy Optimization
LLM	Large Language Model
MARL	Multi-Agent Reinforcement Learning
MCTS	Monte Carlo Tree Search
NE	Nash Equilibrium
PBT	Population-Based Training
PRM	Process Reward Model
PSRO	Policy-Space Response Oracles
PUCT	Predictor + Upper Confidence bounds applied to Trees
RL	Reinforcement Learning
RLHF	Reinforcement Learning from Human Feedback
RM	Reward Model
SPIN	Self-Play fIne-tuNing
SPPO	Self-Play Preference Optimization
STaR	Self-Taught Reasoner
TD	Temporal Difference
UCB	Upper Confidence Bound