Artificial Intelligence 280 (2020) 103216

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

The Hanabi challenge: A new frontier for AI research

Nolan Bard[a,*,1], Jakob N. Foerster[b,1,2], Sarath Chandar[c], Neil Burch[a], Marc Lanctot[a], H. Francis Song[d], Emilio Parisotto[e,3], Vincent Dumoulin[c], Subhodeep Moitra[c], Edward Hughes[d], Iain Dunning[d], Shibl Mourad[f], Hugo Larochelle[c], Marc G. Bellemare[c], Michael Bowling[a]

[a] *DeepMind, Edmonton, Canada*
[b] *University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland*
[c] *Google Brain, Montreal, Canada*
[d] *DeepMind, London, United Kingdom of Great Britain and Northern Ireland*
[e] *Carnegie Mellon University, Pittsburgh, United States of America*
[f] *DeepMind, Montreal, Canada*

A R T I C L E I N F O

Article history:
Received 16 January 2019
Received in revised form 19 October 2019
Accepted 18 November 2019
Available online 27 November 2019

Keywords:
Multi-agent learning
Challenge paper
Reinforcement learning
Games
Theory of mind
Communication
Imperfect information
Cooperative

A B S T R A C T

From the early days of computing, games have been important testbeds for studying how well machines can do sophisticated decision making. In recent years, machine learning has made dramatic advances with artificial agents reaching superhuman performance in challenge domains like Go, Atari, and some variants of poker. As with their predecessors of chess, checkers, and backgammon, these game domains have driven research by providing sophisticated yet well-defined challenges for artificial intelligence practitioners. We continue this tradition by proposing the game of Hanabi as a new challenge domain with novel problems that arise from its combination of purely cooperative gameplay with two to five players and imperfect information. In particular, we argue that Hanabi elevates reasoning about the beliefs and intentions of other agents to the foreground. We believe developing novel techniques for such theory of mind reasoning will not only be crucial for success in Hanabi, but also in broader collaborative efforts, especially those with human partners. To facilitate future research, we introduce the open-source Hanabi Learning Environment, propose an experimental framework for the research community to evaluate algorithmic advances, and assess the performance of current state-of-the-art techniques.

1. Introduction

Throughout human societies, people engage in a wide range of activities with a diversity of other people. These multi-agent interactions are integral to everything from mundane daily tasks, like commuting to work, to operating the

* Corresponding author.
E-mail addresses: nbard@google.com (N. Bard), jnf@fb.com (J.N. Foerster), chandar@google.com (S. Chandar), burchn@google.com (N. Burch), lanctot@google.com (M. Lanctot), songf@google.com (H.F. Song), eparisot@cs.cmu.edu (E. Parisotto), vdumoulin@google.com (V. Dumoulin), smoitra@google.com (S. Moitra), edwardhughes@google.com (E. Hughes), iaindunning@gmail.com (I. Dunning), shibl@google.com (S. Mourad), hugolarochelle@google.com (H. Larochelle), bellemare@google.com (M.G. Bellemare), bowlingm@google.com (M. Bowling).
[1] Equal contribution.
[2] Work done at DeepMind.
[3] Work done at Google Brain.

https://doi.org/10.1016/j.artint.2019.103216

organisations that underpin modern life, such as governments and economic markets. With such complex multi-agent interactions playing a pivotal role in human lives, it is desirable for artificially intelligent agents to also be capable of cooperating effectively with other agents, particularly humans.

Multi-agent environments present unique challenges relative to those with a single agent. In particular, the ideal behaviour for an agent typically depends on how the other agents act. Thus, for an agent to maximise its utility in such a setting, it must consider how the other agents will behave, and respond appropriately. Other agents are often the most complex part of the environment: their policies are commonly stochastic, dynamically changing, or dependent on private information that is not observed by everyone. Furthermore, agents generally need to interact while only having a limited time to observe others.

While these issues make inferring the behaviour of others a daunting challenge for AI practitioners, humans routinely make such inferences in their social interactions using *theory of mind* [1,2]: reasoning about others as agents with their own mental states – such as perspectives, beliefs, and intentions – to explain and predict their behaviour.[4] Alternatively, one can think of theory of mind as the human ability to imagine the world from another person's point of view. For example, a simple real-world use of theory of mind can be observed when a pedestrian crosses a busy street. Once some traffic has stopped, a driver approaching the stopped cars may not be able to directly observe the pedestrian. However, they can reason about why the other drivers have stopped, and infer that a pedestrian is crossing.

In this work, we examine the popular card game Hanabi, and argue for it as a new research frontier that, at its very core, presents the kind of multi-agent challenges where humans employ theory of mind. Hanabi won the prestigious *Spiel des Jahres* award in 2013 and enjoys an active community, including a number of sites that allow for online gameplay [4,5]. Hanabi is a *cooperative* game of *imperfect information* for two to five players, best described as a type of team solitaire. The game's imperfect information arises from each player being unable to see their own cards (*i.e.* the ones they hold and can act on), each of which has a colour and rank. To succeed, players must coordinate to efficiently reveal information to their teammates, however players can only communicate though *grounded* hint actions that point out all of a player's cards of a chosen rank or colour. Importantly, performing a hint action consumes the limited resource of *information tokens*, making it impossible to fully resolve each player's uncertainty about the cards they hold based on this grounded information alone. For AI practitioners, this restricted communication structure also prevents the use of "cheap talk" communication channels explored in previous multi-agent research [6–8]. Successful play involves communicating extra information implicitly through the choice of actions themselves, which are observable by all players.

Hanabi is different from the adversarial two-player zero-sum games where computers have reached super-human skill, *e.g.*, chess [9], checkers [10], go [11], backgammon [12] and two-player poker [13,14]. In those games, agents typically compute an equilibrium policy (or equivalently, a strategy) such that no single player can improve their utility by deviating from the equilibrium. While two-player zero-sum games can have multiple equilibria, different equilibria are *interchangeable*: each player can play their part of different equilibrium profiles without impacting their utility. As a result, agents can achieve a meaningful worst-case performance guarantee in these domains by finding any equilibrium policy. However, since Hanabi is neither (exclusively) two-player nor zero-sum, the value of an agent's policy depends critically on the policies used by its teammates. Even if all players manage to play according to the same equilibrium, there can be multiple locally optimal equilibria that are relatively inferior.[5] For algorithms that iteratively train independent agents, such as those commonly used in the multi-agent reinforcement learning literature, these inferior equilibria can be particularly difficult to escape and so even learning a good policy for all players is challenging.

The presence of imperfect information in Hanabi creates another challenging dimension of complexity for AI algorithms. As has been observed in domains like poker, imperfect information entangles how an agent should behave across multiple observed states [17,18]. In Hanabi, we observe this when thinking of the policy as a *communication protocol*[6] between players, where the efficacy of any given protocol depends on the entire scheme rather than how players communicate in a particular observed situation. That is, how the other players will respond to a chosen signal will depend upon what other situations use the same signal. Due to this entanglement, the type of single-action exploration techniques common in reinforcement learning (*e.g.*, ϵ-greedy, entropy regularisation) can incorrectly evaluate the utility of such exploration steps as they ignore their holistic impact.

Humans appear to be approaching Hanabi differently than most multiagent reinforcement learning approaches. Even beginners with no experience will start signalling playable cards, reasoning that their teammates' perspective precludes them from knowing this on their own. Furthermore, beginners will confidently play cards that are only partially identified as playable, recognising that the intent in the partial identification is sufficient to fully signal its playability. This all happens on the first game, suggesting players are considering the perspectives, beliefs, and intentions of the other players (and expecting the other players are doing the same thing about them). While hard to quantify, it would seem that theory of

[4] Dennett [3] uses the phrase *intentional stance* to refer to this "strategy" for explanation and prediction.

[5] One such equilibrium occurs when players do not intentionally communicate information to other players, and ignore what other players tell them (historically called a pooling equilibrium in pure signalling games [15], or a babbling equilibrium in later work using cheap talk [16]). In this case, there is no incentive for a player to start communicating because they will be ignored, and there is no incentive to pay attention to other players because they are not communicating.

[6] In pure signalling games where actions are purely communicative, policies are often referred to as *communication protocols*. Though Hanabi is not such a pure signalling game, when we want to emphasise the communication properties of an agent's policy we will still refer to its communication protocol.

mind is a central feature in how the game is first learned. We can see further evidence of theory of mind in the descriptions of advanced conventions[7] used by experienced players. The descriptions themselves often include the rationale behind each "agreement" explicitly including reasoning about other players' beliefs and intentions.

> C should assume that D is going to play their yellow card. C must do something, and so they ask themselves: "Why did B give that clue?". The only reason is that C can actually make that card playable. [19]

Such conventions then enable further reasoning about other players' beliefs and intentions. For example, the statement that "C should assume that D is going to play their yellow card", is itself the result of reasoning that partial identification of a playable card is sufficient to identify it as playable.

From human play we can also see that the goal itself is multi-faceted. One challenge is to learn a policy for the entire team that has high utility. Most of the prior AI research on Hanabi has focused on this challenge, which we refer to as the *self-play* setting. Human players will often strive toward this goal, pre-coordinating their behaviour either explicitly using written guides or implicitly through many games of experience with the same players. As one such guide states, though, "Hanabi is very complicated, so it is impossible to write a guide on how to best solve each individual situation."[20]. Even if if such a guide existed it is impractical for human Hanabi players to memorise nuanced policies or expect others to do the same. However, humans also routinely play with *ad-hoc teams* that may have players of different skill levels and little or no pre-coordination amongst everyone on the team. Even without agreeing on a complete policy or a set of conventions, humans are still able to achieve a high degree of success. It appears that human efforts in both goals are aided by theory of mind reasoning, and AI agents with similar capabilities — playing well in both pre-coordinated self-play and in uncoordinated ad-hoc teams — would signal a useful advance for the field.

The combination of cooperation, imperfect information, and limited communication make Hanabi an ideal challenge domain for learning in both the self-play and ad-hoc team settings. In Section 2 we describe the details of the game and how humans approach it. In Section 3 we present the Hanabi Learning Environment open source code framework (Section 3.1) and guidelines for evaluating both the self-play (Section 3.2) and ad-hoc team (Section 3.3) settings. We evaluate the performance of current state-of-the-art reinforcement learning methods in Section 4. Our results show that although these learning techniques can achieve reasonable performance in self-play, they generally fall short of the best known hand-coded agents (Section 4.3). Moreover, we show that these techniques tend to learn extremely brittle policies that are unreliable for ad-hoc teams (Section 4.4). These results suggest that there is still substantial room for technical advancements in both the self-play and ad-hoc settings, especially as the number of players increases. Finally, we highlight connections to prior work in Section 5.

2. Hanabi: the game

Hanabi is a game for two to five players, best described as a type of cooperative solitaire. Each player holds a *hand* of four cards (or five, when playing with two or three players). Each card depicts a rank (**1** to **5**) and a colour (**red**, **green**, **blue**, **yellow**, and **white**); the deck (set of all cards) is composed of a total of 50 cards, 10 of each colour: three **1**s, two **2**s, **3**s, and **4**s, and finally a single **5**. The goal of the game is to play cards so as to form five consecutively ordered stacks, one for each colour, beginning with a card of rank **1** and ending with a card of rank **5**. What makes Hanabi special is that, unlike most card games, players can only see their partners' hands, and not their own.

Players take turns doing one of three actions: giving a hint, playing a card from their hand, or discarding a card. We call the player whose turn it is the *active player*.

Hints. On their turn, the active player can give a hint to any other player. A hint consists of choosing a rank or colour, and indicating to another player all of their cards that match the given rank or colour. Only ranks and colours that are present in the player's hand can be hinted for. For example, in Fig. 1, the active player may tell Player 2, "Your first and third cards are **red**." or "Your fourth card is a **3**." To make the game interesting, hints are in limited supply. The game begins with the group owning eight information tokens, one of which is consumed every time a hint is given. If no information tokens remain, hints cannot be given and the player must instead play or discard.

Discard. Whenever fewer than eight information tokens remain, the active player can discard a card from their hand. The discarded card is placed face up (along with any unsuccessfully played cards), visible to all players. Discarding has two effects: the player draws a new card from the deck and an information token is recovered.

Play. Finally, the active player may pick a card from their hand and attempt to play it. Playing a card is successful if the card is the next in the sequence of its colour to be played. For example, in Fig. 1 Player 2's action would be successful if they play their **yellow 3** or their **blue 1**; in the latter case forming the beginning of the blue stack. If the play is successful,

[7] We use the word *convention* to refer to the parts of a communication protocol or policy that interrelate. Technically, these can be thought of as constraints on the policy to enact the convention. Conventions can communicate information either with or without an explicit signal, viz., conventions not only specify situations when a signal is used and how a teammate responds to that signal, but the absence of a signal also indicates that players are not in one of these situations. While we will also refer to the conventions of a learned policy, note that this is merely a convenient abstraction to aid discussion. The learning agents we examine learn a policy without explicitly encoding conventions. More examples of human conventions in Hanabi will be discussed in Section 2.

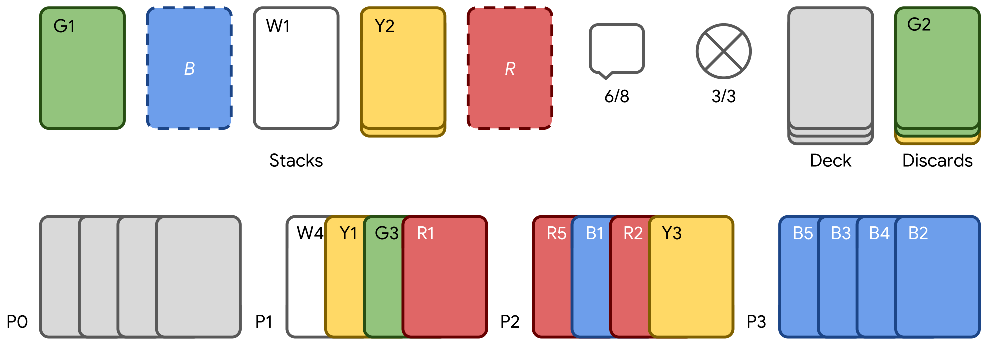


Fig. 1. Example of a four player Hanabi game from the point of view of player 0. Player 1 acts after player 0 and so on. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

the card is placed on top of the corresponding stack. When a stack is completed (the **5** is played) the players also receive a new information token (if they have fewer than eight). The player can play a card even if they know nothing about it; but if the play is unsuccessful, the card is discarded (without yielding an information token) and the group loses one life, possibly ending the game. In either circumstances, a new card is drawn from the deck.

Game over. The game ends in one of three ways: either because the group has successfully played cards to complete all five stacks, when three lives have been lost, or after a player draws the last card from the deck and every player has taken one final turn. If the game ends before three lives are lost, the group scores one point for each card in each stack, for a maximum of 25; otherwise, the score is 0.[8]

2.1. Basic strategy

There are too few information tokens to provide complete information (*i.e.*, the rank and colour) for each of the 25 cards that can be played through only the grounded information revealed by hints.[9] While the quantity of information provided by a hint can be improved by revealing information about multiple cards at once, the value of information in Hanabi is very context dependent. To maximise the team's score at the end of the game, hints need to be selected based on more than just the quantity of information conveyed. For example in Fig. 1, telling Player 3 that they hold four blue cards reveals more information than telling Player 2 that they hold a single rank-**1** card, but lower-ranked cards are more important early on, as they can be played immediately. A typical game therefore begins by hinting to players which cards are **1**s, after which those players play those cards; this both "unlocks" the ability to play the same-colour **2**s and makes the remaining **1**s of that colour useful for recovering information tokens as players can discard the redundant cards.

Players are incentivized to avoid unsuccessful plays in two ways: first, losing all three lives results in the game immediately ending with zero points; second, the card itself is discarded. Generally speaking, discarding all cards of a given rank and colour is a bad outcome, as it reduces the maximum achievable score. For example, in Fig. 1 both **green 2**s have been discarded, an effective loss of four points as no higher rank green cards will ever be playable. As a result, hinting to players that are at risk of discarding the only remaining card of a given rank and colour is often prioritised. This is particularly common for rank-**5** cards since there is only one of each colour and they often need to be held for a long time before the card can successfully be played.

2.2. Implicit communication

While explicit communication in Hanabi is limited to the hint actions, every action taken in Hanabi is observed by all players and can also implicitly communicate information. This implicit information is not conveyed through the impact that an action has on the environment (*i.e.*, what happens) but through the very fact that a player decided to take this action (*i.e.*, why it happened). This requires that players can reason over the actions that another player would have taken in a number of different situations, essentially reasoning over the intent of the agent. Human players often exploit such reasoning to convey more information through their actions. Consider the situation in Fig. 1 and assume the active player (Player 0) knows nothing about their own cards, and so they choose to hint to another player. One option would be to tell Player 1 about the **1**s in their hand. However, that information is not particularly actionable, as the **yellow 1** is not currently

[8] Note that while scoring zero when the team runs out of lives agrees with the game's published rules, much of the prior research on Hanabi (discussed in Section 5.1) scores this as the number of cards successfully played prior to failure.

[9] The deck contains 50 cards, 25 of which players aim to play. When the deck runs out the players will always hold at least 10 cards that cannot recover usable information tokens, but each player could hold one of the 25 cards they want to play. This means at most 17 cards can be discarded to recover information tokens. Combined with the eight initial information tokens, and the four recovered through completing colour stacks, this means players can hint at most 29 times during the game (and fewer times as the number of players increases).

playable. Instead, they could tell Player 1 about the **red** card, which is a **1**. Although Player 1 would not explicitly know the card is a **1**, and therefore playable, they could infer that it is playable as there would be little reason to tell them about it otherwise, especially when Player 2 has a **blue 1** that would be beneficial to hint. They may also infer that because Player 0 chose to hint with the colour rather than the rank, that one of their other cards is a non-playable **1**.

An even more effective, though also more sophisticated, tactic commonly employed by humans is the so-called "finesse" move. To perform the finesse in this situation, Player 0 would tell Player 2 that they have a **2**. By the same pragmatic reasoning as above, Player 2 could falsely infer that their **red 2** is the playable **white 2** (since both **green 2**s were already discarded). Player 1 can see Player 2's **red 2** and realise that Player 2 will make this incorrect inference and mistakenly play the card, leading Player 1 to question why Player 0 would have chosen this seemingly irrational hint. Even without established conventions, players could reason about this hint assuming others are intending to communicate useful information. Consequently, the only rational explanation for the choice is that Player 1 themselves must hold the **red 1** (in a predictable position, such as the most recently drawn card) and is expected to rescue the play. Using this tactic, Player 0 can reveal enough information to get two cards played using only a single information token. There are many other moves that rely on this kind of reasoning about intent to convey useful information (*e.g.*, bluff, reverse finesse) [19,20]. We will use *finesse* to broadly refer to this style of move.

3. Hanabi: the challenge

We propose using Hanabi as a challenging benchmark problem for AI. It is a multi-agent learning problem, unlike, for example, the Arcade Learning Environment [21]. It is also an imperfect information game, where players have asymmetric knowledge about the environment state, which makes the game more like poker than chess, backgammon, or Go. The cooperative goal of Hanabi sets it further apart from all of these other challenge problems, which have players competing against each other. This combination of partial observability and cooperative rewards creates unique challenges around the learning of policies and communication. Unlike signalling games [15] the communication in Hanabi does not use a separate channel, but rather mixes communication and environment actions. Finally, the resulting coordination and communication problem in Hanabi was designed to be challenging to human players.

How humans play the game suggest two different challenges presented by Hanabi. The first, likely easier, challenge is to learn a fixed policy for all players through *self-play*. In this case, the learning process is in control of all players, and the objective is to maximise the expected utility of the resulting joint policy. This represents the case where human players can pre-coordinate their play. The second challenge, *ad-hoc* team play, is learning to play with a set of unknown partners, with only a few games of interaction.

In the spirit of the proposed evaluation protocols [22] for the Arcade Learning Environment, which discuss recommendations for learning in Atari games, we will make some recommendations on how research should be carried out under each challenge.

3.1. Open source environment

To help promote consistent, comparable, and reproducible research results, we have released an open source Hanabi reinforcement learning environment called the *Hanabi Learning Environment* [23]. Written in Python and C++, the code provides an interface similar to OpenAI Gym [24]. It includes an environment state class which can generate observations and rewards for an agent, and can be advanced by one step given agent actions. An agent only needs to be able to generate an integer action, given an observation bitstring.

The default agent observation in our Hanabi environment includes card knowledge from previous hint actions, which includes both positive and negative information (*e.g.*, "the first card is **red**" also says all other cards are not **red**). This removes the memory task from the challenge, but humans tend to find remembering cards to be an uninteresting distraction, and the experimental results in Section 4 show that the game remains challenging without requiring agents to learn how to remember card knowledge. For researchers interested in memory, we provide the option to request a minimal observation, which does not include this remembered card knowledge.

For debugging purposes, the code includes an environment for a small game with only two colours, two cards per player, three information tokens, and a single life token. There is also a very small game with a single colour.

3.2. Challenge one: self-play learning

The self-play challenge is focused on finding a joint policy that achieves a high expected score entirely through self-play learning. A practical advantage of the Hanabi benchmark is that the environment is extremely lightweight, both in terms of memory and compute requirements, and fast (around 0.1 ms per turn on a CPU). It can therefore be used as a testbed for RL methods that require a large number of samples without causing excessive compute requirements. However, developing sample efficient algorithms is also an important goal for RL algorithms in its own right. With this in mind we propose two different regimes for the Hanabi benchmark:

Sample limited regime (SL). In the *sample limited regime*, we are interested in pushing the performance of sample-efficient algorithms for learning to play Hanabi. To that extent, we propose to limit the number of environment steps that the agent can experience to be at most 100 million. Here *environment steps* count the total number of turns taken during training. If the current episode does not end at 100 million steps, then we let the agent finish the episode before terminating training. This regime is similar to the evaluation scheme for Atari 2600 games proposed by Machado and colleagues [22]. The 100 million step limit was chosen based on the learning curves of the Rainbow agent presented in Section 4 to make sure that the current state-of-the-art agents can achieve a decent score in the given amount of time. Our intention with the sample limited regime is to highlight general techniques that efficiently learn to play capably. Consequently, we strongly encourage researchers not just to pursue techniques that notably improve performance at the 100 million step horizon, but to also demonstrate algorithms that achieve good performance with dramatically fewer samples, closer to how human teams might experience the game.

Unlimited regime (UL). In the *unlimited regime* there are no restrictions on the amount of time or compute. The unlimited regime describes research where the focus is on asymptotic performance, such as achieving high performance using large-scale computation. However, we encourage all work on Hanabi to include details about the compute requirements and run-time of their methods alongside the final results.

For every k-player game (where $k \in \{2, 3, 4, 5\}$), we recommend the following details be reported. Here the *best agent* is the training run with the highest average score under test conditions at the end of training, *e.g.*, when disabling exploration and picking the greedy action.

- Training curves for all random number generator seeds, highlighting the best agent.
- A histogram of game scores for the best agent and the percentage of perfect games.
- The mean and standard deviation of the performance of the best agent, computed as an average performance across at least 1000 trials (*i.e.*, full games). In the future, as performance increments become smaller this number should be increased to allow for significant results.

As we will show in Section 4, Hanabi is difficult for current learning algorithms. Even when using a large amount of data and computation time (UL regime), learning agents have trouble approaching the performance of hand-crafted rules in four player games, and fall far short of such rules for three and five players.

3.3. Challenge two: ad-hoc teams

The second challenge Hanabi poses is that of ad-hoc team play. The ultimate goal is agents that are capable of playing with other agents or even human players. For this, a policy which achieves a high score in self-play is of little use if it must be followed exactly by teammates. Good strategies are not unique, and a robust player must learn to recognise intent in other agents' actions and adapt to a wide range of possible strategies.

We propose to evaluate ad-hoc team performance by measuring an agent's ability to play with a wide range of teammates it has never encountered before. This is measured via the score achieved by the agent when it is paired with teammates chosen from a held-out pool of agents.[10] The composition of that pool should be such that the players exhibit diverse strategies, which can be hard-coded or learned by self-play.

We recommend the evaluated agents be given ten random self-play games of its ad-hoc teammates prior to play. While other alternatives may be more challenging (e.g., ten "warm-up" games, or average performance in the first ten games with no prior information), this focuses the challenge on an agent's ability to recognise intent in other agents' behaviour, as they can observe examples of the intended properly coordinated behaviour.

We recommend that the mean and standard deviation of the performance be reported across at least 1000 trials for each hold-out team. Specifically, each trial for a particular hold-out team should be evaluated by giving the agent a set of ten self-play games for the team, followed by the agent playing a *single* game in place of a player from the hold-out team in a random position, and finally resetting the agent so it does not retain memory across trials. The 1000 trials should also involve at least 100 different random sets of self-play games provided to the agent. These results should be reported along with mean and standard deviations of the performance of the hold-out teams in self-play (as a baseline comparison). We further recommend crosstables of the hold-out teams' performance when paired with each other as a method of assessing the diversity of the pool (e.g., see Fig. 5).

In the future we expect to see canonical agent pools of pre-trained or hard-coded self-play agents be made available for training and hold-out sets to allow for consistent comparisons.

4. Hanabi: state of the art

Hanabi presents interesting multi-agent learning challenges for both learning a good self-play policy and adapting to an ad-hoc team of players. In this section, we provide empirical evidence that both tasks are challenging for state-of-the-art learning algorithms, even with an abundance of computational resources.

[10] For two-player games forming a team is straightforward: we pair the evaluated agent with a random player from the pool and randomly permute their order in the team. For three to five-player games, the presence of more than one player from the pool is a potential confounding factor: it could be that the team fails because of the interaction between these players and through no fault of the evaluated agent. We therefore recommend to limit teams to two unique players — the evaluated agent, in one seat, and one agent from the pool — which is replicated in the remaining seats.

As in many domains, we would ultimately like to contrast current machine learning techniques with human performance. Unfortunately, empirically evaluating human performance is difficult due to a variety of factors: identifying and recruiting players, gathering a statistically meaningful quantity of data without having players suffer from fatigue, and mitigating other potentially confounding issues. For instance, from informally looking at online Hanabi game rooms, human players often abandon games part way through when they are unable to achieve perfect scores, causing bias in the data. Moreover, humans are also able to exploit other non-verbal cues such as eye movements [25] and even the time it takes for their partners to make decisions. Instead of human-level "self-play", which one might consider as the performance of an established team of humans, we contrast performance with hand-coded "bots" that make use of human-style conventions.

In the self-play setting, we examine the performance of two modern multi-agent reinforcement learning algorithms using deep learning for function approximation. We contrast these methods with a few of the best known hand-coded Hanabi "bots", and show that the learning agents fall somewhere between being competitive with hand-coded rules and being significantly outperformed by them. This failure is not just a matter of poor data efficiency: even with an intentionally unreasonable amount of experience and computation, the learning agents are outperformed by hand-coded rules.

We also show self-play results for a recent reinforcement learning algorithm that was designed to tackle the joint learning problem posed by Hanabi, explicitly reasoning about both public information and what other agents might have privately observed. This agent was only run for two player Hanabi, but achieved the best reported self-play performance in this case.

Finally, in the ad-hoc setting we show the performance for the two modern reinforcement learning algorithms, using agents from multiple independent runs. In this case, combinations of different learning agents only score slightly more than zero points.

We begin our empirical analysis with an overview of the different learning agents and rule-based bots used in our experiments.

4.1. Learning agents

Actor-Critic-Hanabi-Agent. The family of asynchronous advantage actor-critic algorithms [26] demonstrates stability, scalability and good performance on a range of single-agent tasks, including the suite of games from the Arcade Learning Environment [21], the TORCS driving simulator [27], and 3D first-person environments [28]. In the original implementation, the policy is represented by a deep neural network, which also learns a value function to act as a baseline for variance reduction. Experience is accrued in parallel by several copies of the agent running in different instantiations of the environment. Learning gradients are passed back to a centralised server which holds the parameters for a deep neural network.

Since the environment instantiations and the server interact asynchronously, there is a potential for the learning gradients to become stale, which impacts negatively on performance. ACHA uses the Importance Weighted Actor-Learner variant to address the stale gradient problem [29] by adjusting the stale off-policy updates using the V-trace algorithm. The variant has been successfully applied to the multi-agent task of Capture-the-Flag, achieving human-level performance [30,31]. ACHA also incorporates population-based training [32], providing automatic hyperparameter optimisation.

For our experiments, ACHA was run with a population size of 30 to 50 per run, 100 actors generating experience in parallel, and hyperparameter evolution over the learning rate and entropy regularisation weight. ACHA also used parameter-sharing across the different players in combination with an agent-specific ID that is part of the input. Parameter sharing is a standard method which increases learning speed, while the agent-specific ID allows for some level of specialisation between agents. Our neural architecture consisted of the following. All observations were first processed by an MLP with a single 256-unit hidden layer and ReLU activations, then fed into a 2-layer LSTM with 256 units in each layer. The policy π was a softmax readout of the LSTM output, and the baseline was a learned linear readout of the LSTM output. We refer to this method as the Actor-Critic-Hanabi-Agent (ACHA). To demonstrate what is possible in the unlimited regime, we trained ACHA agents for 20 billion steps. We estimate the computation took 100 CPU years for a population size of 30, however this is likely an overestimate as CPU usage was not saturated and jobs were able to be preempted by the cluster's scheduler.

Rainbow-Agent. Rainbow [33] is a state of the art agent architecture for deep RL on the Arcade Learning Environment. It combines some of the key innovations that have been made to Deep Q-Networks (DQN) [34] over the last few years, resulting in a learning algorithm that is both sample efficient and achieves high rewards at convergence. In our benchmark we use a multi-agent version of Rainbow, based on the Dopamine framework [35]. In our code the agents controlling the different players share parameters. Our Rainbow agent is feedforward and does not use any observation stacking outside of the last action, which is included in the current observation.

Our Rainbow agent uses a 2-layer MLP of 512 hidden units each to predict value distributions using distributional reinforcement learning [36]. Our batch size, *i.e.*, the number of experience tuples sampled from the replay buffer per update, is 32, and we anneal the ϵ of our ϵ-greedy policy to 0 over the course of the first 1000 training steps. We use a discount factor, γ, of 0.99 and apply prioritised sampling [37] for sampling from the replay buffer. Finally, our value distributions are approximated as a discrete distribution over 51 uniformly spaced atoms. Training a Rainbow agent for the sample limited regime of 100 million steps took approximately seven days using an NVIDIA V100 GPU.

BAD-Agent [38]. For the two player self-play setting we also include the results of the Bayesian Action Decoder since it constitutes state-of-the-art for the two-player unlimited regime. Rather than relying on implicit belief representations such as RNNs, the Bayesian Action Decoder (BAD) uses a Bayesian belief update that directly conditions on the current policy of the acting agent. In BAD all agents track a public belief, which includes everything that is common knowledge about the cards, including the posterior that is induced from observing the actions different agents take. BAD also explores in the space of deterministic policies, which ensures informative posteriors while also allowing for randomness required to explore. Further details for the BAD agent are provided in [38].

4.2. Rule-based approaches

For benchmarking we provide results of a number of independently implemented rule-based strategies. Unlike the previous learning agents, which learn a policy without explicitly encoding conventions for behaviour, these rule-based strategies directly encode conventions through their rules. These bots provide examples of the quality of play that can be achieved in Hanabi. We focus our benchmarking on the following rule-based strategies because they outperform other prior works on Hanabi (which we discuss in Section 5.1), most of which also exploit various prespecified rules in some manner.

SmartBot [39]. SmartBot is a rule-based agent that tracks the publicly known information about each player's cards. Tracking public knowledge allows SmartBot to reason about what other players may do, and what additional knowledge it gains from its specific view of the game. Among other things, this enables SmartBot to play/discard cards that its partners do not know that it knows are safe to play/discard, thereby preventing partners from wasting a hint to signal as much. However, this tracking assumes all other players are using SmartBot's policy. When this assumption does not hold, as in the ad-hoc team setting, SmartBot can fall into false or impossible beliefs. For example, SmartBot can believe one of its cards has no valid value as all possible cards are inconsistent with the observed play according to SmartBot's convention. Finally, note that SmartBot has a parameter specifying if it should attempt uncertain plays that may cost a life. Risking lives increases the frequency of perfect games while reducing average score, except in two player games where it is better on both criteria. Our SmartBot results only risks lives in the two player setting.

HatBot [40] and WTFWThat [41]. HatBot uses a technique often seen in coding theory and "hat puzzles". When giving hints, HatBot uses a predefined protocol to determine a recommended action for all other players (*i.e.*, play or discard for one of a player's cards). This joint recommendation is then encoded by summing the indices for the individual recommendations and using modular arithmetic. The encoded joint recommendation is mapped to different hints that HatBot could make, specifically, whether it reveals the colour or rank of a card for each other player. Since each player can view everyone's cards but their own, they can reconstruct the action recommended to them by figuring out what would have been recommended to the other players based on HatBot's convention, which HatBot assumes they know and use. Although this convention is not very intuitive for human players, it would still be possible for humans to learn and follow. Cox and colleagues also introduce an "information strategy" using a similar encoding mechanism to directly convey information about each player's cards (as opposed to a recommended action), however it requires additional bookkeeping that makes it impractical for humans to use. As originally proposed, both the recommendation and information strategies were tailored for playing 5-player Hanabi. However, a variant of the information strategy, called *WTFWThat* [41], can play two through five players. These benchmarks more provide a lower-bound for optimal play than a baseline suggestive of human performance.

FireFlower [42]. FireFlower implements a set of human-style conventions (detailed in Appendix A.1). The bot keeps track of both private and common knowledge, including properties of cards that are implied by the common knowledge of what the conventions entail. Using this, FireFlower performs a 2-ply search over all possible actions with a modelled probability distribution over what its partner will do in response, and chooses the action that maximises the expected value of an evaluation function. The evaluation function takes into account the physical state of the game as well as the belief state. For example, if there is a card in the partner's hand that is commonly known (due to inference from earlier actions) to be likely to be playable, then the evaluation function's output will be much higher if it is indeed playable than if it is not. FireFlower uses a few additional conventions for three and four players, but avoids the hat-like strategies in favour of conventions that potentially allow it to partner more naturally with humans. According to Fireflower's creator, it is designed with a focus on maximising the win probability, rather than average score.

4.3. Experimental results: self-play

Table 1 shows the experimental results of the baseline agents and state-of-the-art learning algorithms with each number of players.

First, note that neither the Rainbow nor ACHA agent reaches the performance of the best hand-coded agent in the two-player setting (SmartBot), and neither learning agent reaches the performance of any of the hand-coded agents with more than two players. There is a large performance gap in what is possible (as demonstrated by the hand-coded agents) and what state-of-the-art learning algorithms achieve. Even rule-based strategies that codify more human-like conventions achieve scores higher than the learning algorithms, particularly in the three and five player setting. Experienced human teams are generally considered stronger than such bots, suggesting the gap between these learning agents and superhuman performance in self-play is even larger still.

The final learning agent, BAD, only reports results for the two player setting, but achieves the best reported performance in that setting. Because BAD was only tested on two player Hanabi and a small synthetic game, we can not make a strong claim that BAD is an algorithmic improvement for a general class of games, rather than just being good at Hanabi. However, we find it suggestive that BAD's tracking of all agents' beliefs leads to a marked improvement in performance for an agent learning to play.

Table 1
Shown are the results for the three learning agents, Rainbow, ACHA and BAD, compared to the rule-based agents, SmartBot, FireFlower and HatBot, for different numbers of players in *self-play*. For each algorithm and number of players we show mean performance of the best agent followed by (standard error of the mean) and percentage of perfect (*i.e.*, 25 point) games. Error of the mean differs based on different number of evaluation games.

Regime	Agent	2P	3P	4P	5P
–	SmartBot	22.99 (0.00)	23.12 (0.00)	22.19 (0.00)	20.25 (0.00)
		29.6%	13.8%	2.076%	0.0043%
–	FireFlower	22.56 (0.00)	21.05 (0.01)	21.78 (0.01)	-
		52.6%	40.2%	26.5%	
–	HatBot	–	–	–	22.56 (0.06)
					14.7 %
–	WTFWThat	19.45 (0.03)	**24.20** (0.01)	**24.83** (0.01)	**24.89** (0.00)
		0.28%	**49.1**%	**87.2**%	**91.5**%
SL	Rainbow	20.64 (0.11)	18.71 (0.10)	18.00 (0.09)	15.26 (0.09)
		2.5 %	0.2%	0 %	0 %
UL	ACHA	22.73 (0.12)	20.24 (0.15)	21.57 (0.12)	16.80 (0.13)
		15.1%	1.1%	2.4%	0%
UL	BAD	**23.92** (0.01)	–	–	–
		58.6%			

Notice that both BAD and the ACHA agent require vast amounts of training data to establish good performance. As such, the self-play challenge, even for two player Hanabi, is fertile ground for further innovation. Furthermore, theory-of-mind-inspired methods still need to be scaled to settings with more than two players.

Comparing the more traditional RL approaches, the ACHA agent in the unlimited regime (using over 20 billion steps of experience for each learner in the population) achieved higher scores than Rainbow (using 100 million steps of experience) across all number of players. This quite naturally may be due to ACHA using more training experience, but may also be due to Rainbow's feed-forward network architecture with no history of past actions possibly making it harder to learn multi-step conventions. Both agents saw a decline in performance as the number of agents increased with Rainbow's more gradual, while ACHA saw a precipitous drop in performance with five players.

Fig. 2 shows training curves for one run of ACHA showing the performance of the multiple "agents" within the population. Note that these curves are linked together through parameter evolution and are not independent. In all but the four player setting, it appears ACHA has found a local minimum in policy space and it is unable to escape even with more training. However, parameter evolution is hiding the true extent of the local minima problem. Fig. 3 gives ACHA training curves for two and four player games when evolution is disabled, so each curve is an independent self-play trial, using the same fixed hyper-parameters found as the best in the earlier experiment. Here we can see that independent learning trials find a wide array of different local minima, most of which are difficult to escape. For example, in the two player setting, roughly a third of the independent agents are below 15 points and appear to no longer be improving.

We further find that even ACHA runs with similar final performance can learn very different conventions. For example, one agent uses colour hints to indicate that the 4th card of the teammate can likely be discarded, while another agent uses the different colour hints to indicate which card of the teammate can be played. Different agents also use the rank-hints to indicate playability of cards in different slots. Details examining specific examples of learned policies are in Appendix A.2.

In contrast to ACHA, the Rainbow agents exhibit low variance across different independent training runs, as shown by the learning curves in Fig. 4. In this case each line represents an independent training trial. In addition, Rainbow agents tend to converge to similar strategies, seemingly identifying the same local minima. In particular, Rainbow agents are 3-4 times less likely to hint for colour than ACHA, and when they do there is no evidence of specific conventions associated with the colour hinted. Instead all Rainbow agents we looked at primarily hint for rank, typically indicating that the most recent card is playable. See Appendix A.2 for details. We speculate that two factors contribute to this consistency across different runs. First, Rainbow has a one-step memory for the past action and no memory of past observations. This drastically reduces the space of possible strategies. Second, Rainbow is a value-based method and starts out with a high exploration rate. During the initial exploration, since agents fail to successfully finish any games without running out of lives, Q-values will tend towards zero. This might cause agents to learn from effectively the same starting Q-values, even across independent runs.

4.4. Experimental results: ad-hoc team play

The policies learned by the aforementioned ACHA, BAD and Rainbow agents are moderately effective in self-play. We now investigate these agents in the ad-hoc team setting, where teammates are using different conventions. In particular, we examine the performance of different ad-hoc teams of ACHA and Rainbow agents. Since we established in Section 4.3 that Rainbow agents learn nearly identical strategies across different runs, for the rest of the section we consider a population of agents made up of the best performing Rainbow agent and a pool of independently trained ACHA agents taken from the top ten agents according to final training-time performance from Fig. 3.

We do not use the hand-coded agents in the ad-hoc setting, for a mix of practical and technical reasons. As written, none of the hand-coded agents interface with the learning environment or are even written in the same language, with a

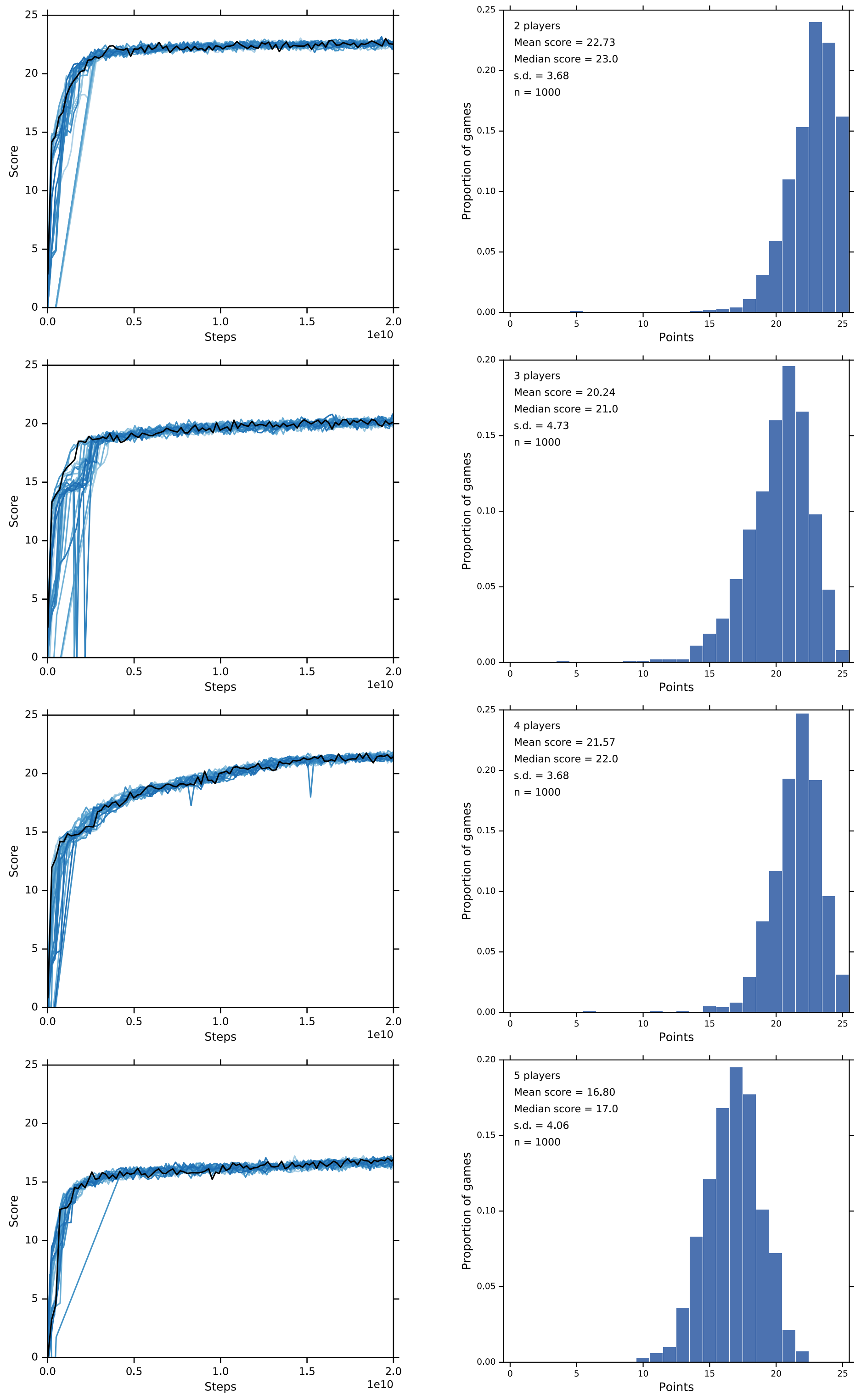


Fig. 2. ACHA results for two to five players, from top to bottom respectively. Performance curves (left) are training-time results from the current soft-max policy. Average scores and distributions (right) are test-time results from 1000 episodes generated using the greedy policy from the agent with the best training score.

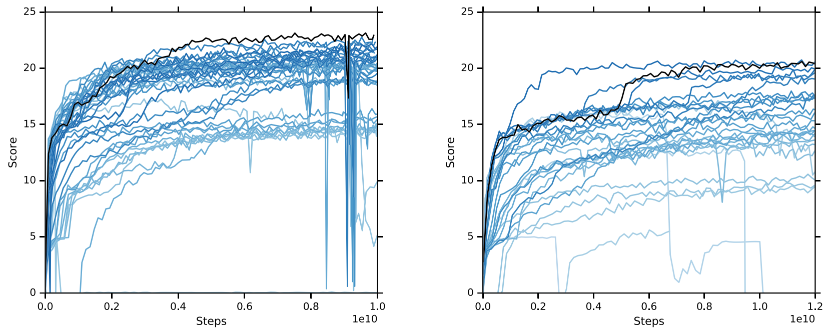


Fig. 3. ACHA results without evolution, using 50 independent runs for two players (left), and 30 independent runs for four players (right).

different non-trivial (and possibly error-prone) translation required to have the learning agents play with each hand-coded agent. We made an early effort to play games with combinations of ACHA agents and SmartBot, but these games always ended with SmartBot crashing because ACHA makes moves SmartBot assumes should never be made. Simple fixes only led to a score of zero points. Given the effort to connect different agents, and the complicated action coding of agents like HatBot and WTFWThat, we assume without further evidence that other combinations of hand-coded agents would have equally bad performance.

As neither of ACHA and Rainbow make use of sample play of their ad-hoc teammates, the ad-hoc team's performance will simply depend on the compatibility of the different protocols. The purpose of this section, then, is primarily to illustrate the difficulty in ad-hoc team play, and suggest a source for creating a diverse pool of agents for future evaluation.

Two players. Fig. 5a shows a crosstable of the agents' test performance in the ad-hoc setting. The entry for the i-th row and j-th column shows the mean performance of agent i playing in an ad-hoc team consisting of agent j, evaluated for 1000 games with a random player starting each game. The scores of 20 or greater along the diagonal entries show that the agents indeed perform well in self-play. However, when paired with other agents, performance drops off sharply, with some agents scoring essentially zero in any of these ad-hoc teams.

Four players. We observe analogous results when evaluating ad-hoc teams in the four player setting. We used the top ten ACHA agents from Fig. 3. Similar to the two player ad-hoc results, the entry for the i-th row and j-th column of Fig. 5b shows the mean performance of agent i when playing with an ad-hoc team consisting of three other agent j players, evaluated for 1000 games with a random player starting each game. As in the two player results, the agents fare relatively well in self-play but performance dramatically decreases once we introduce a second unique agent to the team.

Developing agents that can learn from, adapt, and play well with unknown teammates represents a formidable challenge.

5. Hanabi: related work

This challenge connects research from several communities, including reinforcement learning, game theory, and emergent communication. In this section, we discuss prior work on the game of Hanabi. We also briefly discuss notable work from these other communities.

5.1. Prior work on Hanabi AI

To the best of our knowledge, the earliest published work on Hanabi was in 2015. Osawa [43] described some of the unique elements of Hanabi, and showed that simulated strategies that try to recognise the intention of the other players performed better than a fixed set of static strategies in two-player Hanabi. Later in the same year, Cox et al. [40] developed the hat strategy described in Section 4.2.

Several studies have focused on techniques to achieve strong Hanabi play. First, van den Bergh et al. described fixed rules whose trigger thresholds were optimised by manual tuning via simulation [44]; the action selection is also improved during play using Monte Carlo planning. Walton-Rivers and colleagues [45] evaluated several rule-based and Monte Carlo tree search agents and observed a predictor version that modelled the other players performed better than bots without this ability. Finally, Bouzy [46] examined several heuristic players and found that combining search with the hat strategy yielded the best-performing agents. In the five-player game when ignoring the rule that hints cannot refer to an empty set of cards, they reported achieving 24.92 points and a perfect score 92% of the time on average. While some of these ideas are used in the agents that we describe in Section 4.2, the agents we benchmark have superior performance under the complete Hanabi rules to the numbers reported in these papers.

There are three related works that are not about (directly) increasing performance. Baffier and colleagues [47] examine the complexity of the generalised game and found optimal gameplay in Hanabi to be NP-hard, even with a centralised cheating player playing all seats with perfect information of all cards, including the order of cards in the deck. However, without a centralised cheating player, Hanabi is an instance of a decentralised Markov decision process (DEC-MDP) since

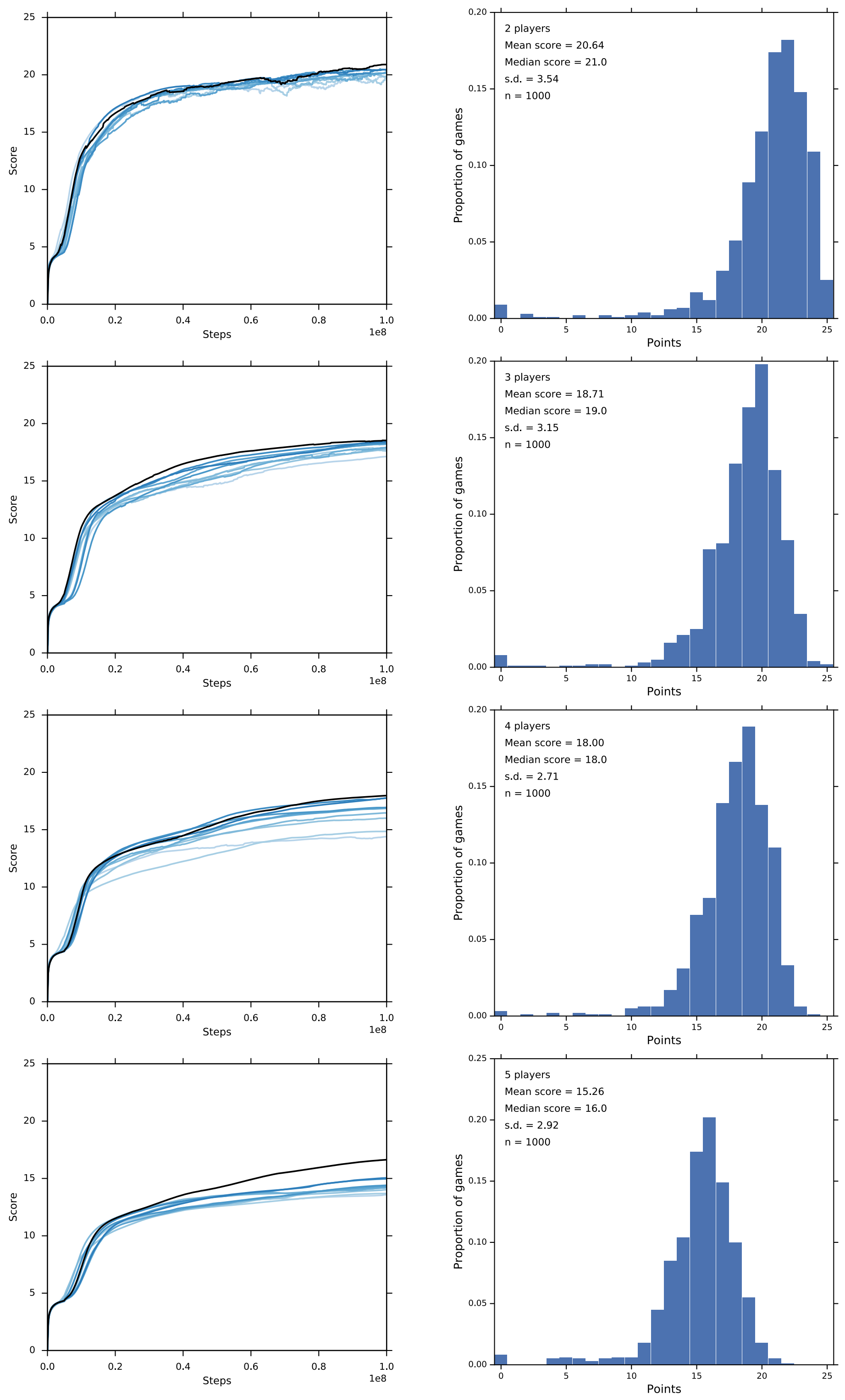
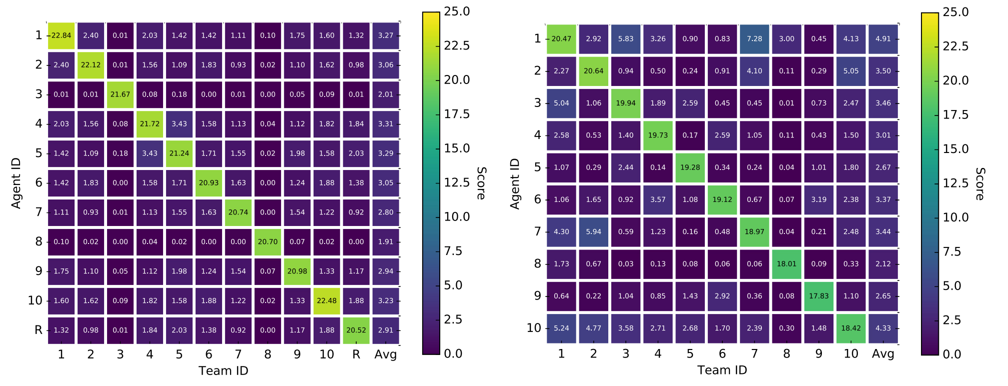


Fig. 4. Rainbow results for two to five players, from top to bottom respectively. Performance curves (left) are training-time results from the current policy. Average scores and distributions (right) are test-time results from 1000 episodes generated using the agent with the best training score. ϵ in ϵ-greedy for all agents is set to zero.



(a) Ad-hoc results for two players. (b) Ad-hoc results for four players.

Fig. 5. Ad-hoc team results. Teams were constructed using the 10 best independently trained ACHA agents (**1–10**; see Fig. 3) and the best Rainbow agent (**R**; see Fig. 4). Mean scores over 1000 trials are shown. Standard error of the mean was less than 0.09 and 0.255 for two and four players, respectively.

the players jointly observe the full state of the game. Bernstein and colleagues [48] showed that solving DEC-MDPs is in the nondeterministic exponential time (NEXP) complexity class, i.e., requiring exponential time even if P=NP. Liang and colleagues [49] compared the performance of artificial agents that implicate additional meaning through their hints with agents that simply hint to minimise entropy, and found that humans were more likely to believe the implicature-based agent was also a human. Eger and Martens [50] describe how to model knowledge and how it is revealed through actions using dynamic epistemic logic. These epistemic formalisations within game theory attempt to quantify what players know and assume others know, how players can reason about this knowledge, and what rationality means in this context [51]. These ideas are also reflected in the BAD agent [38], where they are combined with scalable deep multi-agent RL, and have resulted in the strongest two-player Hanabi agent.

Recently, there was a Hanabi competition that took place at the 2018 IEEE Computational Intelligence in Games Conference in Maastricht [52]. There were a total of five agents, three submissions and two samples . There were two tracks, called "Mirror" and "Mixed" which broadly match the two categories we propose in Section 3. Similarly to van den Bergh, the second-place player used a genetic algorithm to evolve a sequence of rules from a fixed rule set [53]. This agent achieved an average score of 17.52 points in the Mirror competition, while the first place agent, "Monte Carlo NN", achieved a score of 20.57. According to the website these scores are averaged across the results for two to five players. The winning agent, developed by James Goodman [54], won both the Mirror and Mixed tracks; it used Information Set Monte Carlo Tree Search [55] as a base algorithm coupled with a re-determinisation technique that re-samples consistent world states for everyone but the player acting at a node. This entry also used a neural network to represent a policy and value function trained via self-play. Goodman contrasts the self-play performance of this approach using different configurations with our results and prior works under each of the two through five player settings. Like many prior Hanabi agents, Goodman's approach exploited some fixed rules and conventions: selecting actions from a fixed set of nine rules and using the convention that cards are playable when hinted. In the Mixed track, the rollouts of other agents incorporated some agent modelling by using Bayesian predictions about which of a set of policies the other agents were using. Although Goodman's approach was fairly effective in the Mixed track, it is unclear how well it would fare with other agents, particularly those that are not built using published rules or rules encoded by the competition's code framework. Critically, like many prior works on Hanabi, the competition scored failed games (where all the lives were used) as the number of cards successfully played prior to failure, whereas we give such failed games a score of zero. While this distinction may have relatively little impact in self-play, as there is no risk of misinterpreting actions, our experiments suggested this has considerable impact in the ad-hoc teamwork setting.

5.2. Reinforcement Learning (RL) and multi-agent RL

While there has been much work in rule-based and search-based players using various heuristics based on domain knowledge, we are unaware of any prior approaches on learning to play Hanabi directly from experience given only the rules of the game. The framework for this approach is *reinforcement learning*, where an agent chooses actions in its environment, receives observations and rewards [56]. The goal is to learn a policy that achieves a high expected sum of rewards, *i.e.* score.

The setting with multiple reinforcement learning agents was first investigated in competitive games [57]. This was the start of the foundational work on *multiagent* reinforcement learning (MARL), which focused on algorithms and convergence properties in Markov/stochastic (simultaneous move) games. Independent learners in the cooperative case, even in these simpler game models, already face several coordination problems [58]. Several years of work on MARL gave rise to many algorithms, extensions, and analyses [59–63]. For more on multi-agent deep reinforcement learning, Hernandez-Leal and colleagues [64] provide a recent survey.

Model-free methods, which learn a direct mapping from observations to actions, are appealing in traditional RL because the agent need not understand the dynamics of the environment in order to act. In games, however, the perfect model (*i.e.*, the rules) is given. This leads to methods than can combine planning with RL, as first demonstrated by TD-Leaf [65] and TreeStrap [66]. Recently, Monte Carlo tree search [67] was combined with deep neural networks in AlphaGo [68], a computer Go-playing agent that was stronger than any preceding program and defeated the best human Go players. Shortly thereafter AlphaGo was surpassed by AlphaGo Zero, which used less knowledge [69], and then AlphaZero which also learned to play chess and shogi at super-human levels [70].

Many of the above successes were in applications of RL to perfect information games (*e.g.*, Go, chess, shogi). Imperfect-information environments offer new challenges since the world cannot be perfectly simulated: agents must reason about information that is not known to them. In the competitive case, such as poker, strong play required new algorithmic foundations [71,72,14,13].

Finally, there has been recent interest in the problem of emergent communication. These problems include an arbitrary or structured communication channel, and agents must learn to communicate to solve a cooperative problem. Algorithms have been developed to learn to solve riddles and referential games [6,73], gridworld games requiring coordination [74], object identification via question-and-answer dialog [75], and negotiation [7,8]. The main difference in Hanabi is that there is no "cheap-talk" communication channel: any signalling must be done through the actions played. It is therefore more similar in spirit to learning how to relay information to partners in the bidding round of bridge [76].

5.3. Pragmatics, beliefs, and agent modelling

Hanabi provides AI practitioners with an interesting multi-agent learning challenge where they can explore agent communication not just between artificial agents, but also with human partners. Understanding human communication seems important if artificial agents are to collaborate effectively with humans, as they will likely need to communicate with us on our terms. When communicating with natural language, humans exploit theory of mind through their use of *pragmatics*: conveying meaning based on not only what was literally said, but also what is implicated (*i.e.*, suggested) by the speaker based on the context [77]. By relying on a listener to disambiguate a speaker's intended meaning through mechanisms like pragmatic reasoning, communication can be more efficient [78], as in the case with humans hinting about playable cards in Hanabi. Notably, computational models of human pragmatic reasoning depend on speakers being cooperative and having the *intention* of communicating useful information [79].

There has been much work that proposes to model beliefs about the intentions or plans of other agents. Several formalisms have been proposed for this, such as I-POMDPs [80] and Bayesian games [81]. However, algorithms to compute exact solutions quickly become intractable for complex problems.

Classical models to predict human behaviour in games include ways to deal with imperfect rationality [82,83]. In recent years, several mechanisms have been proposed to learn these models using deep learning, from human data in one-shot games [84], learning players with expert features [85], and end-to-end prediction of fixed policies in gridworld games [2]. When predictions are used to exploit or coordinate, this is often called *opponent/teammate modelling* or more generally *agent modelling* [86]. Several MARL algorithms have been recently proposed that learn from models of others whether by using a training regime based on cognitive hierarchies [87], by defining a learning rule that shapes the anticipated learning of other agents [88], or by training architectures that incorporate agent identification as part of the objective and conditioning the policy on these predictions [89]. Lastly, an approach in poker (DeepStack) introduced function approximators that represent belief-conditional values and explicitly representing and reasoning about manipulating agents' beliefs [13], similar to the approach of BAD [38].

6. Conclusion

The combination of cooperative gameplay and imperfect information make Hanabi a compelling research challenge for machine learning techniques in multi-agent settings. We evaluate state-of-the-art reinforcement learning algorithms using deep neural networks and demonstrate that they are largely insufficient to even surpass current hand-coded bots when evaluated in a self-play setting. Furthermore, we show that in the ad-hoc team setting, where agents must play with unknown teammates, such techniques fail to collaborate at all. Theory of mind appears to play an important role in how humans learn and play Hanabi. We believe improvements to both learning in self-play and adapting to unknown teammates will help us understand better the role theory of mind reasoning might play for AI systems that learn to collaborate with other agents and humans. To promote effective and consistent comparison between techniques, we provide a new open source code framework for Hanabi and propose evaluation methodology for practitioners.

Declaration of competing interest

The authors confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

We would like to thank many people: Matthieu d'Epenoux of Cocktail Games and Antoine Bauza, who designed Hanabi, for their support on this project; Alden Christianson for help with coordinating across three different time zones, and discussions with Cocktail Games; Angeliki Lazaridou for discussion and feedback on pragmatics and theory of mind; Ed Lockhart for many discussions on small Hanabi-like games and belief-based reasoning; Daniel Toyama for assistance with writing clear, readable code for the Hanabi research environment used in our experiments; Kevin Waugh for helpful comments and feedback; David Wu for providing details on the FireFlower bot; and Danny Tarlow for early feedback on the project.

Appendix A

A.1. FireFlower details

Implemented conventions include the following:

- Hints generally indicate play cards, and generally newer cards first.
- Hints "chain" on to other hints, e.g., if A hints to B a playable **red 2** as **red**, then B might infer that it is indeed the **red 2**, and then hint back to A a **red 3** in A's hand as red, expecting A to believe it is a **red 3** to play it *after* B plays its **red 2**.
- When discarding, discard provably useless cards, otherwise the oldest "unprotected" card.
- Hints about the oldest unprotected card "protect" that card, with many exceptions where it instead means play.
- Hints about garbage cards indicate "protect" cards older than it.
- Deliberately discarding known playable cards signals to the partner that they have a copy of that card (with enough convention-based specificity on location that they may play it with absolutely no actual hints).
- In many cases, hints about cards that have already been hinted change the belief about those cards in various ways such that the partner will likely change what they do (in accordance with the broader heuristic that one usually should give a hint if and only if they would like to change the partner's future behaviour).

A.2. Conditional probability tables

Tables A.2–A.6 show conditional probability summaries of the learned policies from different runs of ACHA and Rainbow in the two player game. See Section 4 for a discussion of these policy summaries.

Table A.2
Conditional action probabilities for a first two player Rainbow agent.

$P_{\%}(a_{t+1}\|a_t)$		Discard					Play					Hint colour					Hint rank				
		1	2	3	4	5	1	2	3	4	5	R	Y	G	W	B	1	2	3	4	5
a_t		a_{t+1}																			
Discard	1	4	4	3	1	10	2	1	1	0	1	1	6	2	2	1	12	14	11	12	11
	2	2	6	5	2	12	1	0	0	0	0	1	4	2	2	1	14	18	13	11	5
	3	2	6	11	2	13	1	1	1	1	2	1	3	2	2	1	18	17	11	4	1
	4	1	2	5	4	19	0	1	0	0	0	1	6	1	6	2	18	14	10	6	2
	5	1	1	1	0	30	1	0	0	0	0	0	3	2	2	2	10	12	16	13	6
Play	1	4	5	3	1	18	4	2	1	1	2	0	2	1	1	1	13	10	10	11	9
	2	4	6	5	1	22	3	2	1	0	1	0	2	1	1	0	11	12	11	11	6
	3	4	5	2	1	26	4	2	2	1	2	0	3	1	1	1	7	6	8	14	10
	4	3	3	4	0	0	3	5	1	1	1	0	3	0	0	0	49	10	1	12	3
	5	2	4	5	1	26	2	1	0	0	0	0	3	1	1	0	12	14	12	10	5
Hint Colour	R	6	31	25	9	3	0	0	0	0	1	9	4	2	5	2	1	2	0	1	1
	Y	19	31	12	5	8	1	0	1	0	0	1	7	2	3	1	2	3	2	2	2
	G	20	30	18	1	9	2	0	1	0	0	0	4	3	1	1	0	2	2	2	2
	W	18	18	16	10	15	0	0	0	0	0	1	4	1	8	0	2	2	1	2	1
	B	12	8	11	6	37	2	0	0	0	0	0	2	2	4	3	1	1	3	3	4
Hint Rank	1	3	3	6	8	1	8	9	0	8	50	0	0	0	0	0	3	0	1	0	0
	2	3	6	1	0	0	8	23	0	0	52	0	0	0	0	0	3	1	0	0	0
	3	1	5	2	0	1	11	20	10	0	47	0	0	0	0	0	0	1	1	0	0
	4	1	4	3	1	1	13	22	6	1	37	0	0	0	0	0	6	2	1	1	0
	5	2	4	0	0	2	29	19	7	2	28	0	1	0	0	0	0	1	1	1	2
$P_{\%}(a_t)$		3	5	4	2	14	5	7	2	1	17	0	2	1	1	1	9	8	7	7	4

Table A.3
Conditional action probabilities for second two player Rainbow agent.

$P_{\%}(a_{t+1}\|a_t)$		Discard					Play					Hint colour					Hint rank				
		1	2	3	4	5	1	2	3	4	5	R	Y	G	W	B	1	2	3	4	5
a_t		a_{t+1}																			
Discard	1	4	1	1	1	22	1	1	1	0	1	1	2	3	2	0	10	21	9	10	7
	2	2	3	1	0	19	1	1	1	1	1	2	2	3	3	0	9	14	19	11	9
	3	4	1	6	1	19	2	0	0	0	2	3	3	4	3	3	21	11	9	2	5
	4	3	1	1	2	29	1	0	1	0	0	2	3	3	6	2	12	15	11	5	4
	5	1	1	0	0	32	1	0	0	0	0	0	3	2	4	0	10	13	14	11	5
Play	1	4	2	1	1	24	3	2	2	0	3	1	1	1	1	0	11	12	10	11	10
	2	4	2	1	1	28	3	2	1	0	2	0	1	1	1	0	12	14	12	8	7
	3	4	2	1	0	23	4	2	1	0	4	1	2	1	2	0	11	13	7	12	9
	4	3	2	0	0	8	4	2	1	0	6	0	0	1	1	0	35	23	5	5	4
	5	3	1	1	1	33	2	2	1	0	1	0	2	2	1	0	11	15	12	8	5
Hint	R	15	23	8	9	5	1	0	0	0	0	11	0	7	4	4	3	1	1	1	5
Colour	Y	28	15	11	9	13	1	1	0	0	1	1	4	3	1	0	2	2	3	2	3
	G	22	19	5	4	17	1	1	1	0	1	3	2	8	2	0	2	1	2	2	6
	W	9	7	2	4	60	1	1	1	0	1	2	2	3	4	0	1	1	1	1	2
	B	17	7	21	17	4	1	0	0	1	5	7	0	0	1	10	1	4	1	2	0
Hint	1	2	3	2	6	1	9	11	6	5	54	0	0	0	0	0	0	0	0	0	0
Rank	2	8	2	1	0	1	7	21	0	0	48	0	0	0	0	0	8	3	0	0	0
	3	2	2	1	0	1	12	18	11	0	48	0	0	0	0	0	2	1	1	0	0
	4	1	1	0	0	1	24	14	9	1	43	0	0	0	0	0	0	0	1	1	1
	5	8	4	4	2	2	15	19	7	2	23	0	1	2	0	0	2	1	1	3	3
$P_{\%}(a_t)$		4	2	1	1	19	5	7	3	1	17	0	1	2	2	0	8	9	8	6	4

Table A.4
Conditional action probabilities for third two player Rainbow agent.

$P_{\%}(a_{t+1}\|a_t)$		Discard					Play					Hint colour					Hint rank				
		1	2	3	4	5	1	2	3	4	5	R	Y	G	W	B	1	2	3	4	5
a_t		a_{t+1}																			
Discard	1	9	1	2	1	15	1	1	0	0	0	3	2	2	0	0	19	19	11	9	5
	2	8	4	2	1	16	1	0	0	0	0	6	2	2	1	1	16	20	9	8	2
	3	6	1	2	0	24	1	1	0	0	1	3	3	2	0	1	12	16	14	7	4
	4	7	1	2	2	27	1	1	0	1	1	3	3	2	1	1	14	13	10	7	3
	5	2	0	1	0	32	1	0	0	0	0	3	3	3	0	0	11	12	14	13	4
Play	1	9	1	2	1	21	4	3	1	1	2	1	1	0	0	1	10	12	11	12	8
	2	4	1	2	1	25	5	5	1	1	3	1	3	1	0	1	6	7	9	13	12
	3	8	2	2	1	22	2	2	1	1	3	1	1	0	0	0	17	13	10	10	4
	4	5	1	1	2	2	6	3	2	1	8	0	1	0	0	0	33	13	1	15	3
	5	7	1	1	1	29	2	1	1	0	1	2	2	1	0	0	12	15	11	10	4
Hint	R	49	15	20	3	4	0	0	0	0	0	0	1	0	0	1	0	3	1	0	1
Colour	Y	35	5	8	3	12	3	4	1	1	0	1	4	2	0	3	2	2	2	5	7
	G	12	4	7	3	55	1	1	0	0	0	1	3	3	1	2	2	1	1	2	1
	W	14	13	8	9	27	0	0	0	1	0	2	0	2	12	1	0	2	4	2	1
	B	31	2	13	3	12	0	2	1	1	1	2	3	3	1	14	2	1	2	2	3
Hint	1	4	2	5	8	2	10	0	9	6	47	0	0	0	0	0	4	0	1	0	0
Rank	2	9	0	1	0	0	17	0	15	1	48	0	0	0	0	0	4	2	0	0	0
	3	2	1	1	0	0	14	9	19	0	50	0	0	0	0	0	1	1	1	0	0
	4	3	3	2	1	1	19	14	8	1	38	0	0	0	0	0	5	3	1	2	1
	5	2	1	1	0	2	31	14	8	1	30	1	1	0	0	1	1	0	1	2	3
$P_{\%}(a_t)$		7	1	2	1	16	7	3	5	1	17	1	2	1	0	0	9	9	7	7	3

Table A.5
Conditional action probabilities for first two player ACHA agent.

$P_{\%}(a_{t+1}\|a_t)$		Discard					Play					Hint colour					Hint rank				
		1	2	3	4	5	1	2	3	4	5	R	Y	G	W	B	1	2	3	4	5
a_t											a_{t+1}										
Discard	1	3	4	7	3	6	4	3	1	0	1	6	6	5	6	6	11	7	7	5	9
	2	4	5	7	2	6	4	2	2	0	0	6	6	5	6	5	12	6	8	5	9
	3	4	4	9	2	7	5	3	1	0	0	5	6	5	6	5	10	6	9	5	8
	4	4	4	7	3	12	4	3	1	0	0	4	5	3	4	6	11	8	6	5	9
	5	1	2	4	1	9	4	3	1	42	0	2	2	2	2	2	3	6	7	5	2
Play	1	3	3	5	3	6	9	5	2	0	2	4	3	4	5	4	8	8	7	6	11
	2	4	4	6	4	7	8	5	1	0	1	4	4	4	5	3	9	7	8	5	11
	3	3	3	4	2	7	5	3	1	0	1	5	4	4	4	4	12	15	8	4	9
	4	4	4	5	4	9	7	4	1	0	1	4	5	4	5	5	9	9	9	5	5
	5	5	5	8	5	10	13	7	1	0	0	3	3	3	3	3	6	7	6	4	5
Hint Colour	R	4	5	3	38	1	16	14	1	0	0	2	1	1	1	2	3	3	2	2	3
	Y	4	3	3	45	1	15	12	1	0	0	1	2	1	1	1	3	3	2	1	2
	G	5	4	2	37	1	19	14	1	0	0	1	1	2	1	2	3	3	2	1	2
	W	3	3	3	44	1	15	13	1	0	0	2	1	1	1	1	3	2	2	1	2
	B	3	5	1	44	1	15	12	1	0	0	1	1	1	1	2	3	2	2	2	2
Hint Rank	1	0	0	0	0	0	9	6	9	12	64	0	0	0	0	0	0	0	0	0	0
	2	3	4	2	1	0	2	4	4	22	42	0	0	0	1	0	8	2	1	0	1
	3	2	3	1	16	1	5	6	3	26	24	1	1	1	0	1	6	3	1	0	1
	4	1	1	0	3	0	2	2	2	55	31	0	0	0	0	0	0	0	0	0	1
	5	2	2	2	18	0	6	3	1	55	4	1	0	1	1	1	0	0	1	1	1
$P_{\%}(a_t)$		3	3	4	10	5	8	5	2	10	9	3	3	2	3	3	7	6	5	3	5

Table A.6
Conditional action probabilities for second two player ACHA agent.

$P_{\%}(a_{t+1}\|a_t)$		Discard					Play					Hint colour					Hint rank				
		1	2	3	4	5	1	2	3	4	5	R	Y	G	W	B	1	2	3	4	5
a_t											a_{t+1}										
Discard	1	2	2	0	1	31	6	0	0	0	0	9	3	2	3	4	4	2	6	8	16
	2	2	2	1	1	33	2	1	1	1	1	8	2	3	4	4	2	2	3	8	17
	3	0	2	5	2	0	53	3	4	1	2	5	1	5	2	1	2	2	1	1	10
	4	0	0	1	1	0	0	71	0	0	0	4	2	1	1	2	2	2	1	2	11
	5	2	1	0	1	31	1	0	0	0	0	11	3	2	6	4	5	3	6	8	16
Play	1	3	3	1	1	27	6	2	1	0	2	7	3	3	3	3	7	2	5	8	12
	2	4	3	1	1	27	5	2	1	0	2	6	4	2	4	3	4	3	5	9	13
	3	3	2	1	1	25	4	1	1	0	1	8	4	5	4	3	8	3	5	9	12
	4	4	2	2	2	31	3	1	0	0	1	7	3	3	7	4	5	3	4	8	11
	5	2	1	1	1	33	3	2	0	0	1	11	4	3	5	4	5	3	5	8	9
Hint Colour	R	0	0	0	0	0	0	0	0	0	97	0	0	0	0	0	0	0	0	0	1
	Y	0	0	0	1	0	0	94	0	0	0	1	0	0	0	0	1	0	0	0	1
	G	0	0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0
	W	0	0	0	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	0	0
	B	0	0	0	1	0	0	95	0	0	0	0	0	0	0	0	1	0	0	0	1
Hint Rank	1	0	0	0	1	0	23	0	73	0	0	1	0	0	0	0	0	0	0	0	1
	2	0	0	0	0	0	47	0	0	0	44	1	1	0	1	1	2	0	1	0	1
	3	28	4	1	1	5	43	0	1	0	1	1	1	1	1	1	2	1	2	3	1
	4	11	32	1	2	9	17	0	3	0	0	1	2	2	3	2	5	1	3	4	2
	5	0	0	0	0	0	0	0	0	0	97	0	0	0	0	0	0	0	0	0	1
$P_{\%}(a_t)$		3	3	1	1	19	6	6	3	5	15	6	2	2	3	3	4	2	4	6	8

References

[1] D. Premack, G. Woodruff, Does the chimpanzee have a theory of mind? Behav. Brain Sci. 1 (1978) 515–526.
[2] N.C. Rabinowitz, F. Perbet, H.F. Song, C. Zhang, S.M.A. Eslami, M. Botvinick, Machine theory of mind, in: Proceedings of the 35th International Conference on Machine Learning (ICML), 2018, pp. 4215–4224, http://proceedings.mlr.press/v80/rabinowitz18a.html.
[3] D.C. Dennett, The Intentional Stance, The MIT Press, Cambridge, MA, 1987.
[4] Board Game Arena, Board game arena: play board games online! https://en.boardgamearena.com/#!gamepanel?game=hanabi. (Accessed 5 December 2018).
[5] J. Nesta, Hanabi Live, https://hanabi.live, 2017. (Accessed 12 June 2019).
[6] J.N. Foerster, Y.M. Assael, N. de Freitas, S. Whiteson, Learning to communicate with deep multi-agent reinforcement learning, CoRR, arXiv:1605.06676.
[7] M. Lewis, D. Yarats, Y. Dauphin, D. Parikh, D. Batra, Deal or no deal? End-to-end learning of negotiation dialogues, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2017, pp. 2433–2443.
[8] K. Cao, A. Lazaridou, M. Lanctot, J.Z. Leibo, K. Tuyls, S. Clark, Emergent communication through negotiation, CoRR, arXiv:1804.03980.
[9] M. Campbell, A.J. Hoane Jr, F.-h. Hsu, Deep blue, Artif. Intell. 134 (1–2) (2002) 57–83.
[10] J. Schaeffer, R. Lake, P. Lu, M. Bryant, Chinook the world man-machine checkers champion, AI Mag. 17 (1) (1996) 21.
[11] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.
[12] G. Tesauro, Temporal difference learning and TD-Gammon, Commun. ACM 38 (3) (1995) 58–68.
[13] M. Moravčík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, M. Bowling, Deepstack: Expert-level artificial intelligence in heads-up no-limit poker, Science 356 (6337) (2017) 508–513.
[14] N. Brown, T. Sandholm, Superhuman AI for heads-up no-limit poker: Libratus beats top professionals, Science 359 (6374) (2018) 418–424.
[15] D. Lewis, Convention: A Philosophical Study, John Wiley & Sons, 2008.
[16] V.P. Crawford, J. Sobel, Strategic information transmission, Econometrica 50 (6) (1982) 1431–1451, http://www.jstor.org/stable/1913390.
[17] N. Burch, M. Johanson, M. Bowling, Solving imperfect information games using decomposition, in: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014, pp. 602–608, http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8407.
[18] N. Burch, Time and Space: Why Imperfect Information Games Are Hard, Ph.D. thesis, University of Alberta, Alberta, Canada, 2017.
[19] A. Iraci, Conventions for Hanabi, http://hanabi.pythonanywhere.com/static/Hanabi.pdf, 2018. (Accessed 17 October 2019).
[20] J. Nesta, Github - zamiell/hanabi-conventions: Hanabi conventions for the hyphen-ated group, https://github.com/Zamiell/hanabi-conventions, 2018. (Accessed 5 December 2018).
[21] M.G. Bellemare, Y. Naddaf, J. Veness, M. Bowling, The Arcade Learning Environment: an evaluation platform for general agents, J. Artif. Intell. Res. 47 (2013) 253–279.
[22] M.C. Machado, M.G. Bellemare, E. Talvitie, J. Veness, M. Hausknecht, M. Bowling, Revisiting the Arcade Learning Environment: evaluation protocols and open problems for general agents, J. Artif. Intell. Res. 61 (2018) 523–562.
[23] The Hanabi Learning Environment, https://github.com/deepmind/hanabi-learning-environment, 2019. (Accessed 1 February 2019).
[24] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, W. Zaremba, Openai gym, CoRR, arXiv:1606.01540.
[25] E.T. Gottwald, M. Eger, C. Martens, I see what you see: integrating eye tracking into hanabi playing agents, in: Joint Proceedings of the AIIDE 2018 Workshops co-located with 14th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2018), 2018, http://ceur-ws.org/Vol-2282/EXAG_112.pdf.
[26] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T.P. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in: Proceedings of the 33rd International Conference on Machine Learning (ICML), 2016, pp. 1928–1937.
[27] B. Wymann, C. Dimitrakakis, A. Sumner, E. Espié, C. Guionneau, Torcs: The open racing car simulator, 2015.
[28] C. Beattie, J.Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, J. Schrittwieser, K. Anderson, S. York, M. Cant, A. Cain, A. Bolton, S. Gaffney, H. King, D. Hassabis, S. Legg, S. Petersen, Deepmind lab, CoRR, arXiv:1612.03801.
[29] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, K. Kavukcuoglu, IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures, CoRR, arXiv:1802.01561.
[30] M. Jaderberg, W.M. Czarnecki, I. Dunning, L. Marris, G. Lever, A.G. Castañeda, C. Beattie, N.C. Rabinowitz, A.S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J.Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, T. Graepel, Human-level performance in first-person multiplayer games with population-based deep reinforcement learning, CoRR, arXiv:1807.01281.
[31] M. Jaderberg, W.M. Czarnecki, I. Dunning, L. Marris, G. Lever, A.G. Castañeda, C. Beattie, N.C. Rabinowitz, A.S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J.Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, T. Graepel, Human-level performance in 3d multiplayer games with population-based reinforcement learning, Science 364 (6443) (2019) 859–865, https://doi.org/10.1126/science.aau6249, https://science.sciencemag.org/content/364/6443/859.
[32] M. Jaderberg, V. Dalibard, S. Osindero, W.M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, C. Fernando, K. Kavukcuoglu, Population based training of neural networks, CoRR, arXiv:1711.09846.
[33] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M.G. Azar, D. Silver Rainbow, Combining improvements in deep reinforcement learning, CoRR, arXiv:1710.02298.
[34] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[35] P.S. Castro, S. Moitra, C. Gelada, S. Kumar, M.G. Bellemare, Dopamine: a research framework for deep reinforcement learning, arXiv.
[36] M.G. Bellemare, W. Dabney, R. Munos, A distributional perspective on reinforcement learning, in: Proceedings of the International Conference on Machine Learning, 2017.
[37] T. Schaul, J. Quan, I. Antonoglou, D. Silver, Prioritized experience replay, in: International Conference on Learning Representations, 2016.
[38] J.N. Foerster, F. Song, E. Hughes, N. Burch, I. Dunning, S. Whiteson, M. Botvinick, M. Bowling, Bayesian action decoder for deep multi-agent reinforcement learning, CoRR, arXiv:1811.01458.
[39] A. O'Dwyer, Github - quuxplusone/hanabi: framework for writing bots that play hanabi, https://github.com/Quuxplusone/Hanabi, Online; accessed 19-September-2018 (2018).
[40] C. Cox, J.D. Silva, P. Deorsey, F.H.J. Kenter, T. Retter, J. Tobin, How to make the perfect fireworks display: two strategies for Hanabi, Math. Mag. 88 (5) (2015) 323–336.
[41] J. Wu, Github - wuthefwasthat/hanabi.rs: Hanabi simulation in rust, https://github.com/WuTheFWasThat/hanabi.rs, 2018. (Accessed 29 November 2018).
[42] D. Wu, Github - lightvector/fireflower: a rewrite of hanabi-bot in scala, https://github.com/lightvector/fireflower, 2018. (Accessed 3 December 2018).
[43] H. Osawa, Solving Hanabi: estimating hands by opponent's actions in cooperative game with incomplete information, in: Proceedings on the 2015 AAAI Workshop on Computer Poker and Imperfect Information, 2015, pp. 37–43.
[44] M.J.H. van den Bergh, A. Hommelberg, W.A. Kosters, F.M. Spieksma, Aspects of the cooperative card game Hanabi, in: BNAIC 2016: Artificial Intelligence. 28th Benelux Conference on Artificial Intelligence, in: CCIS, vol. 765, Springer, Cham, 2016, pp. 93–105.

[45] J. Walton-Rivers, P.R. Williams, R. Bartle, D. Perez-Liebana, S.M. Lucas, Evaluating and modelling Hanabi-playing agents, in: Proceedings of the IEEE Conference on Evolutionary Computation (2017), 2017.
[46] B. Bouzy, Playing Hanabi near-optimally, in: Advances in Computer Games: ACG 2017, in: LNCS, vol. 10664, Springer, Cham, 2017, pp. 51–62.
[47] J.-F. Baffier, M.-K. Chiu, Y. Diez, M. Korman, V. Mitsou, A. Renssen, M. Roeloffzen, Y. Uno, Hanabi is NP-hard, even for cheaters who look at their cards, Theor. Comput. Sci. 675 (2017) 43–55.
[48] D.S. Bernstein, R. Givan, N. Immerman, S. Zilberstein, The complexity of decentralized control of Markov decision processes, Math. Oper. Res. 27 (4) (2002) 819–840, https://doi.org/10.1287/moor.27.4.819.297.
[49] C. Liang, J. Proft, E. Andersen, R.A. Knepper, Implicit communication of actionable information in human-ai teams, in: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019, https://rpal.cs.cornell.edu/docs/LiaEtal_CHI_2019.pdf.
[50] M. Eger, C. Martens, Practical specification of belief manipulation in games, in: Proceedings, the Thirteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-17), 2017.
[51] E. Pacuit, O. Roy, Epistemic foundations of game theory, in: E.N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, Summer 2017 Edition, Metaphysics Research Lab, Stanford University, 2017.
[52] J. Walton-Rivers, P. Williams CIG, Hanabi fireworks competition, http://hanabi.aiclash.com/, 2018, https://project.dke.maastrichtuniversity.nl/cig2018/competitions/, 2018, retrieved Sept. 27th, 2018.
[53] R. Canaan, H. Shen, R.R. Torrado, J. Togelius, A. Nealen, S. Menzel, Evolving agents for the Hanabi 2018 CIG competition, in: IEEE Conference on Computational Intelligence and Games (CIG), 2018, pp. 1–8.
[54] J. Goodman, Re-determinizing information set Monte Carlo tree search in hanabi, CoRR, arXiv:1902.06075.
[55] P.I. Cowling, E.J. Powley, D. Whitehouse, Information set Monte Carlo tree search, IEEE Trans. Comput. Intell. AI Games 4 (2012) 120–143.
[56] R. Sutton, A. Barto, Reinforcement Learning: An Introduction, 2nd edition, MIT Press, 2018.
[57] M.L. Littman, Markov games as a framework for multi-agent reinforcement learning, in: Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann, 1994, pp. 157–163.
[58] L. Matignon, G.J. Laurent, N.L. Fort-Piat, Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems, Knowl. Eng. Rev. 27 (01) (2012) 1–31.
[59] M. Bowling, M. Veloso, Simultaneous adversarial multi-robot learning, in: IJCAI, vol. 3, 2003, pp. 699–704.
[60] L. Panait, S. Luke, Cooperative multi-agent learning: the state of the art, Auton. Agents Multi-Agent Syst. 11 (3) (2005) 387–434, https://doi.org/10.1007/s10458-005-2631-2, http://dx.doi.org/10.1007/s10458-005-2631-2.
[61] L. Busoniu, R. Babuska, B.D. Schutter, A comprehensive survey of multiagent reinforcement learning, IEEE Trans. Syst. Man Cybern., Part C, Appl. Rev. 38 (2) (2008) 156–172.
[62] A. Nowé, P. Vrancx, Y.-M.D. Hauwere, Game theory and multi-agent reinforcement learning, in: Reinforcement Learning: State-of-the-Art, Springer, 2012, pp. 441–470, Ch. 14.
[63] D. Bloembergen, K. Tuyls, D. Hennes, M. Kaisers, Evolutionary dynamics of multi-agent learning: a survey, J. Artif. Intell. Res. 53 (2015) 659–697.
[64] P. Hernandez-Leal, B. Kartal, M.E. Taylor, Is multiagent deep reinforcement learning the answer or the question? a brief survey, CoRR, arXiv:1810.05587.
[65] J. Baxter, A. Tridgell, L. Weaver, Knightcap: a chess program that learns by combining TD(lambda) with game-tree search, in: Proceedings of the 15th International Conference on Machine Learning, Morgan Kaufmann, 1998, pp. 28–36.
[66] J. Veness, D. Silver, A. Blair, W. Uther, Bootstrapping from game tree search, in: Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, A. Culotta (Eds.), Advances in Neural Information Processing Systems, vol. 22, Curran Associates, Inc., 2009, pp. 1937–1945.
[67] L. Kocsis, C. Szepesvári, Bandit Based Monte-Carlo Planning, 2006, pp. 282–293.
[68] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of Go with deep neural networks and tree search, Nature 529 (2016) 484–489.
[69] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, T.H. Arthur Guez, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, D. Hassabis, Mastering the game of Go without human knowledge, Nature 550 (2017) 354–359.
[70] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, D. Hassabis, A general reinforcement learning algorithm that masters chess, shogi, and go through self-play, Science 362 (6419) (2018) 1140–1144, https://doi.org/10.1126/science.aar6404, https://science.sciencemag.org/content/362/6419/1140.
[71] M. Zinkevich, M. Johanson, M. Bowling, C. Piccione, Regret minimization in games with incomplete information, in: Advances in Neural Information Processing Systems 20 (NIPS 2007), 2008.
[72] M. Bowling, N. Burch, M. Johanson, O. Tammelin, Heads-up Limit Hold'em Poker is solved, Science 347 (6218) (2015) 145–149.
[73] A. Lazaridou, A. Peysakhovich, M. Baroni, Multi-agent cooperation and the emergence of (natural) language, CoRR, arXiv:1612.07182.
[74] S. Sukhbaatar, A. Szlam, R. Fergus, Learning multiagent communication with backpropagation, CoRR, arXiv:1605.07736.
[75] H. de Vries, F. Strub, S. Chandar, O. Pietquin, H. Larochelle, A. Courville, Guesswhat?! Visual object discovery through multi-modal dialogue, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[76] C. Yeh, H. Lin, Automatic bridge bidding using deep reinforcement learning, CoRR, arXiv:1607.03290.
[77] H.P. Grice, Logic and conversation, in: P. Cole, J.L. Morgan (Eds.), Syntax and Semantics, vol. 3, Academic Press, New York, 1975.
[78] S.T. Piantadosi, H. Tily, E. Gibson, The communicative function of ambiguity in language, Cognition 122 (3) (2012) 280–291, https://doi.org/10.1016/j.cognition.2011.10.004, http://www.sciencedirect.com/science/article/pii/S0010027711002496.
[79] M.C. Frank, N.D. Goodman, Predicting pragmatic reasoning in language games, Science 336 (6084) (2012), https://doi.org/10.1126/science.1218633, 998–998, http://science.sciencemag.org/content/336/6084/998.
[80] P.J. Gmytrasiewicz, P. Doshi, A framework for sequential planning in multi-agent settings, J. Artif. Intell. Res. 24 (2005) 49–79.
[81] Y. Shoham, K. Leyton-Brown, Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations, Cambridge University Press, 2009.
[82] W. Yoshida, R.J. Dolan, K.J. Friston, Game theory of mind, PLoS Comput. Biol. 4 (12) (2008) 1–14.
[83] J.R. Wright, K. Leyton-Brown, Predicting human behavior in unrepeated, simultaneous-move games, Games Econ. Behav. 106 (2017) 16–37.
[84] J. Hartford, J.R. Wright, K. Leyton-Brown, Deep learning for predicting human strategic behavior, in: Thirtieth Annual Conference on Neural Information Processing Systems (NIPS 2016), 2016.
[85] H. He, J. Boyd-Graber, K. Kwok, H.D. III, Opponent modeling in deep reinforcement learning, in: Proceedings of the International Conference on Machine Learning (ICML), 2016.
[86] S.V. Albrecht, P. Stone, Autonomous agents modelling other agents: a comprehensive survey and open problems, Artif. Intell. 258 (2018) 66–95.
[87] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Perolat, D. Silver, T. Graepel, A unified game-theoretic approach to multiagent reinforcement learning, in: Advances in Neural Information Processing Systems, 2017.
[88] J.N. Foerster, R.Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, I. Mordatch, Learning with opponent-learning awareness, CoRR, arXiv:1709.04326.
[89] A. Grover, M. Al-Shedivat, J.K. Gupta, Y. Burda, H. Edwards, Learning policy representations in multiagent systems, in: Proceedings of the International Conference on Machine Learning (ICML), 2018.