Dota 2 with Open AI

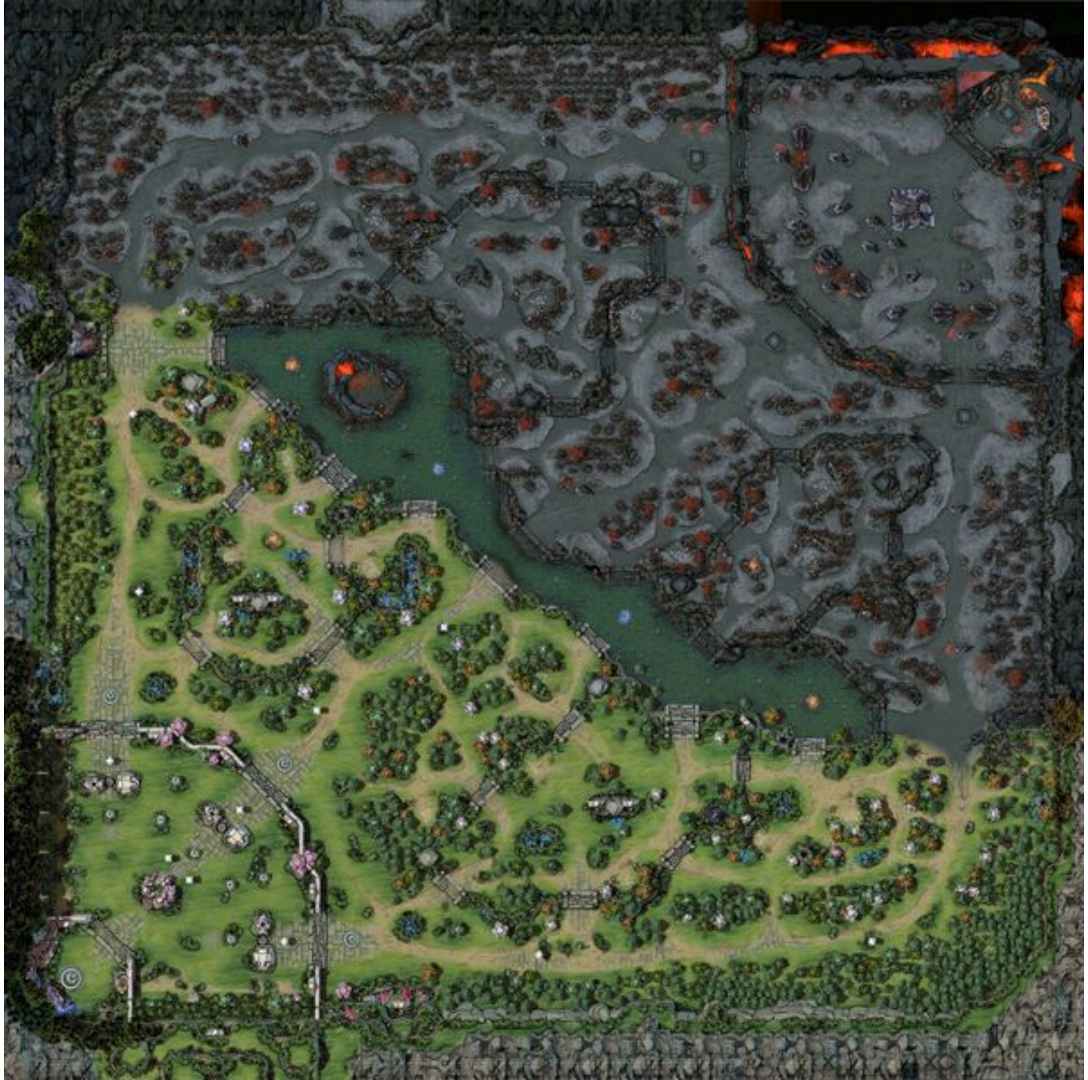
Author: Andon Mitsov

Open AI

- Who are they?
- A R&D company focused on the development of Artificial General Intelligence
- A capped profit company
- Several big investors(Microsoft, Elon Musk and others)
- Their goal is to research ways of how to best utilize AI for the good of humanity

What’s Dota 2

- Multiplayer Online Battle Arena
- A player is part of 5-man team
- Each player chooses a hero from a pool of 117 heroes
- The main objective is to destroy the central structure of the enemy team
- It’s based on a mod of an older game (Warcraft 3)
- Developed by Valve



How do you make a godlike team in Dota 2

- Do you:

a) Train very hard with a team of friends
b) Find the most promising Korean teenagers and pay them thousands of dollars to play Dota 2 all day long
c) Sell your soul to the devil
d) Spend a boatload of cash on the science project of a bunch of Silicon Valley nerds

Here’s a hint: it’s d)

Why Dota 2

- A hard problem for Reinforcement Learning
- Long Time horizons: Games usually last 30 minutes
- Partially-observed state
- High-dimensional action and observation spaces
- Observes around 16,000
- And on the average time step the model can choose from 8000 - 80 000 actions

Constraints

- Opena AI Five has limited hero pool - 17 heroes
- Some items are not supported: Illusion rune, Helm of Dominator
- Scripted actions
- All are necessary to reduce unnecessary complexity

Methods used

- Deep Learning
- Long Short-Term Memory Neural Networks

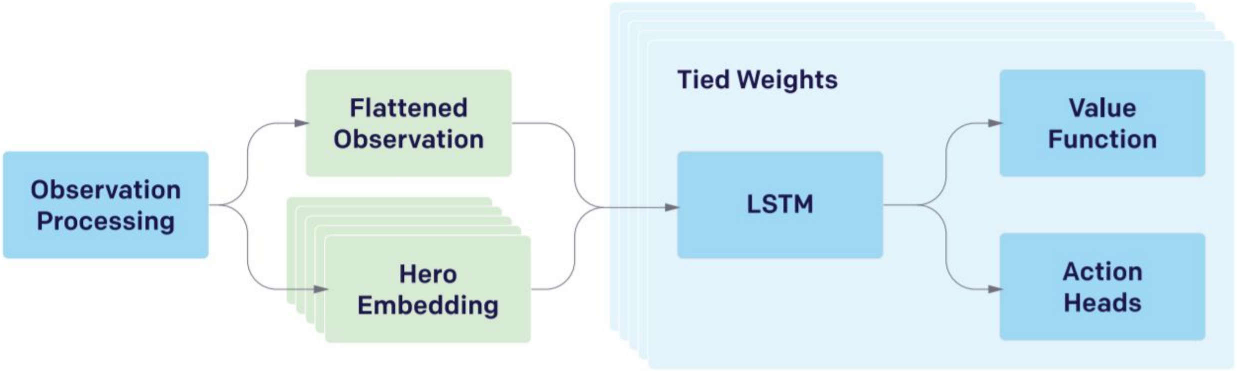
Deep Learning

- A machine learning methodology
- Basically it’s a neural network with a large number of internal layers
- The reason behind this is that it provides model that can extract higher level features from a raw input of data e.g. an image of an animal
- Very computationally expensive

LSTM

- A type of Recurrent Neural Network
- They are well suited for classifying, processing and making predictions on time series data
- Used for classifying handwriting and speech recognition
- They are ideally suited for the task of

Training Overview



- The agent doesn’t use graphical data for an observation
- The timestep is fixed to be every 4th frame rendered by the game engine
- The engine outputs an object containing all of the relevant information for the game in the current frame, such as the position of visible units, the health and position of teammates, etc.

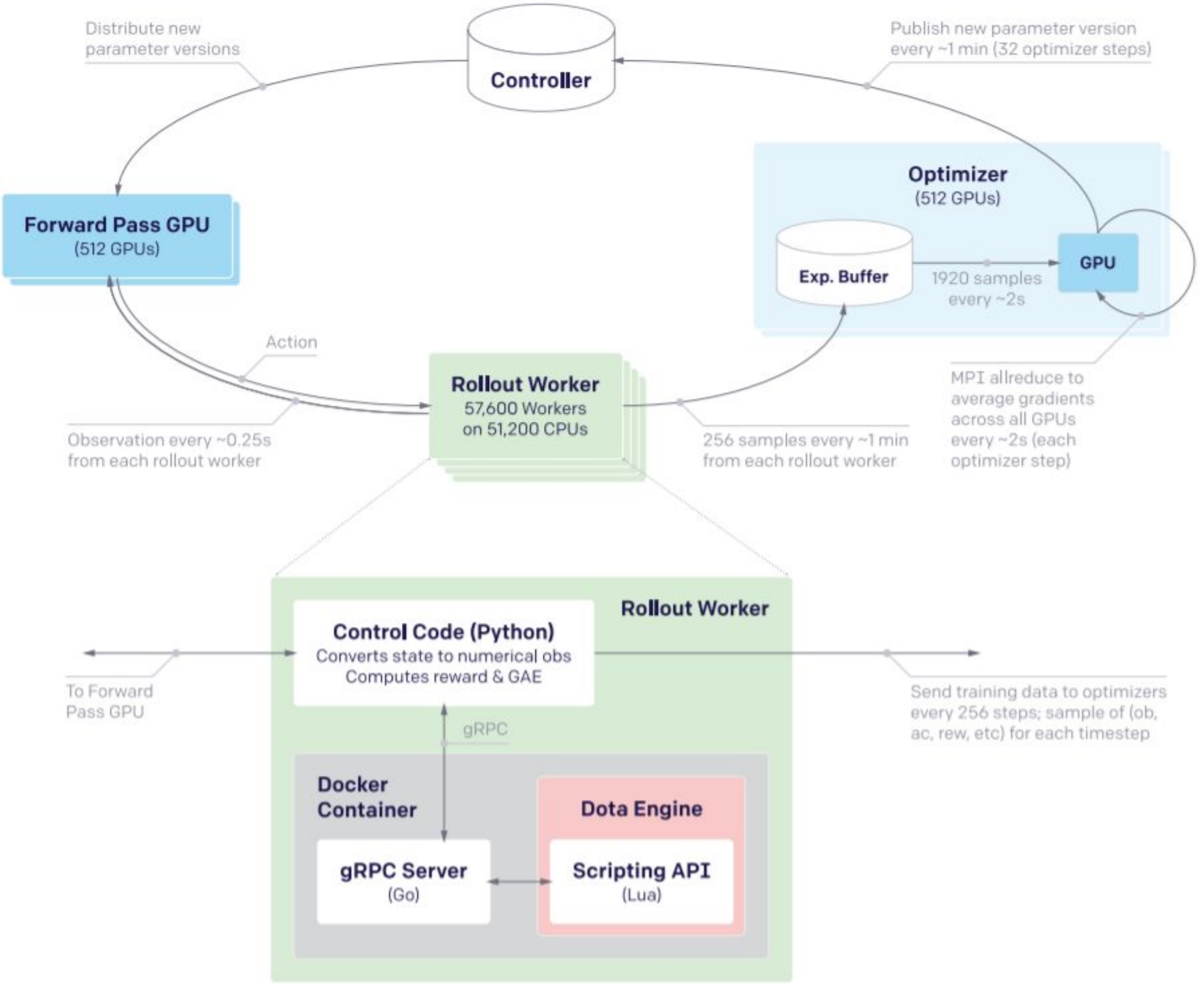
- Some actions were hand-scripted such as: buying items, leveling abilities.
- The policy function π is represented by a model with 4096-unit LSTM neural network
- π is defined as a function as a history of observation to a probability of actions

Observation and Action space

- around 16, 000 values per time step
- every has an observation for himself
- Soma actions are scripted: Iteb buy, leveling up abilities
- More than one million possible actions in the action space

Optimizing the policy

- the agent is trained using Proximal Policy Optimization
- The games used for playing are self-play, i.e. the agent plays against an identical or older version of itself


Controller
Distribute new parameter versions
Publish new parameter version every ~1 min (32 optimizer steps)
Forward Pass GPU
(512 GPUs)
Optimizer
(512 GPUs)
Exp. Buffer
1920 samples every ~2s
GPU
MPI allreduce to average gradients across all GPUs every ~2s (each optimizer step)
Action
Rollout Worker
57,600 Workers on 51,200 CPUs
Observation every ~0.25s from each rollout worker
256 samples every ~1 min from each rollout worker
Rollout Worker
Control Code (Python)
Converts state to numerical obs
Computes reward & GAE
To Forward Pass GPU
Send training data to optimizers every 256 steps; sample of (ob, ac, rew, etc) for each timestep
gRPC
Docker Container
gRPC Server
(Go)
Dota Engine
Scripting API
(Lua)

- the self-play is performed by the Rollout workers playing several games in parallel
- The data for the time steps is sent to the optimizer which stores it asynchronously in an “Experience buffer”
- Then the GPUs sample at random that buffer and use it as a training batch
- They generate gradient which are used to update the model

- Every 32 gradients the optimizers publish those changes to the Controller
- The Forward pass GPUs is a seperate group the runs a forward pass on the model and provides an action based on the observation received from the workers

Surgeries

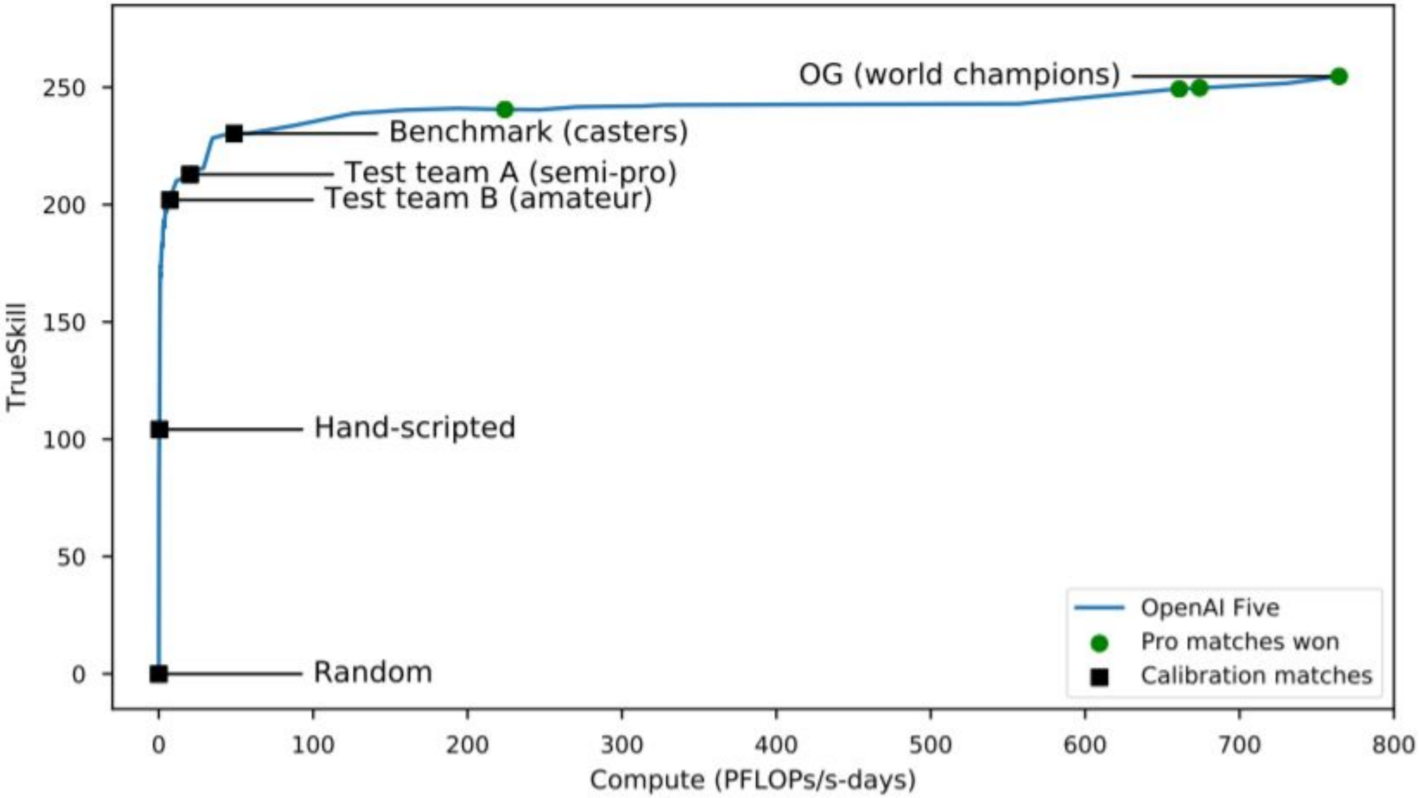
- Changes in the environment, in the code and also knowledge gained durring the training process required changes to the model to be made
- They developed a method, called “surgery”, that modifies the neural network in such a way that doesn’t negatively impact on the training result
- Their goal was to preserve the performance achieved in the old policy and continue improving from there
- This process enabled them to run the training process for over 10 months

Performance improvement

- The entire training run for OpenAI Five was from June 30th 2018 to April 19th 2019
- In mid August 2018 it was playing at top levels and went against two teams of pro players.
- The first one was a group of players from different teams and they lost.
- The second team was a full fledged Dota 2 pro-player team. And they were able to defeat OpenAI Five

But…

- On 12th April 2019 OpenAI Five was able to defeat OG, the reigning champions of Dota2
- And in 4-day event, from the 18th to 21s April, it managed to defeat 99.4% of all human participants, in both cooperative and competitive matches



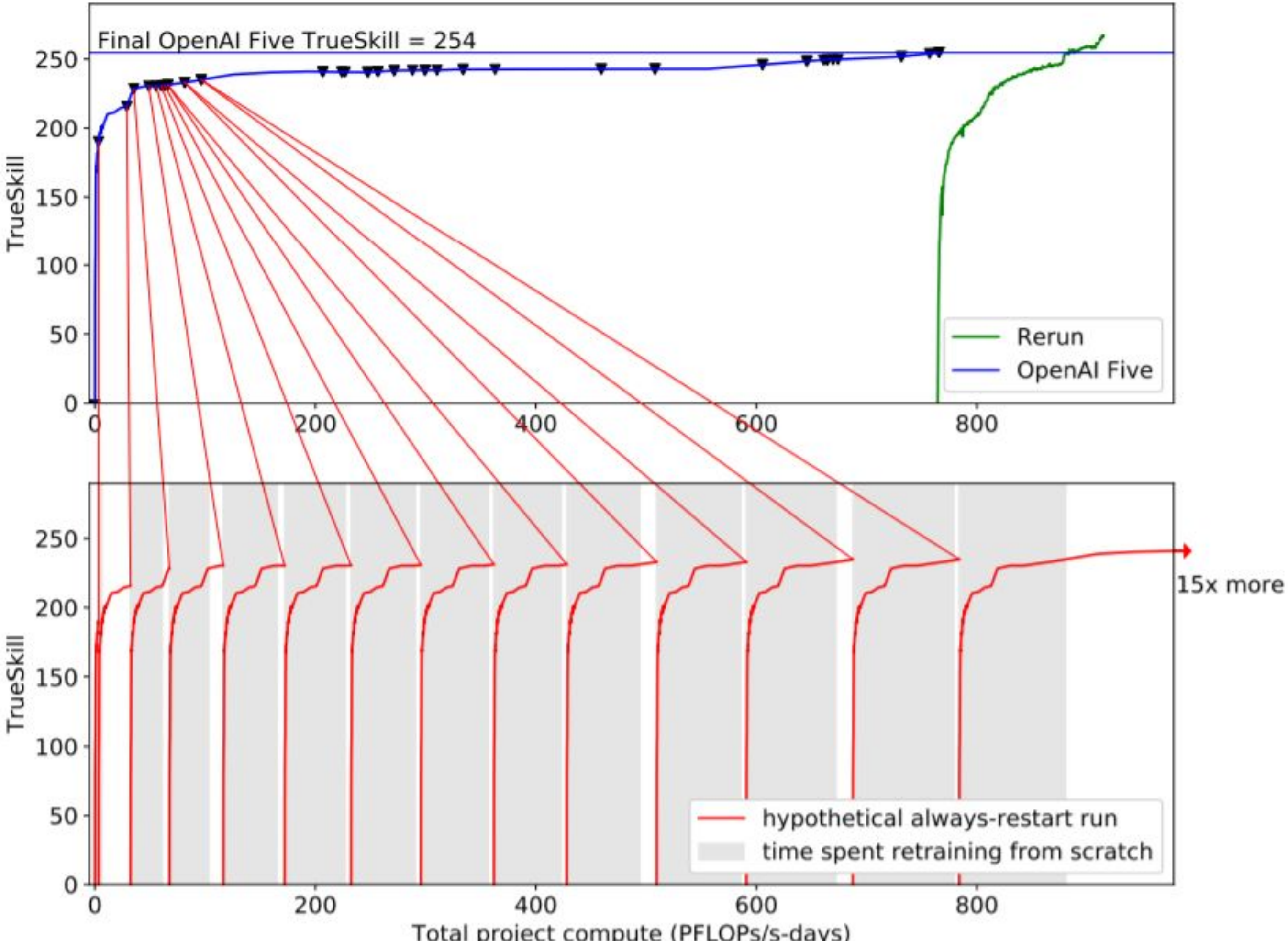
OG (world champions)
Benchmark (casters)
Test team A (semi-pro)
Test team B (amateur)
Hand-scripted
Random
TrueSkill
0
50
100
150
200
250
Compute (PFLOPs/s-days)
0
100
200
300
400
500
600
700
800
OpenAI Five
Pro matches won
Calibration matches

Metrics used

- TrueSkill: A rating system developed by Microsoft Research
- It quantifies a players True skill level based on Baysean Inference Algorithm
- Compute: The amount of processing power spent on training the agent. Measured in PFLOPS/s-day

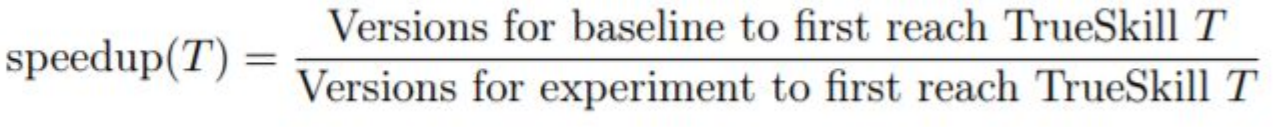
Rerun

- in order to verify the validity of the training results another training run was performed in the last month of the development of OpenAI Five
- It used the final version of all hyperparameters
- The result was it exceeded the performance of the original
- Although fresh reruns offer the best chance for improvement they are also very time consuming and slow down the development process
- What we can take away from this is that it’s best to restart the training when you are certain you have discovered a very good configuration of the model in order to get the best results

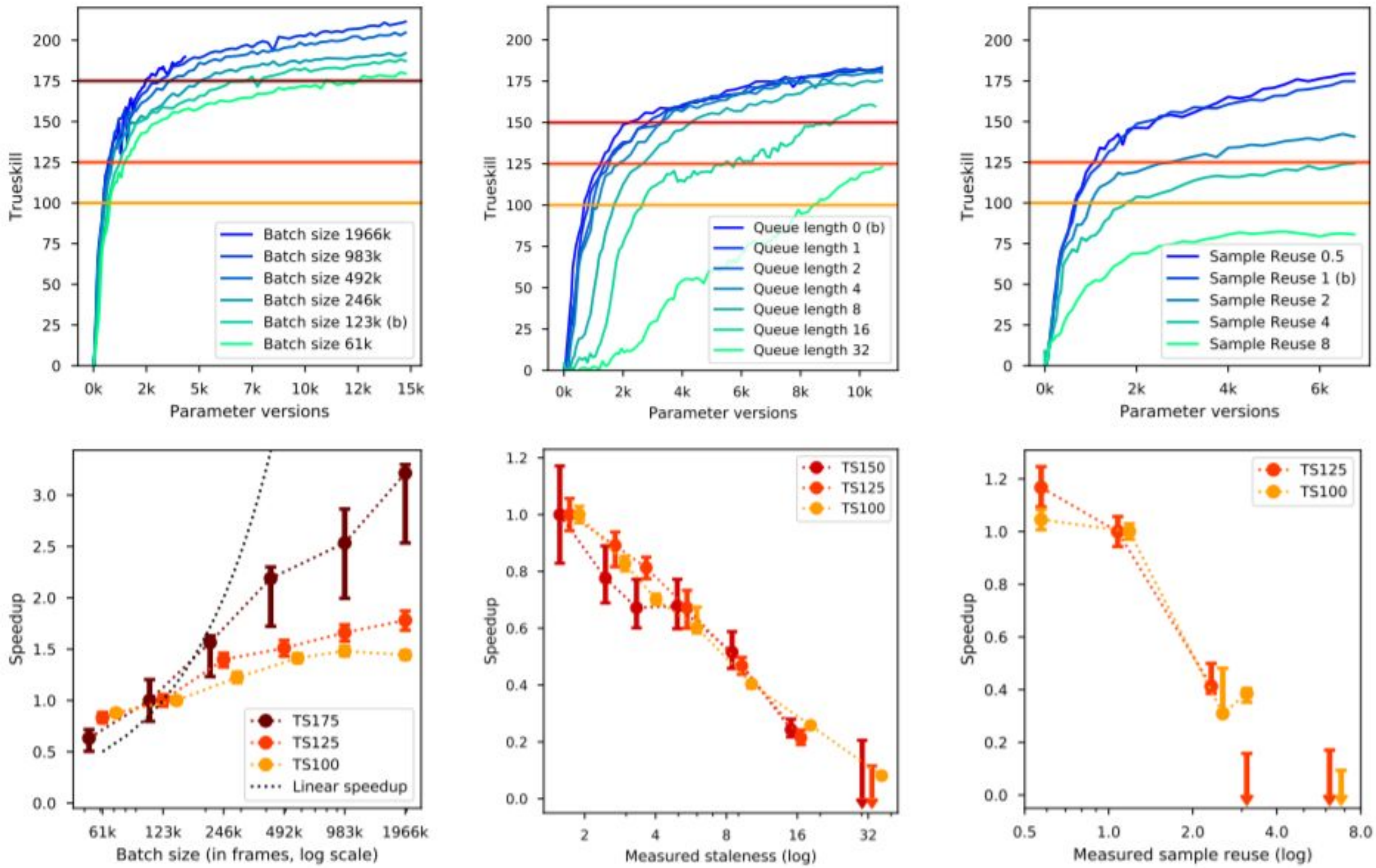

Final OpenAI Five TrueSkill = 254
TrueSkill
0
50
100
150
200
250
0
200
400
600
800
Rerun
OpenAI Five
TrueSkill
0
50
100
150
200
250
0
200
400
600
800
15x more
hypothetical always-restart run
time spent retraining from scratch
Total project compute (PFLOPs/s-days)

Other parameters

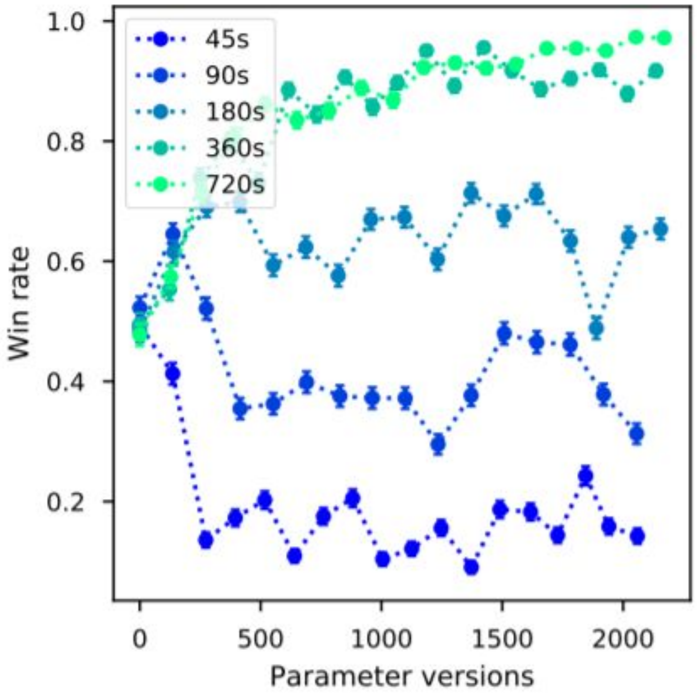
- Batch Size
- Data Stalleness
- Sample Reuse
- All three have different impacts on both the quality and speed of training



$$\text{speedup}(T) = \frac{\text{Versions for baseline to first reach TrueSkill } T}{\text{Versions for experiment to first reach TrueSkill } T}$$


Trueskill
Parameter versions
Batch size 1966k
Batch size 983k
Batch size 492k
Batch size 246k
Batch size 123k (b)
Batch size 61k
Queue length 0 (b)
Queue length 1
Queue length 2
Queue length 4
Queue length 8
Queue length 16
Queue length 32
Sample Reuse 0.5
Sample Reuse 1 (b)
Sample Reuse 2
Sample Reuse 4
Sample Reuse 8
Speedup
Batch size (in frames, log scale)
61k
123k
246k
492k
983k
1966k
TS175
TS125
TS100
Linear speedup
Measured staleness (log)
TS150
TS125
TS100
Measured sample reuse (log)
TS125
TS100

Effect of horizon on agent performance



Results

- it was able to play on par with professional players with a response time 217ms, compared to the typical human response time of 250ms
- this may be different for professional players
- It had a play style which was in some way similar to human play but in other very different
- some behaviours learned early on in the training remained till the end
- In the end the play style was very close to human play but a few key differences

- it tends to move heroes around the map way more
- it takes calculated risks even with low health heroes
- Uses resources far more greedily

What can we gain from this?

- The Surgery method is a very promising set of tools which will greatly improve AI development, for both practical situations and academic research
- It’s possible to train an agent that can plan for the long term
- The semantic representation of the data does give some advantage to the AI compared to human perception
- The AI can detect new strategies, that humans either haven’t thought about or deemed too risky

Things to discuss

- The training results
- The final LSTM networks
- How this AI can be improved