Introduction to Reinforcement Learning

CS 4/5789 Spring 2025

Administrative stuff…

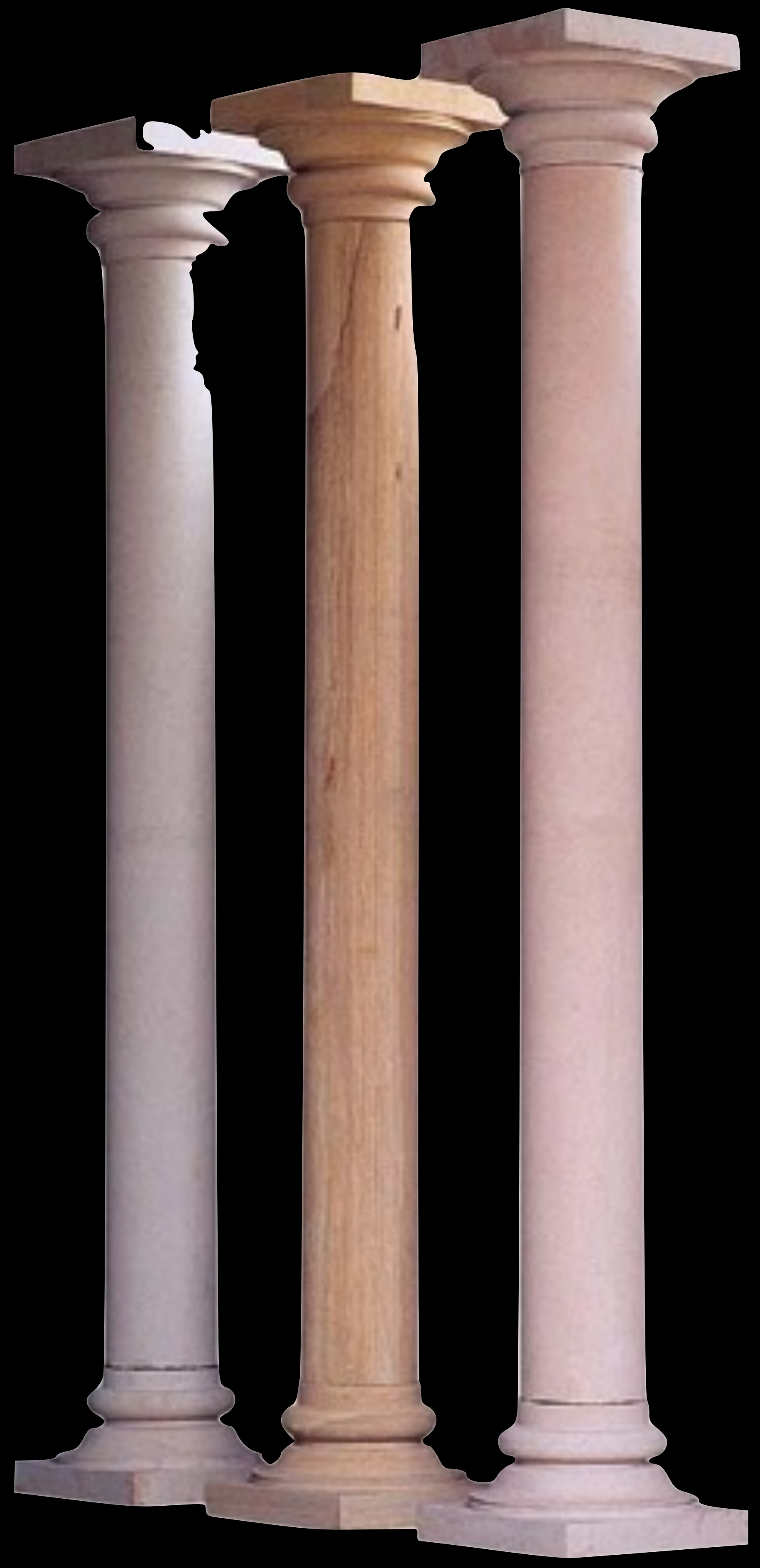
(To get it out of the way)

Course Overview

- Instructor:
  – Wen Sun
- Homepage:
  – https://wensun.github.io/CS4789_spring_2025.html
- TAs:
  – Not that many but they are really good
- Office Hours / Recitations:
  - TA Office Hours: (hopefully) **Every day** (Details will be posted soon.)
  - Leave Feedback: contact Wen
  - Prof. Office Hours: Every Thursday 3-4 (starting next week)
- Questions:
  - Post all questions on ED (you can make them private)
  - **Do not email directly (except in an emergency or need for privacy)**

Prerequisites

- Introduction to ML (CS4/5780)
  - **Statistics / Probability**
  - **Linear Algebra**
  - **Multivariate Calculus**



Homework 0

- Due Jan 29!!!
- Available later today on Canvas/Gradescope
- it’s designed to help you assess your comfort with prerequisite material and introduce you some RL related concepts
- If you find aspects of the HW0 challenging/unfamiliar please reach out

Homework

- Roughly 6 assignments (subject to change)
- Due every 1-2 weeks
- Can work in groups of up to **2**
- Submitted via Gradescope
- Primarily theoretical, they reinforce concepts from class and provide practice for the exams

Exams

- Midterm and Final
- Closed book

Programming

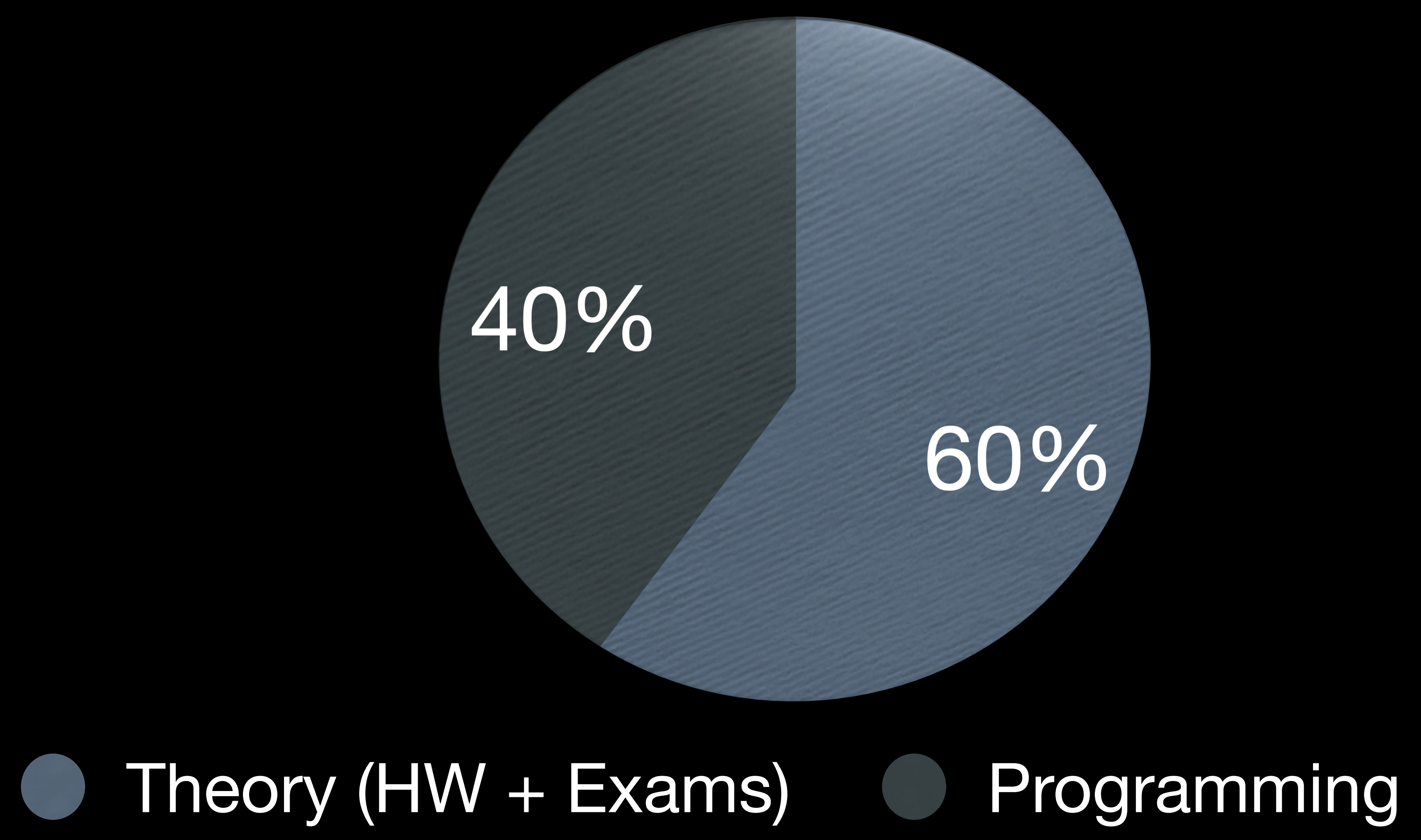
- There will be 4 projects (use RL to control simulated robots and play video games)
- Roughly **2-3 weeks** per project
- Submission details coming soon

For those in 5789

- Intermittent paper comprehension quizzes
- Read and answer questions on relevant RL papers
- Helps build “research comprehension” in the field
- Quizzes completed on Canvas
- Required for everyone in 5789, if you are in 4789 you can complete them if you like

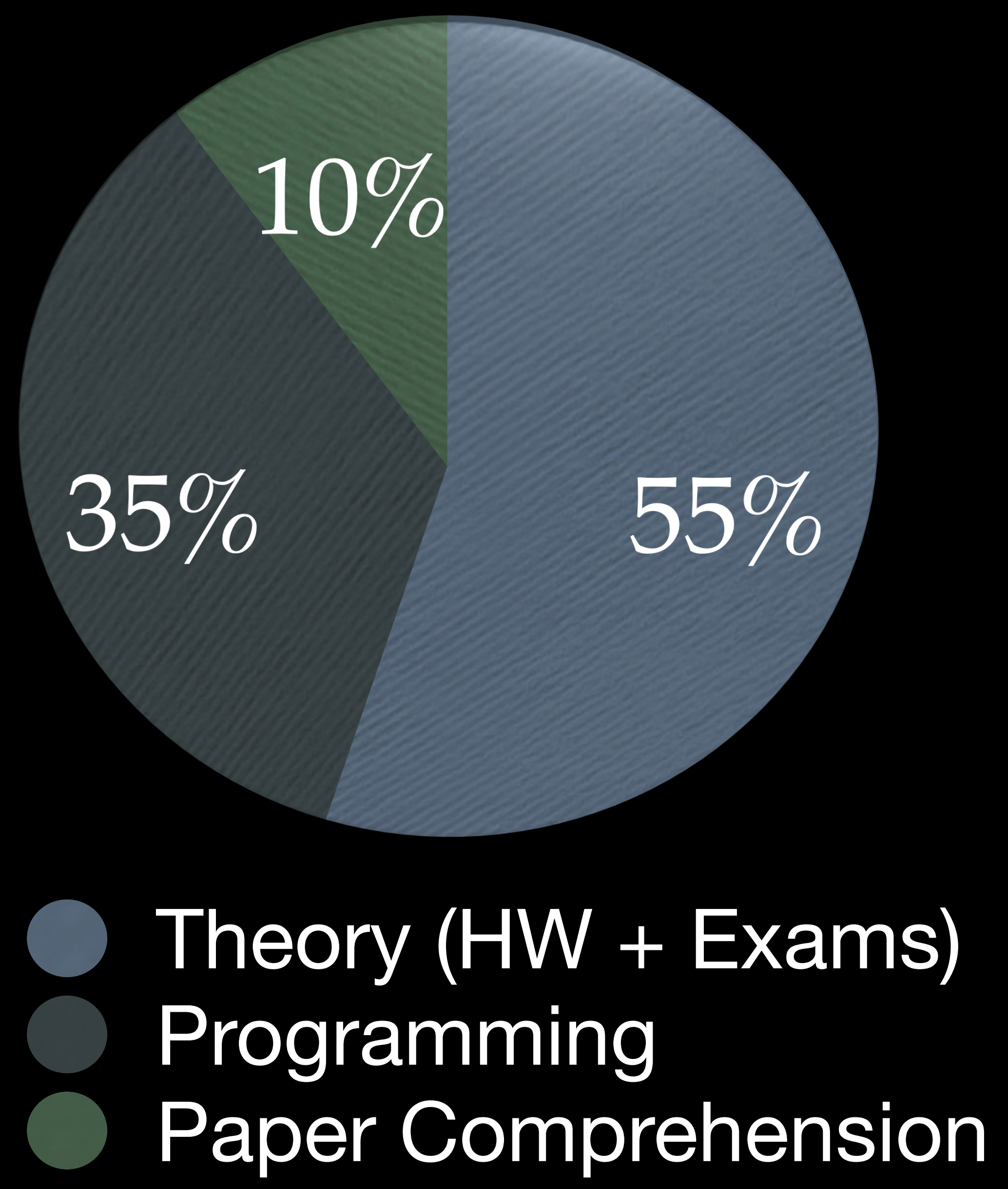
Course Grade Breakdown 4789

- 50% **Theory: Midterm + Final**
  - Closed book
  - No cheat sheets!
  - No personal notes
- 40% **Programming Assignments**
  - Up to **2** members in each team
  - **2** days extension **per team per project**
  - *Extra credit available at times*
- 10% **Homeworks**
  - Up to **2** members in each team
  - 2 days extension per team per assignment
  - Preparation for exam



Course Grade Breakdown 5789

- 45% **Theory: Midterm + Final**
  - Closed book
  - No cheat sheets!
  - No personal notes
- 35% **Programming Assignments**
  - Up to **2** members in each team
  - **2** days extension **per team per project**
  - *Extra credit available at times*
- 10% **Paper Comprehension (mandatory)**
  - Original Research Papers in RL
  - Canvas Quizzes
- 10% **Homeworks**
  - Up to **2** members in each team
  - **2** days extension **per team per homework**
  - Preparation for exam



About this course

- Take this course if …
  - you are interested in Reinforcement Learning
  - you are comfortable with a decent amount of mathematics
  - you are not scared of programming
- Don’t take this course if …
  - Matrices/expectation/conditional expectation scare you
  - you don’t know gradients
  - You never heard about what is dynamic programming

Academic Integrity

- Zero tolerance policy: all occurrences will be reported
- We **actively** look for academic conduct violations
- We check for plagiarism
- AI tool policy: feel free to use it, but need to provide references and include the details in the submissions (e.g., what prompt you used, how did you used the solutions from the tool, what’s the difference between you wrote and the solution provided by the tool;)

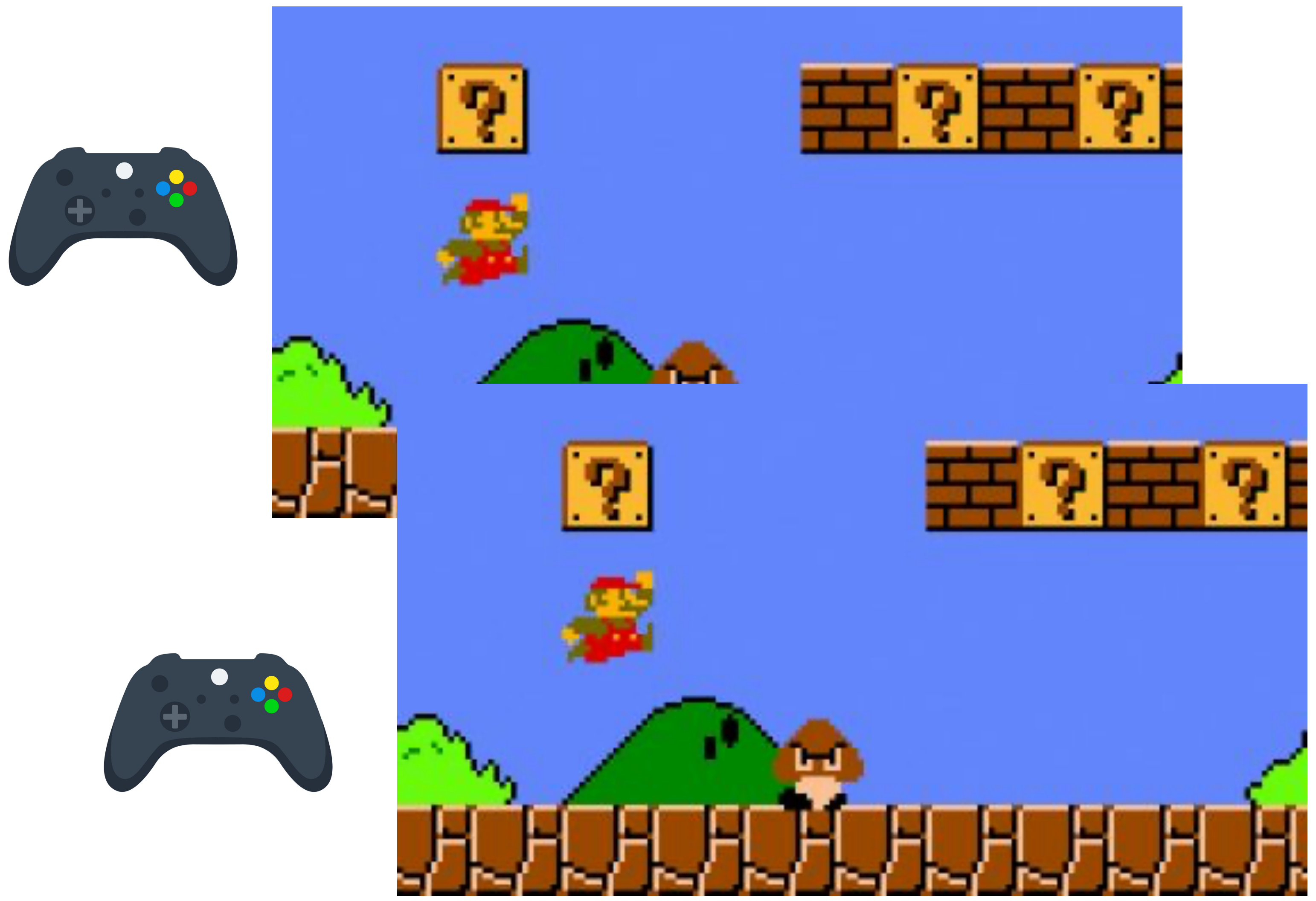
What is Reinforcement Learning

Reinforcement Learning vs
Traditional Machine Learning


SUPER
MARIO BROS.

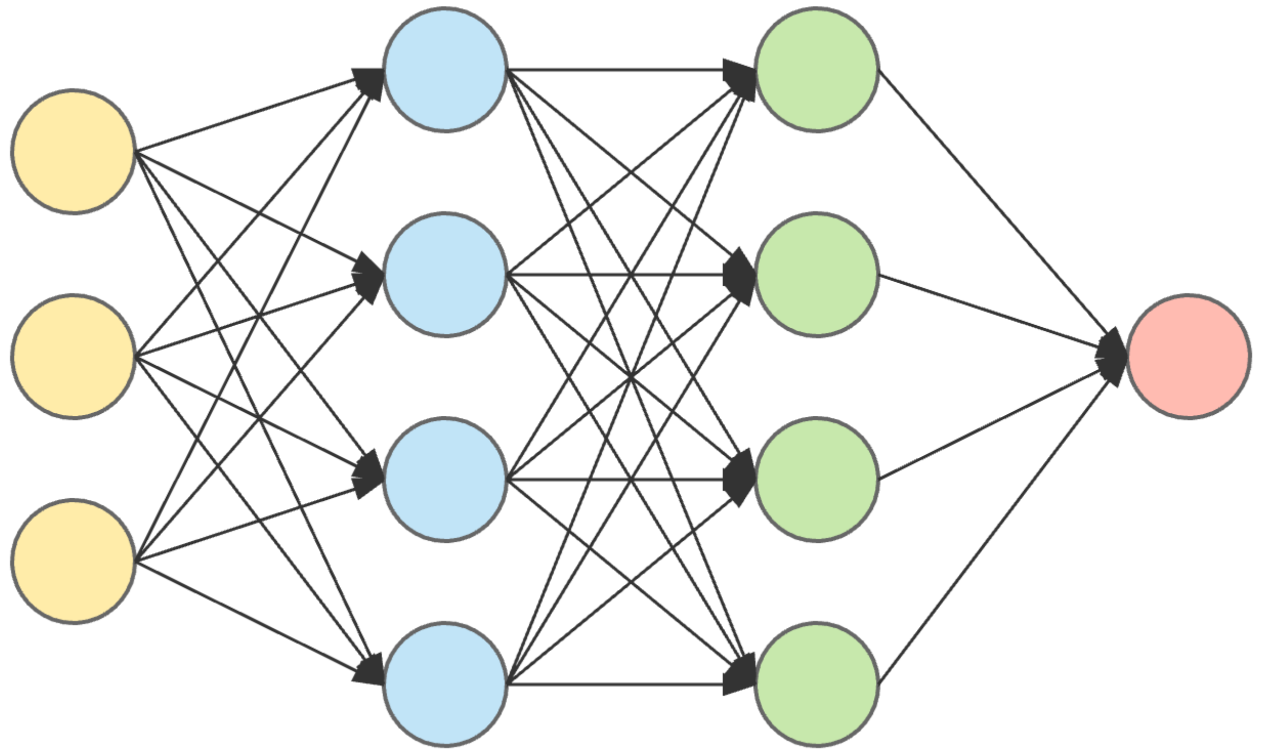
Traditional ML approach

Human demonstrations



....

ML
algorithm



"a program” (e.g.,
a neural network)

AgentLinear

Selected Actions:

RIGHT SPEED

Traditional ML fails, why?

i.i.d assumption does not hold at all

What we act now affects what we gonna see in the future

Traditional ML fails, why?

Training distribution not equal to test distribution

ML algorithms can fail to learn true causation

Example: using ML to learn when to brake from human driving data

ML algorithm thinks: ***brake light on*** => ***brake***

ML algorithms is good at learning spurious correlations;

Correlations ≠ Causation

What we need?

Algorithms that can:

learn from non-i.i.d data

reason about future (if I tried this action, the future would be…)

learn from trial and error (continuously learn from its own experiences)

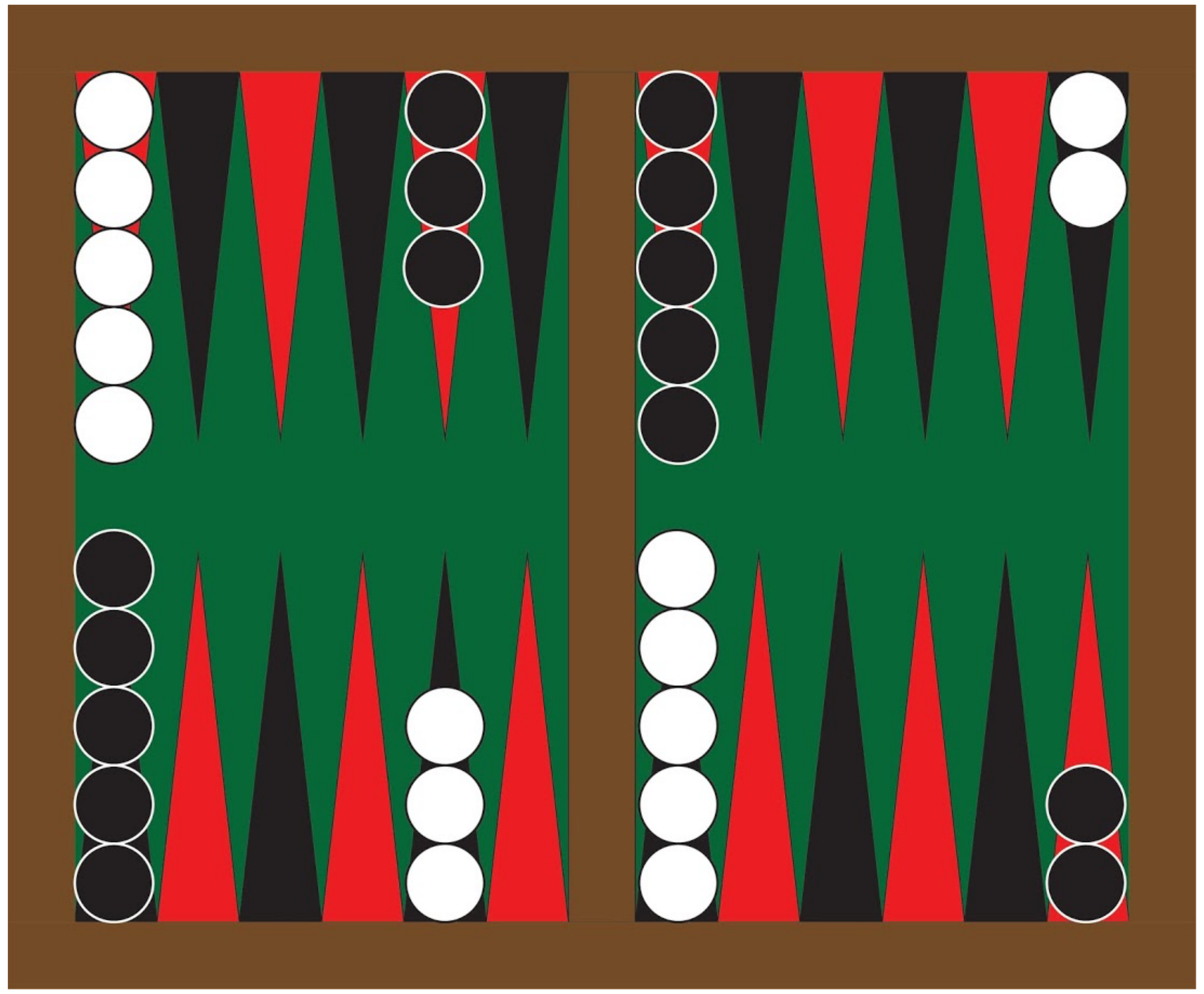
Selected Actions:

RIGHT SPEED

	Learn from Experience	Generalize	Interactive	Exploration	Credit assignment
Supervised Learning	✓	✓			
Reinforcement Learning	✓	✓	✓	✓	✓

Table content based on slides from Emma Brunskill

The very successful stories of ML are based on RL…

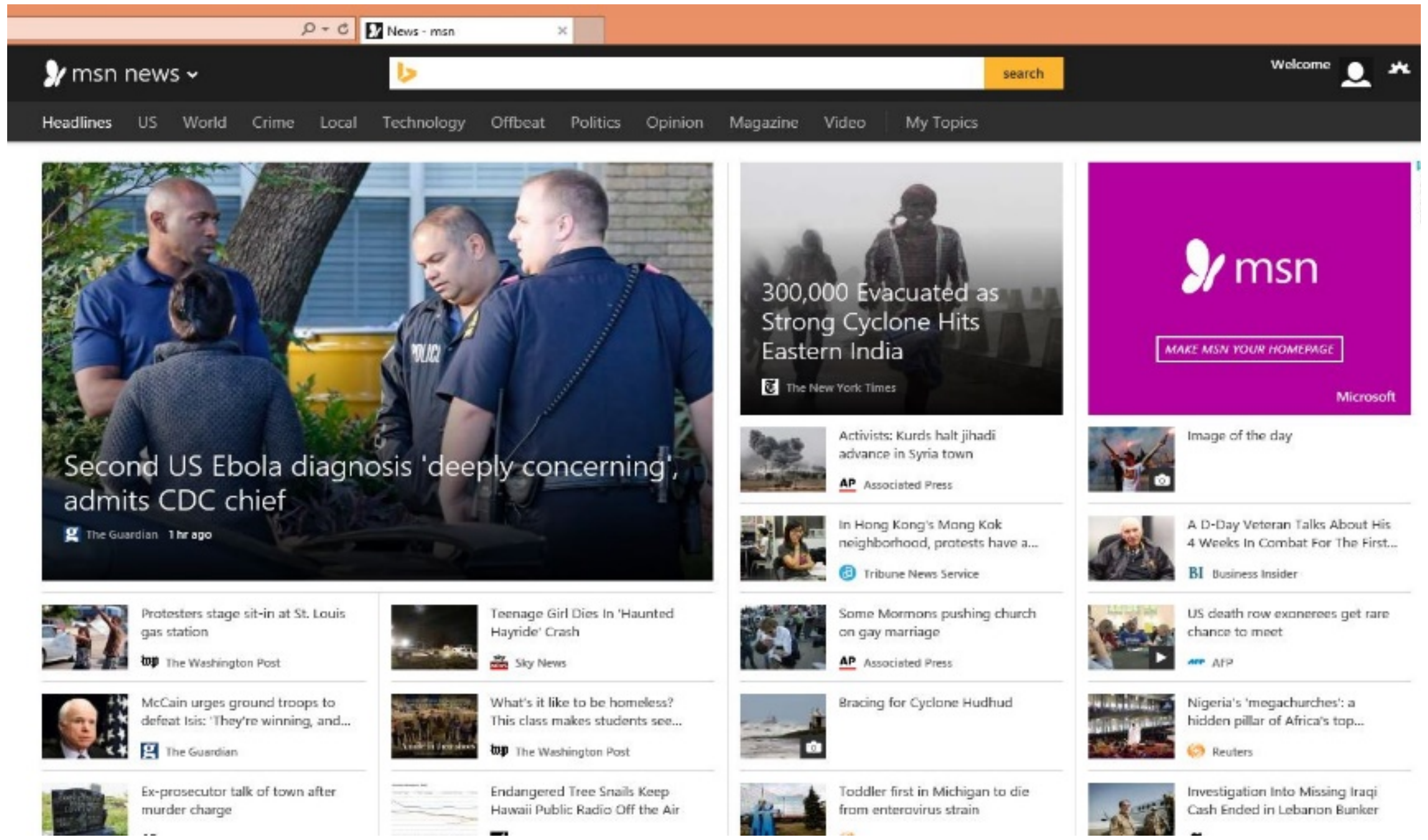


TD GAMMON [Tesauro 95]

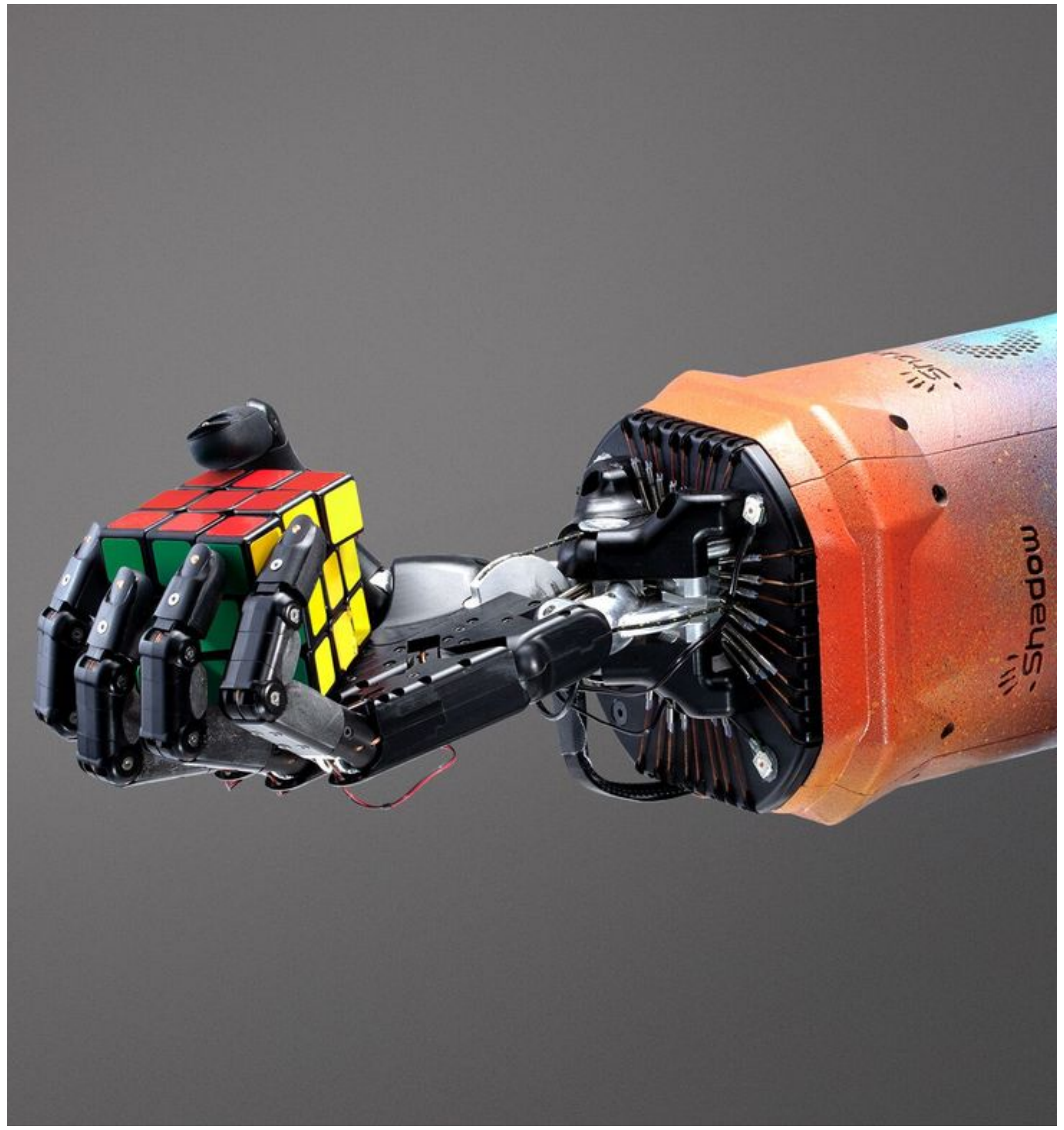
[AlphaZero, Silver et.al, 17]

[OpenAI Five, 18]

RL in Real World:



Recommendation systems



Robotics

RL in Real World:

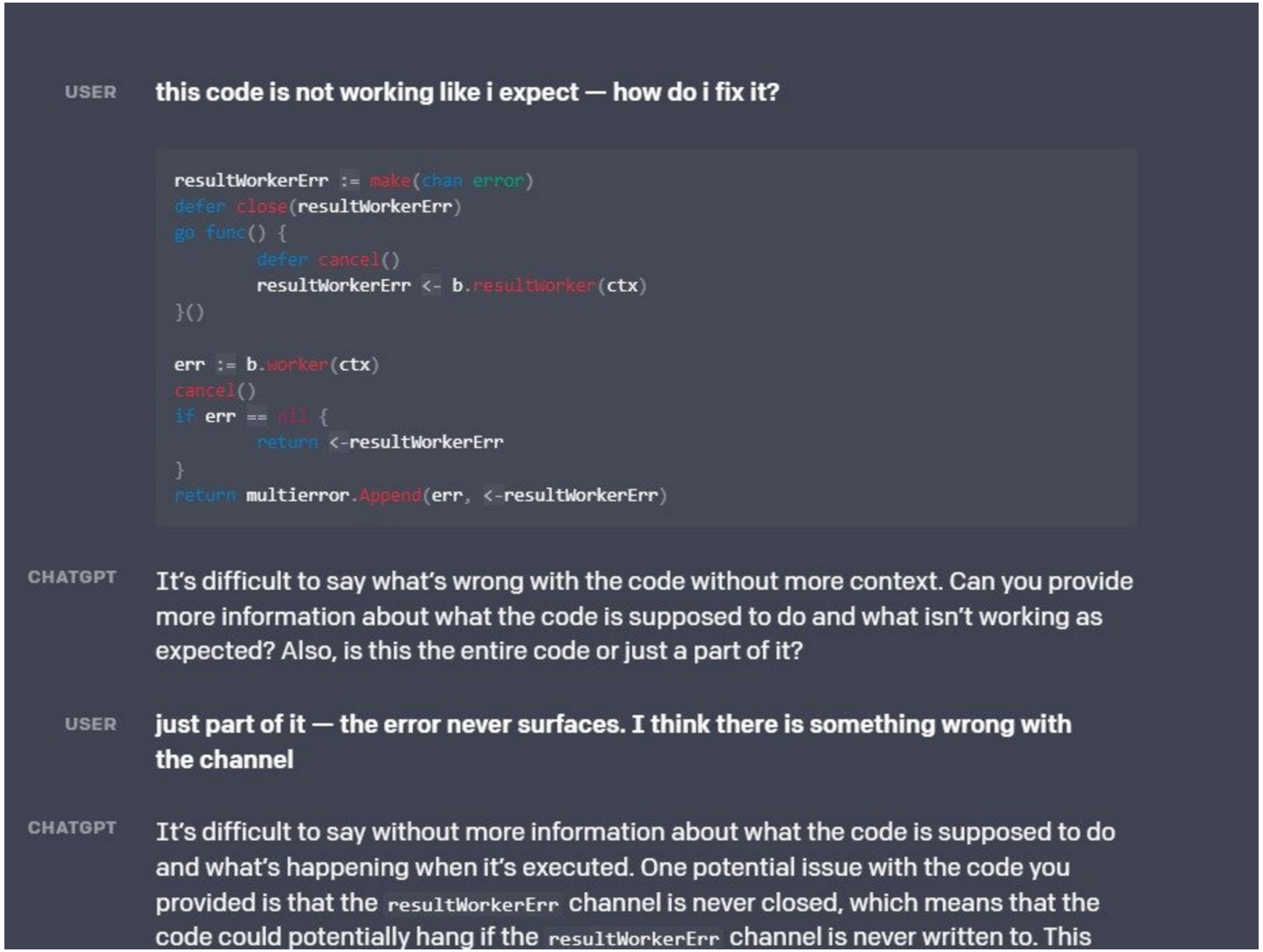
RL discovers new matrix multiplication algorithm

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} \end{pmatrix} \times \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} & b_{1,5} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} & b_{2,5} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} & b_{3,5} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} & b_{4,5} \\ b_{5,1} & b_{5,2} & b_{5,3} & b_{5,4} & b_{5,5} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} & c_{1,5} \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} & c_{2,5} \\ c_{3,1} & c_{3,2} & c_{3,3} & c_{3,4} & c_{3,5} \\ c_{4,1} & c_{4,2} & c_{4,3} & c_{4,4} & c_{4,5} \end{pmatrix}$$

Standard: 100 multiplications; reduce to 80 with human ingenuity;

AlphaTensor found a way with 76 multiplications

RL in the era of generative AI



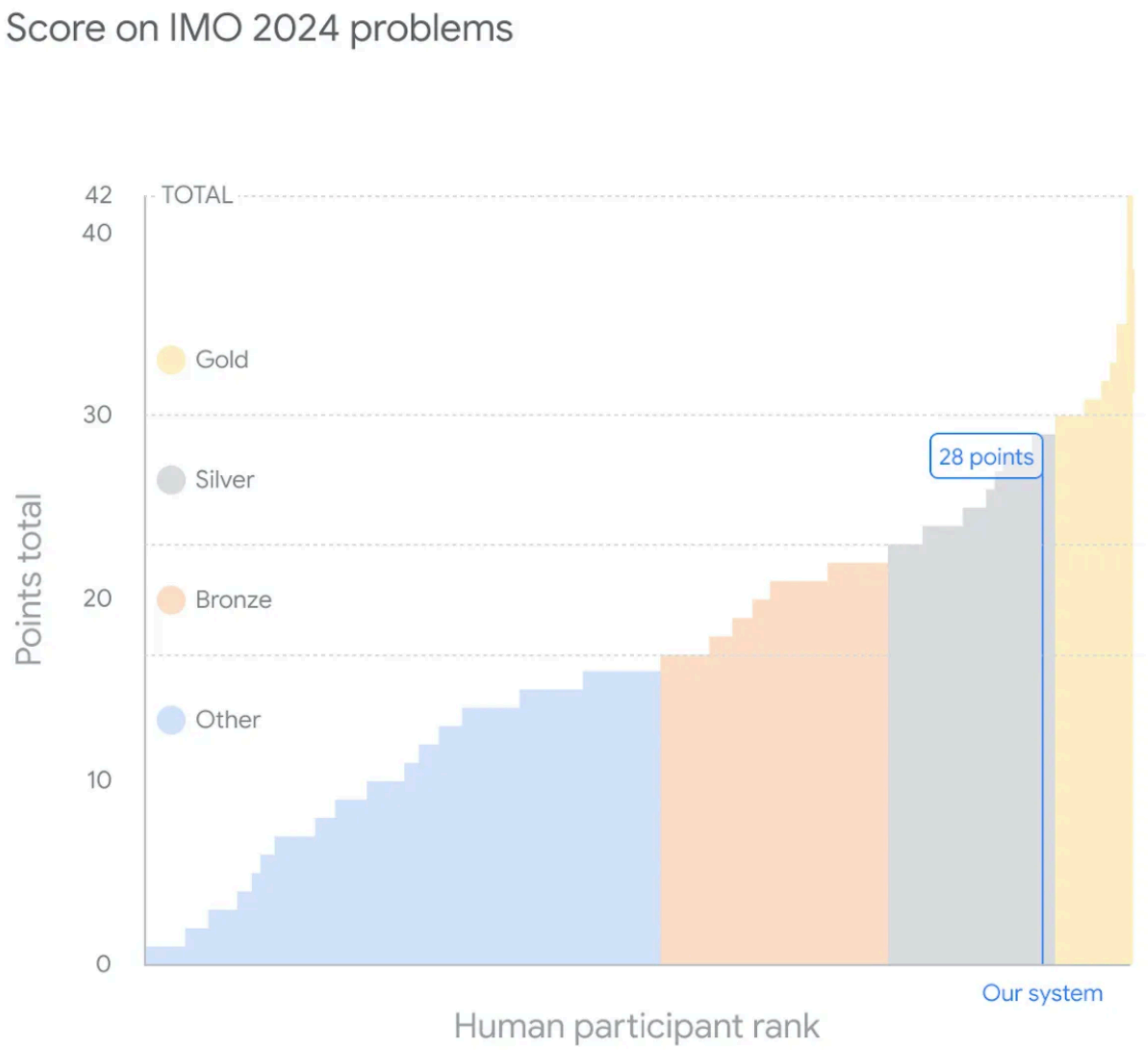
RL makes LLMs able to interact with humans better

RL in the era of generative AI

RL enables image generator to generate novel images

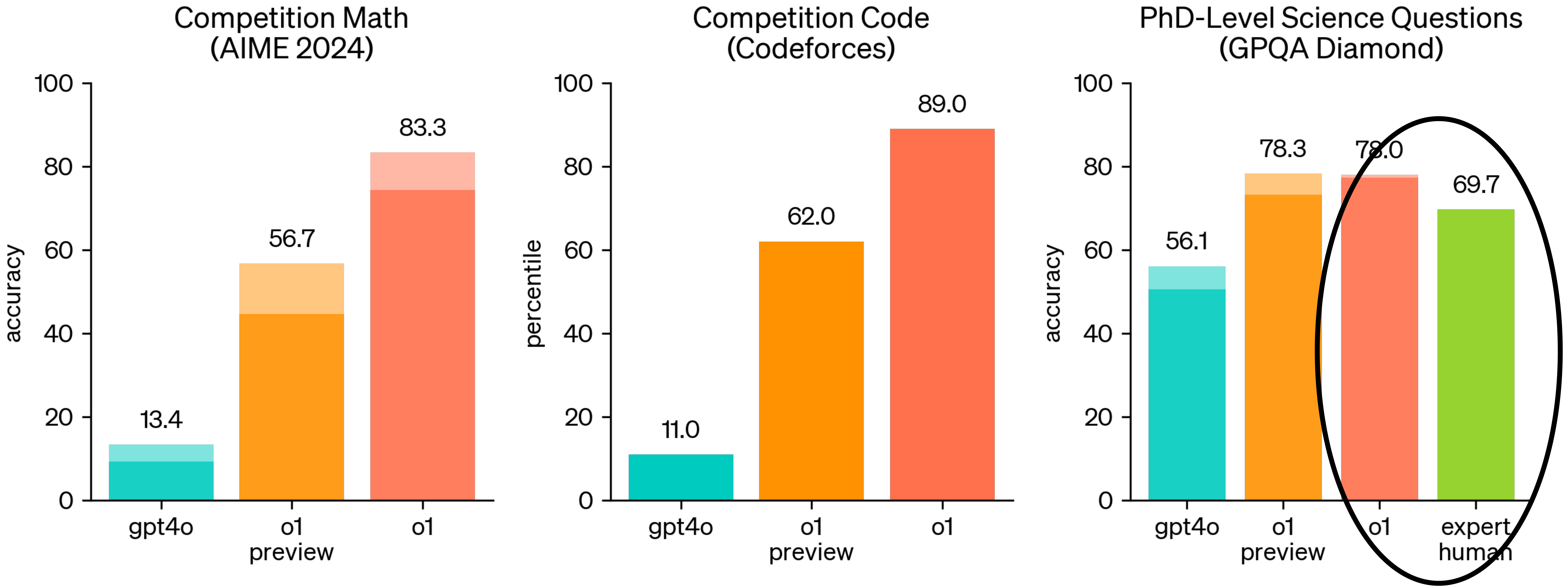
RL in the era of generative AI

RL enables LLMs to solve complicated math questions



RL in the era of generative AI

RL enables LLMs (openAI-o1) to reason



Better than human

At the end of this semester

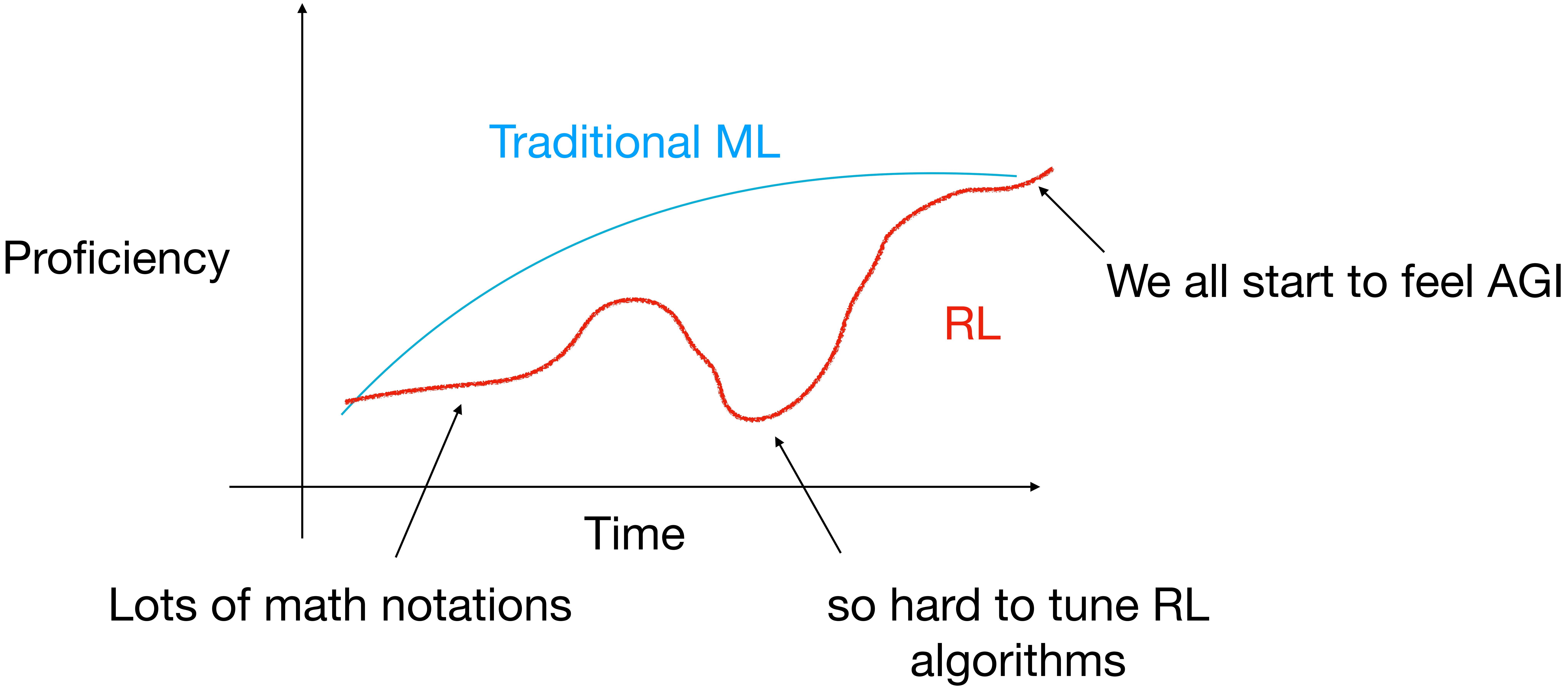
You will learn

The RL algorithms that power ChatGPT, AlphaGo, AlphaProof, etc

You will practice

these algorithms on simplified but interesting domains (robot simulation and video games)

But RL is much harder to learn than traditional ML



Summary

Traditional ML is very limited

RL is more suitable for most challenging real world problems (e.g., i.i.d assumption never holds in reality, we often have to perform sequential decisions)

RL is one of the very few approaches that train AI systems/agents to outperform humans