Automated Curriculum Learning for Reinforcement Learning

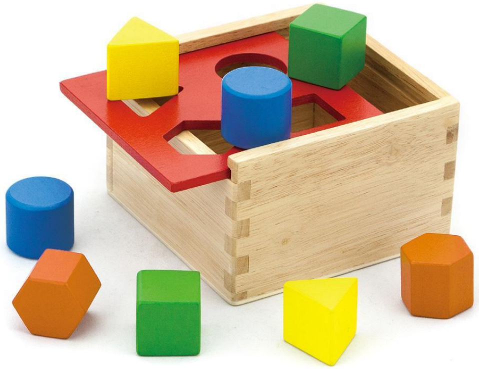
Feryal Behbahani

Jeju Deep Learning Camp 2018

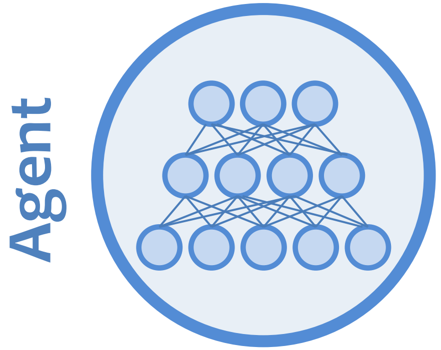
Shape sorter?

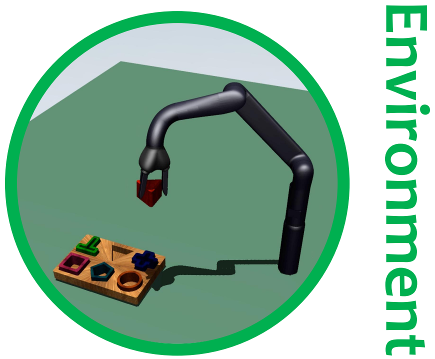
- Simple children toy: **put shapes in the correct holes**
  - Trivial for adults
  - Yet children cannot fully solve until 2 years old (!)

⇒ Can we use Deep Reinforcement Learning to solve it?

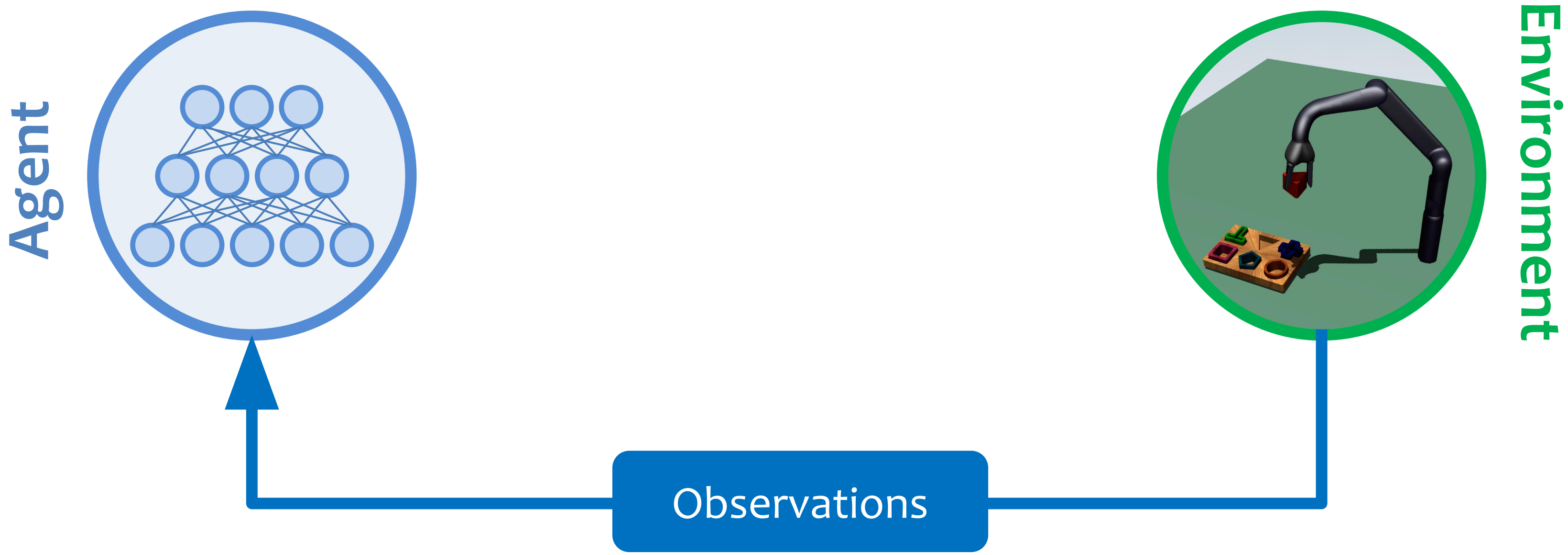


Deep Reinforcement Learning for control

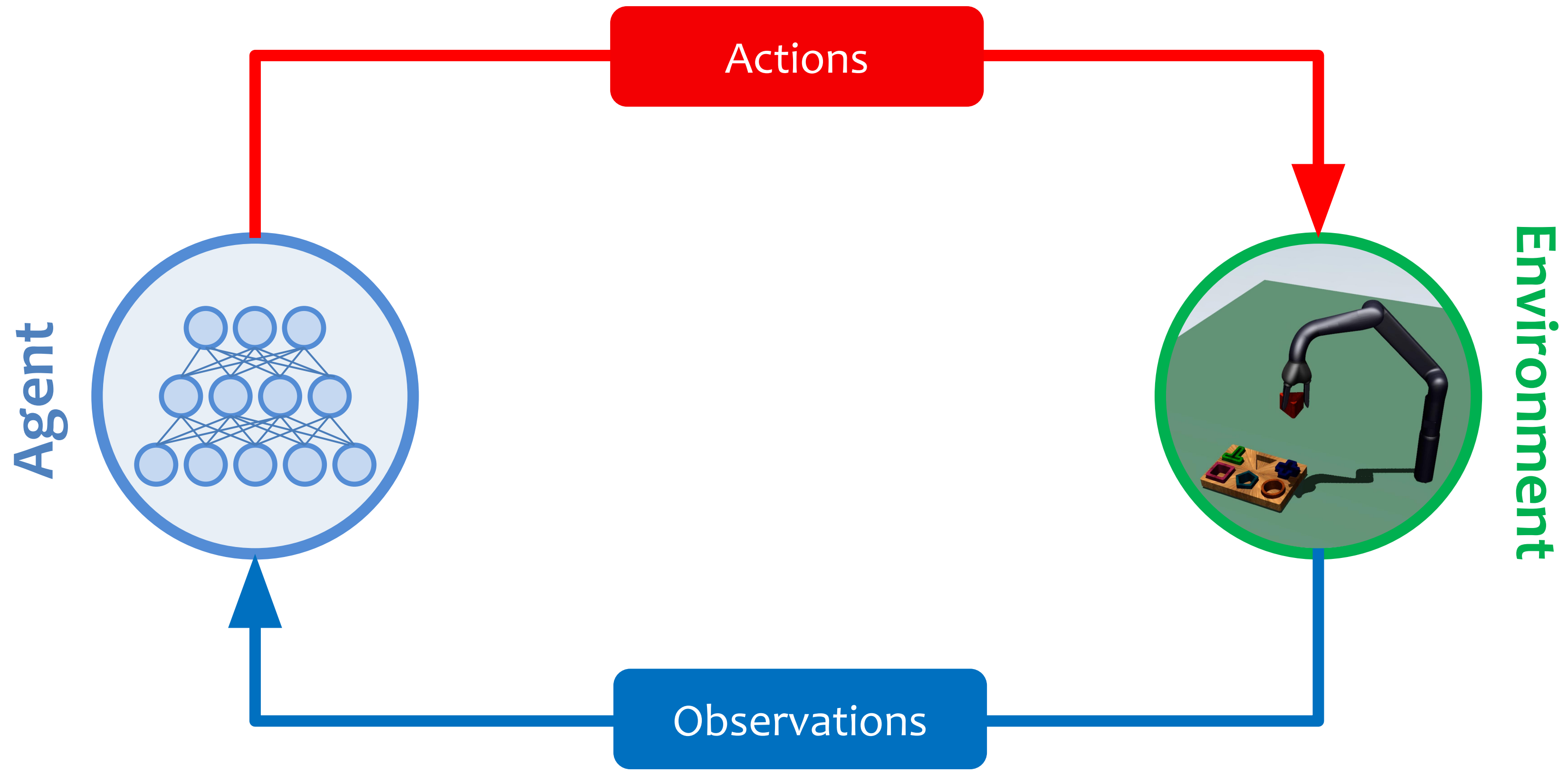




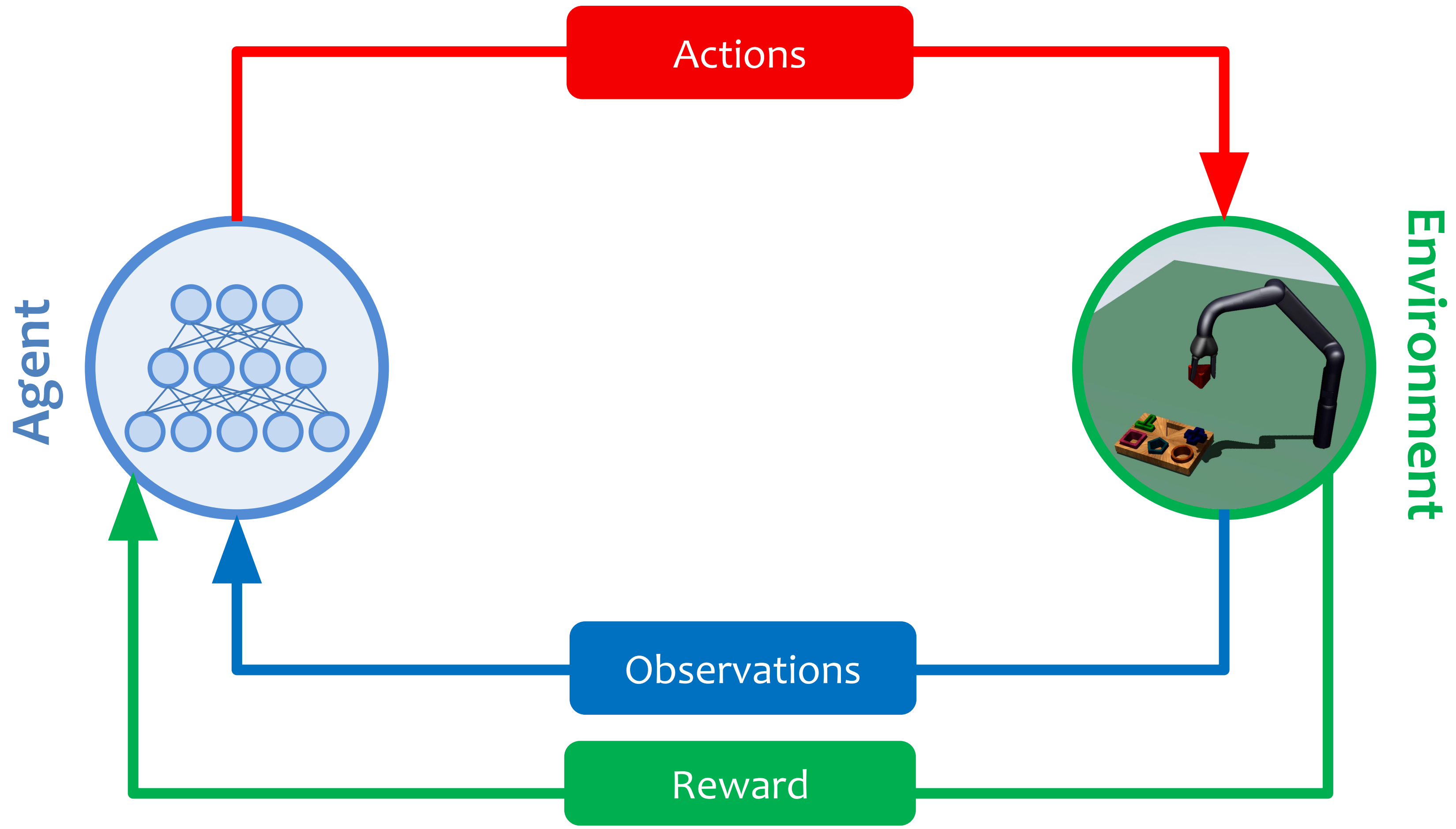
Deep Reinforcement Learning for control



Deep Reinforcement Learning for control



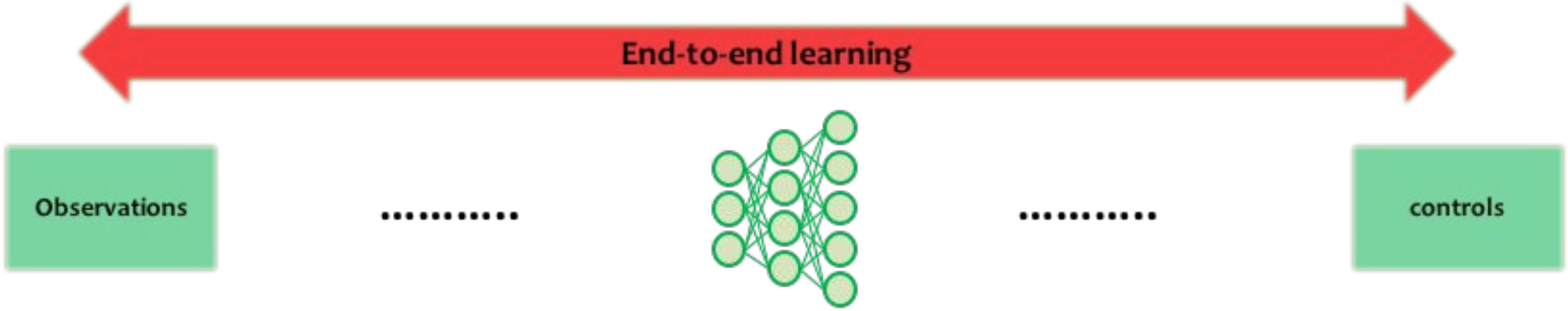
Deep Reinforcement Learning for control



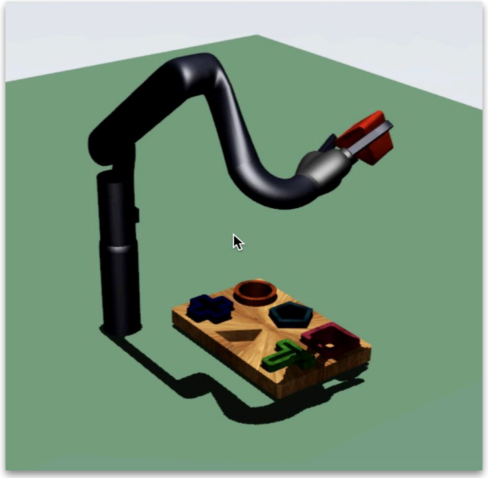
Can we use Deep Reinforcement Learning to directly solve it?

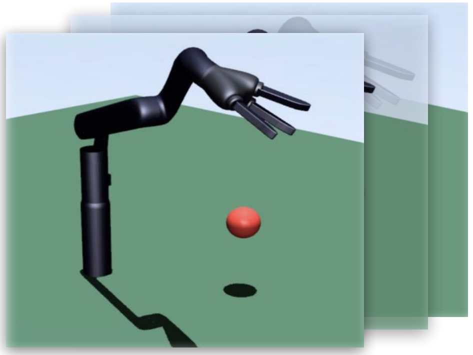
Unlikely…

- Very sample inefficient
- Complex task does not provide learning signal early on



✚ Automatic generation of curriculum of simpler subtasks





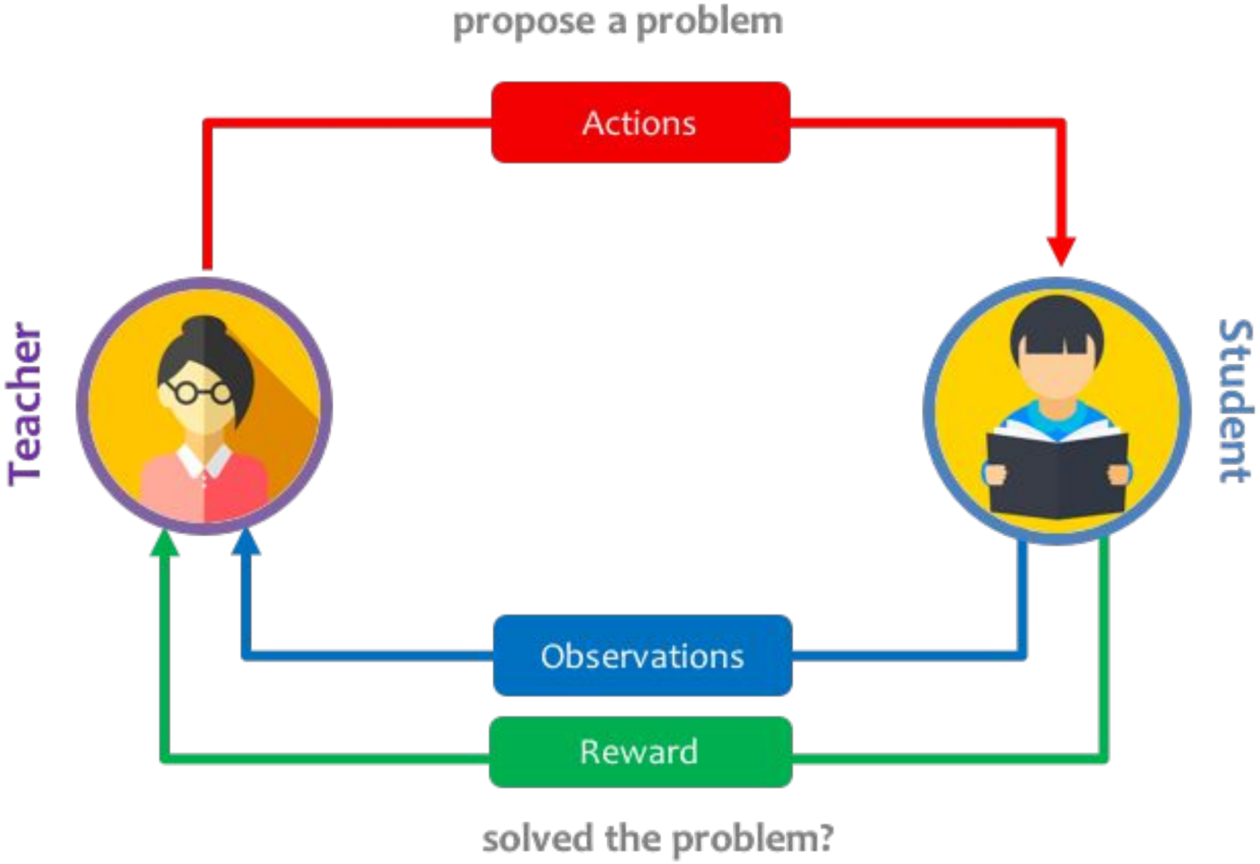
Reach

Push

Grasp

Place

...



Design a sequence of tasks for the agent to train on, in order to improve final performance or learning speed.

Each stage of this curriculum should be tailored to the current ability of the agent in order to promote learning new, complex behaviours.

Environment

Simpler environment with possibility of procedurally generating many hierarchical tasks with sparse reward structure?

[Andreas et al, 2016]

Environment

Crafting and navigation in 2D environment:

- **Move around**
- **Items to pick up and keep in inventory**
- **Transform things at workshops**

Different tasks requiring different actions:

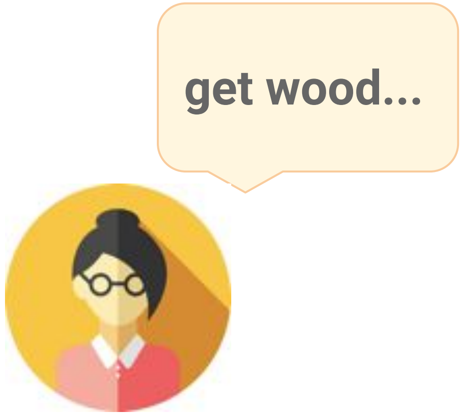
<u>Get wood</u>

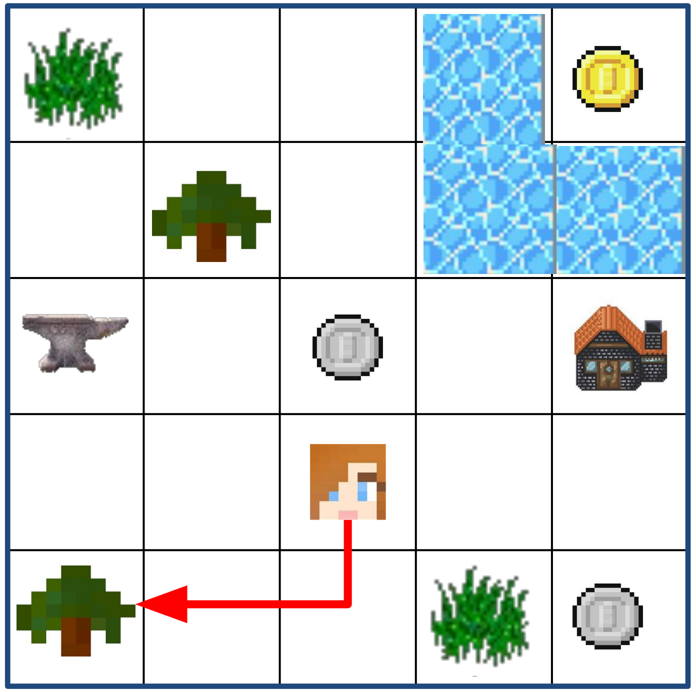
Make plank: Get wood → Use workbench

Make bridge: Get wood → Get iron → Use factory

Get gold: Make bridge → Use bridge on water

...





[Andreas et al, 2016]

Environment

Crafting and navigation in 2D environment:

- Move around
- Items to pick up and keep in inventory
- Transform things at workshops

Different tasks requiring different actions:

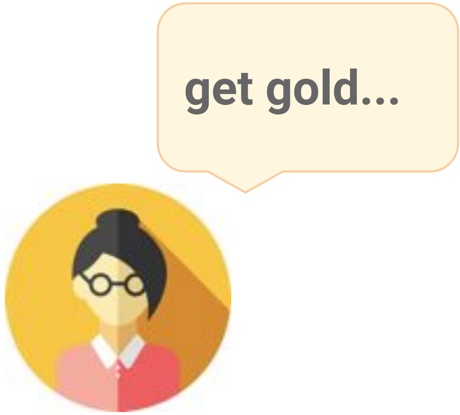
Get wood

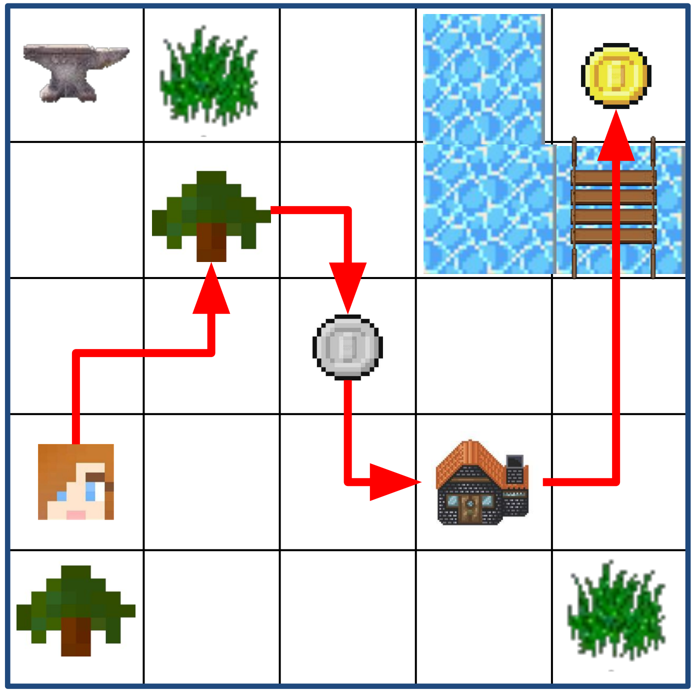
Make plank: Get wood → Use workbench

Make bridge: Get wood → Get iron → Use factory

Get gold: Make bridge → Use bridge on water

...





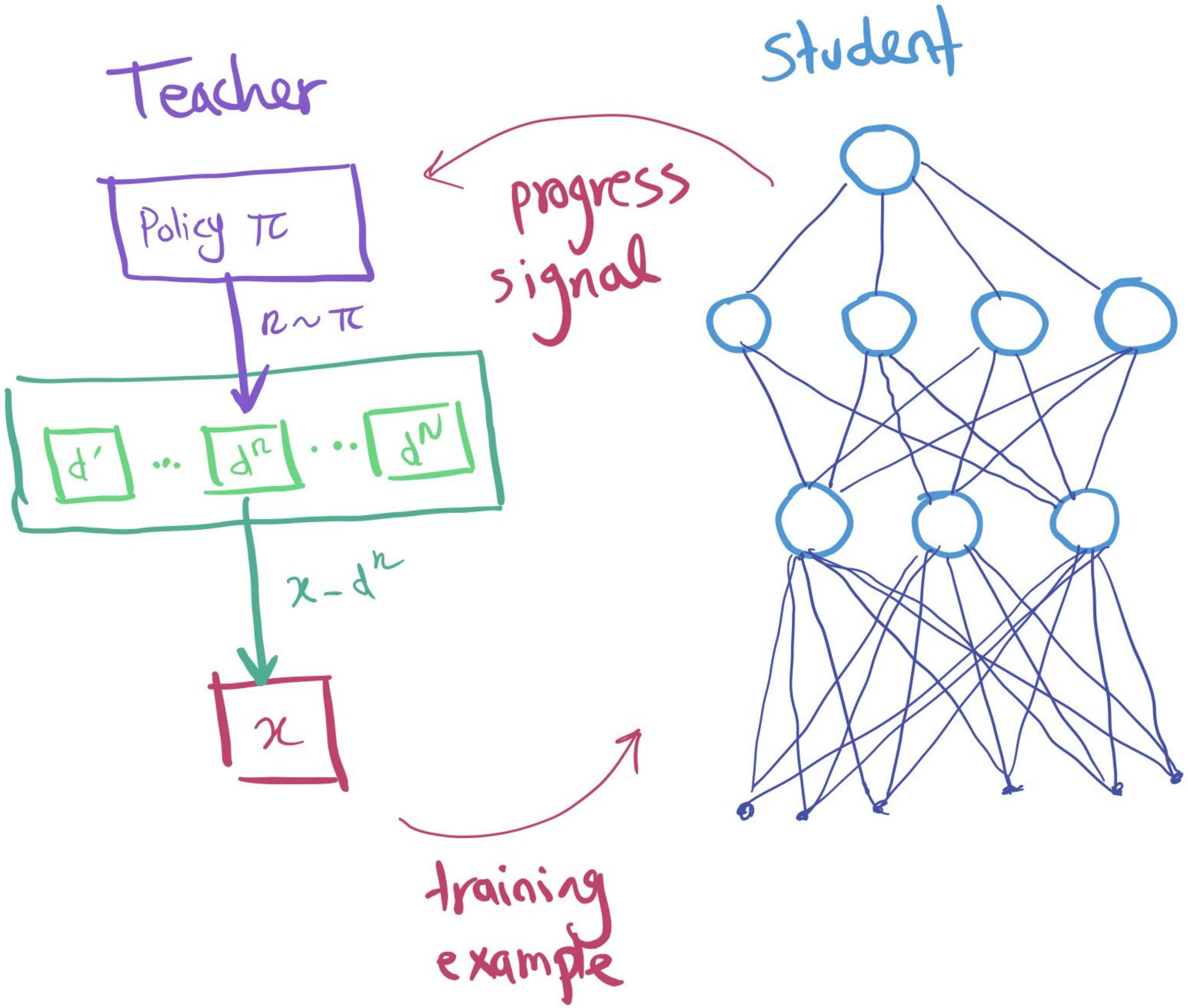
Environment

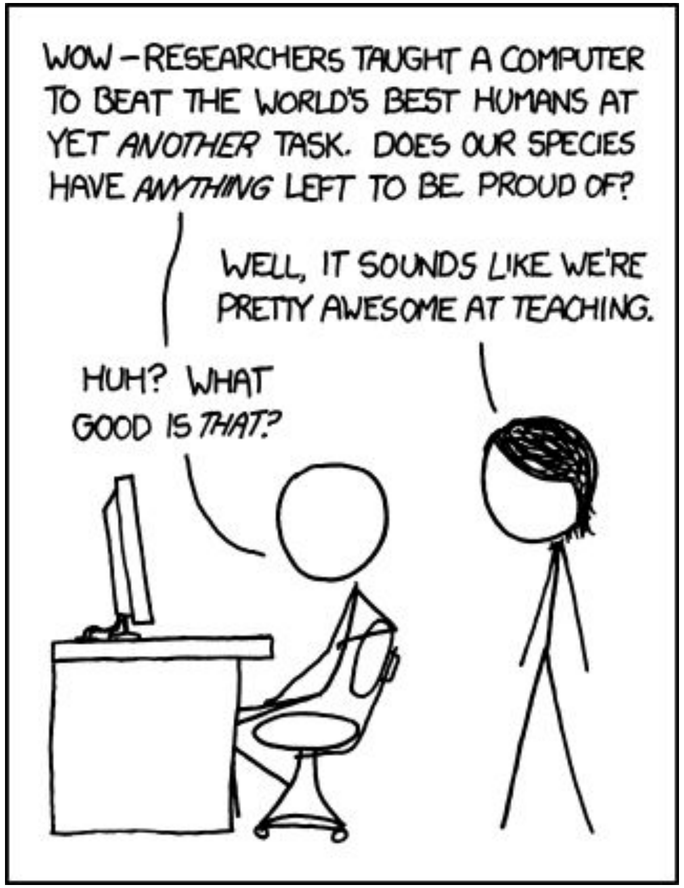
17 tasks - different "difficulties"

Easy	Get wood	
	Get grass	
	Get iron	
Medium	Make plank	Get wood → Use workbench
	Make stick	Get wood → Use anvil
	Make cloth	Get grass → Use factory
	Make rope	Get grass → Use workbench
	Make bridge	Get wood → Get iron → Use factory
	Make bundle	Get wood → Get wood → Use anvil
Complex	Get gold	Make bridge → Use bridge on water
	Make flag	Make stick → Get grass → Use factory
	Make bed	Make plank → Get grass → Use workbench
	Make axe	Make stick → Get iron → Use workbench
	Make shears	Make stick → Get iron → Use anvil
Hard!	Make ladder	Make stick → Make plank → Use factory
	Get gem	Make axe → Cut trees→ Get gem
	Make golden arrow	Make stick → Get gold → Use workbench

random agent

Setup

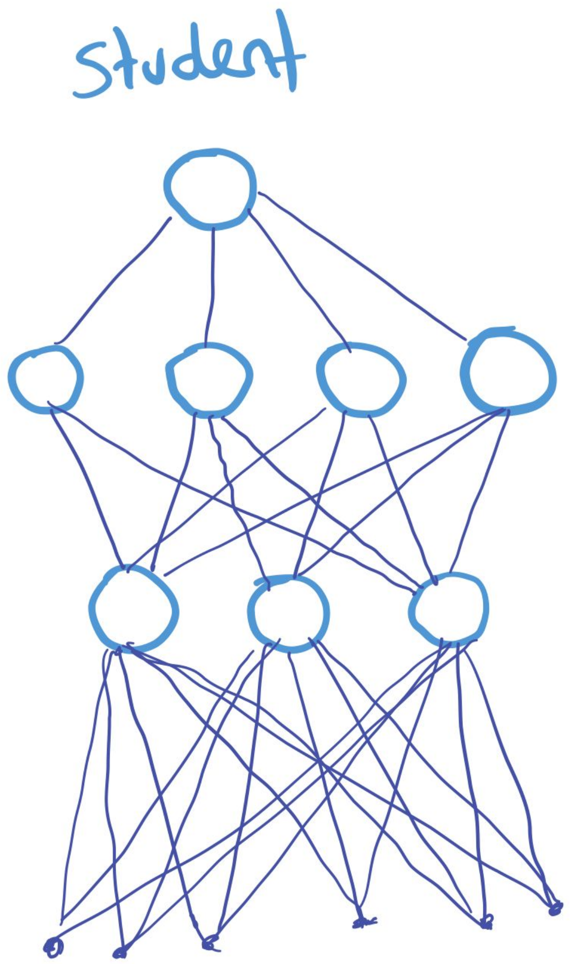




[Comic from: xkcd.com]

[Schematic of Teacher-Student Setup inspired by Marc Bellemare's talk at ICML 2017]

Student Network

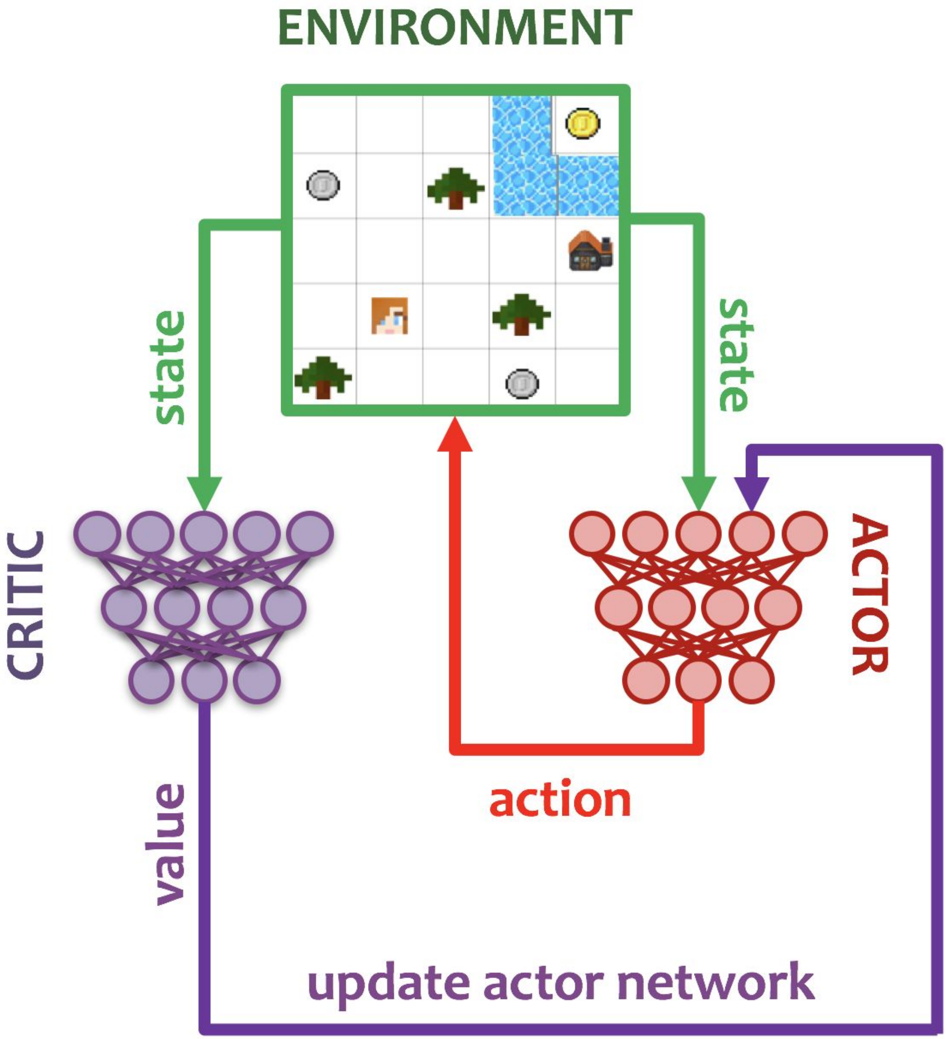


- Will be given a task and associated environment.
- Should learn to perform the task, given sparse rewards.
- Will be trained end-to-end.

- **Choice:** **IMPALA** Scalable agent (DeepMind)
  - Advantage Actor Critic method
  - Off-policy V-Trace correction
  - Many actors, can be distributed
  - Trains on GPU with high throughput
  - *Open-source released recently*

[Espeholt et al, 2018]

Actor-Critic Policy Gradient Method



Agent acts for T timesteps (e.g., T=100)

For each timestep t, compute

$$\hat{R}_t = r_t + \gamma r_{t+1} + \ldots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} \hat{V}(s_T)$$

$$\hat{A}_t = \hat{R}_t - \hat{V}(s_t)$$

Compute loss gradient:

$$g = \nabla_\theta \sum_{t=1}^{T} -\log \pi(a_t|s_t)\hat{A}_t + \left(\hat{V}(s_t) - \hat{R}_t\right)^2$$

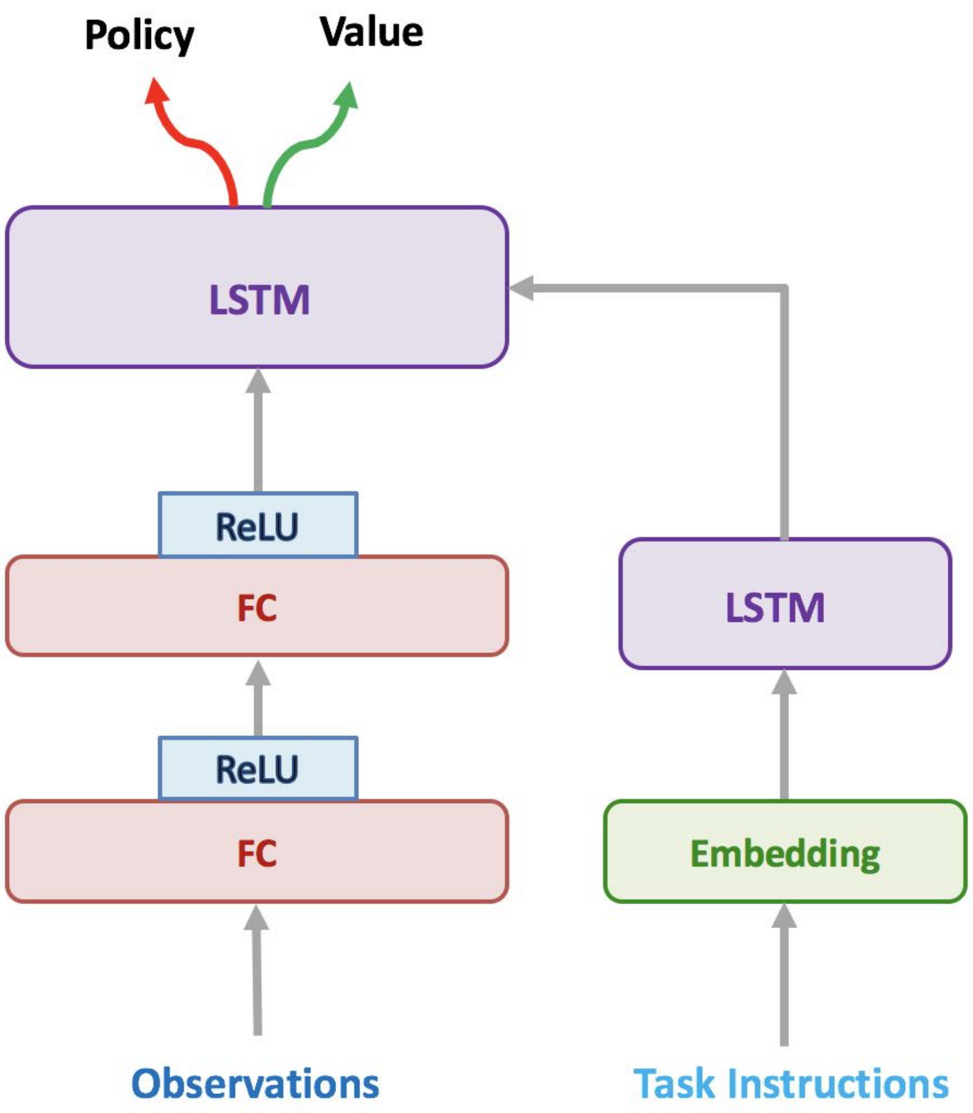
Plug g into a stochastic gradient descent optimiser (e.g. RMSprop)

Multiple actors interact with their own environments and send data back to learner

This helps with robustness and experience diversity

[Mnih et al, 2016]

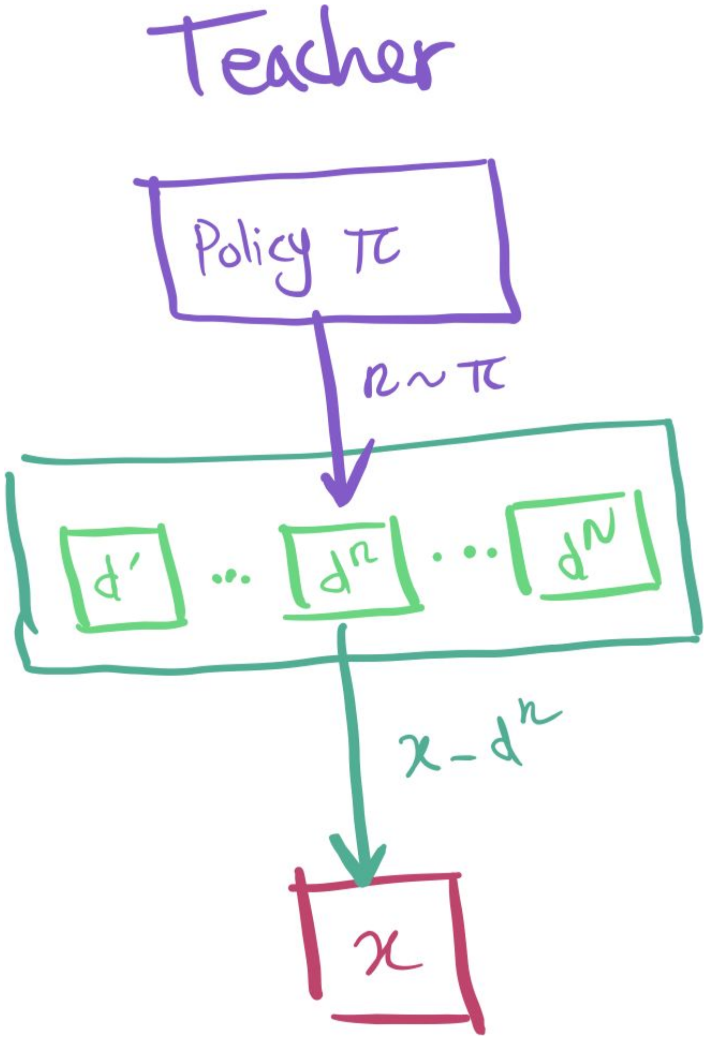
Agent architecture



- **Inputs:**
  - Observations:
    5x5 egocentric view, 1-hot features & inventory
  - Task instructions: strings
- **Observation processing:**
  - 2x fully connected with 256 units
- **Language processing:**
  - Embedding: 20 units
  - LSTM for words: 64 units
- **LSTM** (recurrent core)
  - 64 units
- **Policy**
  - Softmax (5 possible actions : Down/Right/Left/Up/Use)
- **Value**
  - Linear layer to scalar

[Based on Espeholt et al, 2018]

Teacher



- Should propose tasks and monitor the student *progress signal.*
- Need to adapt to student learning.
- Need to explore tasks space well.

- **Choice:** **<u>Multi-armed bandit</u>** EXP3 algorithm
  - Well studied.
  - Proofs of optimality of exploration/exploitation trade-offs.
  - Has been explored in the context of curriculum design before.

[Graves et al, 2017]

Teacher: Multi-armed Bandit

[Zhou et al, 2015]

Learns a model of outcomes	Multi-armed bandits	Reinforcement Learning
Given model of stochastic outcomes	Decision theory	Markov Decision Process
	Actions do not affect the state of the world	Actions change state of the world dynamically

- Given K tasks, propose task with highest expected "reward".
  - *reward* = "progress of student"
- Use EXP3 "Exponential-weight algorithm for Exploration and Exploitation"
  - Optimizes minimum regret.

$$P(\text{pick task } k) = (1-\gamma)\frac{w_k(t)}{\sum_{i=1}^{K} w_i(t)} + \frac{\gamma}{K}$$

$$w_k(t+1) = \begin{cases} w_k(t)e^{\gamma\hat{r}(t)/K} & \text{selected task} \\ w_k(t) & \text{other tasks} \end{cases}$$

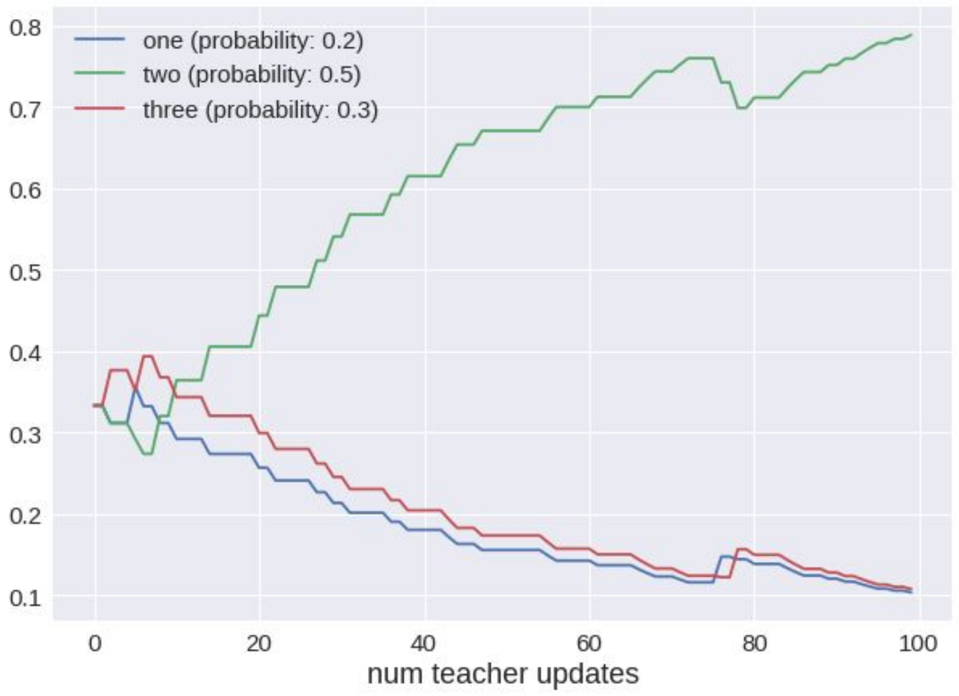
Octopus figure from https://tech.gotinder.com/smart-photos-2/

[Auer et al, 2001]

Teacher: Adversarial Multi-armed Bandit

Toy example on fixed reward situation:

- 3 tasks, rewards = 0.2, 0.5 and 0.3.
  - Explore early, random choices.
  - When enough evidence collected, exploits 2nd arm!



Which "progress signal" to chose?

- Many exist in literature
- Explored two in context of RL:
  - "Return gain"
  - Gradient prediction gain

Progress Signal	Definition
Prediction gain (PG)	$\nu_{PG} := L(x,\theta) - L(x,\theta')$
Gradient prediction gain (GPG)	$\nu_{GPG} := \|\nabla L(x,\theta)\|_2^2$
Self prediction gain (SPG)	$\nu_{SPG} := L(x',\theta) - L(x',\theta') \quad x' \sim D_k$
Target prediction gain (TPG)	$\nu_{TPG} := L(x',\theta) - L(x',\theta') \quad x' \sim D_N$
Mean prediction gain (MPG)	$\nu_{MPG} := L(x',\theta) - L(x',\theta') \quad x' \sim D_k, k \sim U_N$
Gradient variational complexity gain (GVCG)	$\nu_{GVCG} := [\nabla_{\phi,\psi} KL(P_\phi \| Q_\psi)]^\top \nabla_\phi \mathbb{E}_{\theta \sim P_\phi} L(x,\theta)$
L2 gain (L2G)	$L_{L2}(x,\theta) = L(x,\theta) + \frac{\alpha}{2}\|\theta\|_2^2$

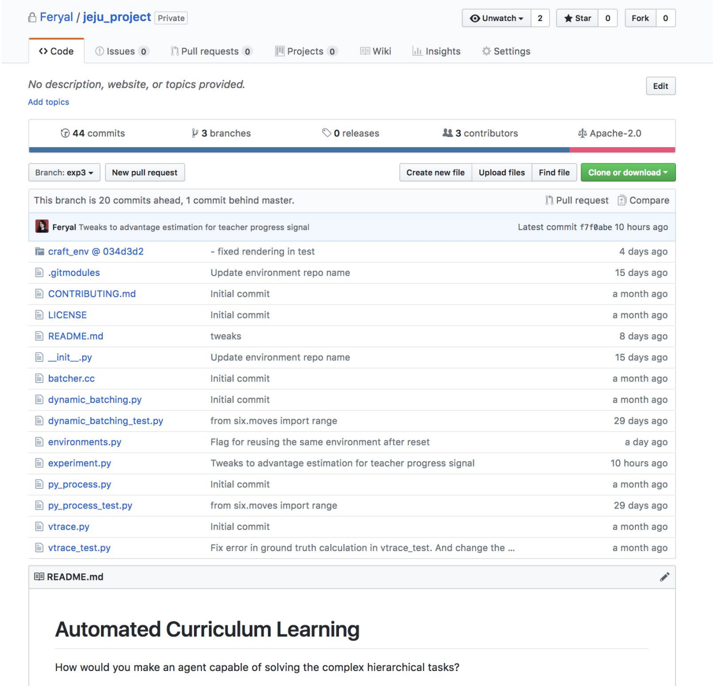
[Extensively studied in ***Graves et al, 2017*** in supervised & unsupervised Learning settings]

Implementation

- Codebase, based on IMPALA , extensively modified:
  a. Handle new Craft environment, adapted from [Andreas et al, 2016], procedurally creating gridworld tasks given a set of rules.
  b. Support "switchable" environments, to change tasks on the fly.
  c. Teacher implementing EXP3 and possible variations with several progress signals.
  d. Evaluation built-in during training, extensive tracking of performance.
  e. Graphical visualisation of behaviour for trained models.
  f. Jupyter notebooks for analysis

Released on Github with accompanying report shortly!

Implementation



Feryal / jeju_project Private

Unwatch 2 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

No description, website, or topics provided. Edit

Add topics

44 commits | 3 branches | 0 releases | 3 contributors | Apache-2.0

Branch: exp3 | New pull request | Create new file | Upload files | Find file | Clone or download

This branch is 20 commits ahead, 1 commit behind master. Pull request | Compare

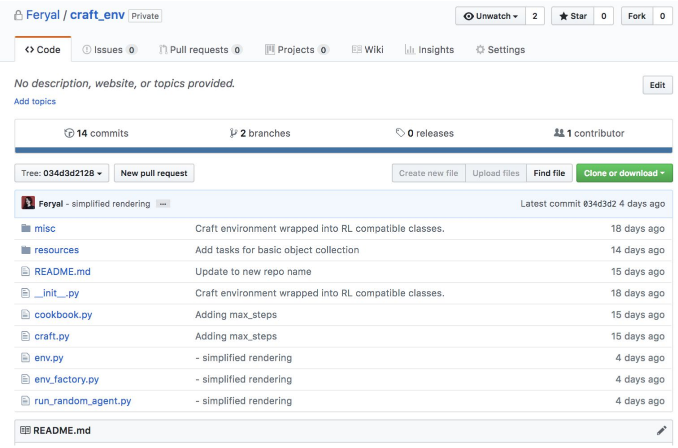
Feryal Tweaks to advantage estimation for teacher progress signal — Latest commit f7f0abe 10 hours ago

craft_env @ 034d3d2	- fixed rendering in test	4 days ago
.gitmodules	Update environment repo name	15 days ago
CONTRIBUTING.md	Initial commit	a month ago
LICENSE	Initial commit	a month ago
README.md	tweaks	8 days ago
__init__.py	Update environment repo name	15 days ago
batcher.cc	Initial commit	a month ago
dynamic_batching.py	Initial commit	a month ago
dynamic_batching_test.py	from six.moves import range	29 days ago
environments.py	Flag for reusing the same environment after reset	a day ago
experiment.py	Tweaks to advantage estimation for teacher progress signal	10 hours ago
py_process.py	Initial commit	a month ago
py_process_test.py	from six.moves import range	29 days ago
vtrace.py	Initial commit	a month ago
vtrace_test.py	Fix error in ground truth calculation in vtrace_test. And change the ...	a month ago

README.md

Automated Curriculum Learning

How would you make an agent capable of solving the complex hierarchical tasks?



Feryal / craft_env Private

Unwatch 2 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

No description, website, or topics provided. Edit

Add topics

14 commits | 2 branches | 0 releases | 1 contributor

Tree: 034d3d2128 | New pull request | Create new file | Upload files | Find file | Clone or download

Feryal - simplified rendering ... — Latest commit 034d3d2 4 days ago

misc	Craft environment wrapped into RL compatible classes.	18 days ago
resources	Add tasks for basic object collection	14 days ago
README.md	Update to new repo name	15 days ago
__init__.py	Craft environment wrapped into RL compatible classes.	18 days ago
cookbook.py	Adding max_steps	15 days ago
craft.py	Adding max_steps	15 days ago
env.py	- simplified rendering	4 days ago
env_factory.py	- simplified rendering	4 days ago
run_random_agent.py	- simplified rendering	4 days ago

README.md

Automated Curriculum Learning

Towards creating agents that can teach themselves!

FERYAL BEHBAHANI
Jeju DL Camp
feryal.github.io

July 2018

Open Source Code
GitHub Repo

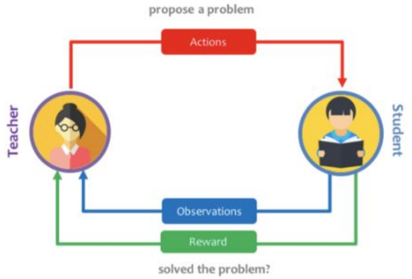
Download
PDF

Abstract

Imagine a problem that is complex and requires a collection of skills, which are extremely hard to learn in one go with sparse rewards (e.g. solving complex object manipulation in robotics). Hence, one might need to learn to generate a curriculum of simpler tasks, so that overall a student network can learn to perform a complex task efficiently. In this project, I set out to train an automatic curriculum generator using a Teacher network which keeps track of the progress of the student network, and proposes new tasks as a function of how well the student is learning.

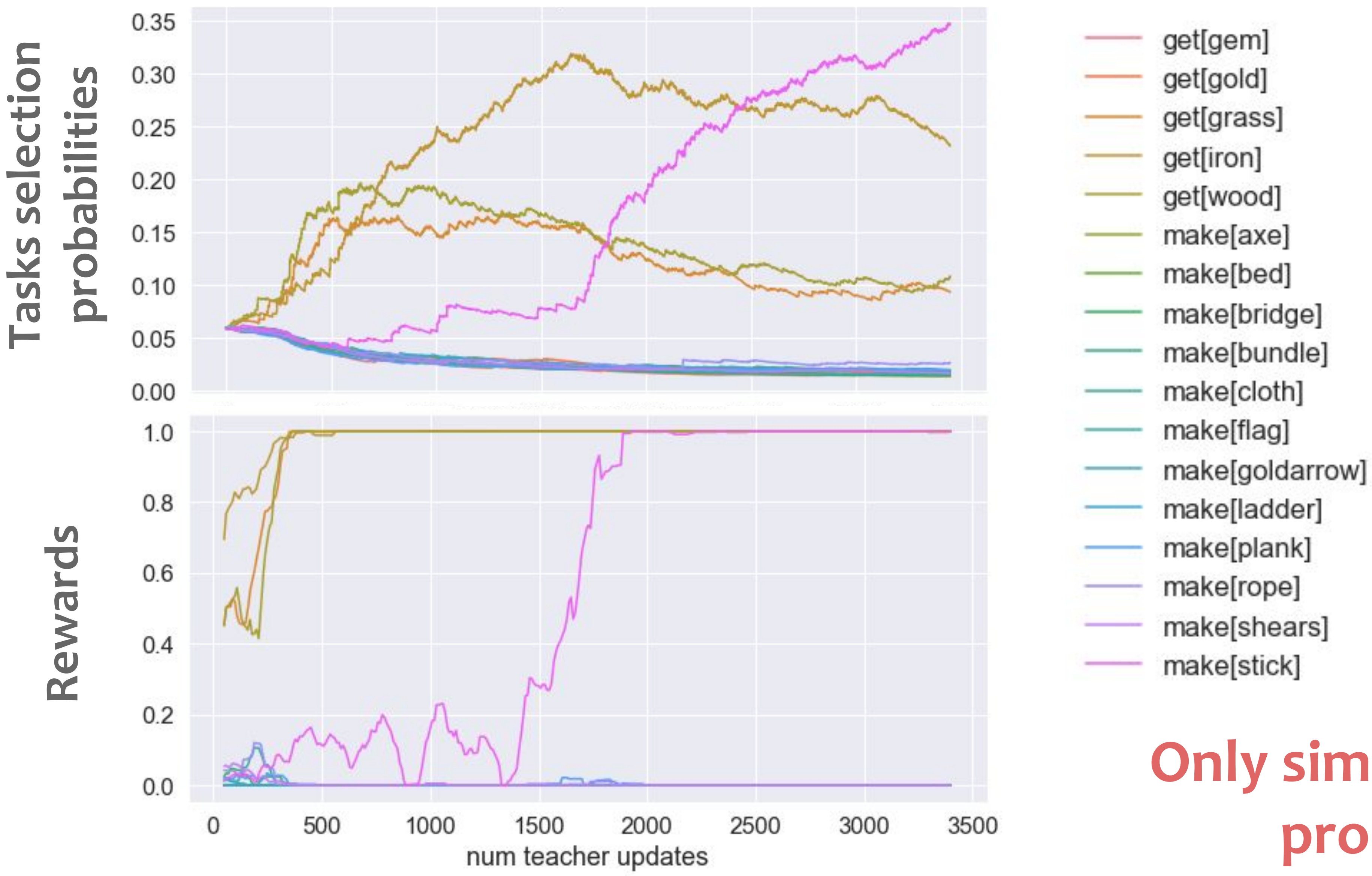
Introduction

How would you make an agent capable of solving the complex hierarchical tasks?



A schematic figure of teacher-student setup.

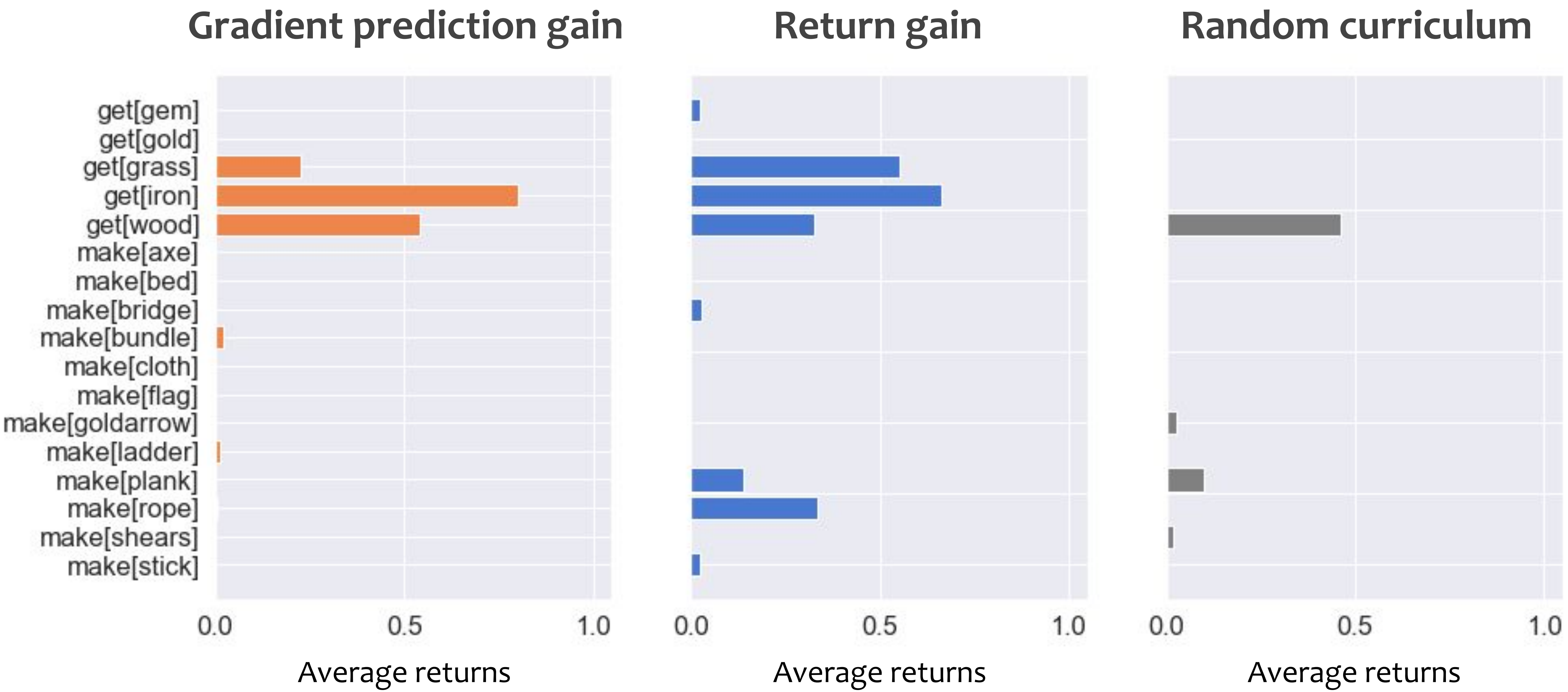
Results: Gradient prediction gain



Only simple tasks are proposed?!

Results: progress signals comparison

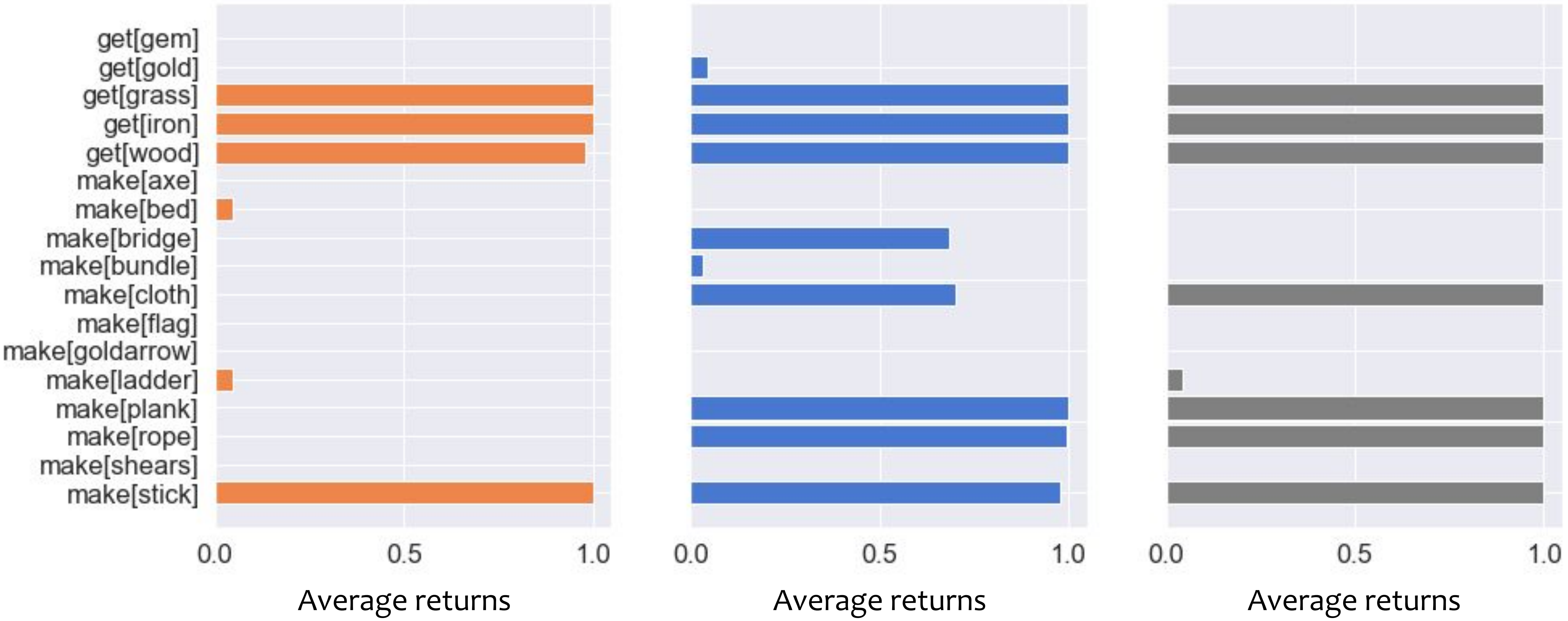
Early during training: 50k steps



Results: progress signals comparison

Mid-training:
30M steps

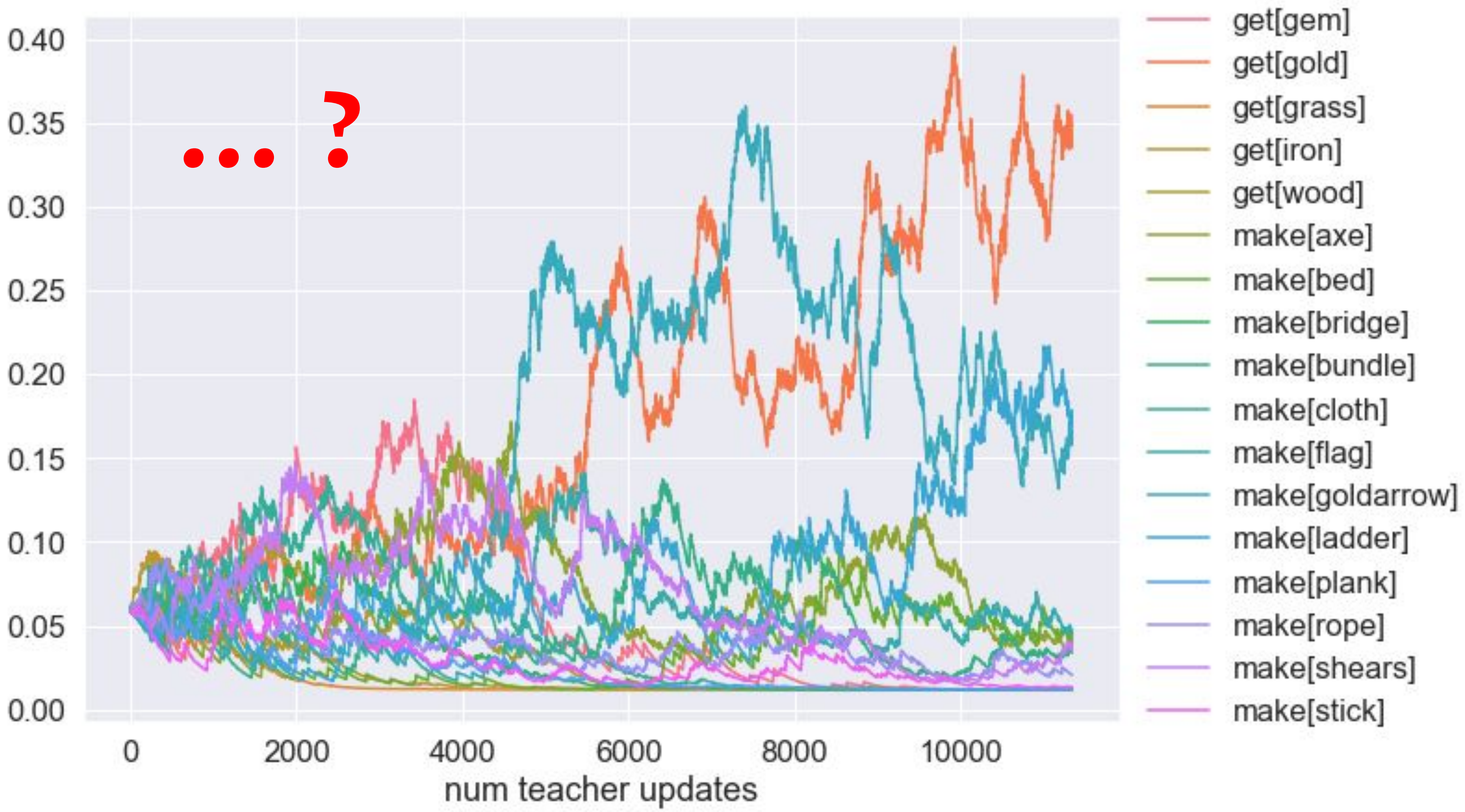
Gradient prediction gain | Return gain | Random curriculum



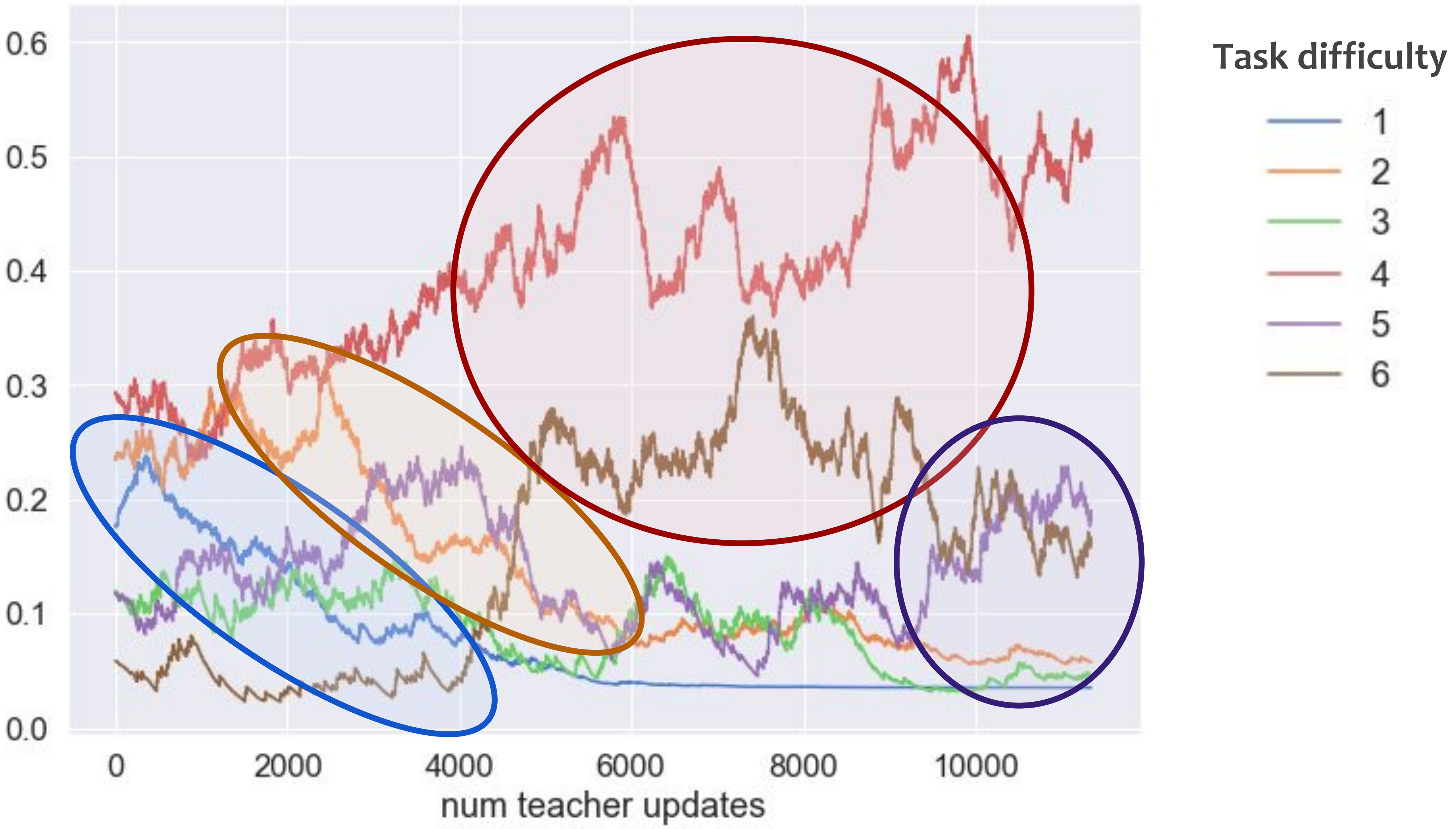
Results: progress signals comparison

Late in training:
100M steps

Return gain - task proposals through training



Return gain - task proposals through training



Results: trained policy on selected tasks

Summary

- Teacher with Return gain successfully taught Student many tasks.
  - Interesting teaching dynamics
  - Just like kids learning, allows the model to learn incrementally, solve simple tasks and transfer to more complex settings
- Bandit teacher could be improved to take other signals into account
  - e.g. safety requirements (Multi Objective Bandit extension)
- More work needed to:
  - Explore Student architecture for more complex tasks
  - Analyse effect of progress signals in the dynamics of learning
  - Teacher proposing “sub-tasks” for the Student: extensions to HRL.

Maybe if our agents become good at teaching, they can optimise how we learn as well!?

Feryal Behbahani feryal.github.io @feryalmp @feryal feryal.mp@gmail.com

Thank you

Great advice and discussions with Taehoon Kim and Eric Jang...

Soonson, Terry and all the other organisers and sponsors for this great opportunity...

Bitnoori for her patience with us!

My new friends from the camp for all the memories and *memes*!



feryal.github.io @feryalmp @feryal feryal.mp@gmail.com