COMPUTER SCIENCE

Human-level performance in 3D multiplayer games with population-based reinforcement learning

Max Jaderberg*†, Wojciech M. Czarnecki*†, Iain Dunning†, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, Thore Graepel

Reinforcement learning (RL) has shown great success in increasingly complex single-agent environments and two-player turn-based games. However, the real world contains multiple agents, each learning and acting independently to cooperate and compete with other agents. We used a tournament-style evaluation to demonstrate that an agent can achieve human-level performance in a three-dimensional multiplayer first-person video game, *Quake III Arena* in Capture the Flag mode, using only pixels and game points scored as input. We used a two-tier optimization process in which a population of independent RL agents are trained concurrently from thousands of parallel matches on randomly generated environments. Each agent learns its own internal reward signal and rich representation of the world. These results indicate the great potential of multiagent reinforcement learning for artificial intelligence research.

End-to-end reinforcement learning (RL) methods (*1–5*) have so far not succeeded in training agents in multiagent games that combine team and competitive play owing to the high complexity of the learning problem that arises from the concurrent adaptation of multiple learning agents in the environment (*6*, *7*). We approached this challenge by studying team-based multiplayer three-dimensional (3D) first-person video games, a genre that is particularly immersive for humans (*8*) and has even been shown to improve a wide range of cognitive abilities (*9*). We focused specifically on a modified version (*10*) of *Quake III Arena* (*11*), the canonical multiplayer 3D first-person video game, whose game mechanics served as the basis for many subsequent games and which has a thriving professional scene (*12*).

The task we considered is the game mode Capture the Flag (CTF), which is played on both indoor- and outdoor-themed maps that are randomly generated for each game (Fig. 1, A and B). Two opposing teams consisting of multiple individual players compete to capture each other's flags by strategically navigating, tagging, and evading opponents. The team with the greatest number of flag captures after five minutes wins. The opposing teams' flags are situated at opposite ends of each map—a team's base—and in indoor-themed maps, the base room is colored according to the team color. In addition to moving through the environment, agents can tag opponents by activating their laser gadget when pointed at an opponent, which sends the opponent back to their base room after a short delay, known as respawning. If an agent is holding a flag when they are tagged, this flag is dropped to the floor where they are tagged and is said to be stray. CTF is played in a visually rich simulated physical environment (movie S1), and agents interact with the environment and with other agents only through their observations and actions (moving forward and backward; strafing left and right; and looking by rotating, jumping, and tagging). In contrast to previous work (*13–23*), agents do not have access to models of the environment, state of other players, or human policy priors, nor can they communicate with each other outside of the game environment. Each agent acts and learns independently, resulting in decentralized control within a team.

Learning system

We aimed to devise an algorithm and training procedure that enables agents to acquire policies that are robust to the variability of maps, number of players, and choice of teammates and opponents, a challenge that generalizes that of ad hoc teamwork (*24*). In contrast to previous work (*25*), the proposed method is based purely on end-to-end learning and generalization. The proposed training algorithm stabilizes the learning process in partially observable multiagent environments by concurrently training a diverse population of agents who learn by playing with each other. In addition, the agent population provides a mechanism for meta-optimization.

In our formulation, the agent's policy π uses the same interface available to human players. It receives raw red-green-blue (RGB) pixel input x_t from the agent's first-person perspective at time step t, produces control actions $a_t \sim \pi(\cdot|x_1, \ldots, x_t)$ by sampling from the distribution given by policy π, and receives ρ_t, game points, which are visible on the in-game scoreboard. The goal of RL in this context is to find a policy that maximizes the expected cumulative reward $\mathbb{E}\pi\left[\sum_{t=1}^{T} r_t\right]$ over a CTF game with T time steps. We used a multistep actor-critic policy gradient algorithm (*2*) with off-policy correction (*26*) and auxiliary tasks (*5*) for RL. The agent's policy π was parameterized by means of a multi–time scale recurrent neural network with external memory (Fig. 2A and fig. S11) (*27*). Actions in this model were generated conditional on a stochastic latent variable, whose distribution was modulated by a more slowly evolving prior process. The variational objective function encodes a trade-off between maximizing expected reward and consistency between the two time scales of inference (*28*). Whereas some previous hierarchical RL agents construct explicit hierarchical goals or skills (*29–32*), this agent architecture is conceptually more closely related to work outside of RL on building hierarchical temporal representations (*33–36*) and recurrent latent variable models for sequential data (*37*, *38*). The resulting model constructs a temporally hierarchical representation space in a way that promotes the use of memory (fig. S7) and temporally coherent action sequences.

For ad hoc teams, we postulated that an agent's policy π_1 should maximize the probability $\mathbb{P}(\pi_1\text{'s team wins}|\omega, \boldsymbol{\pi}_{1:N})$ of winning for its team, $\boldsymbol{\pi}_{1:\frac{N}{2}} = \left(\pi_1, \pi_2, \ldots \pi_{\frac{N}{2}}\right)$, which is composed of π_1 itself, and its teammates' policies $\pi_2, \ldots, \pi_{\frac{N}{2}}$, for a total of N players in the game

$$\mathbb{P}(\pi_1\text{'s team wins}|\omega, \pi_{1:N}) = \mathbb{E}_{\tau\sim p_\omega^{\pi_{1:N}}, \epsilon\sim\mathcal{B}(0.5)} \mathbb{1}\left[\mathbb{R}\left(\tau, \pi_{1:\frac{N}{2}}\right) > \mathbb{R}\left(\tau, \pi_{\frac{N}{2}+1:N}\right) + \epsilon - 0.5\right] \quad (1)$$

in which trajectories τ (sequences of actions, states, and rewards) are sampled from the joint probability distribution $p_\omega^{\pi_{1:N}}$ over game setup ω and actions sampled from policies. The operator $\mathbb{1}[x]$ returns 1 if and only if x is true, and $\mathbb{R}(\tau, \pi)$ returns the number of flag captures obtained by agents in π in trajectory τ. Ties are broken by ϵ, which is sampled from an independent Bernoulli distribution with probability 0.5. The distribution Ω over specific game setups is defined over the Cartesian product of the set of maps and the set of random seeds. During learning and testing, each game setup ω is sampled from Ω, $\omega \sim \Omega$. The final game outcome is too sparse to be effectively used as the sole reward signal for RL, and so we learn rewards r_t to direct the learning process toward winning; these are more frequently available than the game outcome. In our approach, we operationalized the idea that each agent has a dense internal reward function (*39–41*) by specifying $r_t = \mathbf{w}(\rho_t)$ based on the available game points signals ρ_t (points are registered for events such as capturing a flag) and, crucially, allowing the agent to learn the transformation $\mathbf{w}$ so that

DeepMind, London, UK.
*Corresponding author. Email: lejlot@google.com (W.M.C.); jaderberg@google.com (M.J.) †These authors contributed equally to this work.

Downloaded from http://science.sciencemag.org/ on July 18, 2019

policy optimization on the internal rewards r_t optimizes the policy "For The Win," giving us the "FTW agent."

Training agents in multiagent systems requires instantiations of other agents in the environment, such as teammates and opponents, to generate learning experience. A solution could be self-play RL, in which an agent is trained by playing against its own policy. Although self-play variants can prove effective in some multiagent games (*14*, *15*, *42*–*46*), these methods can be unstable and in their basic form do not support concurrent training, which is crucial for scalability. Our solution is to train in parallel a population of P different agents $\boldsymbol{\pi} = (\pi_p)_{p=1}^{P}$ that play with each other, introducing diversity among players in order to stabilize training (*47*). Each agent within this population learns from experience generated by playing with teammates and opponents sampled from the population. We sampled the agents indexed by ι for a training game by using a stochastic matchmaking scheme $m_p(\boldsymbol{\pi})$ that biases co-players to be of similar skill to player p. This scheme ensures that—a priori—the outcome is sufficiently uncertain to provide a meaningful learning signal and that a diverse set of teammates and opponents participate in training. Agents' skill levels were estimated online by calculating Elo scores [adapted from chess (*48*)] on the basis of outcomes of training games. We also used the population to meta-optimize the internal rewards and hyperparameters of the RL process itself, which results in the joint maximization of

$$J_{\text{inner}}(\pi_p|\mathbf{w}_p) = \mathbb{E}_{\iota\sim m_p(\boldsymbol{\pi}),\omega\sim\Omega}\mathbb{E}_{\tau\sim p_\omega^{\pi_\iota}}\left[\sum_{t=1}^{T}\gamma^{t-1}\mathbf{w}_p(\rho_{p,t})\right]\forall\pi_p\in\boldsymbol{\pi}$$

$$J_{\text{outer}}(\mathbf{w}_p,\phi_p|\boldsymbol{\pi}) = \mathbb{E}_{\iota\sim m_p(\boldsymbol{\pi}),\omega\sim\Omega}\,\mathbb{P}(\pi_p^{\mathbf{w},\phi}\text{'s team wins}|\omega,\boldsymbol{\pi}_\iota^{\mathbf{w},\phi}) \qquad (2)$$

$$\pi_p^{\mathbf{w},\phi} = \text{optimize}_{\pi_p}(J_{\text{inner}},\mathbf{w},\phi)$$

This can be seen as a two-tier RL problem. The inner optimization maximizes J_{inner}, the agents' expected future discounted internal rewards. The outer optimization of J_{outer} can be viewed as a meta-game, in which the meta-reward of winning the match is maximized with respect to internal reward schemes $\mathbf{w}_p$ and hyperparameters ϕ_p, with the inner optimization providing the meta transition dynamics. We solved the inner optimization with RL as previously described, and the outer optimization with population-based training (PBT) (*49*). PBT is an online evolutionary process that adapts internal rewards and hyperparameters and performs model selection by replacing underperforming agents with mutated versions of better agents. This joint optimization of the agent policy by using RL together with the optimization of the RL procedure itself toward a high-level goal proves to be an effective and potentially widely applicable strategy and uses the potential of combining learning and evolution (*50*) in large-scale learning systems.

Tournament evaluation

To assess the generalization performance of agents at different points during training, we performed a large tournament on procedurally generated maps with ad hoc matches that involved three types of agents as teammates and opponents: ablated versions of FTW (including state-of-the-art baselines), *Quake III Arena* scripted bots of various levels (*51*), and human participants with first-person video game experience. The Elo scores and derived winning probabilities for different ablations of FTW, and how the combination of components provide superior performance, are shown in Fig. 2B and fig. S1.

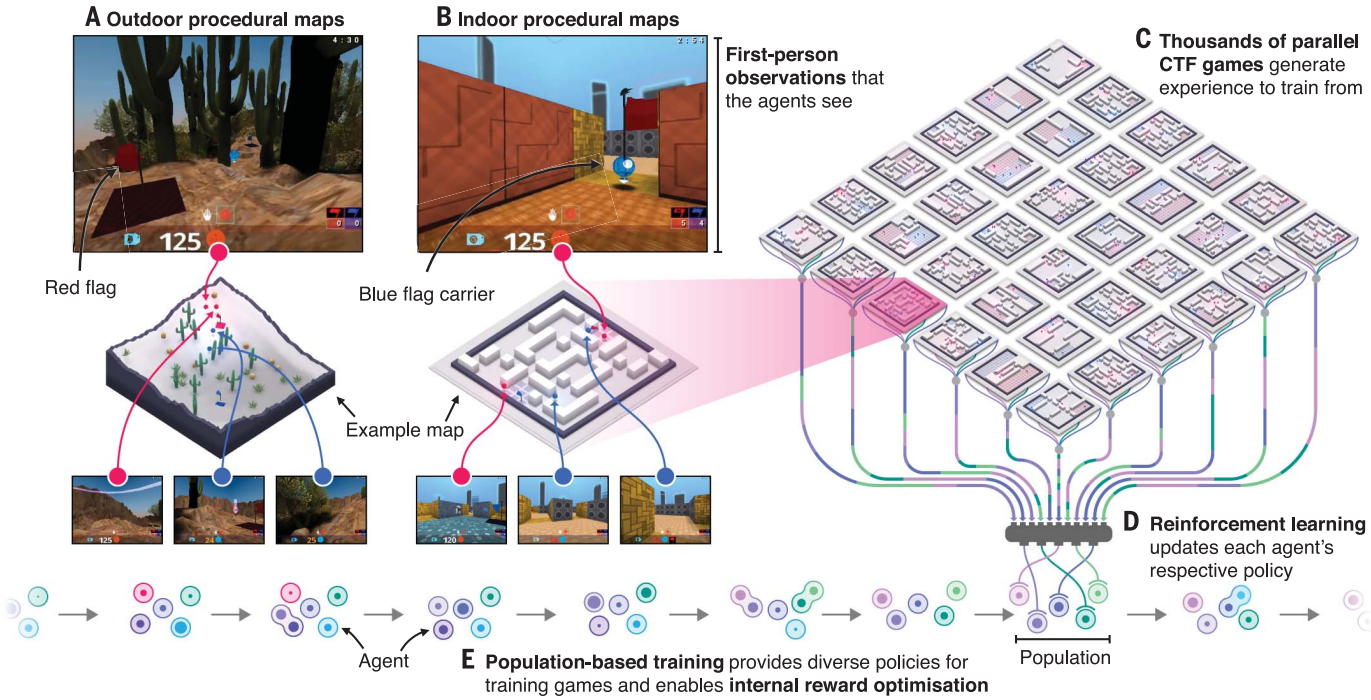

Fig. 1. CTF task and computational training framework. (**A** and **B**) Two example maps that have been sampled from the distribution of (A) outdoor maps and (B) indoor maps. Each agent in the game sees only its own first-person pixel view of the environment. (**C**) Training data are generated by playing thousands of CTF games in parallel on a diverse distribution of procedurally generated maps and (**D**) used to train the agents that played in each game with RL. (**E**) We trained a population of 30 different agents together, which provided a diverse set of teammates and opponents to play with and was also used to evolve the internal rewards and hyperparameters of agents and learning process. Each circle represents an agent in the population, with the size of the inner circle representing strength. Agents undergo computational evolution (represented as splitting) with descendents inheriting and mutating hyperparameters (represented as color). Gameplay footage and further exposition of the environment variability can be found in movie S1.

Downloaded from http://science.sciencemag.org/ on July 18, 2019

The FTW agents clearly exceeded the win-rate of humans in maps that neither agent nor human had seen previously—that is, zero-shot generalization—with a team of two humans on average capturing 16 fewer flags per game than a team of two FTW agents (fig. S1, bottom, FF versus hh). Only as part of a human-agent team did we observe a human winning over an agent-agent team (5% win probability). This result suggests that trained agents are capable of cooperating with never-seen-before teammates, such as humans. In a separate study, we probed the exploitability of the FTW agent by allowing a team of two professional games testers with full communication to play continuously against a fixed pair of FTW agents. Even after 12 hours of practice, the human game testers were only able to win 25% (6.3% draw rate) of games against the agent team (*28*).

Interpreting the difference in performance between agents and humans must take into account the subtle differences in observation resolution, frame rate, control fidelity, and intrinsic limitations in reaction time and sensorimotor skills (fig. S10A) [(*28*), section 3.1]. For example, humans have superior observation and control resolution; this may be responsible for humans successfully tagging at long range where agents could not (humans, 17% tags above 5 map units; agents, 0.5%). By contrast, at short range, agents have superior tagging reaction times to humans: By one measure, FTW agents respond to newly appeared opponents with a mean of 258 ms, compared with 559 ms for humans (fig. S10B). Another advantage exhibited by agents is their tagging accuracy, in which FTW agents achieve 80% accuracy compared with humans' 48%. By artificially reducing the FTW agents' tagging accuracy to be similar to humans (without retraining them), the agents' win rate was reduced though still exceeded that of humans (fig. S10C). Thus, although agents learned to make use of their potential for better tagging accuracy, this is only one factor contributing to their overall performance.

To explicitly investigate the effect of the native superiority in the reaction time of agents compared with that of humans, we introduced an artificial 267-ms reaction delay to the FTW agent (in line with the previously reported discrepancies, and corresponding to fast human reaction times in simple psychophysical paradigms) (*52–54*). This response-delayed FTW agent was fine-tuned from the nondelayed FTW agent through a combination of RL and distillation through time [(*28*), section 3.1.1]. In a further exploitability study, the human game testers achieved a 30% win rate against the resulting response-delayed agents. In additional tournament games with a wider pool of human participants, a team composed of a strong human and a response-delayed agent could only achieve an average win rate of 21% against a team of entirely response-delayed agents. The human participants performed slightly more tags than the response-delayed agent opponents, although delayed agents achieved more flag pickups and captures (Fig. 2). This highlights that even with more human-comparable reaction times, the agent exhibits human-level performance.

Agent analysis

We hypothesized that trained agents of such high skill have learned a rich representation of the game. To investigate this, we extracted ground-truth state from the game engine at each point in time in terms of 200 binary features such as "Do I have the flag?", "Did I see my teammate recently?", and "Will I be in the opponent's base soon?" We say that the agent has knowledge of a given feature if logistic regression on the internal state of the agent accurately models the feature. In this sense, the internal representation of the agent was found to encode a wide variety of knowledge about the game situation (fig. S4). The FTW agent's representation was found to encode features related to the past particularly

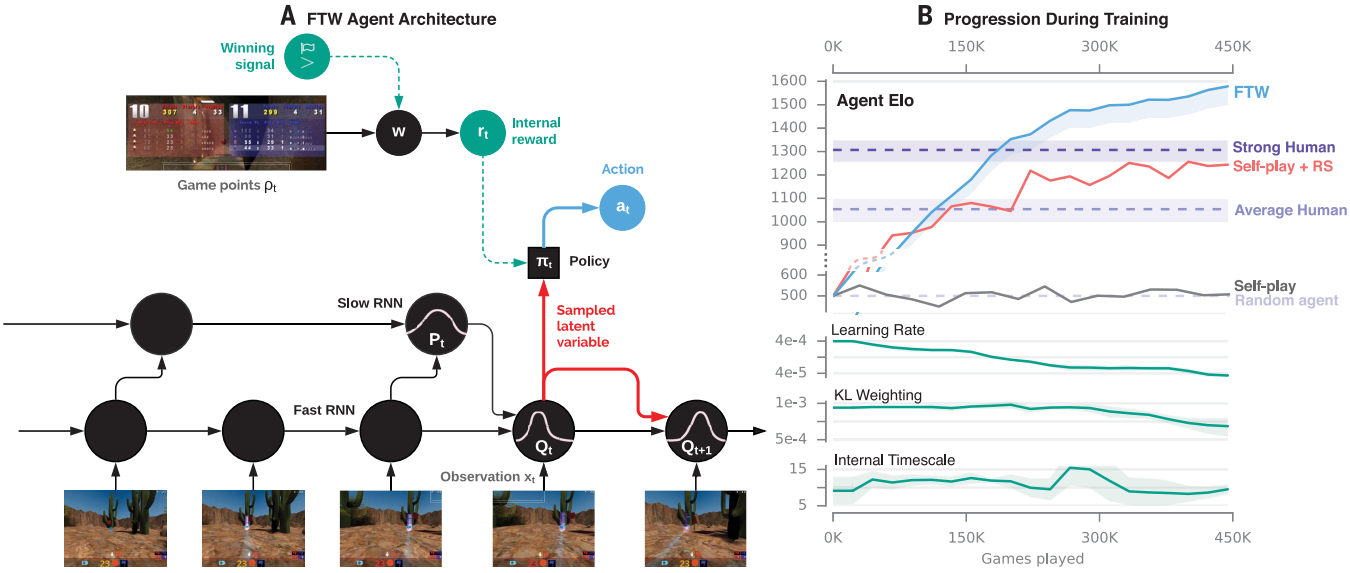

Fig. 2. Agent architecture and benchmarking. (**A**) How the agent processes a temporal sequence of observations x_t from the environment. The model operates at two different time scales, faster at the bottom and slower by a factor of τ at the top. A stochastic vector-valued latent variable is sampled at the fast time scale from distribution $\mathbb{Q}_t$ on the basis of observations x_t. The action distribution π_t is sampled conditional on the latent variable at each time step t. The latent variable is regularized by the slow moving prior $\mathbb{P}_t$, which helps capture long-range temporal correlations and promotes memory. The network parameters are updated by using RL according to the agent's own internal reward signal r_t, which is obtained from a learned transformation **w** of game points ρ_t. **w** is optimized for winning probability through PBT, another level of training performed at yet a slower time scale than that of RL. Detailed network architectures are described in fig. S11. (**B**) (Top) The Elo skill ratings of the FTW agent population throughout training (blue) together with those of the best baseline agents by using hand-tuned reward shaping (RS) (red) and game-winning reward signal only (black), compared with human and random agent reference points (violet, shaded region shows strength between 10th and 90th percentile). The FTW agent achieves a skill level considerably beyond strong human subjects, whereas the baseline agent's skill plateaus below and does not learn anything without reward shaping [evaluation procedure is provided in (*28*)]. (Bottom) The evolution of three hyperparameters of the FTW agent population: learning rate, Kullback-Leibler divergence (KL) weighting, and internal time scale τ, plotted as mean and standard deviation across the population.

Downloaded from http://science.sciencemag.org/ on July 18, 2019

well; for example, the FTW agent was able to classify the state "both flags are stray" (flags dropped not at base) with 91% AUCROC (area under the receiver operating characteristic curve), compared with 70% with the self-play baseline. Looking at the acquisition of knowledge as training progresses, the agent first learned about its own base, then about the opponent's base, and then about picking up the flag. Immediately useful flag knowledge was learned before knowledge related to tagging or their teammate's situation. Agents were never explicitly trained to model this knowledge; thus, these results show the spontaneous emergence of these concepts purely through RL-based training.

A visualization of how the agent represents knowledge was obtained by performing dimensionality reduction of the agent's activations through use of t-distributed stochastic neighbor embedding (t-SNE) (Fig. 3) (*55*). Internal agent state clustered in accordance with conjunctions of high-level game-state features: flag status, respawn state, and agent location (Fig. 3B). We also found individual neurons whose activations coded directly for some of these features—for example, a neuron that was active if and only if the agent's teammate was holding the flag, which is reminiscent of concept cells (*56*). This knowledge was acquired in a distributed manner early in training (after 45,000 games) but then represented by a single, highly discriminative neuron later in training (at around 200,000 games). This observed disentangling of game state is most pronounced in the FTW agent (fig. S8).

One of the most salient aspects of the CTF task is that each game takes place on a randomly generated map, with walls, bases, and flags in new locations. We hypothesized that this requires agents to develop rich representations of these spatial environments in order to deal with task demands and that the temporal hierarchy and explicit memory module of the FTW agent help toward this. An analysis of the memory recall patterns of the FTW agent playing in indoor environments shows precisely that; once the agent had discovered the entrances to the two bases, it primarily recalled memories formed at these base entrances (Fig. 4 and fig. S7). We also found that the full FTW agent with temporal hierarchy learned a coordination strategy during maze navigation that ablated versions of the

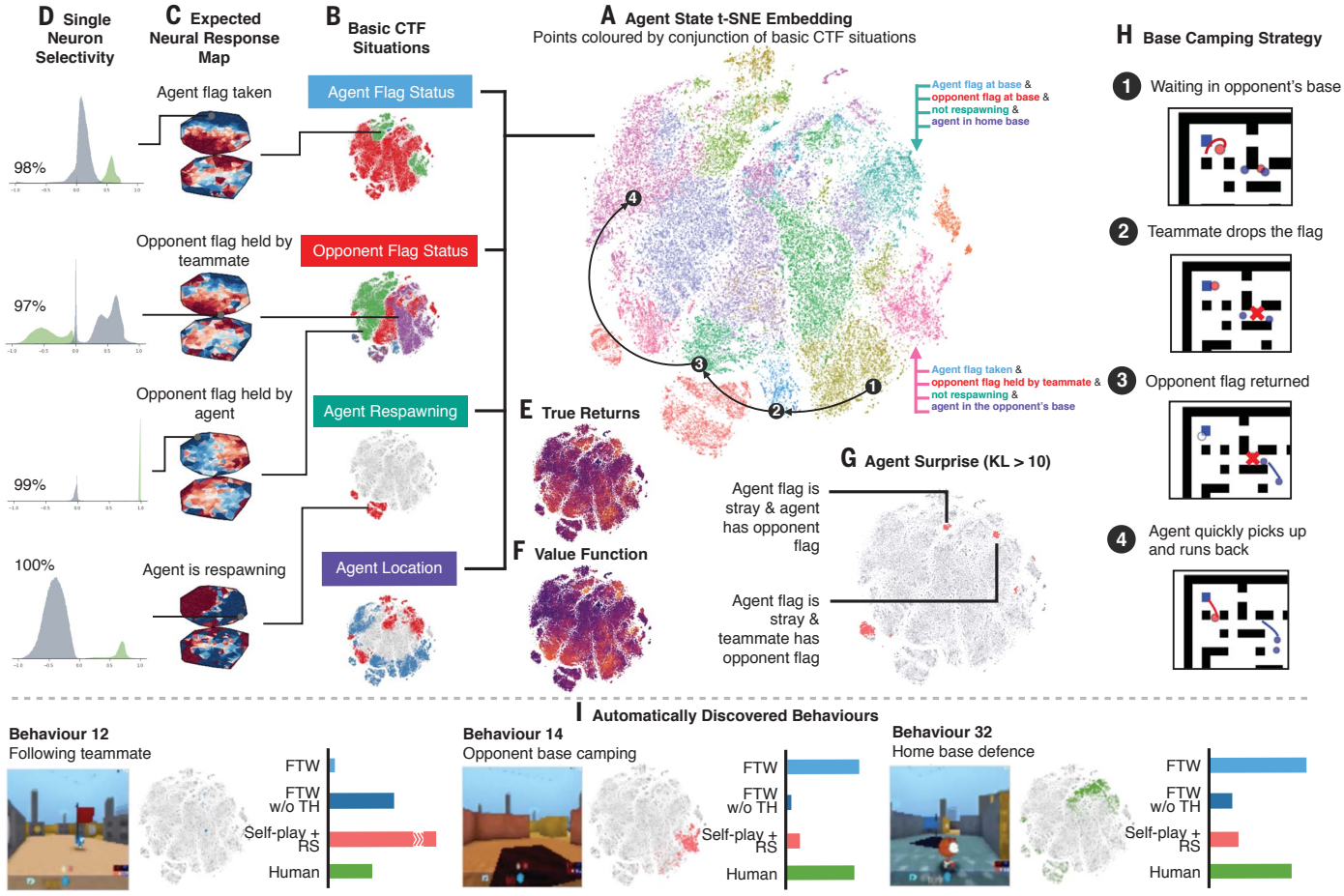

Fig. 3. Knowledge representation and behavioral analysis. (**A**) The 2D t-SNE embedding of an FTW agent's internal states during gameplay. Each point represents the internal state ($\mathbf{h}^p$, $\mathbf{h}^q$) at a particular point in the game and is colored according to the high-level game state at this time—the conjunction of (**B**) four basic CTF situations, each state of which is colored distinctly. Color clusters form, showing that nearby regions in the internal representation of the agent correspond to the same high-level game state. (**C**) A visualization of the expected internal state arranged in a similarity-preserving topological embedding and colored according to activation (fig. S5). (**D**) Distributions of situation conditional activations (each conditional distribution is colored gray and green) for particular single neurons that are distinctly selective for these CTF situations and show the predictive accuracy of this neuron. (**E**) The true return of the agent's internal reward signal and (**F**) the agent's prediction, its value function (orange denotes high value, and purple denotes low value). (**G**) Regions where the agent's internal two–time scale representation diverges (red), the agent's surprise, measured as the KL between the agent's slow– and fast–time scale representations (*28*). (**H**) The four-step temporal sequence of the high-level strategy "opponent base camping." (**I**) Three automatically discovered high-level behaviors of agents and corresponding regions in the t-SNE embedding. (Right) Average occurrence per game of each behavior for the FTW agent, the FTW agent without temporal hierarchy (TH), self-play with reward shaping agent, and human subjects (fig. S9).

Downloaded from http://science.sciencemag.org/ on July 18, 2019

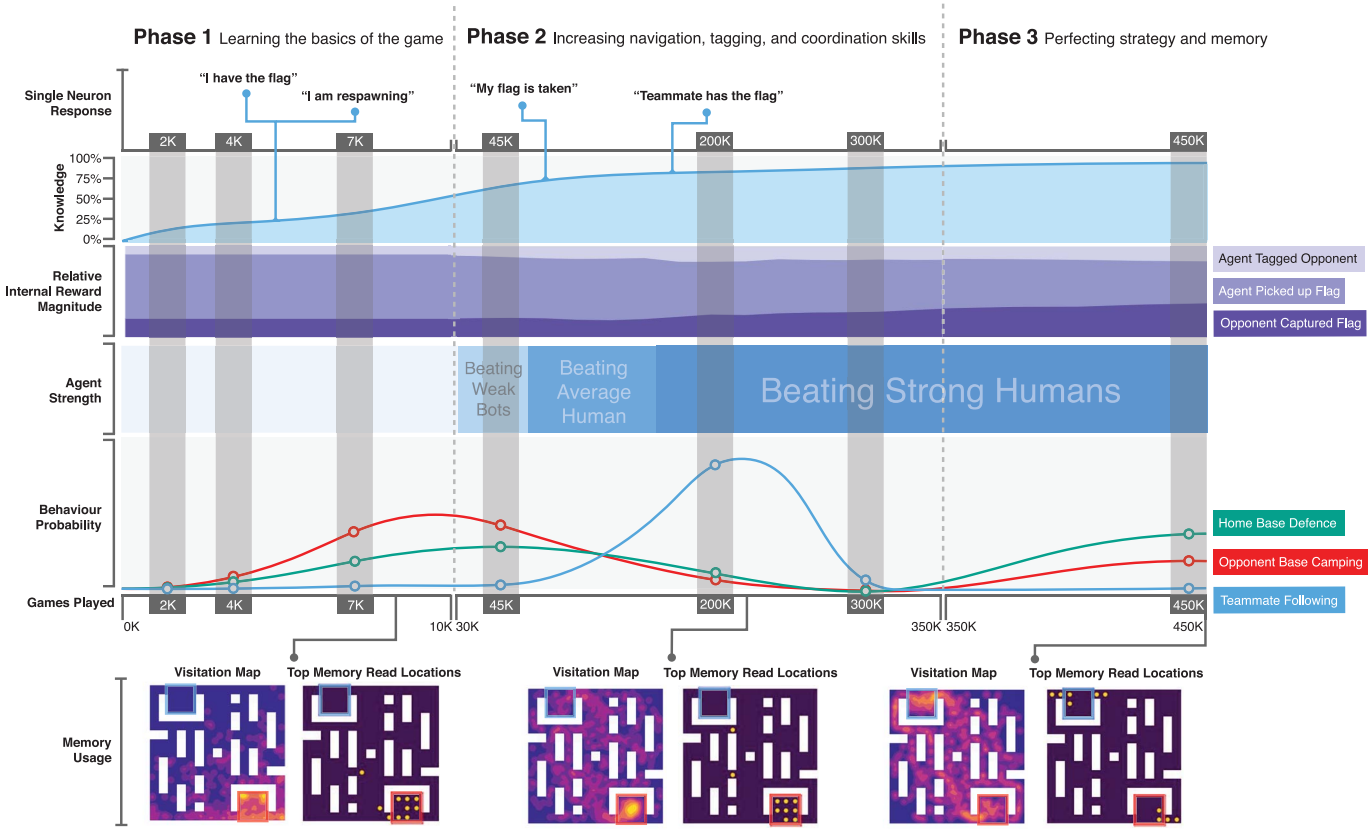

Fig. 4. Progression of agent during training. Shown is the development of knowledge representation and behaviors of the FTW agent over the training period of 450,000 games, segmented into three phases (movie S2). "Knowledge" indicates the percentage of game knowledge that is linearly decodable from the agent's representation, measured by average scaled AUCROC across 200 features of game state. Some knowledge is compressed to single-neuron responses (Fig. 3A), whose emergence in training is shown at the top. "Relative internal reward magnitude" indicates the relative magnitude of the agent's internal reward weights of 3 of the 13 events corresponding to game points ρ. Early in training, the agent puts large reward weight on picking up the opponent's flag, whereas later, this weight is reduced, and reward for tagging an opponent and penalty when opponents capture a flag are increased by a factor of two. "Behavior probability" indicates the frequencies of occurrence for 3 of the 32 automatically discovered behavior clusters through training. Opponent base camping (red) is discovered early on, whereas teammate following (blue) becomes very prominent midway through training before mostly disappearing. The "home base defense" behavior (green) resurges in occurrence toward the end of training, which is in line with the agent's increased internal penalty for more opponent flag captures. "Memory usage" comprises heat maps of visitation frequencies for (left) locations in a particular map and (right) locations of the agent at which the top-10 most frequently read memories were written to memory, normalized by random reads from memory, indicating which locations the agent learned to recall. Recalled locations change considerably throughout training, eventually showing the agent recalling the entrances to both bases, presumably in order to perform more efficient navigation in unseen maps (fig. S7).

agent did not, resulting in more efficient flag capturing (fig. S2).

Analysis of temporally extended behaviors provided another view on the complexity of behavioral strategies learned by the agent (*57*) and is related to the problem a coach might face when analyzing behavior patterns in an opponent team (*58*). We developed an unsupervised method to automatically discover and quantitatively characterize temporally extended behavior patterns, inspired by models of mouse behavior (*59*), which groups short game-play sequences into behavioral clusters (fig. S9 and movie S3). The discovered behaviors included well-known tactics observed in human play, such as "waiting in the opponents base for a flag to reappear" ("opponent base camping"), which we only observed in FTW agents with a temporal hierarchy. Some behaviors, such as "following a flag-carrying teammate," were discovered and discarded midway through training, whereas others such as "performing home base defense" are most prominent later in training (Fig. 4).

Conclusions

In this work, we have demonstrated that an artificial agent using only pixels and game points as input can learn to play highly competitively in a rich multiagent environment: a popular multiplayer first-person video game. This was achieved by combining PBT of agents, internal reward optimization, and temporally hierarchical RL with scalable computational architectures. The presented framework of training populations of agents, each with their own learned rewards, makes minimal assumptions about the game structure and therefore could be applicable for scalable and stable learning in a wide variety of multiagent systems. The temporally hierarchical agent represents a powerful architecture for problems that require memory and temporally extended inference. Limitations of the current framework, which should be addressed in future work, include the difficulty of maintaining diversity in agent populations, the greedy nature of the meta-optimization performed by PBT, and the variance from temporal credit assignment in the proposed RL updates. Our work combines techniques to train agents that can achieve human-level performance at previously insurmountable tasks. When trained in a sufficiently rich multiagent world, complex and surprising high-level intelligent artificial behavior emerged.

Downloaded from http://science.sciencemag.org/ on July 18, 2019

REFERENCES AND NOTES

1. V. Mnih *et al*., *Nature* **518**, 529–533 (2015).
2. V. Mnih *et al*., *Proc. Int. Conf. Mach. Learn.* **48**, pp. 1928–1937 (2016).
3. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms. arXiv:1707.06347 [cs.LG] (2017).
4. T. P. Lillicrap *et al*., Continuous control with deep reinforcement learning. *Proc. Int. Conf. Learn. Rep.* (2016).
5. M. Jaderberg *et al*., Reinforcement learning with unsupervised auxiliary tasks. *Proc. Int. Conf. Learn. Rep.* (2017).
6. D. S. Bernstein, S. Zilberstein, N. Immerman, The complexity of decentralized control of MDPs. *Proc. Conf. Uncert. Artif. Intell.* **16**, pp. 32–37 (2000).
7. L. Matignon, G. J. Laurent, N. Le Fort-Piat, *Knowl. Eng. Rev.* **27**, 1–31 (2012).
8. L. Ermi, F. Mäyrä, *Worlds Play Int. Persp. Dig. Games Res.* **37**, 37–53 (2005).
9. C. S. Green, D. Bavelier, *Curr. Opin. Behav. Sci.* **4**, 103–108 (2015).
10. C. Beattie *et al*., DeepMind Lab. arXiv:1612.03801 [cs.AI] (2016).
11. id Software, *Quake III Arena* (id Software, 1999).
12. QuakeCon 2018, Grapevine, TX, 9 August 2018 to 12 August 2018.
13. G. Tesauro, *Commun. ACM* **38**, 58–68 (1995).
14. D. Silver *et al*., *Nature* **550**, 354–359 (2017).
15. M. Moravčík *et al*., *Science* **356**, 508–513 (2017).
16. J. N. Foerster *et al*., Learning with opponent-learning awareness. arXiv:1709.04326 [cs.AI] (2017).
17. R. Lowe *et al*., Multi-agent actor-critic for mixed cooperative-competitive environments. *Adv. Neur. Inf. Process. Syst.* **30**, 6382–6393 (2017).
18. I. Mordatch, P. Abbeel, Emergence of grounded compositional language in multi-agent populations. *Proc. AAAI Conf. Artif. Intell.* (2018).
19. S. Sukhbaatar, A. Szlam, R. Fergus, Learning multiagent communication with backpropagation. *Adv. Neur. Inf. Process. Syst.*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett, Eds. **29**, 2244–2252 (2016).
20. M. Riedmiller, T. Gabel, *IEEE Symposium on Computational Intelligence and Games, 2007 (CIG 2007)* (IEEE, 2007), pp. 17–23.
21. P. Stone, M. Veloso, in *European Conference on Machine Learning* (Springer, 2000), pp. 369–381.
22. P. MacAlpine, P. Stone, in *RoboCup 2017: Robot Soccer World Cup XXI*, C. Sammut, O. Obst, F. Tonidandel, H. Akyama, Eds., lecture notes in artificial intelligence (Springer, 2018).
23. J. Foerster, I. A. Assael, N. de Freitas, S. Whiteson, Learning to communicate with deep multi-agent reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2016**, 2137–2145 (2016).
24. P. Stone *et al*., Ad hoc autonomous agent teams: Collaboration without precoordination. *Proc. AAAI Conf. Artif. Intell.* (2010).
25. S. Barrett, P. Stone, S. Kraus, 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2 to 6, 2011, vols. 1 to 3 (2011), pp. 567–574.
26. L. Espeholt *et al*., IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. arXiv:1802.01561 [cs.LG] (2018).
27. A. Graves *et al*., *Nature* **538**, 471–476 (2016).
28. Additional information is available as supplementary materials.
29. R. S. Sutton, D. Precup, S. P. Singh, *Artif. Intell.* **112**, 181–211 (1999).
30. T. G. Dietterich, *J. Artif. Intell. Res.* **13**, 227–303 (2000).
31. A. S. Vezhnevets *et al*., FeUdal networks for hierarchical reinforcement learning. *Proc. Int. Conf. Mach. Learn.* **70**, 3540–3549 (2017).
32. P. Bacon, J. Harb, D. Precup, The option-critic architecture. *Proc. AAAI Conf. Artif. Intell.* **17**, 726–1734 (2017).
33. J. Koutník, K. Greff, F. Gomez, J. Schmidhuber, A clockwork RNN. arXiv:1402.3511 [cs.NE] (2014).
34. J. Chung, S. Ahn, Y. Bengio, Hierarchical multiscale recurrent neural networks. *Proc. Int. Conf. Learn. Rep.* (2017).
35. J. Schmidhuber, *Neural Comput.* **4**, 234–242 (1992).
36. S. El Hihi, Y. Bengio, Hierarchical recurrent neural networks for long-term dependencies. *Proc. Ann. Conf. Neur. Inf. Process. Syst.* **400**, 493–499 (1996).
37. J. Chung *et al*., A recurrent latent variable model for sequential data. *Proc. Ann. Conf. Neur. Inf. Process. Syst.* **2015**, 2980–2988 (2015).
38. M. Fraccaro, S. K. Sønderby, U. Paquet, O. Winther, Sequential neural models with stochastic layers. *Adv. Neur. Inf. Process. Syst.* **2016**, 2199–2207 (2016).
39. S. Singh, R. L. Lewis, A. G. Barto, Where do rewards come from? *Proc. Ann. Meet. Cogn. Sci. Soc.* **2009**, 2601–2606 (2009).
40. S. Singh, R. L. Lewis, A. G. Barto, J. Sorg, *IEEE Trans. Auton. Ment. Dev.* **2**, 70–82 (2010).
41. D. H. Wolpert, K. Tumer, An introduction to collective intelligence. arXiv:cs/9908014 [cs.LG] (1999).
42. D. Silver *et al*., *Nature* **529**, 484–489 (2016).
43. T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, I. Mordatch, Emergent complexity via multi-agent competition. *Proc. Int. Conf. Learn. Rep.* (2018).
44. G. W. Brown, in *Activity Analysis of Production and Allocation*, T. Koopmans, Ed. (John Wiley & Sons, 1951), pp. 374–376.
45. M. Lanctot *et al*., A unified game-theoretic approach to multiagent reinforcementlearning. *Adv. Neur. Inf. Process. Syst.* **2017**, 4193–4206 (2017).
46. J. Heinrich, D. Silver, *NIPS Deep Reinforcement Learning Workshop* (2016).
47. C. D. Rosin, R. K. Belew, *Evol. Comput.* **5**, 1–29 (1997).
48. A. E. Elo, *The Rating of Chessplayers, Past and Present* (Arco, 1978).
49. M. Jaderberg *et al*., Population based training of neural networks. arXiv:1711.09846 [cs.LG] (2017).
50. D. Ackley, M. Littman, *Artificial life II* **10**, 487 (1991).
51. J. M. P. Van Waveren, thesis, University of Technology Delft, Delft, Netherlands (2001).
52. P. M. Greenfield, P. DeWinstanley, H. Kilpatrick, D. Kaye, *J. Appl. Dev. Psychol.* **15**, 105–123 (1994).
53. J. Deleuze, M. Christiaens, F. Nuyens, J. Billieux, *Comput. Human Behav.* **72**, 570–576 (2017).
54. M. W. Dye, C. S. Green, D. Bavelier, *Curr. Dir. Psychol. Sci.* **18**, 321–326 (2009).
55. L. J. P. Van der Maaten, G. Hinton, *J. Mach. Learn. Res.* **9**, 2579 (2008).
56. R. Q. Quiroga, *Nat. Rev. Neurosci.* **13**, 587–597 (2012).
57. J. W. Krakauer, A. A. Ghazanfar, A. Gomez-Marin, M. A. MacIver, D. Poeppel, *Neuron* **93**, 480–490 (2017).
58. G. Kuhlmann, W. B. Knox, P. Stone, Know thine enemey: A champion robocup coach agent. *Proc. 21st Natl. Conf. Artif. Intell.* **2006**, 1463–68 (2006).
59. A. B. Wiltschko *et al*., *Neuron* **88**, 1121–1135 (2015).
60. M. Jaderberg *et al*., Supplementary Data for "Human-level performance in 3D multiplayer games with population-based reinforcement learning". Harvard Dataverse (2019); doi: 10.7910/DVN/JJETYE.

ACKNOWLEDGMENTS

We thank M. Botvinick, S. Osindero, V. Mnih, A. Graves, N. de Freitas, N. Heess, and K. Tuyls for helpful comments on the manuscript; A. Grabska-Barwińska for support with analysis; S. Green and D. Purves for additional environment support and design; K. McKee and T. Zhu for human experiment assistance; A. Sadik, S. York, and P. Mendolicchio for exploitation study participation; A. Cain for help with figure design; P. Lewis, D. Fritz, and J. Sanchez Elias for 3D map visualization work; V. Holgate, A. Bolton, C. Hillier, and H. King for organizational support; and the rest of the DeepMind team for their invaluable support and ideas. **Author contributions:** M.J. and T.Gra. conceived and managed the project; M.J., W.M.C., and I.D. designed and implemented the learning system and algorithm with additional help from L.M., T.Gra., G.L., N.S., T.Gre., and J.Z.L.; A.G.C., C.B., and L.M. created the game environment presented; M.J., W.M.C., I.D., and L.M. ran experiments and analyzed data with additional input from N.C.R., A.S.M., and A.R.; L.M. and L.D. ran human experiments; D.S., D.H., and K.K. provided additional advice and management; M.J., W.M.C., and T.Gra. wrote the paper; and M.J. and W.M.C. created figures and videos. **Competing interests:** M.J., W.M.C., and I.D. are inventors on U.S. patent application US62/677,632 submitted by DeepMind that covers temporally hierarchical RL. M.J., W.M.C., and T.G. are inventors on U.S. patent application PCT/EP2018/082162 submitted by DeepMind that covers population based training of neural networks. I.D. is additionally affiliated with Hudson River Trading, New York, NY, USA. **Data and materials availability:** A full description of the algorithm in pseudocode is available in the supplementary materials. The data are deposited in 10.7910/DVN/JJETYE (*60*).

SUPPLEMENTARY MATERIALS

science.sciencemag.org/content/364/6443/859/suppl/DC1
Supplementary Text
Figs. S1 to S12
References (*61–83*)
Pseudocode
Supplementary Data
Movies S1 to S4

3 July 2018; accepted 2 May 2019
10.1126/science.aau6249

Downloaded from http://science.sciencemag.org/ on July 18, 2019

Science

Human-level performance in 3D multiplayer games with population-based reinforcement learning

Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu and Thore Graepel

Science **364** (6443), 859-865.
DOI: 10.1126/science.aau6249

Artificial teamwork

Artificially intelligent agents are getting better and better at two-player games, but most real-world endeavors require teamwork. Jaderberg *et al.* designed a computer program that excels at playing the video game *Quake III Arena* in Capture the Flag mode, where two multiplayer teams compete in capturing the flags of the opposing team. The agents were trained by playing thousands of games, gradually learning successful strategies not unlike those favored by their human counterparts. Computer agents competed successfully against humans even when their reaction times were slowed to match those of humans.

Science, this issue p. 859

ARTICLE TOOLS http://science.sciencemag.org/content/364/6443/859

SUPPLEMENTARY MATERIALS http://science.sciencemag.org/content/suppl/2019/05/29/364.6443.859.DC1

RELATED CONTENT file:/content

REFERENCES This article cites 38 articles, 1 of which you can access for free
http://science.sciencemag.org/content/364/6443/859#BIBL

PERMISSIONS http://www.sciencemag.org/help/reprints-and-permissions

Use of this article is subject to the Terms of Service

Science (print ISSN 0036-8075; online ISSN 1095-9203) is published by the American Association for the Advancement of Science, 1200 New York Avenue NW, Washington, DC 20005. 2017 © The Authors, some rights reserved; exclusive licensee American Association for the Advancement of Science. No claim to original U.S. Government Works. The title *Science* is a registered trademark of AAAS.

Downloaded from http://science.sciencemag.org/ on July 18, 2019