OpenAI Five

JUNE 25, 2018

Our team of five neural networks, OpenAI Five, has started to defeat amateur human teams at Dota 2. While today we play with restrictions, we aim to beat a team of top professionals at The International in August subject only to a limited set of heroes. We may not succeed: Dota 2 is one of the most popular and complex esports games in the world, with creative and motivated professionals who train year-round to earn part of Dota's annual $40M prize pool (the largest of any esports game).

OpenAI Five plays 180 years worth of games against itself every day, learning via self-play. It trains using a scaled-up version of Proximal Policy Optimization running on 256 GPUs and 128,000 CPU cores — a larger-scale version of the system we built to play the much-simpler solo variant of the game last year. Using a separate LSTM for each hero and no human data, it learns recognizable strategies. This indicates that reinforcement learning can yield long-term planning with large but achievable scale — without fundamental advances, contrary to our own expectations upon starting the project.

To benchmark our progress, we'll host a match versus top players on August 5th. Follow us on Twitch to view the live broadcast, or request an invite to attend in person!

OpenAI Five playing the best OpenAI employee team. The match was commentated by professional commentator Blitz and OpenAI Dota team member Christy Dennison, and observed by a crowd from the community.

The problem

One AI milestone is to exceed human capabilities in a complex video game like StarCraft or Dota. Relative to previous AI milestones like Chess or Go, complex video games start to capture the messiness and continuous nature of the real world. The hope is that systems which solve complex video games will be highly general, with applications outside of games.

Dota 2 is a real-time strategy game played between two teams of five players, with each player controlling a character called a "hero". A Dota-

playing AI must master the following:

- **Long time horizons.** Dota games run at 30 frames per second for an average of 45 minutes, resulting in 80,000 ticks per game. Most actions (like ordering a hero to move to a location) have minor impact individually, but some individual actions like town portal usage can affect the game strategically; some strategies can play out over an entire game. OpenAI Five observes every fourth frame, yielding 20,000 moves. Chess usually ends before 40 moves, Go before 150 moves, with almost every move being strategic.
- **Partially-observed state.** Units and buildings can only see the area around them. The rest of the map is covered in a fog hiding enemies and their strategies. Strong play requires making inferences based on incomplete data, as well as modeling what one's opponent might be up to. Both chess and Go are full-information games.
- **High-dimensional, continuous action space.** In Dota, each hero can take dozens of actions, and many actions target either another unit or a position on the ground. We discretize the space into 170,000 possible actions per hero (not all valid each tick, such as using a spell on cooldown); not counting the continuous parts, there are an average of ~1,000 valid actions each tick. The average number of actions in chess is 35; in Go, 250.
- **High-dimensional, continuous observation space.** Dota is played on a large continuous map containing ten heroes, dozens of buildings, dozens of NPC units, and a long tail of game features such as runes, trees, and wards. Our model observes the state of a Dota game via Valve's Bot API as 20,000 (mostly floating-point) numbers representing all information a human is allowed to access. A chess board is naturally represented as about 70 enumeration values (a 8x8 board of 6 piece types and minor historical info); a Go board as about 400 enumeration values (a 19x19 board of 2 piece types plus Ko).

The Dota rules are also very complex — the game has been actively developed for over a decade, with game logic implemented in hundreds of thousands of lines of code. This logic takes milliseconds per tick to execute, versus nanoseconds for Chess or Go engines. The game also

gets an update about once every two weeks, constantly changing the environment semantics.

Our approach

Our system learns using a massively-scaled version of Proximal Policy Optimization. Both OpenAI Five and our earlier 1v1 bot learn entirely from self-play. They start with random parameters and do not use search or bootstrap from human replays.

	OPENAI 1V1 BOT	OPENAI FIVE
CPUs	60,000 CPU cores on Azure	128,000 preemptible CPU cores on GCP
GPUs	256 K80 GPUs on Azure	256 P100 GPUs on GCP
Experience collected	~300 years per day	~180 years per day (~900 years per day counting each hero separately)
Size of observation	~3.3 kB	~36.8 kB
Observations per second of gameplay	10	7.5
Batch size	8,388,608 observations	1,048,576 observations
Batches per minute	~20	~60

RL researchers (including ourselves) have generally believed that long time horizons would require fundamentally new advances, such as hierarchical reinforcement learning. Our results suggest that we haven't been giving today's algorithms enough credit — at least when they're run at sufficient scale and with a reasonable way of exploring.

Our agent is trained to maximize the exponentially decayed sum of future rewards, weighted by an exponential decay factor called γ. During the latest training run of OpenAI Five, we annealed γ from 0.998 (valuing future rewards with a half-life of 46 seconds) to 0.9997 (valuing future rewards with a half-life of five minutes). For comparison, the longest horizon in the PPO paper was a half-life of 0.5 seconds, the longest in the Rainbow paper was a half-life of 4.4 seconds, and the Observe and Look Further paper used a half-life of 46 seconds.

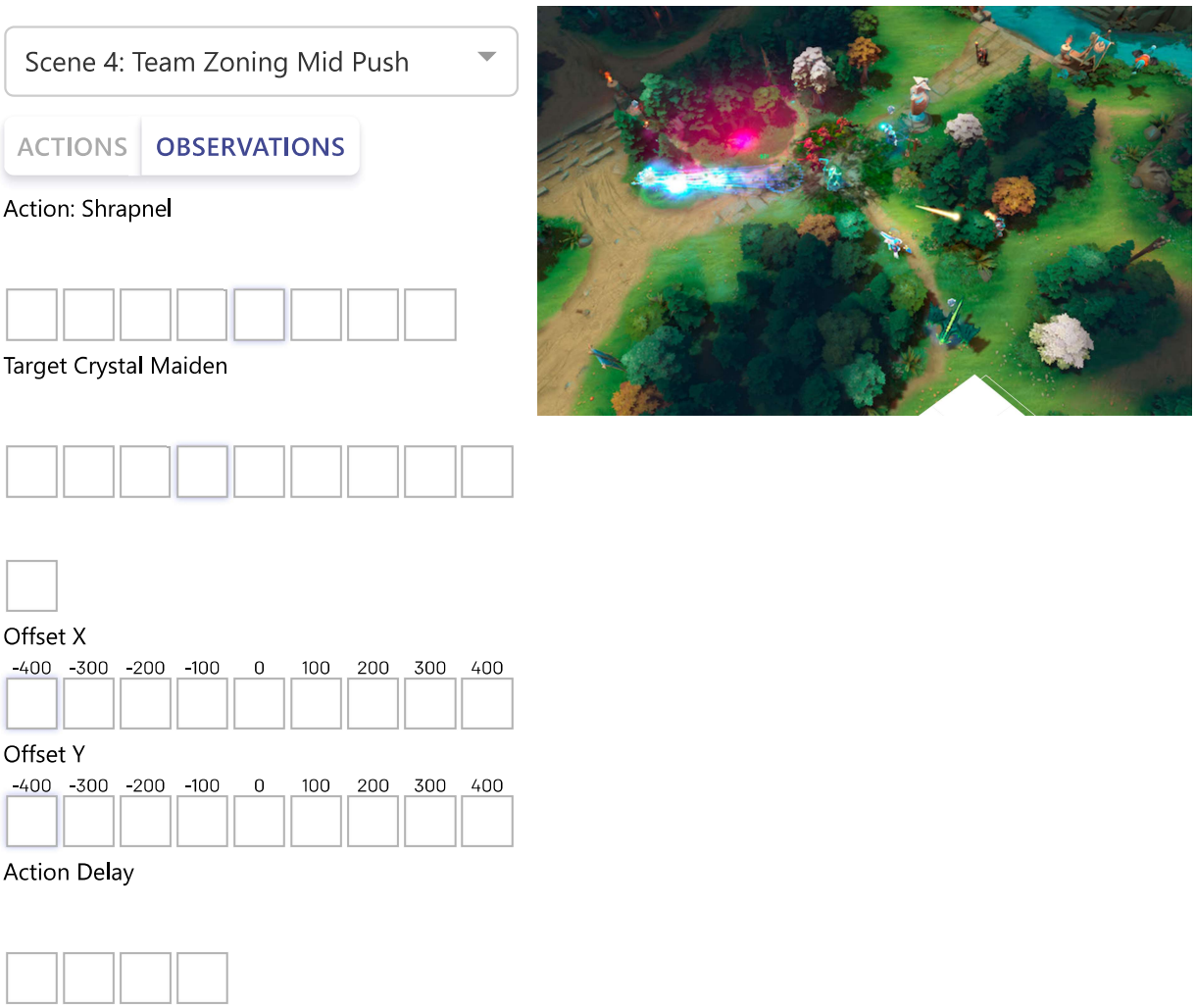
While the current version of OpenAI Five is weak at last-hitting (observing our test matches, the professional Dota commentator Blitz estimated it around median for Dota players), its objective prioritization matches a common professional strategy. Gaining long-term rewards such as strategic map control often requires sacrificing short-term rewards such as gold gained from farming, since grouping up to attack towers takes time. This observation reinforces our belief that the system is truly optimizing over a long horizon.

Model structure

Each of OpenAI Five's networks contain a single-layer, 1024-unit LSTM that sees the current game state (extracted from Valve's Bot API) and emits actions through several possible action heads. Each head has

semantic meaning, for example, the number of ticks to delay this action, which action to select, the X or Y coordinate of this action in a grid around the unit, etc. Action heads are computed independently.

Interactive demonstration of the observation space and action space used by OpenAI Five. OpenAI Five views the world as a list of 20,000 numbers, and takes an action by emitting a list of 8 enumeration values. Select different actions and targets to understand how OpenAI Five encodes each action, and how it observes the world. The image shows the scene as a human would see it.



OpenAI Five can react to missing pieces of state that correlate with what it does see. For example, until recently OpenAI Five's observations did not include shrapnel zones (areas where projectiles rain down on enemies), which humans see on screen. However, we observed OpenAI

Five learning to walk out of (though not avoid entering) active shrapnel zones, since it could see its health decreasing.

Exploration

Given a learning algorithm capable of handling long horizons, we still need to explore the environment. Even with our restrictions, there are hundreds of items, dozens of buildings, spells, and unit types, and a long tail of game mechanics to learn about — many of which yield powerful combinations. It's not easy to explore this combinatorially-vast space efficiently.

OpenAI Five learns from self-play (starting from random weights), which provides a natural curriculum for exploring the environment. To avoid "strategy collapse", the agent trains 80% of its games against itself and the other 20% against its past selves. In the first games, the heroes walk aimlessly around the map. After several hours of training, concepts such as laning, farming, or fighting over mid emerge. After several days, they consistently adopt basic human strategies: attempt to steal Bounty runes from their opponents, walk to their tier one towers to farm, and rotate heroes around the map to gain lane advantage. And with further training, they become proficient at high-level strategies like 5-hero push.

In March 2017, our first agent defeated bots but got confused against humans. To force exploration in strategy space, during training (and only during training) we randomized the properties (health, speed, start level, etc.) of the units, and it began beating humans. Later on, when a test player was consistently beating our 1v1 bot, we increased our training randomizations and the test player started to lose. (Our robotics team concurrently applied similar randomization techniques to physical robots to transfer from simulation to the real world.)

OpenAI Five uses the randomizations we wrote for our 1v1 bot. It also uses a new "lane assignment" one. At the beginning of each training game, we randomly "assign" each hero to some subset of lanes and penalize it for straying from those lanes until a randomly-chosen time in the game.

Exploration is also helped by a good reward. Our reward consists mostly of metrics humans track to decide how they're doing in the game: net worth, kills, deaths, assists, last hits, and the like. We postprocess each agent's reward by subtracting the other team's average reward to prevent the agents from finding positive-sum situations.

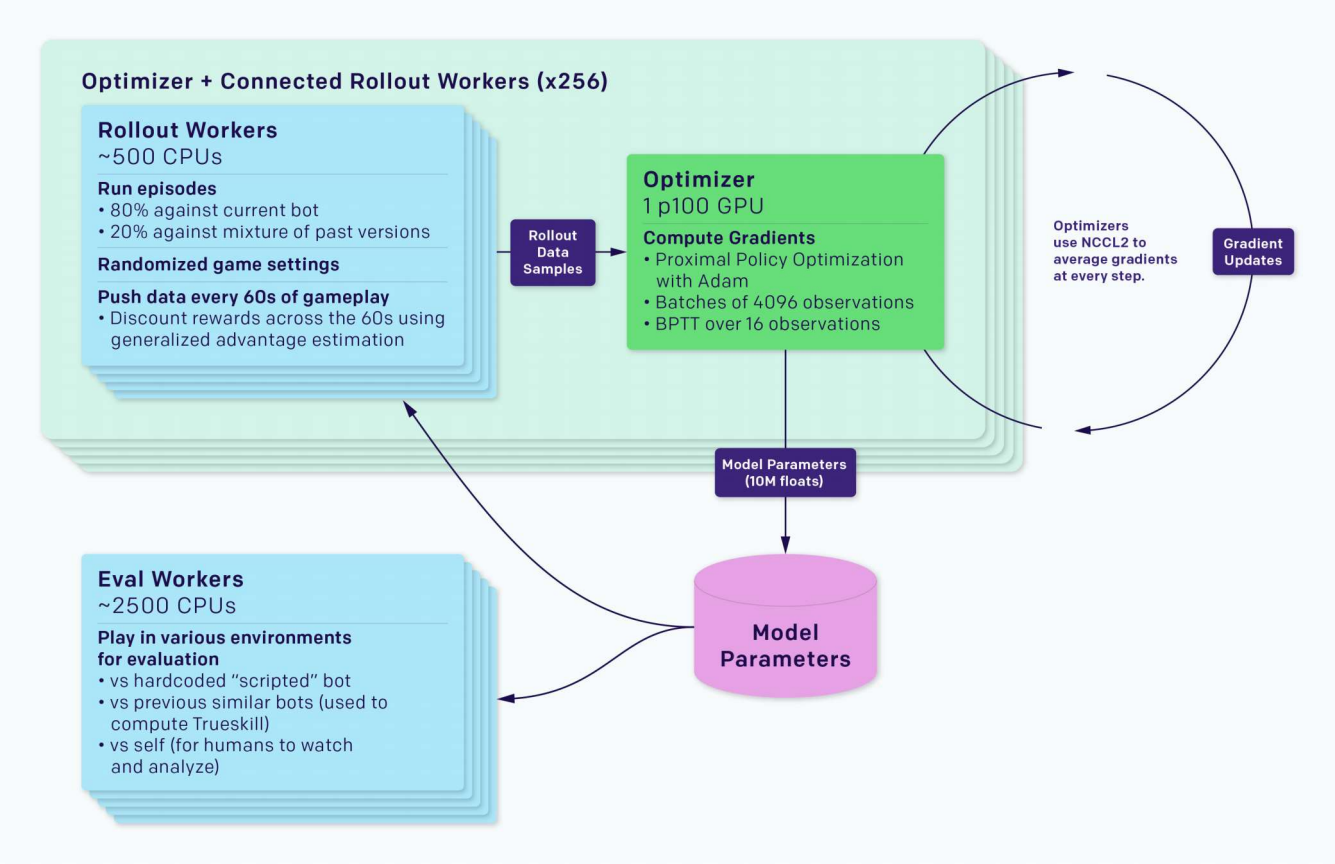
We hardcode item and skill builds (originally written for our scripted baseline), and choose which of the builds to use at random. Courier management is also imported from the scripted baseline.

Coordination

OpenAI Five does not contain an explicit communication channel between the heroes' neural networks. Teamwork is controlled by a hyperparameter we dubbed "team spirit". Team spirit ranges from 0 to 1, putting a weight on how much each of OpenAI Five's heroes should care about its individual reward function versus the average of the team's reward functions. We anneal its value from 0 to 1 over training.

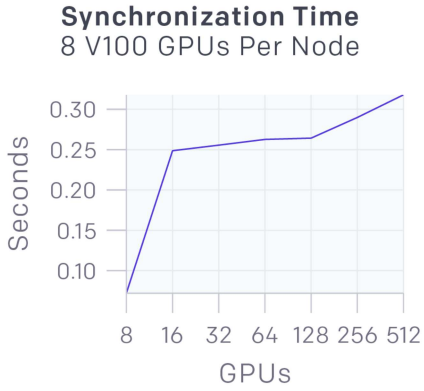
Rapid

Our system is implemented as a general-purpose RL training system called Rapid, which can be applied to any Gym environment. We've used Rapid to solve other problems at OpenAI, including Competitive Self-Play.



The training system is separated into *rollout* workers, which run a copy of the game and an agent gathering experience, and *optimizer* nodes, which perform synchronous gradient descent across a fleet of GPUs. The rollout workers sync their experience through Redis to the optimizers. Each experiment also contains workers evaluating the trained agent versus reference agents, as well as monitoring software such as TensorBoard, Sentry, and Grafana.

During synchronous gradient descent, each GPU computes a gradient on its part of the batch, and then the gradients are globally averaged. We originally used MPI's allreduce for averaging, but now use our own NCCL2 wrappers that parallelize GPU computations and network data transfer. The latencies for synchronizing 58MB of data (size of OpenAI Five's parameters) across different numbers of GPUs are shown on the right. The latency is low enough to be largely masked by GPU computation which runs in parallel with it.



We've implemented Kubernetes, Azure, and GCP backends for Rapid.

The games

Thus far OpenAI Five has played (with our restrictions) versus each of these teams:

1. Best OpenAI employee team: 2.5k MMR (46th percentile)
2. Best audience players watching OpenAI employee match (including Blitz, who commentated the first OpenAI employee match): 4-6k MMR (90th-99th percentile), though they'd never played as a team.
3. Valve employee team: 2.5-4k MMR (46th-90th percentile).
4. Amateur team: 4.2k MMR (93rd percentile), trains as a team.
5. Semi-pro team: 5.5k MMR (99th percentile), trains as a team.

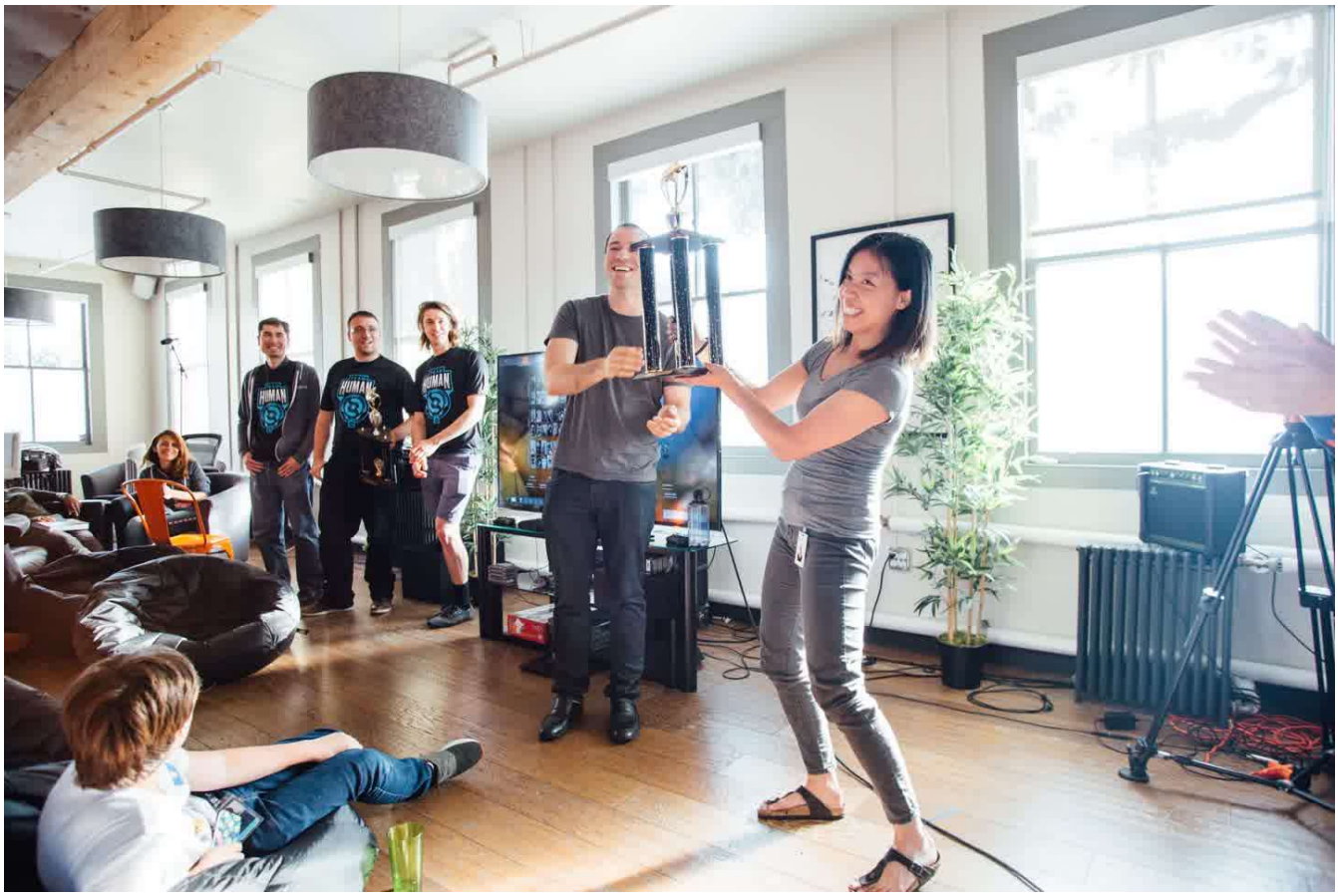
The April 23rd version of OpenAI Five was the first to beat our scripted baseline. The May 15th version of OpenAI Five was evenly matched versus team 1, winning one game and losing another. The June 6th version of OpenAI Five decisively won all its games versus teams 1-3. We set up informal scrims with teams 4 & 5, expecting to lose soundly, but OpenAI Five won two of its first three games versus both.

> *The teamwork aspect of the bot was just overwhelming. It feels like five selfless players that know a good general strategy.*
>
> *— Blitz*

We observed that OpenAI Five:

- Repeatedly sacrificed its own safe lane (top lane for dire; bottom lane for radiant) in exchange for controlling the enemy's safe lane, forcing the fight onto the side that is harder for their opponent to defend. This strategy emerged in the professional scene in the last few years, and is now considered to be the prevailing tactic. Blitz commented that he only learned this after eight years of play, when Team Liquid told him about it.
- Pushed the transitions from early- to mid-game faster than its opponents. It did this by: (1) setting up successful ganks (when players move around the map to ambush an enemy hero — see animation) when players overextended in their lane, and (2) by grouping up to take towers before the opponents could organize a counterplay.

- Deviated from current playstyle in a few areas, such as giving support heroes (which usually do not take priority for resources) lots of early experience and gold. OpenAI Five's prioritization allows for its damage to peak sooner and push its advantage harder, winning team fights and capitalizing on mistakes to ensure a fast win.



Trophies awarded after the match between the best players at OpenAI and our bot team. One trophy for the humans, one trophy for the bots (represented by Susan Zhang from our team!)

Differences versus humans

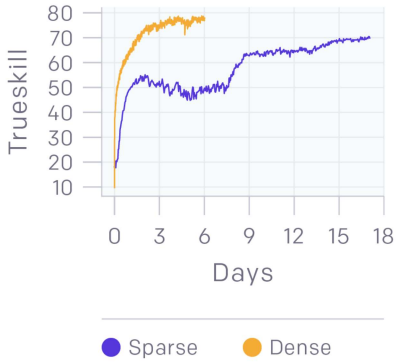
OpenAI Five is given access to the same information as humans, but instantly sees data like positions, healths, and item inventories that humans have to check manually. Our method isn't fundamentally tied to observing state, but just rendering pixels from the game would require thousands of GPUs.

OpenAI Five averages around 150-170 actions per minute (and has a theoretical maximum of 450 due to observing every 4th frame). Frame-perfect timing, while possible for skilled players, is trivial for OpenAI Five. OpenAI Five has an average reaction time of 80ms, which is faster than humans.

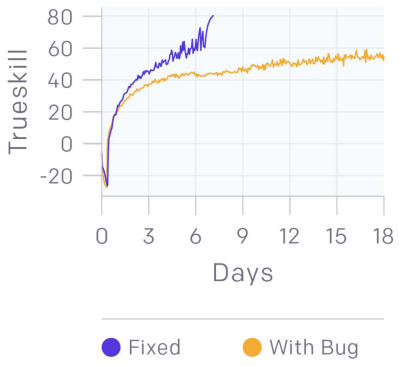
These differences matter most in 1v1 (where our bot had a reaction time of 67ms), but the playing field is relatively equitable as we've seen humans learn from and adapt to the bot. Dozens of professionals used our 1v1 bot for training in the months after last year's TI. According to Blitz, the 1v1 bot has changed the way people think about 1v1s (the bot adopted a fast-paced playstyle, and everyone has now adapted to keep up).

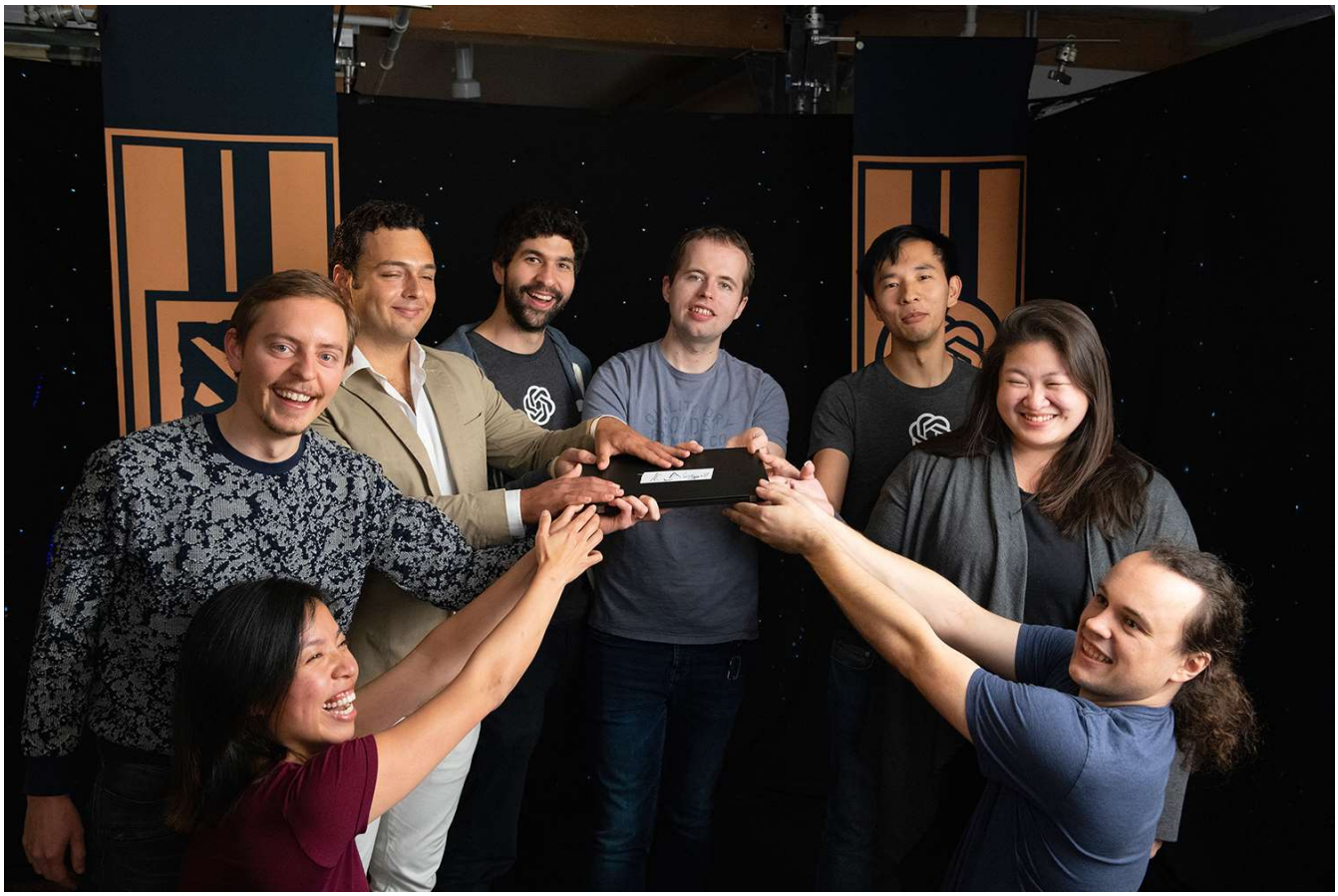
Surprising findings

- **Binary rewards can give good performance.** Our 1v1 model had a shaped reward, including rewards for last hits, kills, and the like. We ran an experiment where we only rewarded the agent for winning or losing, and it trained an order of magnitude slower and somewhat plateaued in the middle, in contrast to the smooth learning curves we usually see. The experiment ran on 4,500 cores and 16 k80 GPUs, training to the level of semi-pros (70 TrueSkill) rather than 90 TrueSkill of our best 1v1 bot).



- **Creep blocking can be learned from scratch.** For 1v1, we learned creep blocking using traditional RL with a "creep block" reward. One of our team members left a 2v2 model training when he went on vacation (proposing to his now wife!), intending to see how much longer training would boost performance. To his surprise, the model had learned to creep block without any special guidance or reward.
- **We're still fixing bugs.** The chart shows a training run of the code that defeated amateur players, compared to a version where we simply fixed a number of bugs, such as rare crashes during training, or a bug which resulted in a large negative reward for reaching level 25. It turns out it's possible to beat good humans while still hiding serious bugs!





A subset of the OpenAI Dota team, holding the laptop that defeated the world's top professionals at Dota 1v1 at The International last year.

What's next

Our team is focused on making our August goal. We don't know if it will be achievable, but we believe that with hard work (and some luck) we have a real shot.

This post described a snapshot of our system as of June 6th. We'll release updates along the way to surpassing human performance and write a report on our final system once we complete the project. Please join us on August 5th virtually or in person, when we'll play a team of top players!

Our underlying motivation reaches beyond Dota. Real-world AI deployments will need to deal with the challenges raised by Dota which are not reflected in Chess, Go, Atari games, or Mujoco benchmark tasks.

Ultimately, we will measure the success of our Dota system in its application to real-world tasks. If you'd like to be part of what comes next, we're hiring!

---

[1] Current set of restrictions:

- *Mirror match of Necrophos, Sniper, Viper, Crystal Maiden, and Lich*
- *No warding*
- *No Roshan*
- *No invisibility (consumables and relevant items)*
- *No summons/illusions*
- *No Divine Rapier, Bottle, Quelling Blade, Boots of Travel, Tome of Knowledge, Infused Raindrop*
- *5 invulnerable couriers, no exploiting them by scouting or tanking*
- *No Scan*

The hero set restriction makes the game very different from how Dota is played at world-elite level (i.e. Captains Mode drafting from all 100+ heroes). However, the difference from regular "public" games (All Pick / Random Draft) is smaller.

Most of the restrictions come from remaining aspects of the game we haven't integrated yet. Some restrictions, in particular wards and Roshan, are central components of professional-level play. We're working to add these as soon as possible.

[2] Thanks to the following for feedback on drafts of this post: Alexander Lavin, Andrew Gibiansky, Anna Goldie, Azalia Mirhoseini, Catherine Olsson, David Dohan, David Ha, Denny Britz, Erich Elsen, James Bradbury, John Miller, Luke Metz, Maddie Hall, Miles Brundage, Nelson Elhage, Ofir Nachum, Pieter Abbeel, Rumen Hristov, Shubho Sengupta, Solomon Boulos, Stephen Merity, Tom Brown, Zak Stone

[3] You can cite this blog post with the following Bibtex:

```
@misc{ OpenAI_dota, author = {OpenAI}, title = {OpenAI Five}, howpublished = {\url{https://blog.openai.com/openai-five/}} year = {2018}}
```

Built by a team of researchers and engineers at OpenAI (order randomized each pageload).

TWEET SHARE

GREG BROCKMAN BROOKE CHAN RAFAŁ JÓZEFOWICZ
MICHAEL PETROV HENRIQUE PONDÉ
JONATHAN RAIMAN PRZEMYSŁAW DĘBIAK
SZYMON SIDOR DAVID FARHI SUSAN ZHANG
JIE TANG JAKUB PACHOCKI FILIP WOLSKI
CHRISTY DENNISON

JACK CLARK LARISSA SCHIAVO ALEC RADFORD
JONAS SCHNEIDER CHRISTOPHER BERNER
SHARIQ HASHME DIANE YOON ERIC SIGLER
ILYA SUTSKEVER SCOTT GRAY CHRISTOPHER HESSE
JOHN SCHULMAN DAVID LUAN PAUL CHRISTIANO
QUIRIN FISCHER

Keep reading

MORE

How AI Training Scales

NO. 16
DEC 14
2018

OpenAI Fellows Summer Class of '18: Final Projects

UPDATES
DEC 19, 2018

Quantifying Generalization in Reinforcement Learning

RESEARCH
DEC 6, 2018