Reinforcement Learning for Fog of War Chess with Action Space Pruning

Kuba Hashemian Leon Liu Sandeep Sethuraman

May 2026

Extended Abstract

Motivation. Fog of War Chess is a partially observable variant of chess in which players can only observe squares visible to their pieces. Unlike standard chess, agents must reason under uncertainty and make decisions without the full game state. This creates two major challenges: uncertain information and a large effective search space. While traditional search algorithms can evaluate many candidate moves, the computational cost grows rapidly under partial observability. We investigate whether a learned action prior can identify strategic and also information-gathering actions, so that search can focus computation on a smaller subset of candidates moves while preserving/improving playing strength.

Method. We propose a two-stage framework consisting of a learned action prior and a restricted search method. The action prior receives a partial state and predicts a score for each legal action. Rather than searching over all legal moves, we retain the top-k actions according to a learned prior and perform lookahead search within this restricted action set. The action prior learns which moves deserve further computation, while search determines which of those candidates should ultimately be played.

Implementation. Board states are represented as a $14 \times 8 \times 8$ tensor encoding all visible information, processed by a CNN–LSTM policy trained with PPO self-play. We study two ways of making the prior information-aware. The first bakes information into the prior during training, adding a reward for revealing hidden squares (Information Prior + Search). The second leaves the policy prior unchanged and instead adds an information bonus at candidate-selection time during inference (Information-Bonus Search). We compare these against a search-free Dense PPO agent, a standard Policy Prior + Search agent, and a Random agent, all under identical search depth and candidate-set size.

Results. In round-robin self-play, the strongest agent was Information-Bonus Search, which augments the standard policy prior with an inference-time information bonus, reaching a 41.7% win rate and edging out Policy Prior + Search (39.7%). Notably, training the prior with information rewards (Information Prior + Search, 33.3%) underperformed the plain policy prior, and both search agents far exceeded the search-free Dense PPO (14.0%) and Random (6.0%). These results indicate that information is most useful when applied at *selection time* rather than encoded into the training objective.

Discussion. Information-gathering actions are central to Fog of War Chess, where reducing uncertainty can be as valuable as immediate tactical gains. However, our experiments show that *how* the information signal is introduced matters: an inference-time information bonus improved play, whereas folding the same signal into the training reward degraded it by distorting the prior's ranking of tactical moves.

Conclusion. We demonstrate that learned action priors provide an effective mechanism for restricted search under partial observability, with search-guided agents dramatically outperforming a search-free policy. We further find that information-awareness is best added at inference time rather than during training. These results highlight the potential of combining reinforcement learning with learned search guidance in environments where uncertainty makes exhaustive search impractical.

Abstract

Fog of War (FoW) Chess is a partially observable variant of chess in which players must make decisions using incomplete information about the board state. This uncertainty creates a challenging search problem, as agents must balance tactical play with information gathering. We investigate whether learned action priors can improve search by identifying promising actions before search is performed, and whether explicitly valuing information acquisition helps. Using a CNN–LSTM policy trained with PPO self-play, we restrict a depth-bounded search to the top-k actions proposed by the prior, and study two ways of injecting an information signal: encoding it into the prior via a training reward (Information Prior + Search), or adding it as an inference-time bonus during candidate selection (Information-Bonus Search). In a round-robin self-play tournament, Information-Bonus Search achieves the highest win rate (41.7%), narrowly surpassing Policy Prior + Search (39.7%), while both decisively beat the search-free Dense PPO agent (14.0%) and Random play (6.0%). Counterintuitively, encoding the information signal directly into the training reward (Information Prior + Search, 33.3%) underperformed the plain policy prior. These results show that search guidance substantially improves play under partial observability, and that information-awareness is most effective when applied at selection time rather than baked into the training objective.

1 Introduction

Recent advances in reinforcement learning have enabled agents to achieve superhuman performance in a variety of games, including Go, Chess, and StarCraft. Systems such as AlphaGo and AlphaZero demonstrated that deep neural networks combined with search techniques can learn highly sophisticated strategies directly from self-play. However, these successes largely rely on environments that provide complete information about the game state. In many real-world applications, agents must instead operate under uncertainty and partial observability.

Fog of War Chess presents a particularly challenging example of such an environment. Unlike traditional chess, players can only observe portions of the board that are visible to their own pieces. Opponent pieces outside the visible region remain hidden, forcing players to make decisions with incomplete information. As a result, the environment can be modeled as a partially observable Markov decision process (POMDP), where optimal decisions depend not only on current observations but also on reasoning about hidden information.

In addition to partial observability, Fog of War Chess presents a large action-space challenge. At each turn, dozens of legal actions may be available, and lookahead search over all of them quickly becomes expensive as depth grows. While only a small subset of these moves are strategically meaningful, an agent must still decide which actions are worth deeper evaluation.

To address this, we adopt a two-stage framework that separates candidate generation from move evaluation. A learned action prior, trained with PPO self-play, scores the legal moves; we retain only the top-k candidates and run a depth-bounded search within this restricted set. This focuses limited search computation on promising actions rather than exploring the full action space. Because reducing uncertainty is itself valuable under partial observability, we further investigate how to make the prior *information-aware*, comparing an information reward applied during training against an information bonus applied at inference-time selection.

Our contributions are as follows:

- We formulate Fog of War Chess as a reinforcement learning benchmark with large action spaces and partial observability.
- We develop a learned action prior that restricts a depth-bounded search to a small top-k candidate set.

- We study two mechanisms for incorporating an information-gain signal — a training-time reward versus an inference-time selection bonus — and analyze their effect on playing strength.
- We evaluate all agents in a round-robin self-play tournament, reporting win rate and search cost, and ablate the candidate-set size k.

2 Related Work

Policy priors and search. Modern game-playing systems often use learned policies to guide search. Building on AlphaGo's combination of deep networks and tree search [8], AlphaZero combines self-play reinforcement learning with Monte Carlo Tree Search (MCTS) [1], using a neural policy prior to focus search on promising actions rather than expanding all moves uniformly [9]. However, these systems are designed for perfect-information games such as chess, shogi, and Go, where the full board state is observable. Our work adapts the policy-prior idea to Fog of War Chess, where the agent must guide search using only a partial observation of the board.

Action masking and action-space reduction. Action masking is commonly used in reinforcement learning to prevent agents from selecting invalid actions. For example, invalid action masking has been shown to be especially important when the full discrete action space is large but only a subset of actions is legal in each state [3]. MaskablePPO applies this idea to PPO by masking illegal actions during policy optimization. Our method extends this idea beyond legality: rather than only removing invalid moves, we learn a prior that ranks legal moves by strategic and informational value, allowing search to focus on a smaller candidate set.

Imperfect-information games. Fog of War Chess shares core challenges with other imperfect-information games such as poker, where agents must reason over hidden state. Counterfactual Regret Minimization (CFR) is a standard method for solving large imperfect-information games and has been successfully applied to poker abstractions [10], and systems such as DeepStack later achieved expert-level play in heads-up no-limit poker by combining such reasoning with learned value functions [5]. However, these methods typically reason explicitly over information sets and require extensive game-tree structure. In contrast, our approach uses reinforcement learning and learned action priors to handle partial observability in a spatial board-game environment.

Belief states and recurrent policies. A common approach to partial observability is to use memory-based models that summarize observation history. Deep Recurrent Q-Networks (DRQN) replace part of a DQN with an LSTM, allowing the agent to integrate information over time in partially observable environments [2]. Our model similarly uses history-aware representations through a CNN-LSTM policy to infer hidden enemy pieces from past observations. However, our main contribution is not only representing hidden information, but using learned priors and information bonuses to decide which actions deserve further search.

3 Environment

Fog of War Chess modifies standard chess by restricting player observations to squares visible from friendly pieces. As a result, players must make decisions without complete knowledge of opponent positions, making the task a partially observable Markov decision process (POMDP) [4].

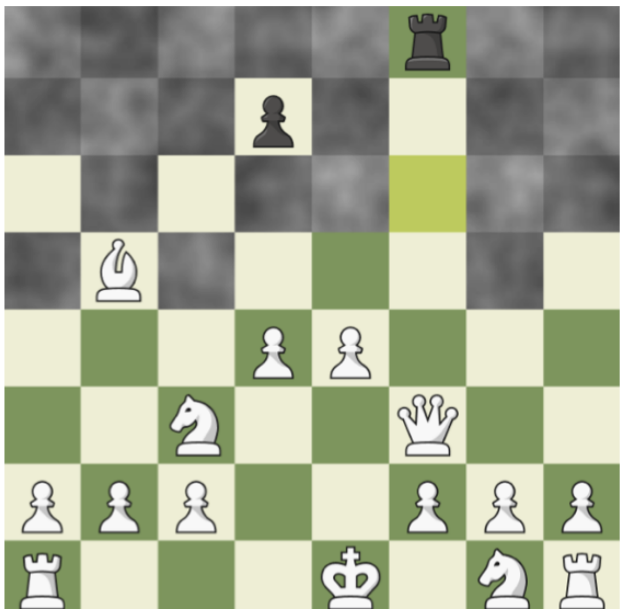

Figure 1: A Fog of War Chess position from the agent's perspective. Squares occupied or attacked by friendly pieces are visible (light/green), while the remaining squares are hidden (gray), so the opponent's pieces are only partially observed.

Each observation consists of:

- Visible friendly pieces.
- Visible opponent pieces.
- Known empty squares.
- Hidden or unobserved regions.

The environment is represented as a tensor encoding piece identities and visibility information. Hidden squares are represented separately from known empty squares to preserve uncertainty information.

We adopt true Fog of War rules: there is no notion of check or checkmate, a player may move into or ignore an attack on their king, and the game is won by *capturing* the opponent's king. Moves are therefore pseudo-legal, and the action space is encoded as $\text{from\_square} \times 64 + \text{to\_square}$, giving a discrete space of 4096 actions (promotions default to queen). Games are truncated at 200 half-moves and scored as draws, as are king-versus-king endings.

Rewards are based primarily on game outcomes: capturing the enemy king yields a terminal reward of $+1$, and the self-play formulation makes the reward zero-sum so that the opponent's gains are penalized symmetrically. To address the sparsity of king captures, we add dense material shaping (a small reward proportional to material captured or promoted). Where indicated, an additional information-gain signal rewards moves that reveal previously hidden squares.

4 Methodology

Figure 2 gives an overview of our approach. A CNN–LSTM actor–critic, trained with PPO self-play, produces an action prior over legal moves. At inference, this prior (optionally augmented with an information bonus) ranks moves; the top-k candidates are passed to a depth-bounded search that selects the final move. We describe each component below.

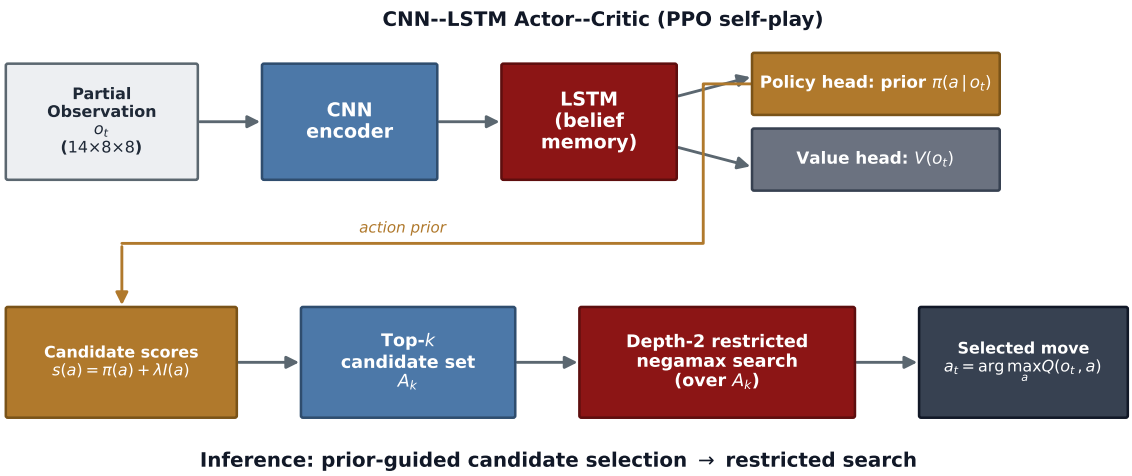

Figure 2: Architecture and decision pipeline. Top: the CNN–LSTM actor–critic that outputs the action prior $\pi(a \mid o_t)$ and value $V(o_t)$. Bottom: at inference, the prior scores legal moves, the top-k form the candidate set A_k, and a depth-2 restricted search selects the move played.

4.1 Baseline PPO Agent

Our baseline agent is trained using Proximal Policy Optimization (PPO) [7], a policy-gradient reinforcement learning algorithm that serves as the foundation for all methods explored in this project. The agent receives a partial observation of the board and directly predicts a distribution over legal actions without performing any explicit search.

Each board state is represented as a $14 \times 8 \times 8$ tensor encoding visible friendly pieces, visible enemy pieces, square visibility information, and side-to-move features. This representation is processed by a CNN-LSTM architecture that captures both spatial relationships on the board and temporal information from previous observations. The recurrent component is particularly important in Fog of War Chess, where hidden opponent pieces must be inferred from historical observations.

The baseline Dense PPO agent is trained through self-play and optimized using PPO's clipped surrogate objective. During inference, actions are selected directly from the learned policy, providing a search-free benchmark against which we evaluate the benefits of action priors and restricted search.

4.2 Action Prior

The central idea is to separate candidate move generation from move evaluation. Rather than allocating search computation to every legal move, an action prior predicts which actions are most deserving of further consideration. Given a partial observation o_t, the prior outputs a score for each legal action a:

$$\pi_{\text{prior}}(a \mid o_t). \tag{1}$$

The default prior is the policy network itself: the same CNN–LSTM trained by PPO self-play to maximize game outcomes is reused to rank moves. We additionally consider an *information-trained* prior, where the policy is trained with an auxiliary reward for revealing hidden opponent squares, so that the resulting prior is biased toward information-gathering actions in addition to tactically strong ones. The next sections describe how the prior restricts search and how an information signal can instead be applied at inference time.

4.3 Restricted Search

Let $A(s)$ denote the set of legal actions available in state s. Rather than performing search over the entire action space, we rank actions using the learned prior and retain only the top-k candidates:

$$A_k(s) = \mathrm{TopK}\Big(\pi_{\mathrm{prior}}(a \mid s)\Big). \tag{2}$$

Search is then performed exclusively over the reduced candidate set $A_k(s)$. By reducing the effective branching factor, the agent can allocate its computational budget toward deeper evaluation of promising actions instead of spending resources on low-value moves.

This approach is inspired by policy-guided search methods such as AlphaZero, but extends them to a partially observable setting where actions must also be evaluated based on their ability to reveal information.

4.4 Inference-Time Information Bonus

An alternative to training an information-aware prior is to inject the information signal only at inference, leaving the policy prior unchanged. When forming the candidate set, we add an information bonus to each action's prior score:

$$s(a) = \pi_{\mathrm{prior}}(a \mid o_t) + \lambda\, I(a), \tag{3}$$

where $I(a)$ measures the information gained by taking action a (the number of previously hidden squares it reveals) and λ controls the relative importance of information acquisition. The top-k candidates are then selected according to $s(a)$, after which the depth-bounded search selects the final move. This biases the candidate set toward uncertainty-reducing moves when several actions appear similarly strong, without altering the trained policy. We refer to this configuration as Information-Bonus Search, in contrast to the Information Prior + Search configuration, which instead encodes the information signal into the prior's weights during training.

5 Experimental Setup

We compare five agents:

- **Random**: selects uniformly among legal moves.
- **Dense PPO**: the trained policy network plays directly, with no search.
- **Policy Prior + Search**: the trained policy is used as the prior; search runs over its top-k candidates.
- **Information Prior + Search**: identical to Policy Prior + Search, but the prior is trained with an auxiliary information-gain reward (training-time information).
- **Information-Bonus Search**: the standard policy prior with an inference-time information bonus added during candidate selection (inference-time information).

Agents are evaluated in a round-robin self-play tournament. Each pair plays 100 games as White and 100 as Black (1,000 games per agent), and to prevent deterministic agents from replaying identical games we begin each game with four uniformly-random legal half-moves before handing control to the agents. All search agents use a candidate-set size of $k = 10$ and a search depth of 2, and all agents act greedily (argmax) at evaluation. We report two metrics: **win rate** and **average search nodes evaluated per game** (a measure of computational cost).

All agents are implemented on top of the Stable-Baselines3 PPO implementation [6]. Hyperparameters used during training are summarized in Table 1.

Parameter	Value
Learning Rate	3×10^{-4}
Rollout Length (n_{steps})	256
Batch Size	128
PPO Epochs	4
Discount Factor γ	0.99
Clip Parameter	0.2
Entropy Coefficient	0.01
LSTM Hidden Size	256
Top-k Value	10
Search Depth	2
Training Steps	500,000

Table 1: Training and search hyperparameters.

6 Results

Table 2 summarizes the performance of all evaluated agents, and Figure 3 visualizes the win rates.

Method	Win Rate (%)	Avg Search Nodes/Game
Random	6.0	0
Dense PPO (no search)	14.0	0
Information Prior + Search	33.3	33,135
Policy Prior + Search	39.7	30,655
Information-Bonus Search	**41.7**	31,150

Table 2: Round-robin performance ($k = 10$, depth $= 2$, 1,000 games per agent). Search nodes are zero for agents that perform no lookahead.

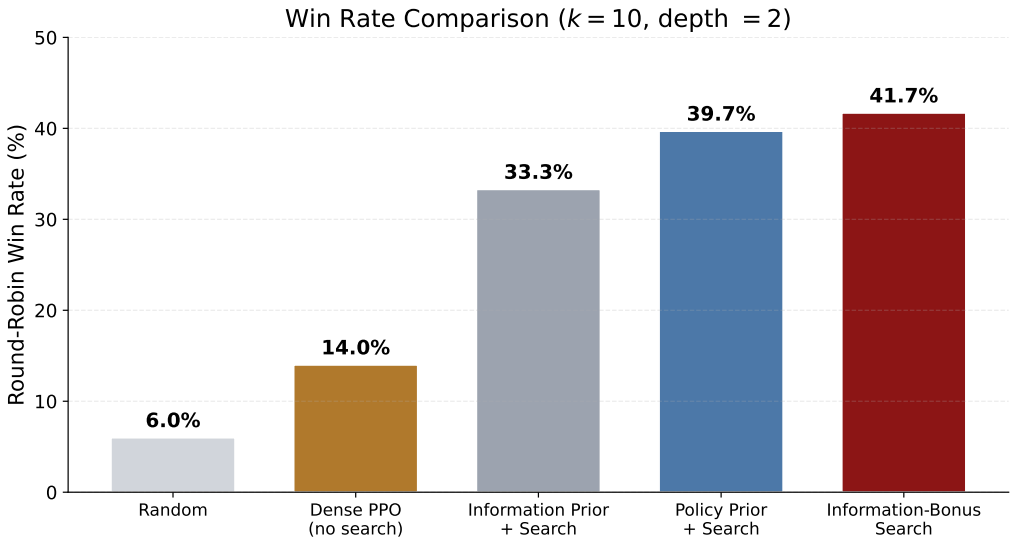

Figure 3: Round-robin win rate by method ($k = 10$, depth $= 2$, 100 games per direction with four random opening plies).

Two clear patterns emerge. First, *search is decisive*: every agent that runs a prior-guided search wins roughly 33–42% of its games, whereas the search-free Dense PPO agent wins only 14.0% and Random 6.0%. Reusing the policy as a prior and searching its top candidates is far more effective than playing the policy

directly.

Second, *where* the information signal is applied is decisive among the search agents. Adding an information bonus at inference time (Information-Bonus Search) yields the highest win rate at 41.7%, edging out the plain Policy Prior + Search (39.7%) at comparable search cost. In contrast, encoding the same information signal into the prior through a training reward (Information Prior + Search) *reduces* performance to 33.3% — the weakest of the three search agents — while also expending the most search nodes.

To understand this gap, Figure 4 (in the Discussion) ablates the candidate-set size k. The information-trained prior is by far the most sensitive to k: it wins only 7.1% at $k = 1$ but climbs steadily to 33.3% at $k = 10$. This indicates that the information reward distorts the prior's *ranking* — the strongest tactical move is frequently displaced from the top position — so good moves are recovered only once the candidate set is wide enough to include them. The policy prior and the inference-time bonus, by contrast, are comparatively flat across k, because their top-ranked move is already strong. The right panel confirms that search cost grows approximately linearly in k for all search agents, while the search-free agents incur none.

7 Discussion

Our results highlight two findings. The first is that a learned action prior is an effective way to allocate limited search: focusing a depth-bounded search on the prior's top candidates raised the win rate from 14.0% for the search-free policy to roughly 40% for the search agents. The quality of the retained candidates, rather than the act of pruning itself, is what drives this improvement.

The second, and more surprising, finding concerns *how* an information signal should be supplied. Both information-aware agents use the same notion of information gain, yet they behave very differently. Applying the signal at inference time as a selection bonus (Information-Bonus Search) gave the best overall result, slightly improving on the plain policy prior, because it nudges the candidate set toward uncertainty-reducing moves only when tactical options are otherwise comparable. Encoding the same signal into the training reward (Information Prior + Search) instead hurt performance: optimizing for revealed squares reshaped the policy's priorities so that strong tactical moves were no longer ranked first, an effect the top-k ablation makes explicit. The lesson is that information acquisition is best treated as a tie-breaker at decision time rather than as a primary training objective that competes with winning.

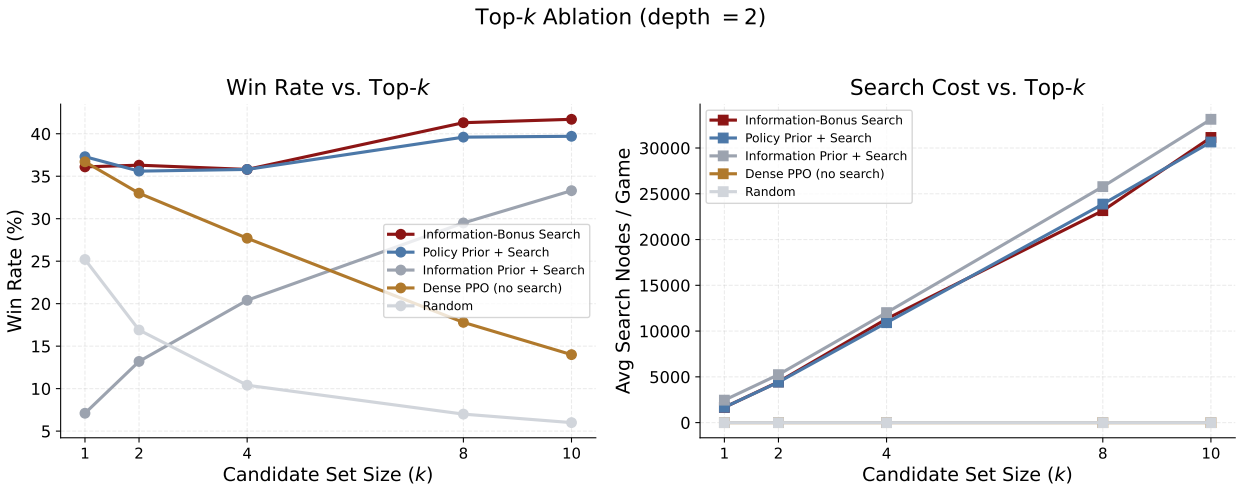

Figure 4: Top-k ablation: win rate (left) and average search nodes per game (right) versus candidate-set size k.

More broadly, these results suggest that learned priors are a robust mechanism for concentrating search

in large action spaces, but that auxiliary objectives intended to encourage exploration of hidden information must be introduced carefully so as not to distort the policy that the search ultimately relies on.

8 Limitations and Future Work

Although our approach achieves strong performance relative to the baselines considered, several limitations remain. First, the learned action prior is trained specifically for Fog of War Chess and may not generalize to other domains without retraining. The effectiveness of the information reward structure is also closely tied to the visibility mechanics of the environment.

Second, our search procedure relies on a fixed candidate set size and search depth. While this simplifies evaluation, different game situations may benefit from dynamically allocating search resources. Future work could investigate adaptive candidate selection strategies that vary the amount of search based on state uncertainty or policy confidence.

Third, while the CNN-LSTM architecture captures some historical information, it does not explicitly maintain a belief state over hidden opponent pieces. More sophisticated approaches, such as learned belief-state tracking, transformer-based memory architectures, or probabilistic world models, may allow the agent to reason more effectively about hidden information.

Finally, our search procedure remains relatively lightweight compared to methods such as Monte Carlo Tree Search, and our evaluation has practical caveats: results come from a single training seed, agents were trained primarily from the White perspective, and search-versus-search games are frequently truncated to draws under the 200-move cap, which compresses the win-rate gaps between the strongest agents. Future work could explore combining information-aware priors with deeper search methods, learned value functions, or model-based planning, as well as multi-seed evaluation and explicit handling of color asymmetry. More broadly, we believe that explicitly reasoning about information gain represents a promising direction for reinforcement learning in partially observable environments.

9 Conclusion

In this work, we investigated how learned action priors and information-awareness affect search in Fog of War Chess, a challenging partially observable strategy game. We combined PPO self-play, a learned action prior, and a depth-bounded search restricted to the prior's top-k candidates, and compared two ways of injecting an information-gain signal: a training-time reward and an inference-time selection bonus.

Experimental results show that prior-guided search dramatically outperforms the search-free policy, lifting the win rate from 14.0% to roughly 40%. Among the search agents, adding an information bonus at inference time produced the strongest play (41.7%), narrowly ahead of the plain policy prior, while encoding the same signal into the training reward proved counterproductive (33.3%) by distorting the prior's ranking of tactical moves.

These findings suggest that learned priors are a promising approach for large-action-space environments where exhaustive search is infeasible, and that information-awareness is most effective as a lightweight inference-time bias rather than a competing training objective.

10 Team Contributions

The final project evolved substantially from our original proposal. Initially, we proposed a PPO agent combined with a hand-designed, rule-based pruning mechanism that would filter actions based on shallow evaluation metrics such as material gain and piece safety. As development progressed, we found that fixed, hand-designed pruning provided limited benefits in a partially observable setting. This led us to redesign

the project around learned action priors, information-aware rewards, and search-based action selection. As a result, responsibilities shifted significantly from the original proposal.

Kuba Hashemian. Kuba led the reinforcement learning infrastructure and experimentation effort. He implemented the PPO training pipeline, managed self-play training, performed hyperparameter tuning, and conducted the majority of large-scale experimental evaluation. Compared to the original proposal, Kuba's role expanded beyond training a baseline PPO agent to include designing and evaluating multiple search-based variants and ablation studies.

Leon Liu. Leon was responsible for environment integration, state representation design, and search implementation. He developed the observation pipeline, constructed the CNN-LSTM state representation, implemented restricted search procedures, and built evaluation tooling for round-robin tournaments. Compared to the original proposal, Leon contributed substantially more to search infrastructure and experimental evaluation as the project shifted from simple action pruning toward search-guided decision making.

Sandeep Sethuraman. Sandeep led the development of the learned action prior framework. He designed the information-aware reward structure, implemented information-gain bonuses, developed candidate-action ranking methods, and conducted experiments evaluating the impact of information-based incentives. This differed significantly from the original proposal, where his primary responsibility was expected to be implementing the rule-based pruning logic. As the project evolved, hand-designed filtering was replaced by a learned information-aware prior, requiring considerably more modeling and experimentation.

Shared Contributions. All team members contributed to project planning, debugging, experimental analysis, poster preparation, and report writing. The transition from a rule-based pruning project to an information-aware search framework required multiple redesigns throughout the quarter, making collaboration across all components essential to the final system.

References

[1] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

[2] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI Fall Symposium Series*, 2015.

[3] Shengyi Huang and Santiago Ontañón. A closer look at invalid action masking in policy gradient algorithms. In *Proceedings of the International FLAIRS Conference*, 2022.

[4] 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.

[5] 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 heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

[6] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

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

[8] 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.

[9] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

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