Reinforcement Learning in Modern Abstract Board Games

Bogdan Gosa

***Abstract*—In this paper we look at various strategies in reinforcement learning algorithms to make an artificial agent learn to play modern abstract board games, specifically Calico. Different reward systems are analyzed, and results are compared against purely statistical strategies, providing insights into the effectiveness of reinforcement learning in stochastic tile-placement, space-optimization games.**

***Index Terms*—Reinforcement Learning, Deep Q-Networks, Abstract Board Games, Tile Placement**

I. Introduction

Modern abstract board games such as Calico, Sagrada, and Harmonies present unique challenges for artificial intelligence due to stochastic elements and indirect competition. While traditional board games like Chess and Go are well-studied in AI research [1], modern tile-placement games remain largely unexplored.

My novel contribution lies in developing a modified Deep Q-Learning or TD algorithm that integrates probabilistic state encoding and adaptive reward smoothing [2] to handle uncertainty in tile draws. This study focuses exclusively on the single-player setting, as extending the approach to multiplayer scenarios, where indirect competition introduces significant additional complexity, is left for future work.

II. Background and Related Work

Existing literature on reinforcement learning in games primarily focuses on perfect-information, zero-sum settings such as Go [1], Othello [3], Backgammon [4], Nine Men's Morris [5], and Hex [6]. In contrast, modern abstract games feature stochastic tile placement, spatial optimization, and complex scoring systems. Some recent studies explore reinforcement learning in non-deterministic environments, but few address modern abstract games specifically.

As shown in [1], reinforcement learning has achieved superhuman performance in Go. Recent work applied Q-Learning to modern board games such as Dominion [7] or Settlers of Catan [8]. The only paper available that discusses this sort of abstract board game uses Monte Carlo Tree Search (MCTS) to play Ceramic, a simplified version of Azul [9]. While their model performs better than random action, it's still very weak in comparison to humans. [9].

Other exploratory works have started addressing more complex imperfect-information and stochastic board games such as Ticket to Ride [10] and broader discussions of RL challenges in tabletop games [11]

III. Novel Approach

While Monte Carlo Tree Search (MCTS) has been applied to abstract tile-placement games such as Ceramic [9], its performance is limited in environments with high stochasticity and combinatorial complexity.

To address these limitations, we propose a novel adaptation of Deep Q-Learning (DQL) and Temporal Difference (TD) methods specifically tailored for stochastic, tile-based board games like Calico. Our approach incorporates probabilistic state encoding, which represents uncertain tile placements and potential patterns as probability distributions, and adaptive reward smoothing, which stabilizes learning in the presence of sparse and highly variable rewards. By learning value estimates directly from experience rather than relying on exhaustive simulations, our method can generalize across unseen board states and maintain stable learning in highly stochastic scenarios.

Compared to the MCTS approach in Ceramic, our method should be both computationally efficient and more robust to randomness, making it well-suited for the complex scoring and spatial optimization challenges present in modern abstract games. This represents the first reinforcement learning approach designed to handle the stochastic, combinatorial nature of tile-placement games beyond simplified variants.

IV. Hypotheses

A modified Deep Q-Learning or Temporal Difference model can achieve higher performance in modern abstract board games than baseline statistical strategies or naive play.

V. Research questions

1. Can a modified Deep Q-Learning or a Temporal Difference model perform better than a set statistical strategy in stochastic modern abstract board games?

2. Can a modified Deep Q-Learning model or a Temporal Difference converge to a optimal policy in a stochastic modern abstract board games?

3. Can the model,on average, have a higher score than a human opponent?

VI. Calico

Calico is a puzzly tile-laying game of quilts and cats.

Behind the cute theme lies a strategical game which involves complex decision-making and risk management. Players are often faced with trade-offs between securing immediate, modest points and pursuing strategies that may yield larger rewards in the long term, given a bit of luck. The tiles are drawn from a shared shop, this means selecting a tile that benefits an opponent may still be strategically justified if it prevents them from scoring.

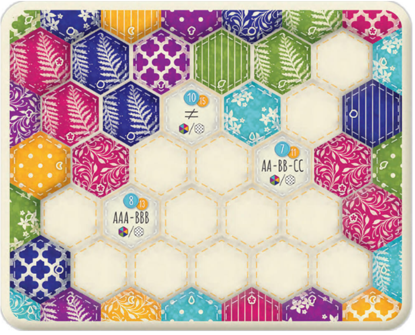

Fig. 1: The calico board.

There are 3 ways of scoring in Calico:

- Buttons: 3 points. To collect a button you need 3 or more tiles of the same color adjacently.
- Cats: 3-11 points. Each cat has 2 preferred patterns and a number of adjacent tiles of that patterns that are required. For example a cat that gives you 5 points needs 4 adjacent tiles of a certain pattern.
- Objectives: 7-15 points. Each objective from the board may score points based on the configuration of tiles surrounding it. You can complete the configuration on color or pattern, but doing both grants extra points.

VII. Reinforcement Learning

Reinforcement learning is a framework for training machine learning models to make sequential decisions, where an agent learns to achieve a goal through interaction with an uncertain and potentially complex environment [12], [13].

Q-Learning is a model-free reinforcement learning algorithm that learns an action-value function $Q(s, a)$, which estimates the expected cumulative reward of taking action a in state s and following the optimal policy thereafter [14]. At each step, the Q-values are updated using the observed reward and the maximum estimated value of the next state. Double Q-Learning reduces overestimation in standard Q-Learning resulting in more stable and reliable learning. [15].

A natural extension of Q-Learning to high-dimensional state spaces is the Deep Q-Network (DQN) [16]. A DQN is a multi-layered neural network that, for a given state s, outputs a vector of action values $Q(s, \cdot; \theta)$, where θ are the network parameters. For an n-dimensional state space and an action space of m actions, the network maps $\mathbb{R}^n \to \mathbb{R}^m$. These techniques allow DQNs to approximate Q-values efficiently even in large state spaces, making them suitable for complex environments such as Calico.

VIII. Methodology

Each agent will be tested on 1000 self play games. The scores for objective, cat and color as well as final score will be stored for each game, forming the dataset.

Agent performance is validated by comparing game scores against the human dataset and other tested models, with the score obtained at the conclusion of the game serving as the key performance indicator.

A secondary performance factor will be the computational cost of the agents, quantified as the average decision latency (milliseconds per move)

A. Human Performance Dataset

To establish a meaningful performance ceiling for our agents, a dataset comprising scores from human players was collected. This dataset serves as the primary benchmark against which the performance of all models is measured. The average score for this human benchmark is 58.5 points, exceeding the score of any baseline agent.

Due to the lack of publicly available standardized data on this game this benchmark was constructed from 41 self-play games played by 15 distinct human participants. We recognize that the limited sample size is susceptible to sampling bias, however, recording more games will increase the reliability of this dataset for future comparison.

IX. Environment

The *Calico* board consists of a grid of 5×5 tiles, while the 3 objectives are set it still leaves 22 positions to be occupied by

$$6 \times 6 \times 3 = 108 \text{ tiles.}$$

possible tiles. Therefore, the number of possible combinations of tiles that can fill the 22 available spaces is extremely large ($\sim 10^{34}$).

The game environment was implemented in Python to simulate the full rule set and mechanics of Calico, consisting of board state, tile market, player hands, and scoring conditions for buttons, cats, and objectives. Each game state is encoded as a numerical representation suitable for input to reinforcement learning algorithms. Each tile is encoded into a number ranging from 1 to 36.

The complete implementation can be found in the GitHub repository in the class `CalicoEnv`. This environment class will be used for all experiments presented in this paper.

X. Conducted Experiments

A. Random Play

Random play serves as the essential low-bound baseline for evaluating agent performance in this paper. Under this deterministic yet completely non-strategic policy, the agent selects moves entirely at random from the set of legal actions throughout the entirety of the game. This method consistently converges to a low average score of just 13.1 points per game.

B. One Step Lookahead Policy

The One-Step Lookahead Policy serves as a foundational approach to agent decision-making in complex state spaces. This deterministic policy operates greedily, choosing the action A^* that maximizes the score returned by a Heuristic Evaluation Function (HEF), $E(S, A)$, applied to the resulting state

S'. Crucially, this policy does not incorporate planning or learning; decisions are based solely on the immediate value assigned by the HEF.

Experimental results confirm the severe limitations of this constrained search depth. When the HEF uses only the True Reward Function (based purely on the game's official rules), the average final score consistently converges to an average of 25 points. This low performance, however, remains a direct consequence of the agent's inability to look beyond the current move, leading it to favor immediate, low-value rewards over complex, multi-step strategies required to fulfill various scoring conditions.

To mitigate this planning deficiency, we introduced a customized Heuristic Evaluation Function. This function integrates features representing future potential, such as partial scoring for partially built objectives, Cat token proximity (P_{Cat}), and color group viability (P_{Color}). This linear combination of features is parameterized by explicit hyperparameters (w_i), which allow for precise tuning of the agent's priorities (e.g., favoring objective completion over immediate scoring). This structural change effectively injects long-term value signals into the agent's immediate decision-making process. This significantly improves the score to 42 points on average when using the untuned Heuristic Evaluation Function (with all feature weights initialized to 1), demonstrating the baseline benefit of integrating future potential features.

To transition from this static performance and overcome the challenge of manual tuning, the weights of the Heuristic Evaluation Function were optimized using a Neuroevolutionary approach (specifically, a Genetic Algorithm). This technique successfully evolved the weight parameters to maximize the agent's average performance, resulting in a final average score of 53 points.

The heuristic evaluation function is defined as a linear combination of four weighted features:

$$\mathbf{E}(\mathbf{S},\mathbf{A}) = \begin{aligned} & w_F \cdot F(S') + w_{CP} \cdot P_{\text{Cat}}(S') \\ & + w_{CC} \cdot P_{\text{Color}}(S') + w_{OP} \cdot P_{\text{Obj}}(S') \end{aligned} \tag{1}$$

Where the selection policy is:

$$\mathbf{A}^* = \underset{A \in \text{LegalActions}(S)}{\text{argmax}} \quad E(S,A) \tag{2}$$

The coefficients w_i are hyperparameters defined by the configuration, config, and the feature functions are described as follows:

- w_F: weight final score
- $F(S')$: The true score obtained from currently completed objectives and groups (`get_total_score_on_board`).
- w_{CP}: weight cat potential score
- $P_{\text{Cat}}(S')$: Heuristic potential score for the proximity and arrangement of tiles to complete Cat token placements (`cat_potential_score`).
- w_{CC}: weight color potential score
- $P_{\text{Color}}(S')$: Heuristic potential score for the size and arrangement of uncompleted color and pattern groups (`color_potential_score`).
- w_{OP}: weight objective potential score
- $P_{\text{Obj}}(S')$: Heuristic score assessing the ease or difficulty of completing remaining objective tokens (`objective_potential_score`).

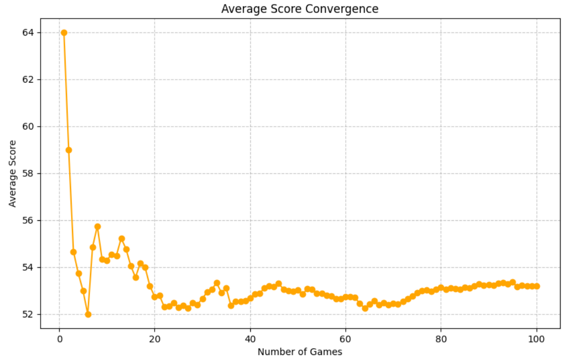

Fig. 2: Convergence of the Neuroevolutionary OSLH Agent. The graph confirms the rapid stabilization of the average score around 53.2 points.

C. Reinforcement Learning Approach

We implemented Q-Learning and Deep Q-Networks (DQN) for the game Calico, as well as a Temporal difference(TD) approach. The state space represents the current board configuration, and the action space includes all possible tile placements as well as all tile buy options. This deep learning approach should yield better results than the MCTS agend presented in [9].

D. Baseline Statistical Strategy

A purely statistical baseline selects moves based on heuristic evaluations of immediate score potential, without long-term planning or learning.

XI. Results

A. Training Performance

Present training curves and their convergence after a few thousand episodes.

Present the average score achieved by each model after 1000 self play games.

B. Comparison with Baseline

The reinforcement learning agent is compared with the baseline random model,as well as the statistical model, and, hopefully, it demonstrates the benefit of learning from experience.

XII. Conclusion

The complete source code and environment definitions for the Calico AI project are publicly available at: https://github.com/bogdangosa/calico-ai.

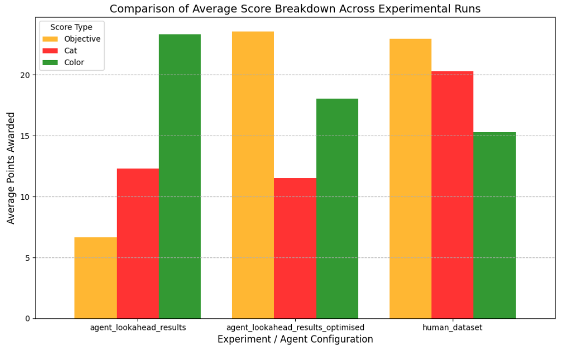

Fig. 3: Comparison between baseline OSLH Agent, Neuroevolutionary OSLH Agent and human performance

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, and et al., “Mastering the game of go without human knowledge,” *Nature*, vol. 550, pp. 354–359, 2017. [Online]. Available: https://doi.org/10.1038/nature24270

[2] V. Lee, P. Abbeel, and Y. Lee, “Dreamsmooth: Improving model-based reinforcement learning via reward smoothing,” 2024. [Online]. Available: https://arxiv.org/abs/2311.01450

[3] M. van der Ree and M. Wiering, “Reinforcement learning in the game of othello: Learning against a fixed opponent and learning from self-play,” in *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, Singapore, 2013, pp. 108–115.

[4] G. Tesauro, “Td-gammon, a self-teaching backgammon program, achieves master-level play,” *Neural Computation*, vol. 6, no. 2, pp. 215–219, 1994.

[5] J. Abukhait, A. Aljaafreh, and N. Al-Oudat, “A multi-agent design of a computer player for nine men’s morris board game using deep reinforcement learning,” in *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, Granada, Spain, 2019, pp. 489–493.

[6] M. Lu and X. Li, “Deep reinforcement learning policy in hex game system,” in *2018 Chinese Control And Decision Conference (CCDC)*, Shenyang, China, 2018, pp. 6623–6626.

[7] G. Angelopoulos and D. Metafas, “Q learning applied on the board game dominion,” in *Proceedings of the 24th Pan-Hellenic Conference on Informatics (PCI ’20)*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 34–37.

[8] K. Xenou, G. Chalkiadakis, and S. Afantenos, “Deep reinforcement learning in strategic board game environments,” in *Multi-Agent Systems, EUMAS 2018, Lecture Notes in Computer Science, vol 11450*, M. Slavkovik, Ed. Springer, Cham, 2019.

[9] G. Quentin and K. Tomoyuki, “Ceramic: A research environment based on the multi-player strategic board game azul,” *2020*, vol. 2020, pp. 155–160, 11 2020. [Online]. Available: https://cir.nii.ac.jp/crid/1050011097117536256

[10] A. TBD, “Reinforcement learning agents playing ticket to ride – a complex imperfect-information board game with delayed rewards,” *Preprint / Conference*, 2023. [Online]. Available: https://www.researchgate.net/publication/371649650_Reinforcement_Learning_Agents_Playing_Ticket_to_Ride__a_Complex_Imperfect_Information_Board_Game_with_Delayed_Rewards

[11] M. Balla *et al.*, “Challenges and opportunities for reinforcement learning in modern tabletop games,” *CoG ’23 (Workshop proceedings)*, 2024, preprint arXiv:2305.xxxxx. [Online]. Available: https://arxiv.org/abs/2405.18123v1

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html

[13] X. Wang, S. Wang, X. Liang, D. Zhao, J. Huang, X. Xu, B. Dai, and Q. Miao, “Deep reinforcement learning: A survey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 4, pp. 5064–5078, 2024.

[14] C. J. C. H. Watkins and P. Dayan, “Q-learning,” *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[15] H. van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double q-learning,” in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*. AAAI Press, 2016, pp. 2094–2100. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/10295

[16] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” 2013. [Online]. Available: https://arxiv.org/abs/1312.5602