OpenAI Five Finals
and Proximal Policy Optimization (PPO)

TensorFlow and Deep Learning Singapore Meetup
30 May 2019

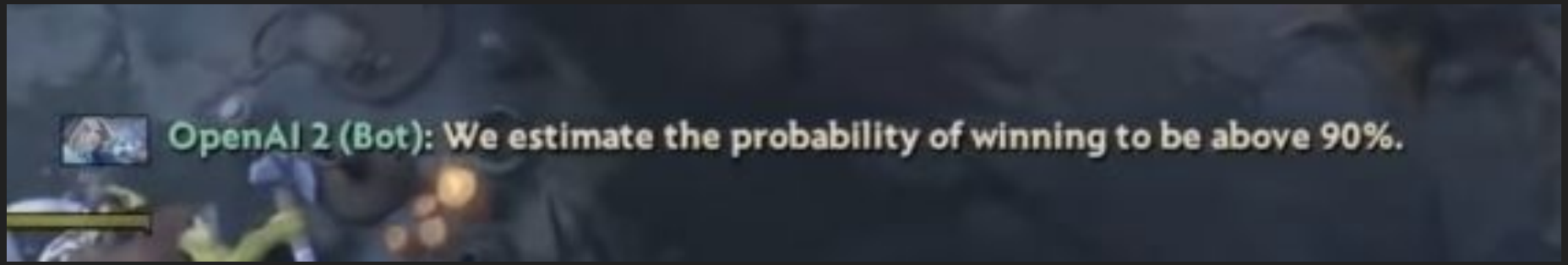
Agenda (many slides, 15 minutes, will skip through details)

- Dota 2 - the game
- Deep reinforcement learning (RL) - high level overview
- Proximal policy optimization (PPO) - high level overview
- Why Dota 2 - OpenAI research
- OpenAI Finals - the event (13 April 2019)
- OpenAI Five
  - The agent
  - Compute
  - Architecture
  - Reward (structure)
- What's next for Dota 2 research
- How to start learning deep RL

Dota 2 - the game

- Sequel to Defense of the Ancients (Dota), officially released in 2013
- Developed by Valve (Half-Life, Team Fortress, CS, L4D, Portal)
- Multiplayer online battle area
- **Two teams** - the Radiant and Dire
- **Five heroes/team** w/ unique abilities per team, collecting experience points, items
- **Winning: destroying a large fortified thing** in the opposing team's territory (the Ancient)
- E-sports: the largest tournament - The International


K/D/A 2 / 1 / 1
LH/DN 26 / 6
9
8:30
3
OG vs OpenAI Five Game 2
(F) LAST HITS/DENIES
40 / 17
26 / 6
25 / 2
19 / 3
18 / 7
18 / 5
17 / 10
11 / 10
6 / 3
2 / 1
SproiNK
OpenAI 4 (Bot)
major first timer
OpenAI 2 (Bot): We estimate the probability of winning to be above 90%.
SVEN
285 / 960
263 / 363
STASH
OpenAI
79


OpenAI 2 (Bot): We estimate the probability of winning to be above 90%.

Deep reinforcement learning - high level overview

Do not code in how to play, coded in how to learn via +/- reward over time

RL - **learning through interaction with the env, finding the optical policy/value fn is key**; no labeled data & no human-made datasets, often need a reward signal (“dopamine”) for achieving a goal related to a desired state (e.g. achieving an objective, winning
RL - **general framework for studying sequential decision-making** (Vlad Mnih) - (e.g. Transformers in AlphaStar (StarCraft 2) by DeepMind)
RL - sequential, evaluative (feedback), *exhaustive* (Miguel Morales)

DL - current best way of making machines perceive the world (Vlad Mnih)

DRL = “(basic) neural net from DL as a function approximator + RL”

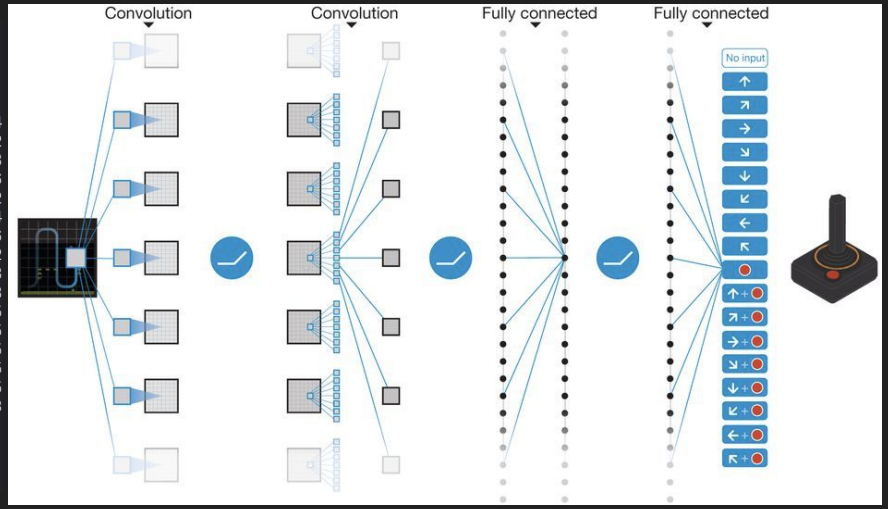
Deep reinforcement learning - high level overview

“observe -> learn (update) -> take action - > evaluate -> …-> find optimal π”

Human View

AI View

3.006	-1.386	-0.4695	0.883	1	0.84
-0.3154	-0.5425	-0.5	0.866	0	0.82
3.11	-1.36	-0.9336	0.3584	1	0.78
-2.324	2.863	0.9746	0.225	0	0.86
3.037	-1.361	-0.7773	0.6294	1	0.82
-1.387	2.951	0.988	0.1565	0	0.74
3.023	-0.9395	0.05234	-0.9985	0	0.66
2.951	-0.5747	0.01746	1	0	0.72
2.963	-1.303	0.3906	0.9204	0	0.68
2.834	-3.164	0.01746	-1	0	0.68
3.127	-1.368	0.6562	0.755	1	0.55
3.088	-1.366	0.4695	0.883	0	0.55
2.984	-1.398	-0.225	0.9746	1	0.55
3.037	-1.391	0.788	0.6157	0	0.55
3.076	-1.438	0.883	0.4695	0	0.55
-2.412	2.846	0.996	0.08716	1	0.3



(Source: OpenAI blog; Nature 2015 - Human Level Control through Deep RL (DeepMind))

REINFORCE - a vanilla PG method (sorry for the greeks)

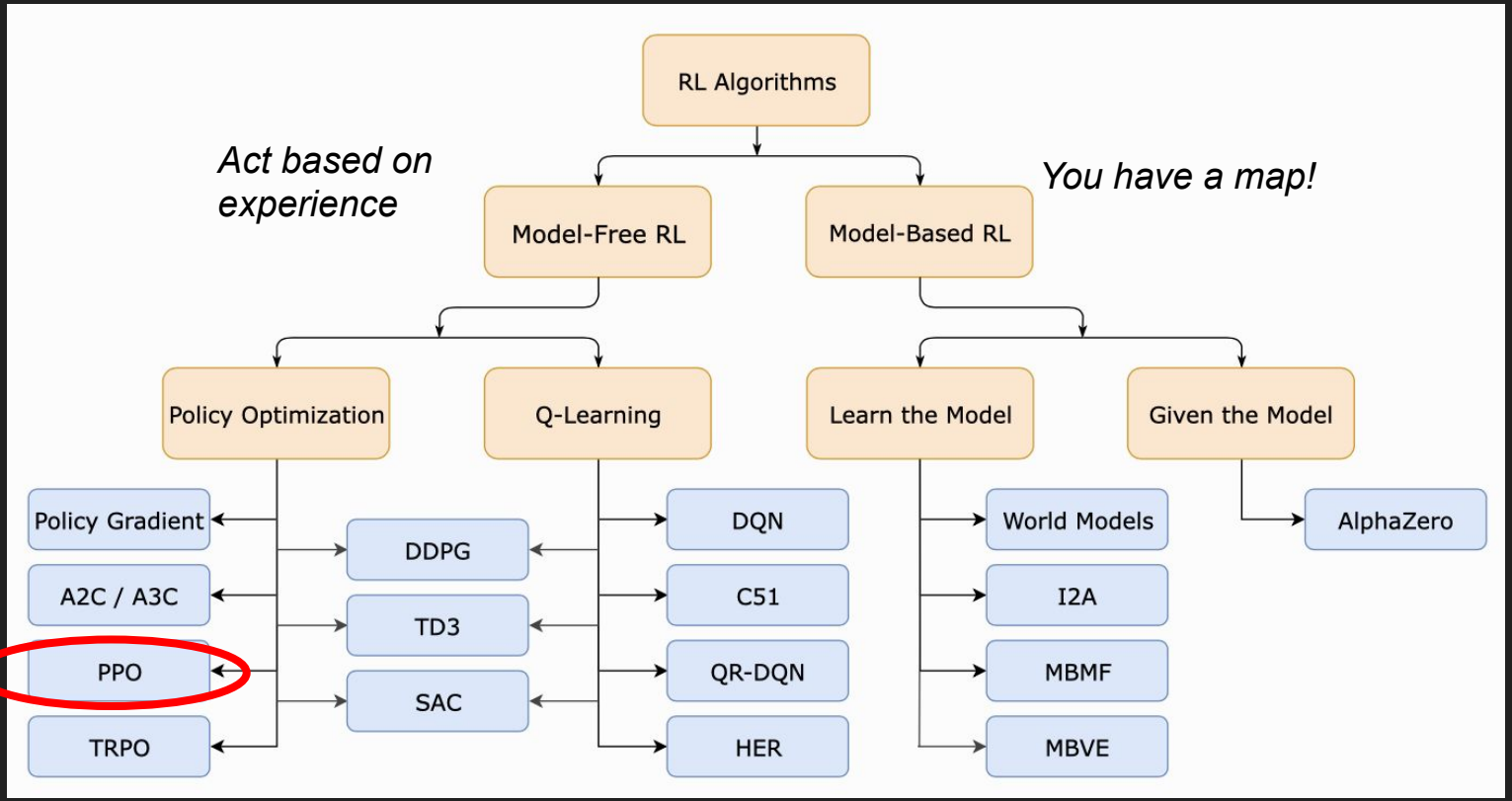
- Initialize a random **policy π_θ**
- Collect **trajectories** τ - **state-action-reward (s, a, R) sequences** with length **H** (**horizon**)
- Compute **sum of rewards from τ = R(τ)** = $r_1 + r_2 + \dots + r_H + r_{H+1}$
- Use the m trajectories τ to **estimate the gradient ∇ U(θ) = g**
  = $(1/m) \times \sum\sum \nabla \log \pi(a_t \mid s_t) R(\tau)$
- **Update the policy gradient *g* and weights θ** with gradient ascent step with LR α:
  $\theta \leftarrow \theta + \alpha \nabla U(\theta)$
- Loop

Goal: find the **weights θ of the neural network** that **maximizes expected return U(θ)**

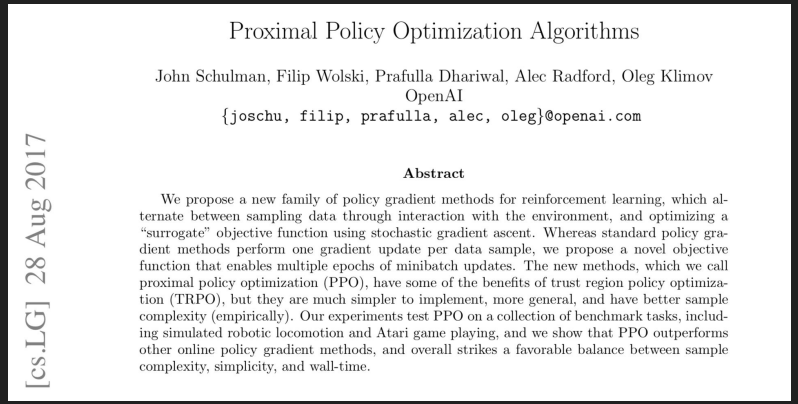
Set the weights θ such that on average the agent experiences trajectories τ with high return - max $U(\theta)$ s.t. $U(\theta) = \sum P(\tau;\theta) R(\tau)$

Expected return = Sum (probability of a trajectory (subject to weights) x Return from a trajectory)

Reinforcement learning (taxonomy by OpenAI)



PPO - high level overview



Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

Abstract

We propose a new family of policy gradient methods for reinforcement learning, which alternate between sampling data through interaction with the environment, and optimizing a "surrogate" objective function using stochastic gradient ascent. Whereas standard policy gradient methods perform one gradient update per data sample, we propose a novel objective function that enables multiple epochs of minibatch updates. The new methods, which we call proximal policy optimization (PPO), have some of the benefits of trust region policy optimization (TRPO), but they are much simpler to implement, more general, and have better sample complexity (empirically). Our experiments test PPO on a collection of benchmark tasks, including simulated robotic locomotion and Atari game playing, and we show that PPO outperforms other online policy gradient methods, and overall strikes a favorable balance between sample complexity, simplicity, and wall-time.

- A policy gradient family of methods
- A simpler version of Trust Region Policy Optimization (TRPO)
- Discrete and continuous action spaces
- Samples data through env interaction, optimizing a "surrogate" objective function using stochastic gradient ascent

- Smoother discount rewards, reducing var to make training smoother/more stable thanks to **generalized advantage estimate (GAE)**
- Gradual policy π update: thanks to **surrogate loss function *L*** - A ratio between new and old probabilities X **advantage *A***
- **Mini-batch update - size *M***: as experiences are sampled - these trajectories τ are broken into random mini batches & the network is gradually updated

PPO - pseudocode (sorry for the greeks)

- Initialize a random **policy π_θ**
- 1. Collect **trajectorie**s τ - state-action-reward (s, a, R) sequences with length H (horizon) and initialize weights $\theta' = \theta$
- **2. Compute the gradient of the clipped surrogate function L using trajectories τ: ∇ Lclip_sur(θ', θ)**
- 3. Update weights θ' using the gradient ascent step with LR α: $\theta' \leftarrow \theta + \alpha \nabla$ Lclip_sur(θ', θ)
- 4. Repeat 2 and 3 without generating new trajectories τ a few times
- 5. Set $\theta = \theta'$ and repeat from 1
- 6. Compute sum of rewards from τ of the trajectory $R(\tau) = r_1 + r_2 + r_3 + \ldots$
- 7. Use the m trajectories to estimate gradient $\nabla U(\theta) = g$ $\nabla U(\theta) = g\_hat = (1/m) \times \sum\sum \nabla \log \pi(a_t \mid s_t) R(\tau)$
- 8. Update the policy gradient *g* and update weights θ' with LR α: $\theta' \leftarrow \theta + \alpha g$
- 9. Loop

PPO - pseudocode

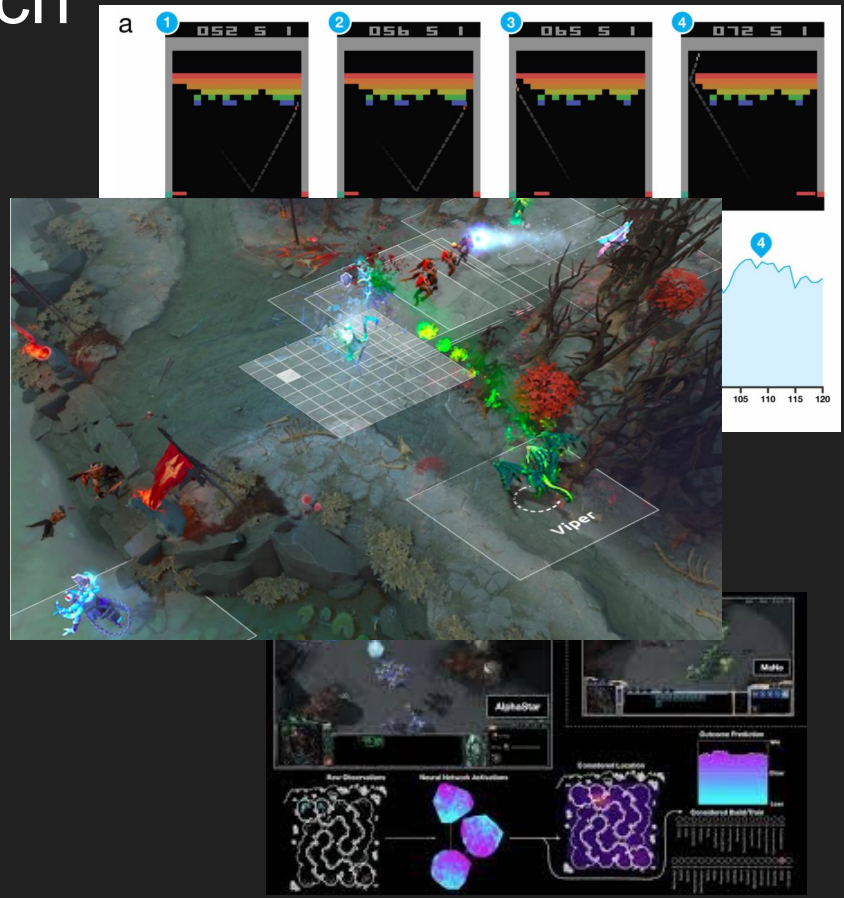
Algorithm 1 PPO, Actor-Critic Style

for iteration=$1, 2, \ldots$ **do**
  for actor=$1, 2, \ldots, N$ **do**
    Run policy $\pi_{\theta_{\text{old}}}$ in environment for T timesteps
    Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
  end for
  Optimize surrogate L wrt θ, with K epochs and minibatch size $M \leq NT$
  $\theta_{\text{old}} \leftarrow \theta$
end for

Why Dota 2 - OpenAI research

Backgammon/TD-Gammon (1992)...
Chess/Deep Blue (1996)...
Atari/DQN (2013-2015)...
Go/original AlphaGo (2015)...
[Chess, Shogi, Go]/AlphaZero (2017-2018)...

Dota 2 OpenAI 1v1 (2017)
Dota 2/OpenAI Five (2018-2019)
StarCraft 2/AlphaStar (2019-)



Why Dota 2 - OpenAI research

- Difficult environment for RL development
- Popular game where there is no strong correlation between action-per-minute and skill (reflex/micro is a secondary skill)
- Complexity:
  - **Long time horizons, hidden information**
  - **High-dim, continuous action space** (~170,000 actions per hero, ~1000 valid actions per tick vs ~250 average number of actions in Go)
  - **High-dim, continuous obs space:** a large map, 5v5 heroes, dozens of buildings, dozens of non-player char (NPC) units + trees, wards, runes
- Linux and Valve Bot API

  => good for model-free PPO

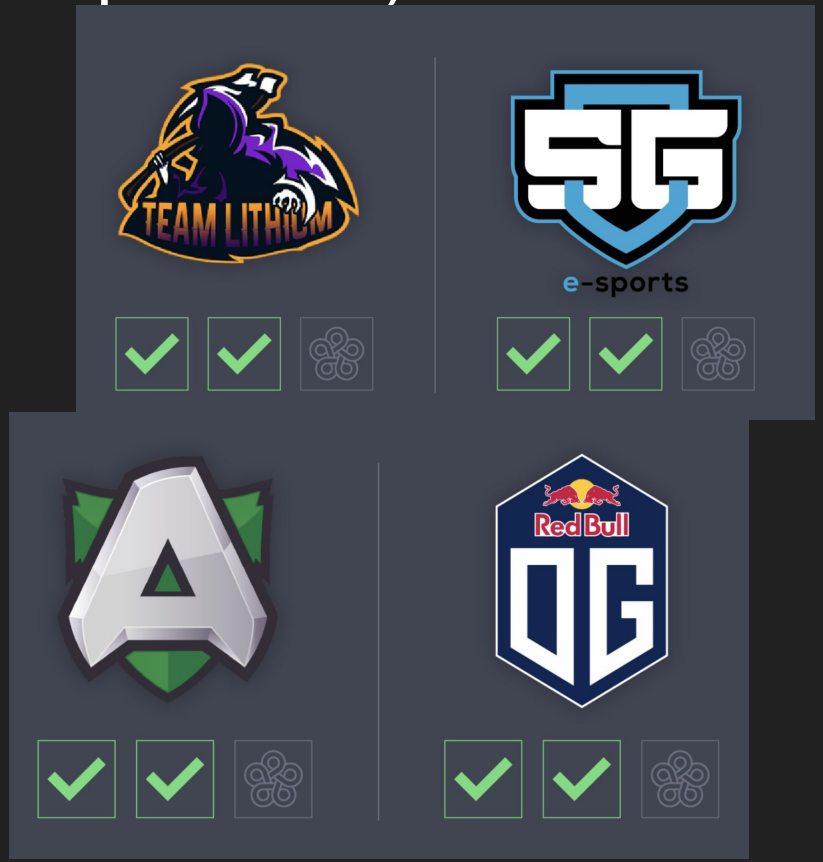
OpenAI Finals - the event (13 April 2019)

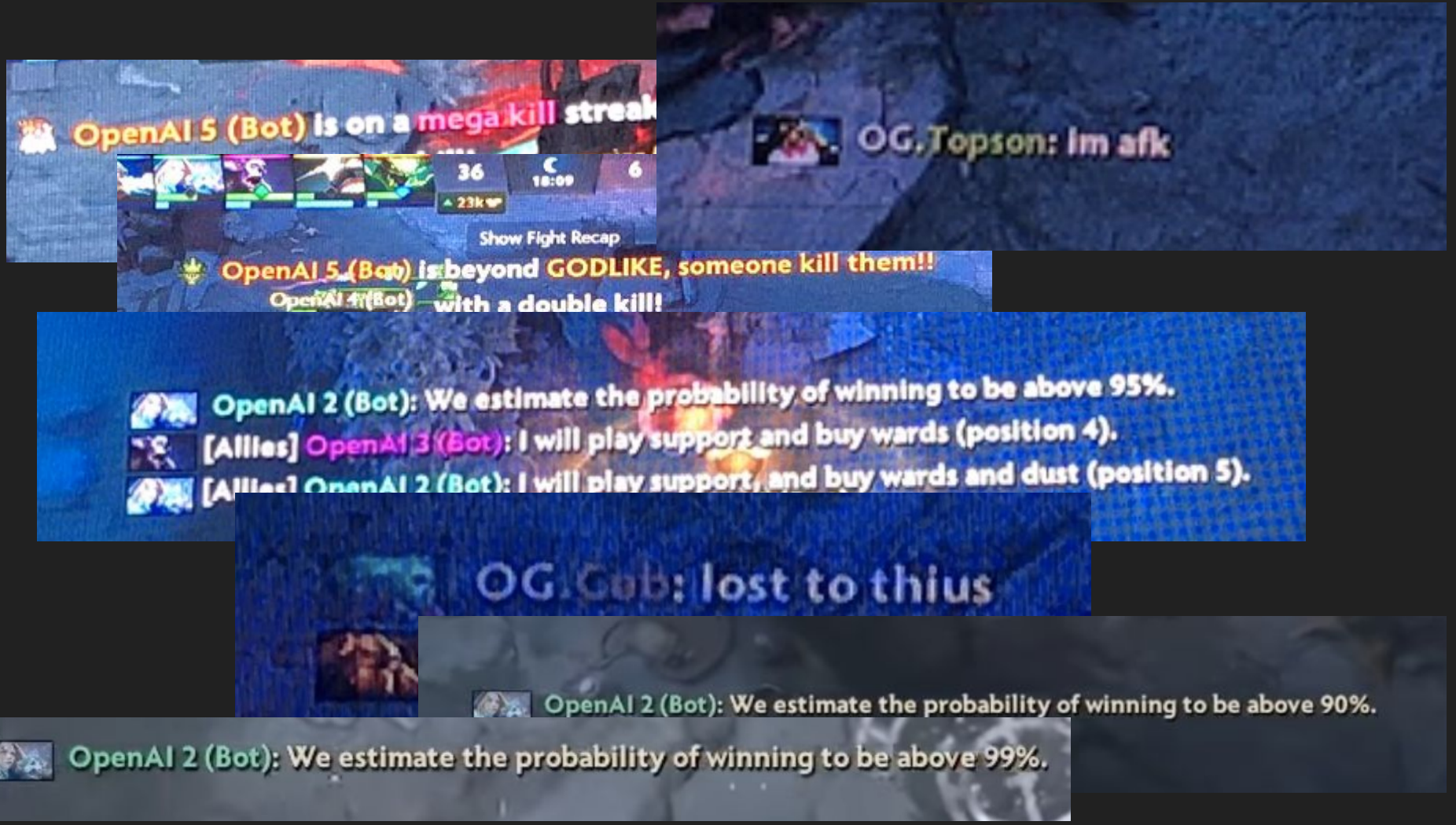
Historic moment

Five vs team OG (2018 champions) live in front of an audience

AlphaStar hasn't beaten world champions live yet

Notable discoveries during another match: becoming cooperative despite the lack of training with humans; using buyback often (quick respawning); being confident of winning




OpenAI 5 (Bot) is on a mega kill streak
36
18:09
6
23k
Show Fight Recap
OpenAI 5 (Bot) is beyond GODLIKE, someone kill them!!
OpenAI 4 (Bot) with a double kill!
OG.Topson: im afk
OpenAI 2 (Bot): We estimate the probability of winning to be above 95%.
[Allies] OpenAI 3 (Bot): I will play support and buy wards (position 4).
[Allies] OpenAI 2 (Bot): I will play support, and buy wards and dust (position 5).
OG.Cub: lost to thius
OpenAI 2 (Bot): We estimate the probability of winning to be above 90%.
OpenAI 2 (Bot): We estimate the probability of winning to be above 99%.


OPENAI FIVE VICTORY
WINS SERIES
2–0
Best of 3

OpenAI Five - agent

"If we train OpenAI Five from scratch,
it takes it 24h before I cannot beat it anymore. "
- Jonathan Raiman

“Five simply has a different way of approaching how it would achieve its goal of winning, which may or may not map to how humans think about "strategy".”
- Susan Zhang

(dev team, reddit AMA)

OpenAI Five - agent

No sophisticated new algorithmic ideas:
"Fundamental improvement we needed for this problem was scale"

A version of PPO at massive scale with a OpenAI's Rapid system

One AI controls the 5 heroes - equivalent to 5 clones of a brain - each knows exactly what the other ones are going to do

“Ever since their birth everything they have seen is together.”
- David Farhi (dev team)

5 networks (5 heroes to control) - each with an LSTM that sees the game via the API and emits the actions through separate action heads.

OpenAI Five - agent

Lots of compute and self-play

No search or bootstrap from human replays

"We have increasing confidence that self-play will be a core part of powerful AI systems in the future." - Competitive Self-Play - Oct 11, 2017 OpenAI Blog

(Each action head is computed independently - has semantic meaning: e.g. no. of ticks to delay an action, which action to select, the X or Y coordinate of this action in a grid around the unit)

OpenAI Five - agent

- **OpenAI Five at Finals was 10 months old (continuous training)** - at The International (Aug 2018) it was 1.5 months old
- **Not training from pixels - seeing the Dota world as numbers (about 20,000 numbers - mostly floats per state)** using Valve's Bot API
  - Learning how it was learning new strategies without sacrificing the compute for pixels
  - Numbers represent what the human can access (vs 400 enumeration values in Go board)

Future rewards valued with a half-life of 46 in June 2018 vs the longest horizon of the PPO paper (July 2017) of 0.5 seconds
Milliseconds s per tick to execute vs nanoseconds in Go engine
Finals game restrictions: 17 heroes, no illusions and summons)
Fixed (scripted) item & skill builds (scripted consumable logic) - experimented with RL

OpenAI Five - the agent

Human View

AI View

```
  3.006   -1.386  -0.4695   0.883  1 0.84
-0.3154  -0.5425    -0.5   0.866  0 0.82
   3.11    -1.36  -0.9336  0.3584  1 0.78
 -2.324    2.863   0.9746   0.225  0 0.86
  3.037   -1.361  -0.7773  0.6294  1 0.82
 -1.387    2.951    0.988  0.1565  0 0.74
  3.023  -0.9395  0.05234 -0.9985  0 0.66
  2.951  -0.5747  0.01746       1  0 0.72
  2.963   -1.303   0.3906  0.9204  0 0.68
  2.834   -3.164  0.01746      -1  0 0.68
  3.127   -1.368   0.6562   0.755  1 0.55
  3.088   -1.366   0.4695   0.883  0 0.55
  2.984   -1.398   -0.225  0.9746  1 0.55
  3.037   -1.391    0.788  0.6157  0 0.55
  3.076   -1.438    0.883  0.4695  0 0.55
 -2.412    2.846    0.996 0.08716  1  0.3
```

OpenAI Five - agent

Self-play: 80% of games vs itself, 20% vs past selves

- attempted 99% vs itself and 50%, but 80-20 prevents strategy collapse + enables getting double the amount of data from the 80% that play itself (source: JR @ OpenaAI)

Pre-Finals: Trained on 18 heroes (17+1 Lich)... trained on more heroes but surgerying failed

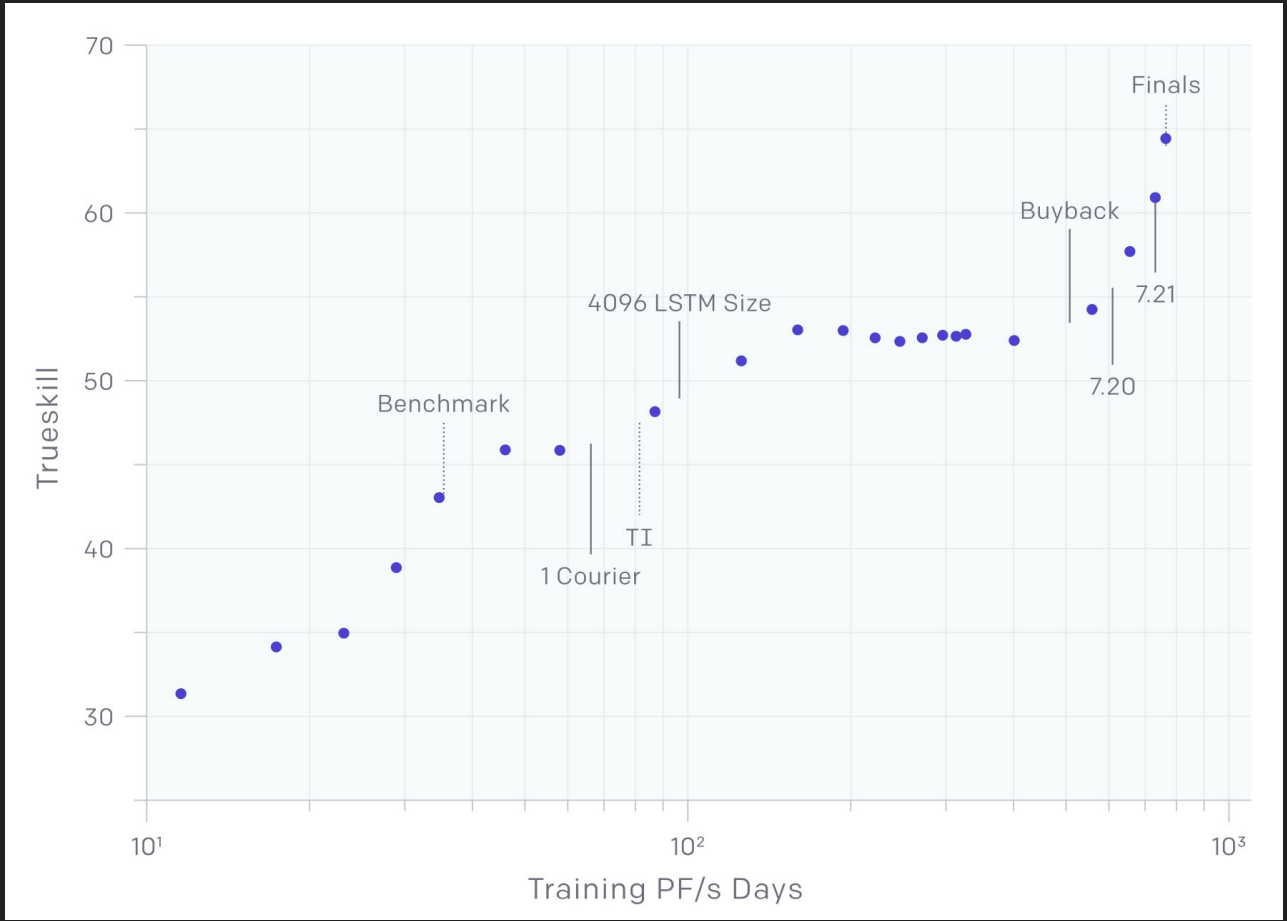
LSTM size - 1024 (1v1) to 2048 (5v5) at TI (Aug 2018) to 4096 units

Experience: total 45,000 years of self-play over 10 realtime months (vs 10,000 years and 1.5 RT months as of TI (Aug 2018)) ... for ave. 250 years of simulated experience per day

... **transfer learning in RL** (continuing training despite the model size change (double the model size since TI, initialize from old params,....)

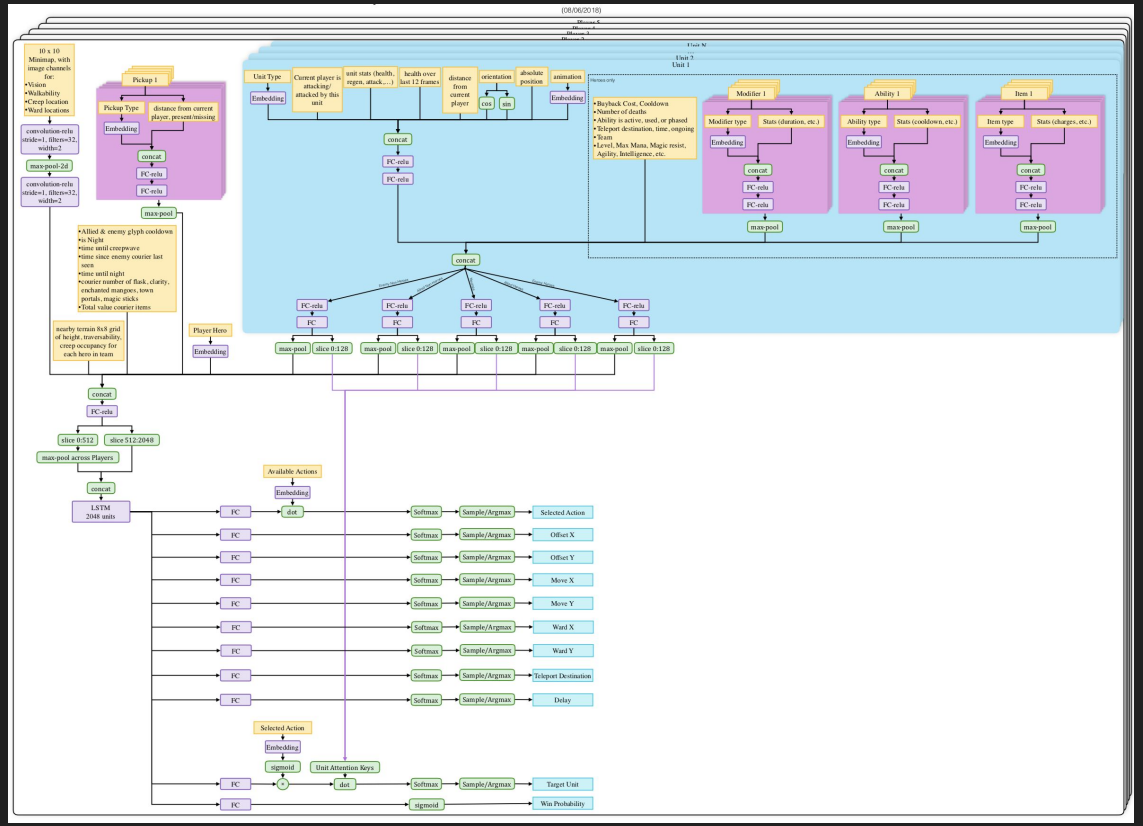
OpenAI Five - compute

“The graph is roughly linear,
meaning that **OpenAI Five benefited continually from additional compute**



Trueskill = a skill-based ranking system for games w/ 2+ players (source: OpenAI blog)

OpenAI Five - architecture



Input:

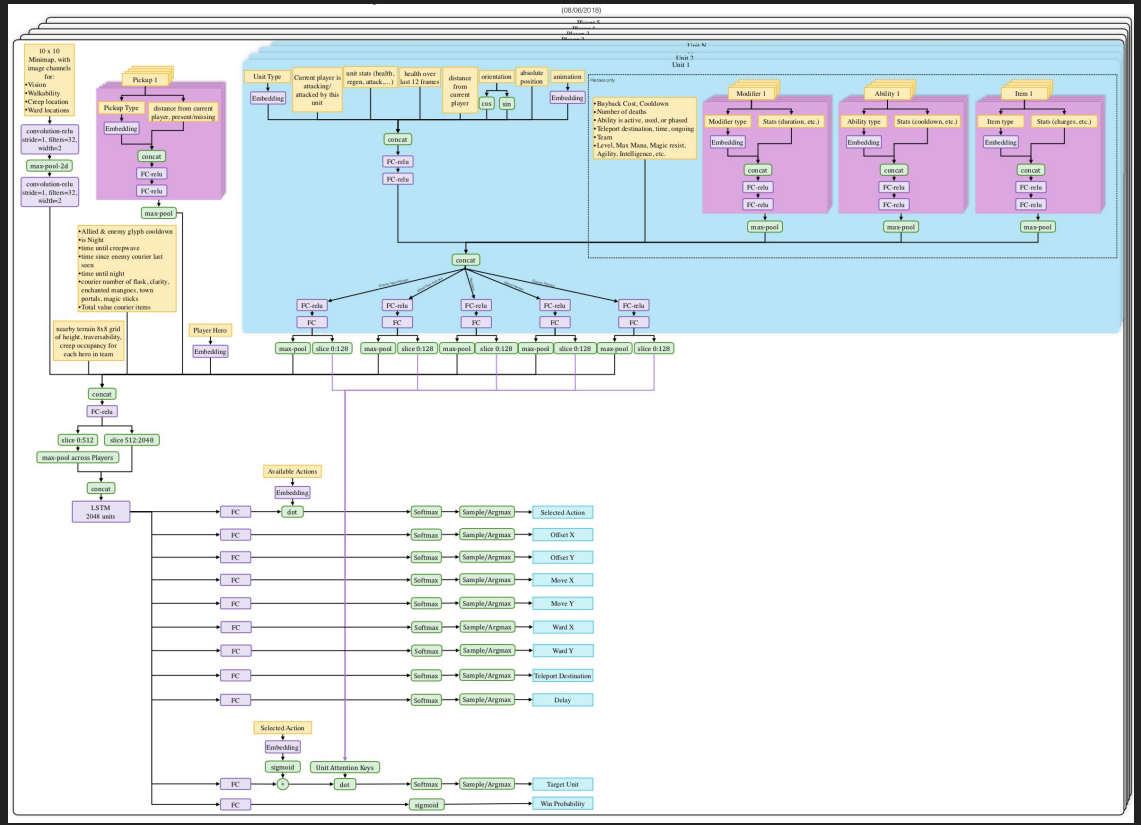
- **Obs:** orientation, health, abs. position, current player attacking/being attacked
- **Heroes' obs:** ability, modifier, item, ...

through FC, ReLU, etc
via **large LSTM**, FC, softmax, etc

Output:

- **Actions:** move, ward, offset, delay, selected action,....
  … **+ win probability**
-

OpenAI Five - architecture



Model size: 167 million parameters

Deployable model size: ~668Mb

To run a game vs Five: 32 CPU cores (Intel Skylake) or 16 physical cores

OpenAI Five - reward (David Farhi - dev team)

Overall goal - winning the game
The agent's **reward** = increase in score; An individual **signal** produces a score.

- Each **hero's score** = linear combination of separate **signals** (before averaging & reweighting)
  - +ve hero score weight per signal: health, experience
    -ve weight: taking out a hero, being defeated
- **Building scores** (alive + health):
  - +ve building score weights per alive building
- **Extra scores:** large weights for defeating last enemy barracks, winning the game
- **Exploration score** - [three] "lane assignment":
  - -ve reward if a hero strays from the randomly assigned lanes early

OpenAI Five - reward (David Farhi - dev team)

Rewards also processed for **competitiveness and cooperation** (***team work***):

- **Zero sum [competitiveness]** - a team's mean reward subtracted from the enemy's rewards
  - **Preventing +ve sum situations**
- **Team spirit [team work]** - taking into account the **teammates' situations** (no greedily optimizing for own reward)
  - Average the across the team's heroes this "team spirit" hyperparameter
- **Time scaling [competitiveness]** - placing **importance on the early part of the game** - scaling up rewards early/scale down rewards
  - The late game portion is longer and the units should be more developed then
  - Early hero/game development is important, can affect negatively in future

What’s next for Dota 2 research

OpenAI Five **was retired** in April after Finals
Work on Dota 2 continues - no plans on branching out yet

Technical paper to be released soon… btw OpenAI is hiring deep learning - reasoning researchers

Deep RL - compute power - decreasing the amount of experience needed in RL

“...The main goal of this project is to research RL, and **we've mainly been focused on getting Five to be the best it can**. We can now take a step back and **figure out why Five works the way it does**, and hopefully **help to make RL more efficient and train better**.”
- Christy Dennison *(dev team, reddit AMA)*

What’s next for Dota 2 research

"In our push to develop Five, there were a **lot of decisions that were made where it wasn't 100% clear** whether or not they benefitted Five's learning curve. We hope to examine each of these in detail and release as much of our findings as we can."
- Susan Zhang

"The thing that surprised me the most was that a lot of problems that we believed will be extremely hard for AI to learn turned out to be not-that-hard in the end.”
- Przemyslaw Debiak

(dev team, reddit AMA)

How to start learning deep RL

“Once you play around with the hyper-parameters, you'll understand why there are complaints on deep reinforcement learning papers reproducibility. ***RL is hard.****”*
- Miguel Morales (author of Grokking Deep RL)

“[Supervised learning] wants to work. Even if you screw something up you’ll usually get something non-random back. ***RL must be forced to work****...****Long story short your failure is more due to the difficulty of deep RL****, and* ***much less due to the difficulty of “designing neural networks”****.”*
- Andrej Karpathy (when at OpenAI):

To get started with DRL - as recommended by Pieter Abbeel:

- OpenAI: Spinning Up in Deep RL (website)
- Deep RL Bootcamp at Berkeley from 2017 (google it)

"Many great developments started as crazy expensive research, and became within everyone's reach once people knew what was possible and started optimizing them. ***The first deep net to ever go into production at Google (for speech recognition) took months to train, and was 100x too slow to run.*** *Then we found tricks to speed it up, improved (and open-sourced) our deep learning infrastructure, and now everybody in the field uses them. SmartReply was crazy expensive, until it wasn't. The list goes on."*

- Vincent Vanhoucke (Google Brain, principal scientist/robotics director, reddit AMA 2018)

Thank you!

Q&A