Reinforcement Learning for Competitive *Magic: The Gathering* Gameplay

Alex Thaikanov[1] **Blair Probst**[2] **Casey Tzao**[3]

Dana Evelyn[4]
[1]MIT [2]Stanford University [3]CMU [4]University of Oxford
{alexr,blairs,caseyt,danae}@example.edu

Abstract

Magic: The Gathering is a long-horizon, stochastic, partially observable, adversarial domain with a combinatorial action space and rules that rival programming languages in expressivity. These characteristics make *Magic: The Gathering* a compelling but underexplored benchmark for modern reinforcement learning (RL). We present a framework for self-play RL in *Magic: The Gathering* that unifies three components: (i) a rules-faithful simulator with hierarchical action abstraction for the priority/stack system, (ii) a model-based self-play learner that combines policy/value learning with search, and (iii) a *meta-game* outer loop for deck optimization co-evolving against a population of opponents. We detail design principles for state and action encoding, curriculum shaping, opponent sampling, and evaluation protocols that account for the volatile metagame. We also outline a results template for reporting ELO, win rates across matchups, generalization to unseen decks, and ablations on action abstraction, search budget, and population diversity. Our work positions *Magic: The Gathering* as a rigorous testbed for planning under uncertainty and large discrete action spaces, and provides a blueprint that can be adopted or extended by future studies.

1 Introduction

Learning to act under uncertainty in multi-agent settings remains a central challenge in RL. Recent successes in games have showcased the benefits of scale, search, and self-play: Go and classical board games with AlphaZero-*style* training (15; 16), real-time strategy in StarCraft (17), and imperfect-information domains in poker (13; 3; 4). These milestones highlight robust mechanisms—self-play, population-based training, and planning—for synthesizing strong decision policies.

Magic: The Gathering introduces distinct obstacles beyond those domains. First, the *rules engine* features priority passing, a LIFO stack of spells/abilities, and replacement/triggered effects, interacting across phases and steps; the official comprehensive rules form a living specification updated frequently (18). Second, *partial observability* and *stochasticity* arise from hidden hands and randomized draws. Third, the *action space* is vast and combinatorial: even a single decision node (e.g., declare attackers/blocks, respond with multiple stackable effects) can induce a combinatorial explosion. Finally, the metagame is dynamic as new sets rotate in and card pools change, reminiscent of non-stationary environments.

This paper contributes a practical, rules-aware RL framework for competitive *Magic: The Gathering* . We (1) formalize a hierarchical action abstraction consistent with the stack and priority system; (2) propose a model-based self-play learner that uses search within a learned latent model in the spirit of MuZero (14) while accommodating partial observability (11); and (3) integrate a population-based,

Preprint. Under review.

co-evolutionary outer loop that jointly optimizes decks and policies, extending ideas from large-scale self-play and population training (2; 10). We position evaluation practices that reflect *Magic: The Gathering* 's realities and outline an experimental scaffold others can replicate.

2 Related Work

Self-play and search. Self-play with value/policy networks accelerated by tree search underpins breakthroughs in Go and board games (15; 16). Model-based variants such as MuZerolearn an implicit dynamics model to plan in latent space (14). Monte Carlo Tree Search (MCTS) with UCT (12) remains a key planning substrate.

Imperfect information and multi-agent RL. Poker systems demonstrate principled handling of hidden information via counterfactual regret minimization and deep search (13; 3; 4). Population-based methods mitigate overfitting to single opponents and reduce exploitability (2; 10). Game-theoretic RL frameworks for general-sum settings (9) inform opponent modeling (7).

Large discrete action spaces. *Magic: The Gathering* 's decision points often involve combinatorial sets (e.g., subsets of attackers/blocks). Methods for large discrete action spaces such as action embeddings and candidate pruning offer scalable approximations (8; 6).

Card games and *Magic: The Gathering* . Prior work has examined computational properties of *Magic: The Gathering* , including undecidability/Turing completeness under idealized constructions (5). While collectible card game AI literature is richer in *Hearthstone* than *Magic: The Gathering* , many insights on deck-building and simulation-based planning transfer. Our focus is competitive *Magic: The Gathering* with a rules-faithful simulator and an RL pipeline purpose-built for the stack/priority mechanics.

3 Methodology

3.1 Environment and State Representation

We implement (or interface with) a rules engine that executes phases/steps, priority passing, stack resolution, and state-based actions as per the comprehensive rules (18). Observations are *partially observable*: each player sees public zones (battlefield, stack, graveyards, exiled cards with public information) and their own private zones (hand). The opponent's hand and libraries are hidden.

States are encoded as:

- **Global scalars:** life totals, poison counters, turn number, phase/step, storm count, available mana, mulligan history.
- **Permanents set:** a permutation-invariant multiset encoder over battlefield objects (card identity, types/subtypes, counters, tapped status, summoning sickness, ongoing continuous effects). We use learned embeddings for card identities with type-aware features.
- **Stack sequence:** ordered encoding of spells/abilities on the stack, including targets and modes.
- **Hand summary:** private hand as a set with embeddings, plus coarse distributional features (cmc histogram, color profile).
- **History:** a truncated action-observation history window; a learned recurrent state (GRU) summarizes long-range context to address partial observability (11).

3.2 Action Space and Hierarchical Abstraction

The raw legal action set at each priority window includes: pass priority; cast spells (with mode/alternative costs/targets); activate abilities; declare attackers/blockers; select targets and modes; perform special actions. We introduce a two-level abstraction:

1. **High-level intents** (*macro-actions*): *"Develop board"*, *"Apply pressure"*, *"Hold up interaction"*, *"Combo execution"*, *"Stabilize"*. Each intent maps to a constrained subspace of primitives.

2. **Grammar-constrained primitives**: parameterized actions (e.g., `Cast(CardID, Mode, Targets, CostChoice)`, `Activate(PermanentID, AbilityID, Targets)`, `DeclareAttackers(Subset)`). For combinatorial choices (attack/block subsets), we use candidate generation and prune with an intent-conditioned scorer (6; 8).

Priority windows (including responses) are handled via a *stack-aware controller* that interleaves policy proposals with on-policy search, enabling timely interaction during the opponent's turn.

3.3 Learning: Model-Based Self-Play with Search

We adopt an actor-critic backbone augmented with a learned dynamics model as in MuZero (14). The system comprises:

- **Representation** h_θ: encodes observations and recurrent hidden state into latent s_0.
- **Dynamics** g_θ: rolls forward $(s, a) \mapsto (s', r)$ in latent space.
- **Prediction** f_θ: outputs policy logits over abstract actions and a value estimate.

Planning uses MCTS with UCT (12), operating in latent space, with action priors from f_θ. Partial observability is addressed by maintaining a belief-augmented latent via recurrence and by sampling opponent hands from a learned generative model calibrated to decklists and draw rules.

Self-play population. To prevent overfitting and promote robustness, agents train in a *population* (2; 10): a mixture of current and historical checkpoints plus distinct deck archetypes. Opponent sampling follows a decaying mixture of uniform and performance-weighted schedules; periodic *fictitious play* style evaluation approximates exploitability trends.

3.4 Deck Optimization Outer Loop

Decks (60-card maindeck with sideboard for best-of-three) are optimized in an outer loop that treats the inner RL policy as a black box. We explore:

- **Evolutionary search**: mutate decklists via card swaps restricted to legal formats; fitness is estimated by cross-play win rate.
- **Bayesian optimization**: represent decklists via card-frequency embeddings; optimize acquisition against a noisy win-rate oracle.
- **Population co-evolution**: jointly evolve decks and policies, encouraging metagame diversity and preventing collapse.

To maintain legality and rules fidelity we validate each candidate against the rules engine and ban/format lists (18).

3.5 Curriculum, Shaping, and Training Details

We stage complexity with a **curriculum** (1): start with creature-centric subsets (reduced triggers), then introduce stack-based interaction and nuanced timing windows, and finally full card sets. **Reward shaping** augments terminal rewards with dense proxies (life differential, card advantage, board control) while preserving correct credit assignment through potential-based shaping.

Optimization. Policies train from reanalyzed search targets; we use prioritized replay, entropy regularization, and PBT for hyperparameters (8; 10). Large discrete action support follows candidate pruning plus a pointer network for target selection (6).

4 Results

4.1 Evaluation Protocol

We recommend:

Table 1: Overall performance (fill with your results).

Agent / Setting	ELO	Win Rate vs. Baseline (%)	Games
Full (Ours)			
No Search (Policy Only)			
No Population (Single Opponent)			
No Deck Outer Loop			

1. **ELO vs. baselines:** Cross-play among population checkpoints, scripted heuristic bots, and ablations.
2. **Matchup matrix:** Win rates across archetypes (e.g., Aggro, Midrange, Control, Combo) with best-of-three sideboarding.
3. **Generalization:** Performance against *unseen* decks and after set rotations.
4. **Data efficiency:** Strength as a function of environment steps and search budget.

4.2 Quantitative Placeholders

4.3 Ablations

Report the marginal contribution of (i) hierarchical action abstraction, (ii) search budget, (iii) opponent population diversity, and (iv) deck co-evolution. Use common seeds and identical training budgets to isolate effects.

4.4 Qualitative Analysis

Include case studies of stack interactions (e.g., baiting removal, counterspell wars), combat puzzles (profitable attack/blocks under tricks), and combo execution timing. Provide playout traces and value-policy heatmaps for insight.

5 Discussion

What makes *Magic: The Gathering* hard for RL? Long horizons, branching factor spikes at timing windows, and the need to reason about hidden information and future priority exchanges. The rules system enables interactions akin to program execution; indeed, idealized *Magic: The Gathering* has been shown Turing complete (5).

Search vs. policy. Planning mitigates myopic errors around stack timing and target selection, while learned policies capture style and heuristics. Balancing compute between online search and offline policy improvement mirrors lessons from AlphaZeroand MuZero (16; 14).

Population and metagame. A diverse opponent population is crucial to avoid exploitability spirals observed in self-play. Co-evolving decks alters the loss landscape, preventing brittle specializations and better reflecting real competitive dynamics (2).

Ethical and practical considerations. Releasing strong agents may affect online ladders and the secondary card economy. We encourage responsible disclosure, stress-testing against exploits, and rate-limiters in public-facing bots.

6 Conclusion

We have outlined a comprehensive framework for competitive *Magic: The Gathering* via self-play RL that marries a rules-faithful simulator, hierarchical action abstraction, model-based planning, and population-driven deck optimization. Beyond card games, the techniques speak to decision-making in partially observable environments with large discrete action spaces and rich domain rules. We hope this blueprint accelerates reproducible research and positions *Magic: The Gathering* as a challenging benchmark that rewards progress in planning, opponent modeling, and meta-level adaptation.

Acknowledgments

[Optional: acknowledge compute grants, lab colleagues, or community resources.]

References

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 41–48, 2009.

[2] Christopher Berner, Greg Brockman, Brooke Chan, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

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

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

[5] Alex Churchill, Stella Biderman, and Austin Herrick. Magic: The gathering is turing-complete. arXiv preprint arXiv:1904.09828, 2019.

[6] Gabriel Dulac-Arnold, Daniel Evans, Hado van Hasselt, et al. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*, 2015.

[7] He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. *arXiv preprint arXiv:1609.05559*, 2016.

[8] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2018.

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

[10] Max Jaderberg, Victor Dalibard, Simon Osindero, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

[11] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

[12] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning*, pages 282–293, 2006.

[13] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in no-limit poker. *Science*, 356(6337):508–513, 2017.

[14] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588:604–609, 2020.

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

[16] David Silver, Thomas Hubert, Julian Schrittwieser, et al. A general reinforcement learning algorithm that masters chess, shogi and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[17] Oriol Vinyals, Igor Babuschkin, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350–354, 2019.

[18] Wizards of the Coast. Magic: The gathering comprehensive rules. `https://magic.wizards.com/en/rules`, 2025. Accessed: insert date of access.