PKU-IAI Technical Report: PKU-IAI-2022-T-0006

Playing Hanabi with ToM and Intrinsic Rewards

Yilue Qian, Yuxuan Luo, Haoxiang Yang, Siwen Xie
Yuanpei College
Peking University
{qyl,2000017426,yyyhhhxxx,swxie}@stu.pku.edu.cn

Abstract

In recent years, artificial agents have made drastic advances in multi-player games such as Go and poker. However, cooperative games with imperfect information are relatively underexplored, despite that such setting is common in daily human-robot interaction. The card game Hanabi is an example, where reasoning about other agents' mental states (*e.g.*, belief and intention) is brought to the foreground. It is a meaningful benchmark because it requires Theory of Mind (ToM) reasoning and challenges an agent's decision-making ability in a partially observable and cooperative setting. Existing work on Hanabi achieves great self-play results. However, they fail to address these essential challenges in the other-play setting. To fill in the gap, we propose two innovative plug-in modules that can be applied to general RL agents. The Hand Card Information Completion module is designed to model other agents' mental states and complement environment information. The Goal-Oriented Intrinsic Reward module encourages agents' exploration and collaboration. We believe such attempts will boost performance in this particular game and facilitate human-robot cooperation in a broader range of interactive scenarios. Our code is available at `https://github.com/LoYuXr/Hanabi_Plugins`.

1 Introduction

Multi-agent cooperation lies at the roots of human society and is indispensable in modern life. However, such a natural human act turns out to be sophisticated for both non-human animals and artificial machines. Studies in psychology and cognitive science have demonstrated that children and chimpanzees have similar cognitive skills regarding the physical world, but perform significantly better in social cognition [7], indicating that human cognitive abilities largely lie in our social skills instead of some kind of general intelligence. This also accounts for the problems that modern AI encounters in human-robot interaction. Effective cooperative interaction requires every participant to understand others' intentions, mutually exchange information, as well as realize the necessity of altruistic actions and be motivated to perform them to eventually reach a shared goal, and thus remains a daunting challenge for AI agents.

How do humans manage to complete such a complicated task in the first place? A major cognitive skill used in social scenarios is Theory of Mind (ToM) [14]. ToM stands for the general ability to attribute mental states to others, or in other words, reason about what other agents are thinking about, including their beliefs, desire, and intention, to explain and predict their actions and make corresponding responses. Intuitively, this means to imagine the world from another person's point of view, isolated from the ground truth. ToM is proven to be a critical component of human social

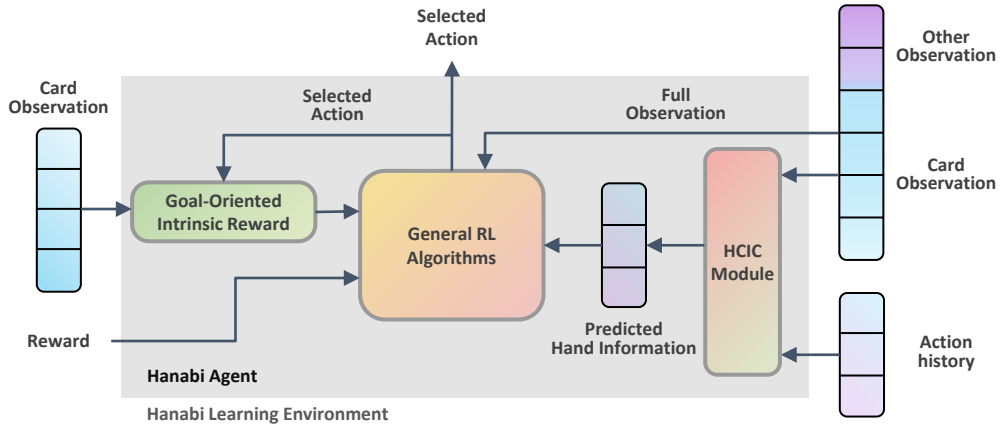


Figure 1: Involvement of HCIC module and GOIR module in general RL models. The HCIC module takes action history and card observation as input, inferring the hand card information. The GOIR module takes the player's partial observation and action in the current state as input. It outputs the intrinsic reward for the action. The model is trained using observation complemented by the HCIC and full rewards, which is the summation of intrinsic and extrinsic rewards.

cognition, as children with Autism do not exhibit this ability [2]. Therefore, it might be the key to better social cooperation between humans and AI.

Having seen the potential of machine ToM, we need a suitable benchmark. The card game Hanabi is proposed for this purpose [1]. It is a purely cooperative game with imperfect information for 2 to 5 players, where a player can see all the other players' cards but not his/her own. Each card depicts a rank (1 to 5) and a color (red, green, blue, yellow, and white); the deck is composed of a total of 50 cards, 10 of each color: three 1s, two 2s, 3s, and 4s, and finally a single 5. The shared goal of all players is to play cards cooperatively, so as to form five consecutively ordered stacks, called fireworks, one for each color. Players take turns to take one of the following 3 actions.

- **Reveal.** Give a hint to any other player, by selecting one specific rank or color, and pointing out all cards that match the selected rank or color in the hinted player's hand. Each hint costs an information token (initially 8, and will never exceed 8). When there is no token left, a hint cannot be made, and the player must choose one of the other two actions.
- **Discard.** Discard a card in hand. Discarded cards will form a stack along with unsuccessfully played ones, which is visible to all players. This action restores back an information token.
- **Play.** Play a card in hand and draw a new card from the deck. Playing a card is successful if the card is the next in the sequence of its color to be played. An unsuccessful play costs a life token (initially 3), and a successful play restores an information token.

The game ends when all 3 life tokens are taken away, with the players scoring 0; or when the card deck runs out, and the players receive 1 point for each card in the played stacks, maximally 25 in total.

Hanabi is special because it creates a multi-agent environment where humans actively employ ToM. Observations on the behavior of human players show that they infer about their partners' mental states throughout the game, especially their belief (*e.g*. What do they know about my cards?) and intention (*e.g*. Is this hint intended for me to play the card or discard it?). Clues for this information lie in their partners' actions, but they cannot provide enough evidence unless combined with observations of the current state, since the mental states behind the same action can vary in different contexts. For example, if Alice gives a hint that one of Bob's cards is a 4, she would likely be intending for him to play it if there are many 3s on the played stacks, but would possibly be hoping him to discard it if there are many 4s.

The imperfect information of Hanabi also challenges modern algorithms. For decision processes such as games, Reinforcement Learning (RL) is commonly used, which learns a policy function or optimal action-value function to pick an optimal action given the observed state. However, Hanabi forms a Partially-Observable MDP (POMDP) problem, which is hard for general reinforcement learning models. Besides, the optimality of a policy largely depends on that of other players, adding uncertainty

and entanglement to the algorithm's exploration. Finally, there is the issue of sparse reward. A typical RL Hanabi agent receives an immediate reward only when it plays a card successfully, which not only harms learning efficiency but also discourages altruistic hinting.

The cooperative and complicated nature of Hanabi makes it an interesting challenge for AI agents. The learning objective of these agents are multi-faceted: they can either learn a policy for the entire team, which is referred to as *self-play*, or learn to play in an ad-hoc team where players have different strategies and little or no pre-coordination, that is, *other-play*. Previous research on Hanabi mainly tackles the former goal, while little attempts for the latter, let alone achieving remarkable results, although humans perform nearly as well in the latter setting.

In this paper, we propose two innovative plug-in modules to help tackle the Hanabi challenge, inspired by human ToM. Our main contributions are as follows.

- We design a plug-in module that infers the belief of other players on observing their actions, imitating human ToM. This module is applicable to any RL codebase, and prospectively helps to boost human-AI cooperation in various scenarios.
- We design a discrete goal-oriented intrinsic reward to the RL algorithm based on the current player's observation and action to encourage exploration and collaboration.
- We run experiments and preliminarily validate the effectiveness of our two modules, especially for three and four players.

2 Related work

2.1 Hanabi agents

Self-play The self-play challenge focuses on finding a joint policy of the team that achieves a high expected score through self-play learning. A main approach to this goal is by deep RL, including (i) the Actor-Critic Hanabi Agent [4] with multiple copies of the agent running in different instantiations of the environment, (ii) the Rainbow Agent [1] based on Rainbow DQN [8] which is an innovated variant of Deep Q-Networks (DQN), and (iii) the BAD Agent [5] which relies on a Bayesian Action Decoder to explicitly track public belief, conditioning on the actions taken by all agents in the environment. Another aproach is by rule-based algorithms that directly encode behavioral conventions in their rules, such as SmartBot [12], HatBot [4], and WTFWThat [17]. The results are summarized by Bard et al. [1], from which we can see that RL agents significantly underperform rule-based ones in 3-to-5-player settings. The other learning agent, BAD, performs the best in a 2-player setting, which can probably be attributed to its explicit modeling of belief. We cannot yet confirm that BAD achieves general improvement, because it is only tested with 2 players.

Other-play Despite playing relatively well in self-play, the RL agents' performance experiences a dramatic drop when paired with other unknown partners, scoring only slightly above 0. The reason is simply that neither ACHA nor Rainbow makes use of the sample play of its partners, and thus they lack the flexibility to adapt and coordinate with an ad-hoc team. Humans, on the other hand, are capable of cooperating with an unseen partner, which motivates researchers to endow Hanabi agents with the same ability. Canaan et al. [3] attempted to pair a Rainbow DQN agent with other (individual or mixed) rule-based agents during training, and discovered that although it plays relatively well with the agents used in training, it still fails to cooperate with novel partners, including itself. Hu et al. [9] adds an OP (Other-Play) algorithm that enhances a self-play agent by looking for more robust and non-specified strategies, achieving significantly better results in other-play setting. In our work, we approach such robustness and flexibility from another perspective, by adding a plug-in module applicable to any RL architecture, that makes use of the other players' actions to infer information of its own hand.

Environment The open-source Hanabi Learning Environment, produced by [4], provides environment interfaces and sample agents using different policies (random, simple heuristic, and Rainbow).

2.2 RL for POMDP

Hanabi is a Partially Observable Markov Decision Process (POMDP) game, where traditional RL methods cannot perform well. No matter what optimization objective it aims at (typically value

or policy), an RL algorithm strongly depends on the observation of state s, while in POMDP this information is defective. To tackle this problem, a popular idea is to complement the missing information by modifying the input state.

Stacking up observations is one primitive attempt. For example, DQN [11] uses multiple recent frames as the observed state for Atari games. The main difficulty of this method is that state stacking leads to infinite dimensions, and thus it is incapable of storing long-term memory. Besides, the missing information cannot always be complemented by backtracking in time, such as a player's hand in Hanabi.

Recurrent Neural Networks (RNNs) can deal with relatively long-term sequential states without exceeding memory limits, and thus are employed to generate latent representations based on observation history, achieving remarkable results, as in Recurrent DQN [6] and the state-of-the-art R2D2 [15]. However, the mapping function from an observation to a latent state relies solely on reward signals back-propagated from the value function. Such representations lack semantic features about historical memories and can lead to overfitting in sparse reward environments. In this paper, we adopt the Transformer [16] architecture, which takes observation history as input, and decodes it to make a prediction of the cards in hand, compensating for the lack of semantic meaning of latent representations in previous RNN-based methods.

2.3 Intrinsic rewards

Traditionally in RL, the agent receives rewards from the environment, which further molded their behaviors. However, from child play to scientific discovery, many activities humans engage in are rewarding in and of themselves. This leads to "extrinsic" and "intrinsic" rewards, which respectively represent the outer effects and self-awareness. We introduce intrinsic reward not only because intelligent organisms such as human beings have self-motivation mechanisms (like dopamine and endorphins), but also because of the consideration of reinforcement learning itself. Sometimes, the extrinsic reward is sparse, which may inhibit exploration and converge the agent to harmless but mediocre policies, such as spinning around the starting point, etc. To handle this issue, researchers borrowed the idea of intrinsic reward to the design of agent models. For example, in a sparse reward environment, to promote exploration (which was similar to "curiosity" for it is intrinsic), Pathak et al. [13] modeled curiosity as the difference between the agent's prediction of a new state in visual feature space and the actual state. They used a traditional A3C algorithm and add a reward function of prediction error as a curiosity reward. This method is confirmed to solve the sparse reward problem and can achieve better generalization in different game environments. Kulkarni et al. [10] proposed hierarchical-DQN (h-DQN), a framework to integrate hierarchical action-value functions with goal-driven intrinsically motivated deep reinforcement learning. A top-level q-value function picks short-term goals to optimize expected future extrinsic reward, while a lower-level function uses both the state and the chosen goal to select actions to optimize intrinsic reward.

3 Methods

We intend to design a plug-in module that can boost the performance of general RL algorithms on the Hanabi environment, either self-play or other-play, both in terms of convergence speed and obtained scores.

As mentioned above, the Hanabi environment is a POMDP game where the exact information of the player's hand card is unknown and independent of time. The reward is also sparse, only the playing action can receive a positive reward. The imbalance of reward between actions may hurt exploration and lead the agent to behave either too boldly (mostly conducting "playing" action) in the early times, or too conservatively (merely revealing others' cards or discarding theirs, for it does no harm to reward). To solve these two problems, we propose the following two modules: The Hand Card Information Completion (HCIC) module, which is to complete POMDP into MDP; and the Goal-Oriented Intrinsic Reward (GOIR) module for mitigating the sparse reward problem. Fig. 1. shows how these two modules are involved in general RL models.

3.1 Hand Card Information Completion

The Hand Card Information Completion (HCIC) module imitates human actions when playing Hanabi by considering observable card information and the cooperators' actions. As we have mentioned before, human players can form their beliefs of other players' intentions behind their actions and

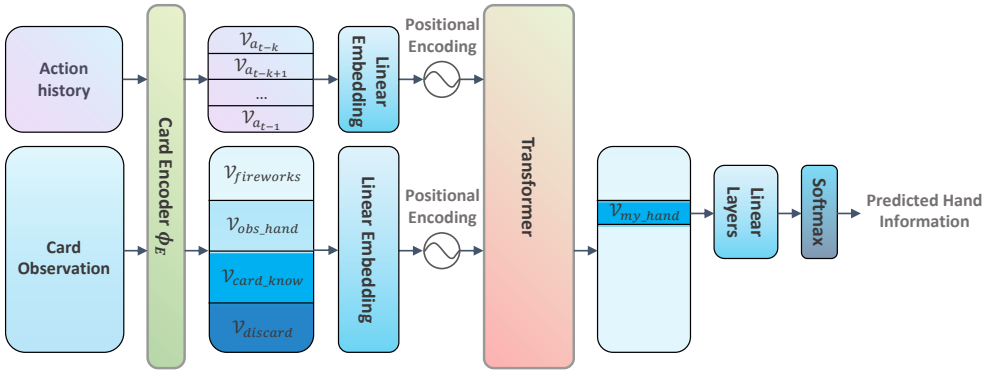


Figure 2: Architecture of Hand Card Information Completion module. Card Encoder ϕ_E encodes action history and card observation into vectors, which are then sent to Transformer after positional encoding. The output of the Transformer provides the predicted hand information.

then further infer their hand card information, which becomes the crucial point in solving the game. Similarly, we are going to model this Theory of Mind behavior.

At each time step t, the HCIC module takes global action history $\mathcal{A} = \{a_{t-k}, a_{t-k+1}, \ldots, a_{t-1}\}$ and card observation $\mathcal{C}_{obs}$ as input, inferring the agent's own hand card information $\mathcal{C}_{my\_hand}$. As Fig. 2 shows, the card encoder ϕ_E first encodes $\mathcal{C}_{obs}$ and $\mathcal{A}$ into vectors. For each observed card $\mathcal{C}_i$ from $\mathcal{C}_{obs}$, the card encoder encodes it into a 25-dim vector, as shown in Alg. 1. When current agent is not able to observe or know a card (*e.g.*, its own hand cards), the unknown properties are sent to Alg. 1 as `null`. Specifically, each one of the fireworks are treated as the card that is expected to be placed on that firework next. We also pad the discard sequence with unknown cards to a max sequence length l. To better represent actions from $\mathcal{A}$, card encoder ϕ_E only encodes `REVEAL` actions which revealing properties of current agent's hand. These actions are also treated as cards so that ϕ_E could encodes them with Alg. 1.

The encoded vectors are embedded and then sent to Transformer after positional encoding. Vectors representing action history $\mathcal{A}$ are sent to the Transformer encoder and vectors representing card observation $\mathcal{C}_{obs}$ are sent to the Transformer decoder. In the output of Transformer, we pick the vectors as the predicted hand vectors $\mathcal{V}_{my\_hand}$ that are in the same position as the vectors representing the observation of the agent's own hand in the input. The vectors are then sent to a linear head and a softmax function to obtain predicted hand information.

Here we would like to explain why we choose Transformer as the backbone of the HCIC module. First, as described in Sec. 2.2, previous methods either display stacked observations or implicitly use RNN for information completion of POMDP. For environments like partial observation mazes, the information gain is rather high by superimposing observations at different locations in different time frames. But the missing hand information in Hanabi is time-invariant, meaning that if other players don't offer hints, you'll never know your hand. Thus, we believe that Transformer, which uses positional encoding to compute information in parallel, is more efficient in training than RNN. Another point is that the cross attention mechanism in Transformer could fully use the strong correlation between `REVEAL` actions and hand card information.

We achieve the training data by running a baseline Rainbow model in the self-play setting from 2 players to 4 players with training iteration 5000. We record the validation game sequences and pick the last 1000 iterations as training data. After the model finishes training, we add the HCIC module into the agent. To avoid information leakage, we will stop updating the parameters in the HCIC module during the training of our target agent.

3.2 Goal-Oriented Intrinsic Reward

The original Hanabi environment only rewards successful plays, which is unfriendly for agents due to their large action spaces. In the environment, the actions of `REVEAL` and `DISCARD` may be seen as philanthropy for they do not directly bring benefits. This can explain our observation of agents playing recklessly at the very beginning, leading to early failures. After trial and error (mostly errors), the model might converge to a conservative and risk-aversion policy (*e.g.*, revealing others' card

Algorithm 1 Card Encoding

Input: Card $\mathcal{C}_i$, with rank $n \in \{0, 1, 2, 3, 4, \texttt{null}\}$
and color $c \in \{\texttt{RED(0)}, \texttt{YELLOW(1)}, \texttt{GREEN(2)}, \texttt{WHITE(3)}, \texttt{BLUE(4)}, \texttt{null}\}$.
Output: The card representation $\mathcal{V} = (v_1, v_2, \ldots, v_{25}) \in \mathbb{R}^{25}$.

$\mathcal{V} \leftarrow (0, 0, \ldots, 0)$
if c is `null` **then**
  if n is `null` **then**
    $v_i \leftarrow 1/25$ **for** $i = 1, 2, \ldots, 25$
  else
    $v_i \leftarrow 1/5$ **for** $i = 5n, 5n + 1, \ldots, 5n + 4$
  end if
else if r == `null` **then**
  $v_i \leftarrow 1/5$ **for** $i = c, c + 5, \ldots, c + 4 \cdot 5$
else
  $v_{5n+c} \leftarrow 1$
end if
return $\mathcal{V}$

information or continuously discarding hands), restraining itself to a local minimum. To improve the learning efficiency of the model, we draw on the idea of intrinsic reward.

And at the same time, inspired by goal-oriented RL, we analyze the goal of a Hanabi game. It can be concluded that the **long-term goal** is to play all suits, from 1 to 5 and the **short-term goal** is to play the following five cards on the current game board. For example, if we have "R1, Y2, G1, W1, B2", then our next short-term goal is to play one "R2, Y3, G2, W2, B3".

Based on the ideas above, We design the Goal-Oriented Intrinsic Reward module (GOIR) as an attachment to the backbone RL agent. Specifically, for actions of revealing cards, we hope the agent can provide new information instead of information already known to the target player. Meanwhile, to keep the game going, the agent should reveal cards that can be played, which are rewarded. For discarding cards, the action of discarding cards that are already in the firework is rewarded, otherwise punished. Lastly, the reckless actions of playing the wrong cards are punished. Note that intrinsic rewards don't change extrinsic rewards given by the environment. The algorithm for calculating GOIR is shown in Alg. 2.

4 Experiments

4.1 Configurations

To ensure the reproducibility and comparability of our results, we use the Hanabi Learning Environment (HLE) for all experimentation. For further details regarding Hanabi and the self-play part of the Hanabi challenge, you can refer to e Bard et al. [1].

We choose Rainbow DQN as the backbone model to host our attachments. The reasons are threefold. First, the DQN-based algorithms are broadly used and are suitable for discrete reinforcement learning problems. Second, Rainbow DQN combines the majority of improvements in the DQN family and has achieved decent scores in the Hanabi self-play setting, which is challenging enough as the baseline model. Third, the model is easy to modify. The HCIC module completes the hand card observation and sends it to RL algorithms, and we update the Q-value networks with the summation of the intrinsic reward from GOIR and the extrinsic reward from the environment. Other RL algorithms such as Policy-Based RL and Actor-Critic RL can be modified in the same way.

In the following experiments, we fix the Q-value net to a two-layer linear network with a dimension of 512. The replay buffer capacity is set to 50000, with a minimal size of 500. The discount factor γ is 0.99, and the ϵ-decay period is 1000. The parameters will update every 500 steps, using the Adam optimizer with the learning rate of 2.5×10^{-5}.

4.2 Ablation Study for HCIC and GOIR

We conduct the ablation study for our proposed HCIC and GOIR modules according to the above configurations. Due to tight computational resources, we train each model within 1000 iterations, and

Algorithm 2 Goal-Oriented Intrinsic Reward

Input: Current player's observation in the current state, o; The action the current player takes in the current state, a;
Output: The intrinsic reward for the action, r;

$r \leftarrow 0$
if $a.type$ == `REVEAL_COLOR` or $a.type$ == `REVEAL_RANK` **then**
    $n_1 \leftarrow$ COUNTREPEATREVEAL(o, a) // Count the number of cards n_1 that are repeatedly revealed
    $r \leftarrow r - 0.1 \times n_1$
    $n_2 \leftarrow$ COUNTREVEALPLAY(o, a) // Count the number of revealed cards n_2 that can be played according to the current firework
    $r \leftarrow r + 0.2 \times n_2$
else if $a.type$ == `DISCARD` **then**
    if DISCARDUSELESS(o, a) == $true$ **then** // The same card is in the current firework
        $r \leftarrow r + 0.2$
    else
        $r \leftarrow r - 0.2$
    end if
else if $a.type$ == `PLAY` **then**
    if PLAYWRONG(o, a) **then** // It is an unsuccessful play
        $r \leftarrow r - 0.3$
    end if
end if
return r

in each iteration, we run 10000 steps, which is enough for us to examine it. The tabular results are shown in Tab. 1. At the same time, we also plot the validation results for every hundred iterations, which is generated by averaging total rewards from 100 validation games. The results are displayed in Fig. 3.

Players	**2**	**3**	**4**
Rainbow (Baseline)	**13.20**	9.90	4.85
Rainbow w. GOIR	12.99	**11.78**	**11.00**
Rainbow w. HCIC	12.73	10.37	9.47
Rainbow w. GOIR & HCIC	12.88	10.85	10.30

Table 1: Experiment results of Rainbow model with and without our modules. The best results are marked as bold.

We find that in the 2-player self-play Hanabi game, HCIC and GOIR only marginally improve the performance. However, for 3 and 4 players, the gap between them gradually widens. The Rainbow w. GOIR model outperforms others, followed by the Rainbow w. GOIR & HCIC model and the Rainbow w. HCIC model. All of them exceed the baseline model.

Here we would like to make a detailed analysis of the results. First, we illustrate the effectiveness of the HCIC module. The module receives a fixed length of global action observation. When the game includes more players, the proportion of teammate's actions will increase, leading to more helpful information for inferring the agent's hand card knowledge. It explains why the HCIC-supported model is more effective than the baseline model as the number of players increases. Second, we explain how the GOIR module affects the policy-learning process. The complexity of multi-agent collaboration rises with more cooperators included, which is hard for common RL algorithms to learn. However, GOIR orients the models to act toward the same sub-goal, so that the models can converge to a better cooperation strategy with less time. Regarding the phenomenon that the model with HCIC and GOIR does not perform as well as the model using solely the GOIR module, we conjecture that this may be related to the poor quality of the training data for the HCIC module. The revealed actions generated from the baseline model may lack a theory of mind process, which is hard for the HCIC to predict their intentions. We believe that the performance of the final result would be much better if the human player dataset were used to train the HCIC model.

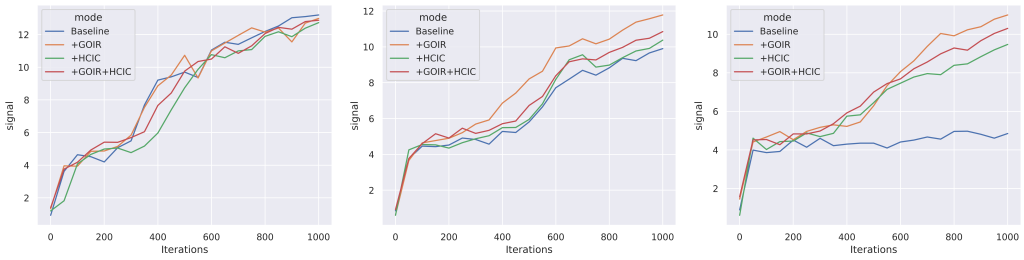


Figure 3: Results of ablation study in 2-player, 3-player, and 4-player games at 1000 iterations. We evaluate each model every 100 iterations, by averaging the reward from the 100 validation games.

Figure 4: Reward results in 2-player, 3-player, and 4-player games in 3000 iterations. The orange lines represent the results of GOIR-aided module, while the blue lines represent the baseline results.

4.3 Further Experiments for GOIR

We hope to verify that Goal-Oriented Intrinsic Reward can alleviate the problem of model exploration bias brought by sparse reward and accelerate model training at an early stage. Therefore, we continue the experiments for the baseline model and the model with GOIR module to 3000 iterations. The average reward results are shown in Fig. 4. We can see that GOIR effectively improves the model performance in the first 1000 iterations. The influence on the model gradually becomes smaller in the later stage. This observation is consistent with the characteristic of intrinsic reward. Therefore, we further explore the role of the GOIR module on the exploration ability of the model in the first 1000 iterations.

We compute the probability of chosen actions in iterations 10, 20, 50, 200, 500, and 1000. The probability serves as a measure of action selection preferences. Theoretically, the environment reward solely encourages playing, while with our GOIR module, the model will adjust its strategy according to the current short-term goal, leading to diverse exploration and fast convergence.

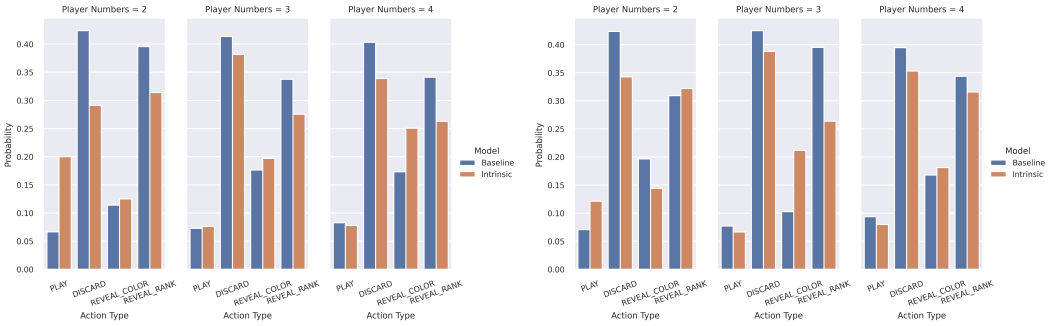
The results of our experiments lie in Fig. 5. Six sub-graphs record the distribution of the action probability of two models in different iterations.

As shown in Fig. 5a and Fig. 5c, in early training iterations, the variance of the action distribution derived from the GOIR-aided model is smaller than that without. In particular, the GOIR module design significantly reduces the unreasonably high number of `REVEAL_RANK` actions, demonstrating its encouragement for extensive exploration. From 500 to 1,000 iterations, we could see that model with intrinsic reward tends to play more frequently, which indicates better learning. We also notice that the improved model has a more stylized strategy (favoring `REVEAL_RANK` over `REVEAL_COLOR`). This characteristic makes the behavior of the agent better predictable and improves interpretability.

5 Conclusion and Future Work

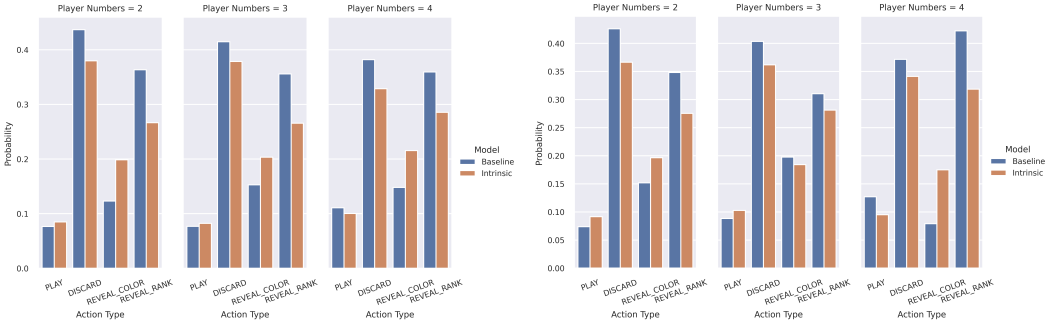
In this paper, we proposed two novel plug-in modules that can be attached to arbitrary RL agents to boost their performance on the Hanabi challenge. We conduct experiments with the Rainbow DQN as the backbone in the self-play setting, and our design substantially improves the average reward of the original model, especially for 3-player and 4-player games.

While these are encouraging steps, there is more work to do. In particular, the model using both plug-ins does not perform as well as expected. In the previous analysis, we believe this is related to the low quality of training data for the HCIC module. Therefore, we need to collect more effective



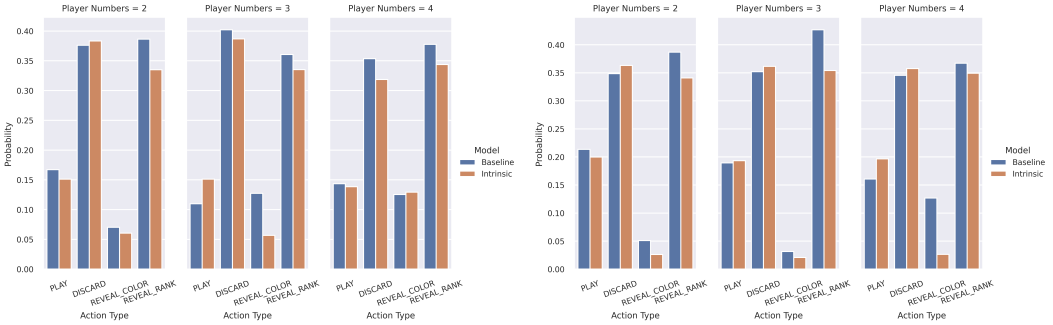
(a) Action distribution at iteration 10.

(b) Action distribution at iteration 20.



(c) Action distribution at iteration 50.

(d) Action distribution at iteration 200.



(e) Action distribution at iteration 500.

(f) Action distribution at iteration 1000.

Figure 5: Distribution of action selection between the baseline model and GOIR-attached model in the early stage of training (0-1000 iterations). The results reveal that our proposed intrinsic reward can promote exploration and accelerate training.

data to better train the HCIC module. We will perform more thorough experiments until the models converge.

Furthermore, we will explore the other-play setting. Our primitive plan is to conduct few-shot finetuning on the HCIC module in order to adapt it to partners with different strategies, imitating human behavior when they gradually adapt to new teammates after a game or two. Other attempts might also be made during experiments.

Author contributions

In this course project, Yilue Qian designed, implemented, trained the HCIC module, and wrote related sections. Yuxuan Luo proposed to solve the Hanabi problem with plug-in modules and wrote the basic framework, designed the GOIR module, ran corresponding experiments, visualized the results, and completed Section 3.2 and Section 4 of the report. Haoxiang Yang implemented the GOIR module, wrote the related description and the algorithm, complemented the code framework, ran the experiments, and analyzed the results. Siwen Xie did the primal research on literature, wrote Sections 1 and 2 of the report, and edited the presentation video. All of them surveyed related work, and together discussed and proposed the methods.

References

[1] Nolan Bard, Jakob N Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020. 2, 3, 6

[2] Simon Baron-Cohen, Alan M Leslie, and Uta Frith. Does the autistic child have a "theory of mind"? *Cognition*, 21(1):37–46, 1985. 2

[3] Rodrigo Canaan, Xianbo Gao, Youjin Chung, Julian Togelius, Andy Nealen, and Stefan Menzel. Evaluating rl agents in hanabi with unseen partners. In *AAAI'20 Reinforcement Learning in Games Workshop*, 2020. 3

[4] Christopher Cox, Jessica De Silva, Philip Deorsey, Franklin HJ Kenter, Troy Retter, and Josh Tobin. How to make the perfect fireworks display: Two strategies for hanabi. *Mathematics Magazine*, 88(5):323–336, 2015. 3

[5] Jakob Foerster, Francis Song, Edward Hughes, Neil Burch, Iain Dunning, Shimon Whiteson, Matthew Botvinick, and Michael Bowling. Bayesian action decoder for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 1942–1951. PMLR, 2019. 3

[6] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *2015 aaai fall symposium series*, 2015. 4

[7] Esther Herrmann, Josep Call, María Victoria Hernández-Lloreda, Brian Hare, and Michael Tomasello. Humans have evolved specialized skills of social cognition: The cultural intelligence hypothesis. *science*, 317(5843):1360–1366, 2007. 1

[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 *Thirty-second AAAI conference on artificial intelligence*, 2018. 3

[9] Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob Foerster. "other-play" for zero-shot coordination. In *International Conference on Machine Learning*, pages 4399–4410. PMLR, 2020. 3

[10] Tejas D. Kulkarni, Karthik R. Narasimhan, Ardavan Saeedi, and Joshua B. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3682–3690, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819. 4

[11] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. 4

[12] A. O'Dwyer. Github - quuxplusone/hanabi: framework for writing bots that play hanabi. `https://github.com/Quuxplusone/Hanabi`. Accessed 19 September 2018. 3

[13] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2778–2787. JMLR.org, 2017. 4

[14] David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and brain sciences*, 1(4):515–526, 1978. 1

[15] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2d2: repeatable and reliable detector and descriptor. *arXiv preprint arXiv:1906.06195*, 2019. 4

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[17] J. Wu. Github - wuthefwasthat/hanabi.rs. https://github.com/WuTheFWasThat/hanabi.rs. Accessed 29 November 2018. 3