MACHINE LEARNING

Mastering the game of Stratego with model-free multiagent reinforcement learning

Julien Perolat*†, Bart De Vylder*†, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer‡, Paul Muller, Jerome T. Connor, Neil Burch, Thomas Anthony, Stephen McAleer, Romuald Elie, Sarah H. Cen, Zhe Wang, Audrunas Gruslys, Aleksandra Malysheva, Mina Khan, Sherjil Ozair, Finbarr Timbers, Toby Pohlen, Tom Eccles, Mark Rowland, Marc Lanctot, Jean-Baptiste Lespiau, Bilal Piot, Shayegan Omidshafiei, Edward Lockhart, Laurent Sifre, Nathalie Beauguerlange, Remi Munos, David Silver, Satinder Singh, Demis Hassabis, Karl Tuyls*†

We introduce DeepNash, an autonomous agent that plays the imperfect information game Stratego at a human expert level. Stratego is one of the few iconic board games that artificial intelligence (AI) has not yet mastered. It is a game characterized by a twin challenge: It requires long-term strategic thinking as in chess, but it also requires dealing with imperfect information as in poker. The technique underpinning DeepNash uses a game-theoretic, model-free deep reinforcement learning method, without search, that learns to master Stratego through self-play from scratch. DeepNash beat existing state-of-the-art AI methods in Stratego and achieved a year-to-date (2022) and all-time top-three ranking on the Gravon games platform, competing with human expert players.

Progress in artificial intelligence (AI) has been measured through the mastery of board games since the inception of the field. Board games allow us to gauge and evaluate how humans and machines develop and execute strategies in a controlled environment. The ability to plan ahead has been at the heart of successes in AI for decades in perfect information games such as chess, checkers, shogi, and Go, as well as in imperfect information games such as poker and Scotland Yard (*1*–*6*). For many years, the Stratego (*7*) board game has constituted one of the next frontiers of AI research (for a visualization of the game phases and game mechanics, see Fig. 1). The game poses two key challenges. First, the game tree of Stratego has 10^{535} possible states, which is larger than both no-limit Texas Hold'em poker, a well-researched imperfect information game with 10^{164} states, and the game of Go, which has 10^{360} states. Second, acting in a given situation in Stratego requires reasoning $>10^{66}$ possible pairs of private deployments at the start of the game, whereas in Texas Hold'em poker, players are dealt one of 10^3 different two-card hands for 10^6 possible private configurations with two players. Perfect information games such as Go and chess do not have a private deployment phase, thus avoiding the complexity that this challenge poses in Stratego. Currently, it is not possible to use state-of-the-art model-based perfect information planning techniques nor state-of-the-art imperfect information search techniques that break down the game into independent situations (*5*, *6*).

For these reasons, Stratego is a major challenge for the AI community and provides a hard benchmark for studying strategic interactions at an unparalleled scale. As in most board games, Stratego tests the ability to make relatively slow, deliberative, and logical decisions sequentially. Additionally, in most imperfect information games, other tactics that better reflect decision-making processes in the real world need to be deployed. As von Neumann described it, "real life consists of bluffing, of little tactics of deception, of asking yourself what is the other man going to think I mean to do" (*8*). Most recent successes in large imperfect information games have been achieved in real-time strategy games such as StarCraft, Dota, and Capture the Flag (*9*–*11*) or in racing simulation video games such as Gran Turismo (*12*), in which most decisions must be made quickly and instinctively and are of a continuous-time nature. Stratego is a game for which little progress has been achieved by the AI research community because of the many complex aspects of its structure. Successes in the game have been limited, with artificial agents only able to play at a level comparable to a human amateur [see, e.g., (*13*–*17*)].

This work introduces a novel game-theoretic method that allows an AI for learning to play Stratego in self-play in a model-free manner without human demonstration and from scratch. This new method resulted in a bot called DeepNash that beat previous state-of-the-art AI agents and achieved human expert–level performance in the most complex variant of the game, Stratego Classic. At the core of DeepNash is a principled, novel, model-free reinforcement learning (RL) algorithm called Regularized Nash Dynamics (R-NaD). Our method resulting in DeepNash combines R-NaD with a deep neural network architecture to learn a strategy that plays at a highly competitive level by aiming to find a Nash equilibrium (*18*) (i.e., an unexploitable strategy in zero-sum two-player games). In earlier work, it was formally shown that an R-NaD approach converges to a Nash equilibrium in several classes of matrix games, including two-player zero-sum games (*19*). The present work suggests that R-NaD at scale converges empirically to an approximate Nash equilibrium in Stratego. Figure 2 illustrates a high-level overview of this approach, which underlies DeepNash.

The performance of DeepNash was systematically evaluated against various state-of-the-art Stratego bots and human expert players on the Gravon games platform. DeepNash convincingly won against all current state-of-the-art bots that have been developed to play Stratego, producing a win rate of >97%, and it achieved a highly competitive level of play against human expert Stratego players on Gravon, where it ranked among the top three players, both on the year-to-date 2022 (determined on 22 April 2022) and on all-time

Table 1. Evaluation of DeepNash against existing Stratego bots. The numbers are reported from DeepNash's point of view. More games (800) were played against bots that could be run automatically.

Opponent	No. of games	Wins	Draws	Losses
Probe	30	100.0%	0.0%	0.0%
Master of the Flag	30	100.0%	0.0%	0.0%
Demon of Ignorance	800	97.1%	1.8%	1.1%
Asmodeus	800	99.7%	0.0%	0.3%
Celsius	800	98.2%	0.0%	1.8%
Celsius1.1	800	97.9%	0.0%	2.1%
PeternLewis	800	99.9%	0.0%	0.1%
Vixen	800	100.0%	0.0%	0.0%

DeepMind Technologies Ltd., London, UK.
*Corresponding author. Email: perolat@deepmind.com (J.P.); bartdv@deepmind.com (B.D.V.); karltuyls@deepmind.com (K.T.)
†These authors contributed equally to this work and are co-lead authors.
‡Independent consultant.

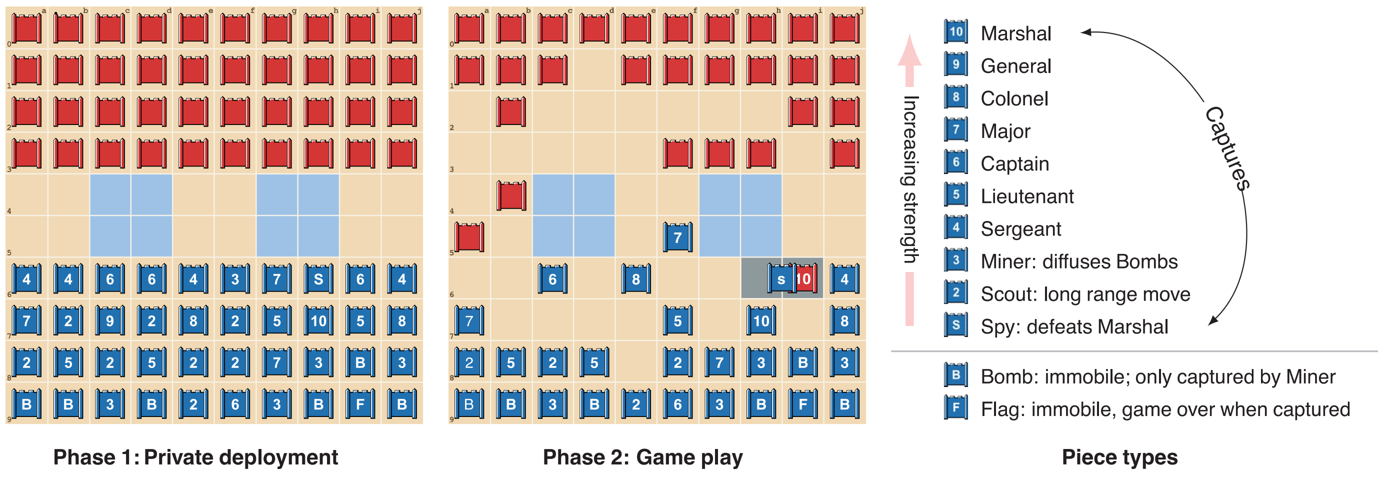
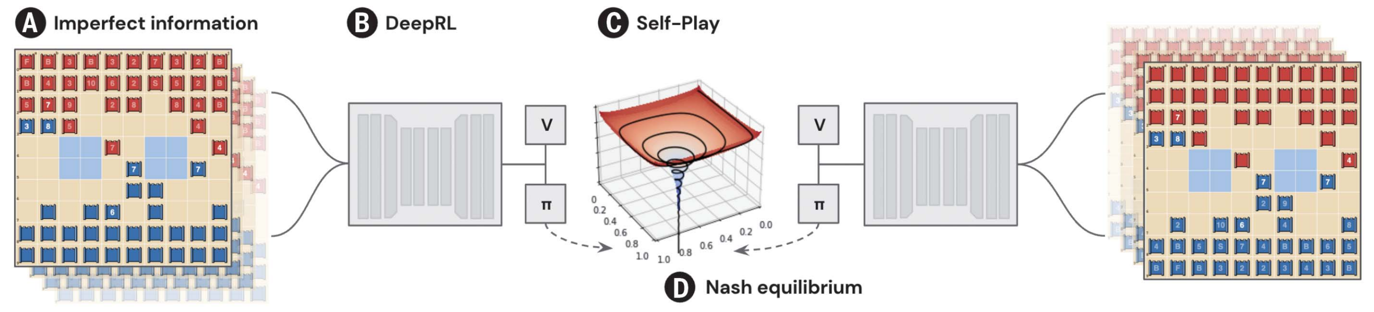


Fig. 1. Stratego is a two-player board game in which players try to capture the opponent's flag. Initially, the players secretly deploy 40 pieces of diverse strengths on the board. Then, they take turns moving pieces, possibly encountering an opponent piece that reveals both piece identities, and then the weaker piece is removed. Two lakes (indicated in blue) cannot be crossed by any piece. The complete rules are defined by the International Stratego Federation.



Replicator dynamics: $\frac{d}{d\tau}\pi^i_\tau(a^i) = \pi^i_\tau(a^i)\left[Q^i_{\pi_\tau}(a^i) - \sum_{b^i}\pi^i_\tau(b^i)Q^i_{\pi_\tau}(b^i)\right]$

Reward transformation: $r^i(\pi^i,\pi^{-i},a^i,a^{-i}) = r^i(a^i,a^{-i}) - \eta\log\left(\frac{\pi^i(a^i)}{\pi^i_{\text{reg}}(a^i)}\right) + \eta\log\left(\frac{\pi^{-i}(a^{-i})}{\pi^{-i}_{\text{reg}}(a^{-i})}\right)$

Fig. 2. Overview of R-NaD. (**A**) Overview of the R-NaD approach at scale underlying DeepNash, which allows for learning to play the imperfect information game Stratego. (**B** to **D**) R-NaD learns a policy represented by a deep neural network (B) through self-play from scratch (C) and aims at converging to a Nash equilibrium (D). The approach relies on two core ideas to reach convergence: replicator dynamics and reward transformation. Their equations are shown for illustrative purposes in their simplest form.

leaderboards, with a win rate of 84%. Therefore, to the best of our knowledge, this is the first time an AI algorithm was able to learn to play Stratego at a human expert level. It is worth mentioning that this performance was achieved without deploying any search method, which was a key ingredient of many milestone achievements in AI for board games in the past.

Methods

R-NaD at scale takes an end-to-end learning approach to solving Stratego by incorporating the learning of the deployment phase, i.e., putting the pieces tactically on the board at the start of a game (Fig. 1), into the learning of the game-play phase using an integrated deep RL and game-theoretic approach. As with much work in two-player zero-sum games, the purpose is to learn an approximate Nash equilibrium through self-play. In the context of two-player zero-sum games, a Nash equilibrium guarantees that the agent will perform well, even against a worst-case opponent. Such robustness typically allows an algorithm to perform well against humans [see, e.g., (*3*–*5*)]. In perfect information games, search techniques aided by RL have provided state-of-the-art superhuman bots in Go and chess (*2*, *20*). However, searching for a Nash equilibrium in imperfect information games requires estimating private information of the opponent from public states (*3*, *6*). Given the vast number of such possible private configurations in a public state, Stratego is computationally too challenging for all existing search techniques because the search space becomes intractable. This work therefore chose a different route, without search, and proposed a new method that combines model-free RL in self-play with a game-theoretic algorithmic idea, R-NaD. The model-free part implies that R-NaD does not build an explicit opponent model–tracking belief space (calculating a likelihood of the opponent's state), and the game-theoretic part is based on the idea that by modifying the dynamical system underpinning the reinforcement-learning algorithm, one can steer the learning behavior of the agent in the direction of the Nash equilibrium. The main advantage of this combined approach is that one does not need to explicitly model private states from public ones. A complex challenge, on the other hand, is to scale up this model-free RL approach with

A

		Player 2	
		Head: H	Tail: T
Player 1	Head: H	1	-1
	Tail: T	-1	1

B

R-NaD Iteration

Start with an arbitrary regularization policy: $\pi_{0,\text{reg}}$

1. Reward transformation: Construct the transformed game with: $\pi_{m,\text{reg}}$
2. Dynamics: Run the replicator dynamics until convergence to: $\pi_{m,\text{fix}}$
3. Update: Set the regularization policy:
$$\pi_{m+1,\text{reg}} = \pi_{m,\text{fix}}$$

Repeat stages until convergence

C

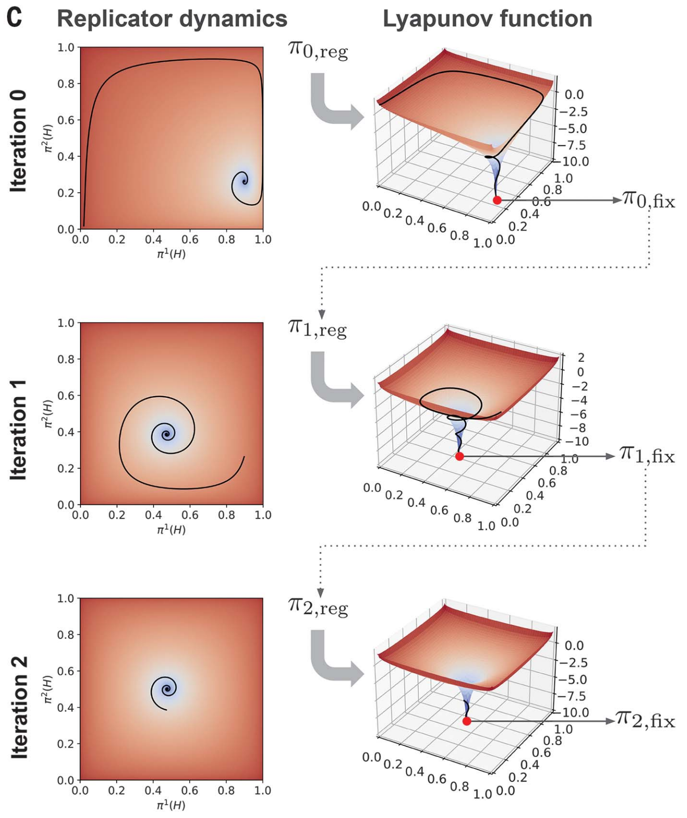


Fig. 3. Illustrating R-NaD using the matching pennies game. (**A**) Payoff table. (**B**) Algorithmic stages. (**C**) Dynamics and Lyapunov function.

R-NaD to make self-play competitive against human expert players in Stratego, which had not been achieved to date. This combined approach is illustrated in Fig. 2.

The following subsections present the learning algorithm behind DeepNash, referring to the supplementary materials for more technical details where relevant.

Learning approach

The approach underpinning DeepNash aims to learn a Nash equilibrium in Stratego through self-play and model-free RL. The idea of combining the two has been tried before, but it has been empirically challenging to stabilize such learning algorithms when scaling up to complex games such as Capture the Flag, Dota, and StarCraft (*9*–*11*). Some empirical work manages to stabilize the learning either by training against past versions of the agent (*9*–*11*) or by adding reward shaping (*10*, *11*) or expert data (*9*) to the training algorithm. Although these approaches help, they lack theoretical foundations, remain difficult to tune, and are rather domain dependent. Furthermore, in a game such as Stratego, it is difficult to define a loss for which minimization would converge to a Nash equilibrium without introducing prohibitive computational obstacles at large scale. For instance, minimizing the exploitability (*21*), a well-known quantity that measures the distance to a Nash equilibrium, requires estimating an agent's best response during training, which is computationally intractable in Stratego. However, it is possible to define a learning update rule that induces a dynamical system for which there exists a so-called Lyapunov function. This function can be shown to decrease during learning and thus guarantees convergence to a fixed point. This is the central idea behind the R-NaD algorithm and is the successful recipe for DeepNash, which scales this approach using a deep neural network.

R-NaD algorithm

The R-NaD learning algorithm used in DeepNash is an actor-critic method based on the idea of regularization for convergence purposes (*19*), which is briefly explained first in the context of zero-sum two-player normal form games (illustrated on the matching pennies game). A normal form game is an abstraction of a decision-making situation involving more than one agent. Each agent (indexed by $i \in \{1,2\}$) needs to simultaneously take an action a^i (in a set of possible actions A^i) according to a policy $\pi^i(.)$ (i.e., a distribution over possible actions A^i), after which it receives a game reward [$r^i(a^1, a^2)$], and then the game is repeated. For convenience, the opponent of player i is indexed by $-i$. R-NaD relies on three key stages (Fig. 3B), described below.

In the first stage, the reward is transformed based on a regularization policy π_{reg}, which induces a modified game with rewards

$$r^i(\pi^i\pi^{-i}a^ia^{-1}) = r^i(a^ia^{-1}) - \eta\log\left(\frac{\pi^i(a^i)}{\pi^i_{\text{reg}}(a^i)}\right) + \eta\log\left(\frac{\pi^{-i}(a^{-i})}{\pi^{-i}_{\text{reg}}(a^{-i})}\right) \quad (1)$$

where $\eta > 0$ is a regularization parameter and i is the player index ($i \in \{1,2\}$). Note that this transformed reward is policy dependent.

Second, in the dynamics stage, the system evolves according to the replicator dynamics system (*22*, *23*) on this modified game. The basic replicator dynamics equations are a descriptive learning process from evolutionary game theory, equivalent to RL algorithms (*23*), which are also equivalent to an instance of follow-the-regularized-leader (*24*) and are defined as follows:

$$\frac{d}{d\tau}\pi^i_\tau(a^i) = \pi^i_\tau(a^i)\left[Q^i_{\pi_\tau}(a^i) - \sum_{b^i}\pi^i_\tau(b^i)Q^i_{\pi_\tau}(b^i)\right] \quad (2)$$

where $Q^i_{\pi_\tau}(a^i)$ is the quality or fitness of an action: i.e., $Q^i_\pi(a^i) = \mathbb{E}_{a^{-i}\sim\pi^{-i}}\left[r^i(\pi^i,\pi^{-i},a^i,a^{-i})\right]$. These dynamics reinforce the probability of taking actions with high fitness (relative to other actions). Because of the reward transformation, this system has a unique fixed point, π *fix*, and convergence to it is guaranteed in zero-sum two-player games [see (*19*)], which can be proven by the Lyapunov function:

$$H_{\pi_{fix}}(\pi) = \sum_{i=1}^{2}\sum_{a^i\in A^i}\pi^i_{fix}(a^i)\log\left(\frac{\pi^i_{fix}(a^i)}{\pi^i(a^i)}\right) \quad (19).$$

However, this fixed point is not yet a Nash equilibrium of the original game.

In the final update stage, the fixed point obtained is used as the regularization policy for the next iteration. These three stages are applied repeatedly, generating a sequence of fixed points that can be proven to converge to a Nash equilibrium of the original (unmodified) game (*19*) in zero-sum two-player games, but not in all general-sum games. Figure 3C illustrates the R-NaD algorithm on the two-player matching pennies game (the payoff table is shown in Fig. 3A). The first iteration starts from $\pi^i_{0,\text{reg}}[H,T] = [0.999, 0.001]$, $(\eta = 0.2)$ and the replicator dynamics converge to $\pi^0_{0,\text{fix}}[H,T] = [0.896, 0.104]$ and $\pi^1_{0,\text{fix}}[H,T] = [0.263, 0.737]$. The right figure shows the evolution of the logarithm of the Lyapunov function and illustrates that it decreases while learning. Three iterations of R-NaD are shown.

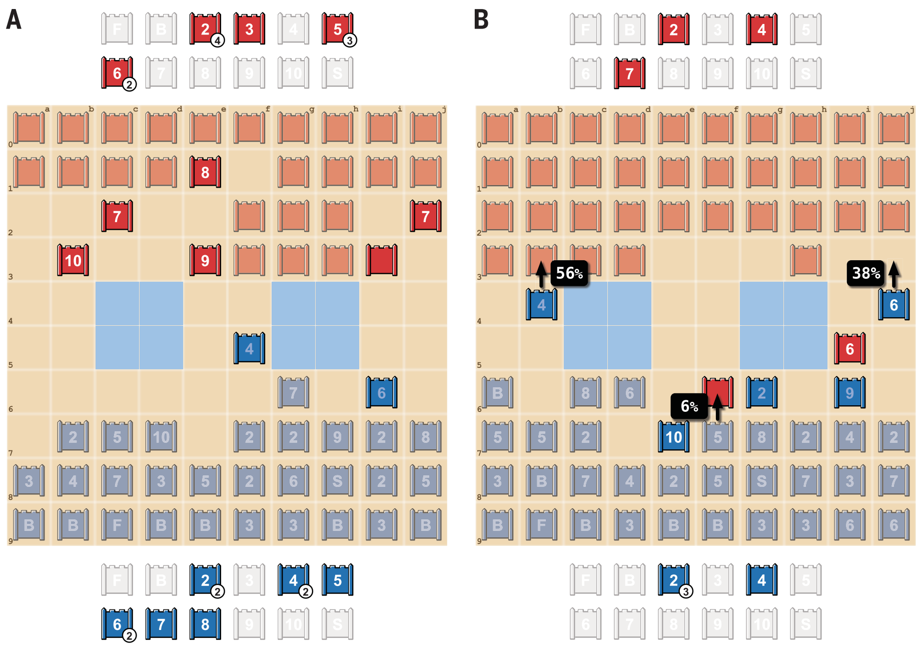


Fig. 4. Illustration of DeepNash's assessment of the relative value of material versus information. Shown is an illustration of DeepNash's assessment of the relative value of material versus information in two human (red) versus DeepNash (blue) matches. (**A**) Although blue is behind a 7 and 8, no pieces are revealed and only two are moved. As a result, DeepNash assesses its chance of winning to still be ~70% (blue indeed won this match). (**B**) Blue to move. DeepNash's policy supports three moves at this state, with the indicated probabilities shown (the move on the right was played in the actual match). Although blue has the opportunity to capture the opponent's 6 with its 9, this move was not considered by DeepNash, likely because the protection of 9's identity was assessed to be more important than the material gain.

R-NaD at scale

Our method consists of three components: (i) a core training component R-NaD, i.e., the model-free RL algorithm presented above, which is implemented using a deep convolutional network; (ii) a component that fine-tunes the learned policy to reduce the residual probabilities of taking highly improbable actions; and (iii) a test-time postprocessing component that uses game-specific knowledge [whereas (i) and (ii) are game agnostic] to filter out low-probability actions and obvious mistakes.

The following section starts by concisely laying out some essential background information on imperfect information games necessary to understand how R-NaD is scaled to a deep learning model. Then, the implementation of the three algorithmic stages of R-NaD are summarized. A detailed description of R-NaD is provided in the supplementary materials.

Imperfect information games

In a two-player zero-sum imperfect information game, two players (player $i = 1$ or $i = 2$) sequentially interact in turns. At turn t, the players receive a reward signal $(r^1_t r^2_t)$, and the current player $i = \Psi_t$ observes the game state through an observation o_t and selects an action a_t according to a parameterized policy function $\pi(.|o_t)$. In model-free RL, the trajectories $T = \left[\left(o_t, a_t, \left(r^1_t r^2_t\right), \pi(.|o_t)\right), \Psi_t\right]_{0\le t<t_{\max}}$ are the only data the agent will leverage to learn the parameterized policy.

Model-free RL with R-NaD

The R-NaD algorithm is scaled by using deep learning architectures. It performs the same three algorithmic stages as before in normal form games: (i) the reward transformation stage, (ii) the dynamics stage that allows for convergence to a fixed point, and (iii) the update stage in which the algorithm updates the policy that defines the regularization function.

The neural architecture consists of the following components: a U-Net torso with residual blocks and skip connections (*25*) and four heads that are smaller replicas of the torso augmented with final layers to generate an output of the appropriate shape. The first DeepNash head outputs the value function as a scalar, and the three remaining heads encode the agent's policy by outputting a probability distribution over its actions at deployment and during game play. The agent architecture is described in detail in the supplementary materials.

The observation is encoded as a spatial tensor consisting of the following components: DeepNash's own pieces, publicly available information about both the opponent's and DeepNash's pieces, and an encoding of the 40 last moves. This public information represents the types each piece can still have given the history of the game. In total, the observation contains 82 stacked frames encoded in a single tensor. The observation's detail is given in the supplementary materials.

Given the regularization policy $\pi_{m,\text{reg}}$ at iteration m and a trajectory, the reward transform used at time step t for player i is $r^i_{t,\pi_{m,\text{reg}}}(a,\pi) = r^i_t - \eta\log\left(\frac{\pi(a|o_t)}{\pi_{m,\text{reg}}(a|o_t)}\right)$ if $i = \Psi_t$ and $r^i_t + \eta\log\left(\frac{\pi(a|o_t)}{\pi_{m,\text{reg}}(a|o_t)}\right)$ if $i \neq \Psi_t$.

The dynamics stage of the method is composed of two parts. The first part estimates the value function, which is done through an adaptation of the ν-trace estimator (*26*) to the two-player imperfect information case, resulting in a parameter update direction $Update_{\text{value}}$. The second part learns the policy through the Neural Replicator Dynamics (NeuRD) update (*27*) using a new estimate of the state action value based on the ν-trace estimator, resulting in a parameter update direction $Update_{\text{policy}}$. These parts are detailed in the supplementary materials.

After a fixed number of learning steps, an approximate fixed point policy, $\pi_{m,\text{fix}}$, is obtained, which is then used as the next regularization

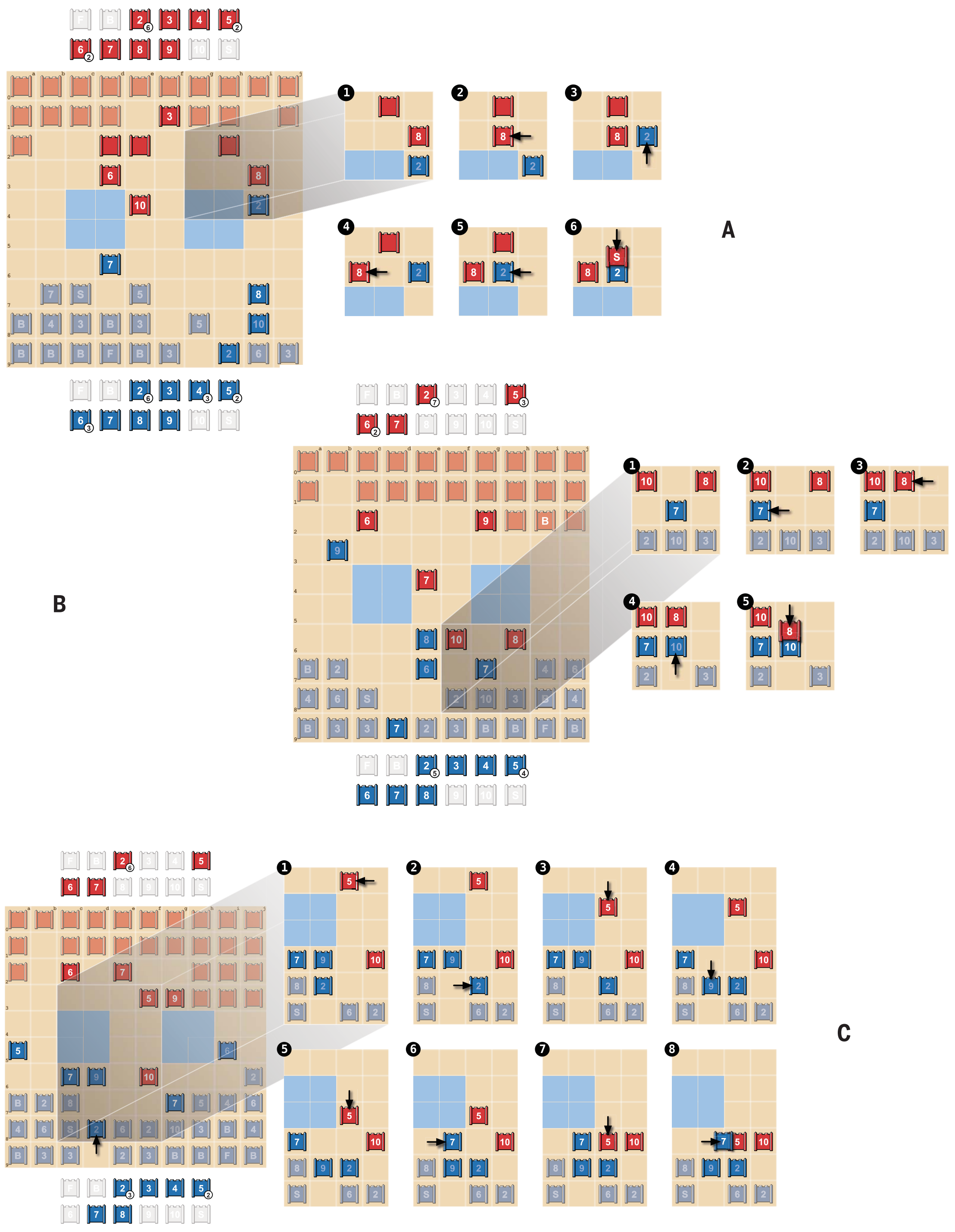


Fig. 5. Illustration of DeepNash bluffing. Shown is an illustration of DeepNash bluffing in three human (red) versus DeepNash (blue) matches. (**A**) Positive bluffing. (**B**) Negative bluffing. (**C**) DeepNash makes a scout (2) behave like a spy and gains material.

policy, $\pi_{m+1,\text{reg}} = \pi_{m,\text{fix}}$. The three stages are repeated using a smooth transition from the reward transformation of iteration m to the one of step $m + 1$.

Directly learning with the above-described method leads to convergence to an empirically satisfying solution, which, however, is slightly distorted by low-probability mistakes. Those mistakes appear because the *softmax* projection used to compute the policy from the logits assigns a nonzero probability to every action. To alleviate this issue, the policy is fine-tuned during training by performing additional thresholding and discretization to the action probabilities. The supplementary materials provide more details on this aspect and also describe a few additional heuristics applied at test time that remove obvious mistakes from the policy. As opposed to the R-NaD model-free training algorithm, these heuristics are Stratego specific. Qualitatively, they do remove rare mistakes in matches against humans, but they do not give notable quantitative improvements in self-play (see the supplementary materials).

Results

This section presents an overview of the evaluation results of DeepNash against both human expert players and current state-of-the-art Stratego bots. For the former, DeepNash has been evaluated on the Gravon platform, a well-known online games server popular among Stratego players. For the latter, DeepNash has been tested against eight known AI bots that play Stratego. A detailed analysis is also presented with regard to some of the capabilities of the agent's game play, including deployment, bluffing, and trading off of material versus information.

Evaluation on Gravon

Gravon is an internet platform for human players that offers several online games, including Stratego. It is by far the largest online platform for Stratego, and is where some of the strongest players compete. For more details on the platform and its ranking system, please refer to the supplementary materials.

DeepNash was evaluated against top human players over the course of 2 weeks in the beginning of April 2022, resulting in 50 ranked matches on Gravon. Of these matches, 42 (84%) were won by DeepNash. In the *Classic Stratego Challenge Ranking 2022* at that time, this corresponded to a rating of 1799, which put DeepNash in third place of all ranked Gravon Stratego players (the top two ratings were 1868 and 1831). In the all-time *Classic Stratego Ranking*, this resulted in a rating of 1778, which also put DeepNash in the third place of all ranked Gravon Stratego players (the top two ratings were 1876 and 1823). The rating for this leaderboard considers all ranked games going back to the year 2002.

These results confirm that DeepNash reaches a human expert level in Stratego only through self-play learning and without bootstrapping from existing human data.

Evaluation against state-of-the-art Stratego bots

DeepNash was also evaluated against several existing Stratego computer programs: Probe was a three-time winner of the Computer Stratego World Championship (2007, 2008, and 2010); Master of the Flag won that championship in 2009; Demon of Ignorance is an opensource implementation of Stratego with an accompanying AI bot; and Asmodeus, Celsius, Celsius1.1, PeternLewis, and Vixen are programs that were submitted in an Australian university programming competition in 2012 (see the supplementary materials for more details).

As shown in Table 1, DeepNash won the overwhelming majority of games against all of these bots despite not having been trained against any of them and only being trained using self-play. Therefore, it is not necessarily expected that the residual losses against some of these bots would vanish even if the exact Nash-equilibrium were reached. For example, in most of the few matches that DeepNash has lost against Celsius1.1, the latter played a high-risk strategy of capturing pieces early on with a high-ranking piece and thus was trying to get a significant material advantage. Most often, this strategy does not work, but occasionally it can lead to a win.

Illustration of DeepNash's abilities

The only goal of the algorithm behind DeepNash is to learn a Nash equilibrium policy and, by doing so, to learn qualitative behavior that one could expect a top player to master.

Indeed, the agent is able to generate a wide range of deployments, which makes it difficult for a human player to find patterns to exploit by adapting their own deployment. We describe DeepNash's deployment behavior in more detail in the supplementary materials (see the additional results section). DeepNash was able to make nontrivial trade-offs between information and material, to execute bluffs, and to take gambles when needed. The rest of this section illustrates these behaviors through matches that were played on Gravon.

For convenience, the behavior is described in a way a human observer might naturally interpret it, including terms such as "deception" and "bluffing," which arguably refer to mental states that the program does not have.

Trade-off between information and material

An important tactic in Stratego is to keep as much information as possible hidden from an opponent to gain an advantage. During certain game situations, there will be trade-offs to be considered in which a player needs to balance the value of capturing an opponent's piece (or even moving a piece), and thus revealing information about their own piece, versus not capturing a piece (or not moving) but keeping the identity of a piece hidden. DeepNash was able to make such trade-offs in extraordinary ways.

Figure 4A shows a situation in which DeepNash (in blue) was behind in pieces (it lost a 7 and an 8) but was ahead in information; the opponent in red has its 10, 9, an 8 and two of its 7's revealed. Valuing information and material in Stratego is nontrivial a priori, but the agent has learned a policy through self-play that seems to naturally make this trade-off. In the above example, DeepNash was behind in material but knew the identity of many of the opponent's high-ranked pieces. On the contrary, almost all of DeepNash's remaining pieces had not yet moved and its opponent was left in the dark. The value function (v = 0.403) credited this information asymmetry as an advantage for DeepNash (with an expected win rate of ~70%) despite having lesser material on the board. This game was won by DeepNash.

The second example in Fig. 4B shows a situation in which DeepNash had the opportunity of capturing the opponent's 6 with its 9, but this move was not considered, probably because protecting the identity of the 9 was deemed more important than the material gain. The situation also illustrates the stochasticity of DeepNash's policy during game-play.

Deceptive behavior and bluffing

In addition to being able to value asymmetry of information, one can also expect the agent to occasionally bluff to deceive its opponent and potentially gain an advantage. The situations shown in Fig. 5, A to C, illustrate this ability. Figure 5A illustrates positive bluffing, in which a player pretended that a piece had higher value than it actually did. DeepNash (blue) chased the opponent's 8 with an unknown piece, a scout (2), pretending it was the 10. The opponent believed that this piece had a high chance of being the 10 and guided it next to its spy (which could capture the 10). In an attempt to capture this piece, however, the opponent lost its spy to DeepNash's scout.

A second type of bluff, called negative bluffing, is shown in Fig. 5B. In contrast to a positive bluff, this tactic entails pretending a piece is of a lower rank. Here, the movement of the unknown 10 of DeepNash (blue) was interpreted by the opponent as a positive bluff because they tried to capture it with a known 8. DeepNash's move could have been interpreted as moving the spy closer to the opponent's 10, for example. The opponent instead encountered DeepNash's 10 and lost an 8.

A more complex bluff is shown in Fig. 5C, where DeepNash (blue) brought its unrevealed scout (2) close to the opponent's 10, which can be easily interpreted as a spy. This tactic actually allowed blue to capture red's 5 with its 7 a few steps later, thereby gaining material but also preventing the 5 from capturing the scout (2), and revealing that it was actually not the spy.

Conclusion

This work introduces a new game-theoretic method at scale that allows for AI to play the imperfect information game Stratego from scratch in self-play up to a human expert level, as illustrated by our bot DeepNash. This model-free learning method combines a deep residual neural network with the game-theoretical R-NaD multiagent learning algorithm. No form of search or explicit opponent modeling is performed during training, and DeepNash only relies on the use of some game-specific heuristics at test time. As such, the method underlying DeepNash takes a contrasting approach to state-of-the-art search-based learning methods that have been successfully applied to other complex games such as Go and chess and to imperfect information games such as poker and Scotland Yard. However, because of their computational toll and the inherent complexity of the Stratego game itself, those methods are not applicable to such an elaborate game.

The core component behind DeepNash is the at-scale implementation of the R-NaD algorithm. It performs three essential stages in an iteration of the algorithm: reward transformation starting from a random regularized policy to define a modified game, subsequently applying the replicator dynamics on this modified game to converge to a fixed point policy, and finally updating the regularization policy to this new fixed point. Repeatedly applying this three-stage process yields a strategy that is empirically difficult to exploit.

Evaluated against other AI bots, DeepNash achieved a minimum win rate of 97%, and in the evaluation against human expert players on the Gravon platform, DeepNash achieved an overall win rate of 84%, which placed it in the top-three rank of both the year-to-date (2022) and all-time leaderboards. This is an extraordinary result that the Stratego community did not believe would have been possible with current techniques, judging by quotes from Thorsten Jungblut (owner of the Gravon platform) and Vincent de Boer, which can be found in the supplementary materials.

Looking forward, at this stage, there are no indications of how R-NaD fares beyond zero-sum two-player settings. However, it is reasonable to assume that it can unlock further applications of RL methods in real-world multiagent problems with astronomical state spaces characterized by imperfect information, which are currently out of reach for state-of-the-art AI methods to be applied in an end-to-end fashion. For example, state-of-the-art methods on two-player poker (*4*) have successfully transferred to six-player poker (*5*). Many applications can be found in this larger class of games, including crowd and traffic modeling, smart grid, auction design, and market problems.

REFERENCES AND NOTES

1. M. Campbell, A. J. Hoane Jr., F. Hse, *Artif. Intell.* **134**, 57–83 (2002).
2. D. Silver *et al.*, *Science* **362**, 1140–1144 (2018).
3. M. Moravčík *et al.*, *Science* **356**, 508–513 (2017).
4. N. Brown, T. Sandholm, *Science* **359**, 418–424 (2018).
5. N. Brown, T. Sandholm, *Science* **365**, 885–890 (2019).
6. M. Schmid *et al.*, Player of games. arXiv:2112.03178 [cs.AI] (2021).
7. Stratego is a trademark of Jumbo Diset Group and is used in this publication for information purposes only.
8. J. von Neumann, O. Morgenstern, *Theory of Games and Economic Behavior* (Princeton Univ. Press, 1947).
9. O. Vinyals *et al.*, *Nature* **575**, 350–354 (2019).
10. C. Berner *et al.*, Dota 2 with large scale deep reinforcement learning. arXiv:1912.06680 (2019).
11. M. Jaderberg *et al.*, *Science* **364**, 859–865 (2019).
12. P. R. Wurman *et al.*, *Nature* **602**, 223–228 (2022).
13. V. de Boer, L. J. M. Rothkrantz, P. Wiggers, *Int. J. Intell. Games Simul.* **5**, 25 (2008).
14. M. Schadd, M. Winands, in *Proceedings of the Twenty-First Benelux Conference on Artificial Intelligence, Eindhoven, Netherlands, 29 to 30 October 2009* (BNAIC, 2009), pp. 225–232.
15. I. Satz, *J. Int. Comput. Games Assoc.* **34**, 27–29 (2011).
16. S. Redeca, A. Groza, in *2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 6 to 8 September 2018* (IEEE, 2018), pp. 97–104.
17. S. Mcaleer, J. Lanier, R. Fox, P. Baldi, in *Advances in Neural Information Processing Systems 34* (NeurIPS, 2020), vol. 33, pp. 20238–20248.
18. J. Nash, *Ann. Math.* **54**, 286–295 (1951).
19. J. Perolat *et al.*, in *Proceedings of the 38th International Conference on Machine Learning, 18 to 24 July 2021* (ICML, 2021), pp. 8525–8535.
20. D. Silver *et al.*, *Nature* **550**, 354–359 (2017).
21. M. Lanctot *et al.*, OpenSpiel: A framework for reinforcement learning in games. arXiv:1908.09453 [cs.LG] (2019).
22. E. Zeeman, *J. Theor. Biol.* **89**, 249–270 (1981).
23. D. Bloembergen, K. Tuyls, D. Hennes, M. Kaisers, *J. Artif. Intell. Res.* **53**, 659–697 (2015).
24. H. B. McMahan, in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, 11 to 13 September 2011* (AISTATS, 2011), vol. 15, pp. 525–533.
25. K. He, X. Zhang, S. Ren, J. Sun, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, Nevada, 26 June to 1 July 2016* (CVPR, 2016), pp. 770–778.
26. L. Espeholt *et al.*, in *Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10 to 15 July 2018* (ICML, 2018), pp. 1406–1416.
27. D. Hennes *et al.*, in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, Auckland, New Zealand, 9 to 13 May 2020* (AAMAS, 2020), pp. 492–501.
28. J. Perolat *et al.*, Figure data for: Mastering the game of Stratego with model-free multiagent reinforcement learning, Zenodo (2022); https://doi.org/10.5281/zenodo.7118519.

ACKNOWLEDGMENTS

We thank the following DeepMind colleagues for their technical support and feedback during the development of DeepNash: S. Mourad, L. Prince, A. Huang, M. Schmidt, M. Bowling, G. Ostrovski, M. Riedmiller, K. Kavukcuoglu, W. Czarnecki, and S. Singh. We also thank T. Jungblut, the Gravon platform owner, for providing access to Gravon, which allowed us to evaluate DeepNash against human expert players, and for allowing us to use matches played online for illustration purposes in this manuscript, as well as the anonymous reviewers for their comments that improved the quality of this article. **Funding:** This research was funded by DeepMind Technologies Ltd. **Author contributions:** J.P., K.T., and B.D.V. initiated and oversaw the project, developed training infrastructure and algorithms, performed agent analysis, wrote the paper, and provided research direction. D.He., E.T., F.S., N.Bu., M.K., and E.L. developed the training infrastructure and algorithms, performed agent analysis, and wrote the paper. P.M., J.T.C., T.A., S.H.C., Z.W., A.G., A.M., S.Oz., M.L., and J.L. developed the training infrastructure and algorithms and performed agent analysis. D.Ha. and V.d.B. performed the agent analysis and provided research direction. S.M., R.E., F.T., T.P., T.E., S.Om., and R.M. performed the agent analysis. L.S. wrote the paper and provided research direction. M.R. and B.P. wrote the paper. N.Be. initiated and oversaw the project. D.S. and S.S. provided research direction. **Competing interests:** US provisional patent application number 63/356,009 related to this work was filed 27 June 2022: "Model-Free Reinforcement Learning with Regularized Nash Dynamics"; assignee: DeepMind Technologies Ltd. The authors declare no competing interests. **Data and materials availability:** Data for the quantitative figures are available on Zenodo (*28*). All other data needed to evaluate the conclusions in this study are present in the main text or the supplementary materials. **License information:** Copyright © 2022 the authors, some rights reserved; exclusive licensee American Association for the Advancement of Science. No claim to original US government works. https://www.science.org/about/science-licenses-journal-article-reuse

SUPPLEMENTARY MATERIALS

science.org/doi/10.1126/science.add4679
Related Work
Additional Methods
Additional Results
Tables S1 to S3
Figs. S1 to S9
References (*29*–*50*)

Submitted 30 June 2022; accepted 3 November 2022
10.1126/science.add4679