Heterogeneous Adversarial Play in Interactive Environments

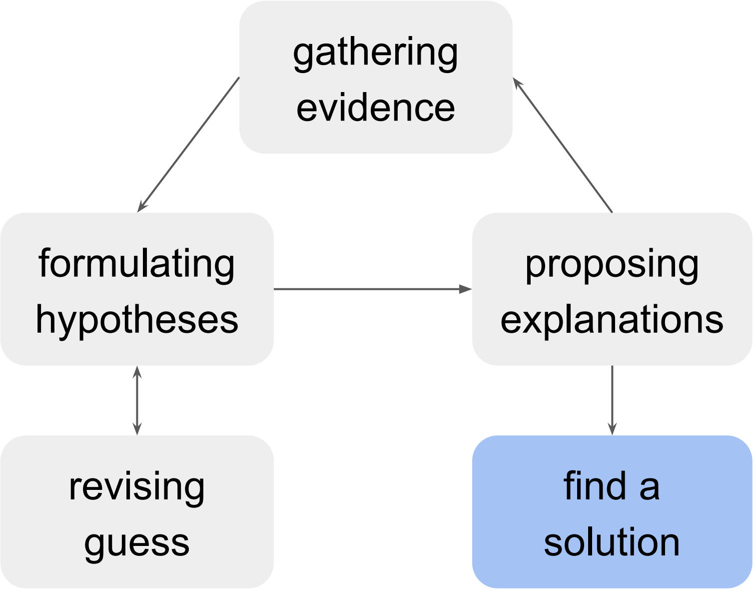
Manjie Xu, Xinyi Yang, Jiayu Zhan, Wei Liang, Chi Zhang, Yixin Zhu

Institute for Artificial Intelligence & School of Psychological and Cognitive Sciences
& State Key Lab of General AI
& Beijing Key Laboratory of Behavior and Mental Health, Peking University
School of Computer Science & Technology & Yangtze Delta Region Academy,
Beijing Institute of Technology
Embodied Intelligence Lab, PKU-Wuhan Institute for Artificial Intelligence

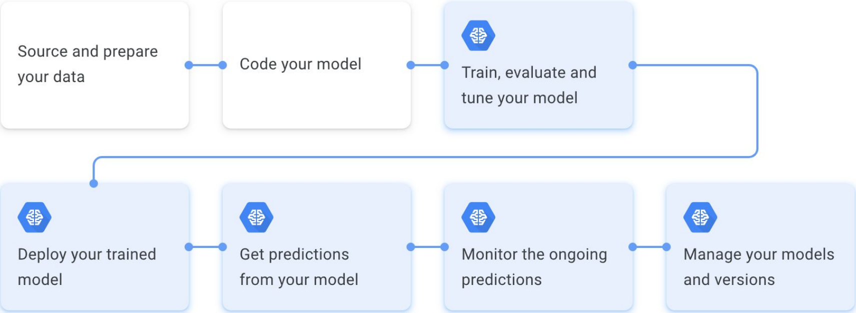
Previous Work

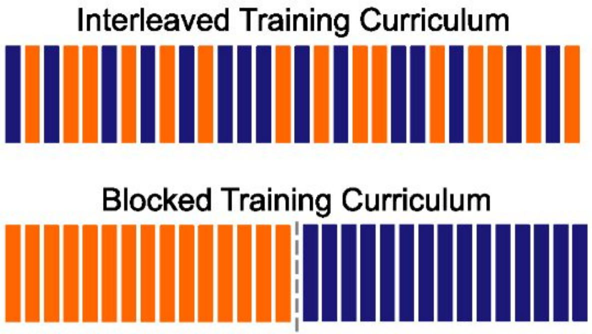
Learning by Actively Interacting with the World

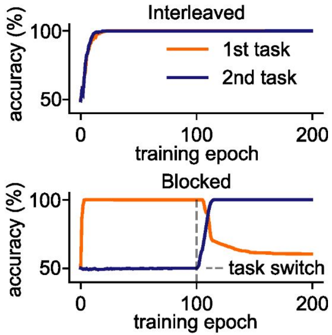
Lori Blahey / epl.ca



We cannot just iterate !







Humans, but not machines, seem to **benefit** from training regimes that blocked one task at a time, especially when they had a prior bias to **represent stimuli** in a way that encouraged **task separation**.

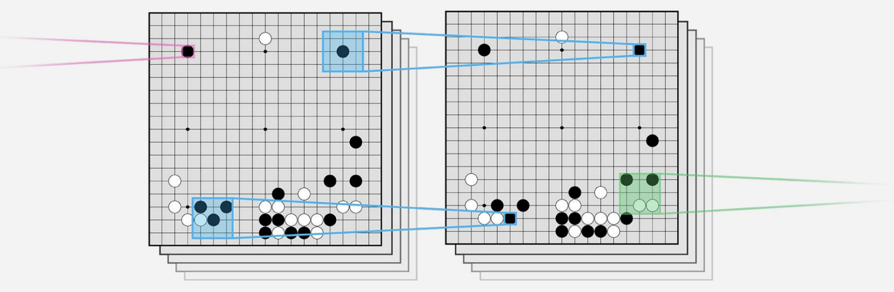
Self-Play
From AlphaGo

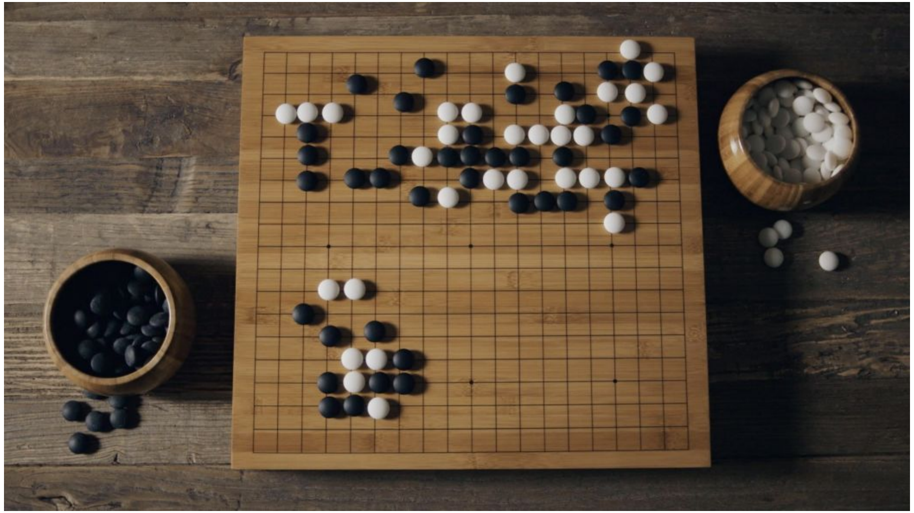
Why is Go hard for computers to play?

Game tree complexity = b^d

Brute force search intractable

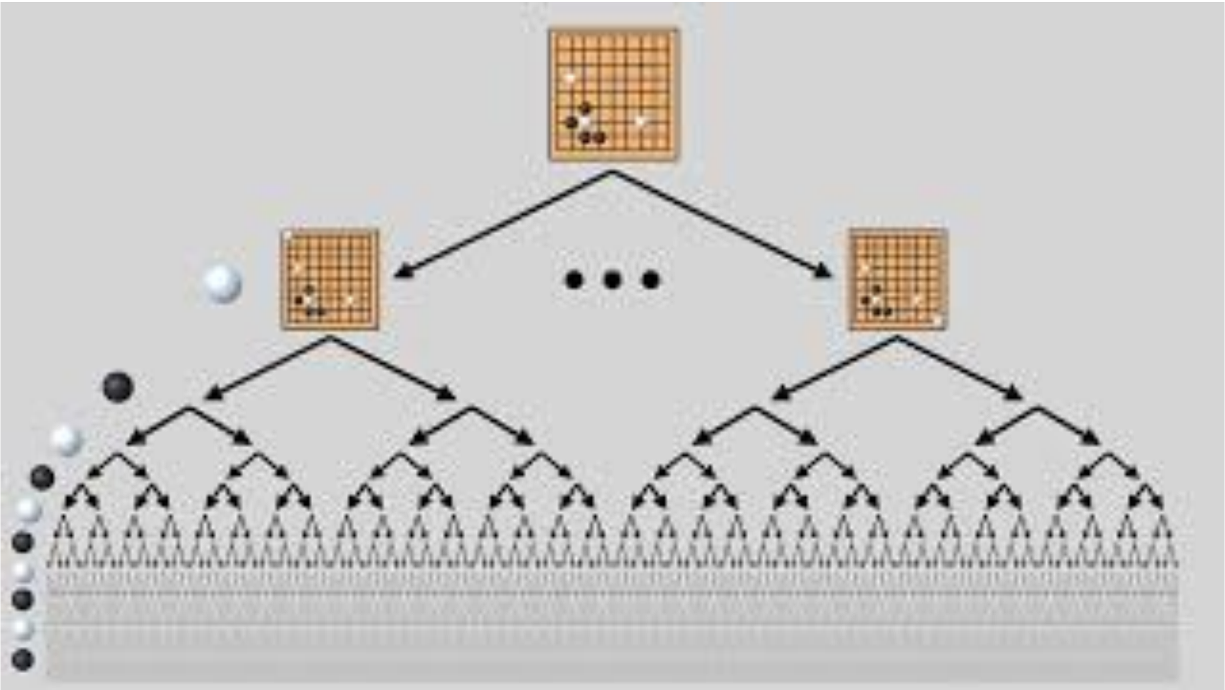
- Search space is huge
- "Impossible" for computers to evaluate who is winning





Source: AlphaGo-tutorial-slides (2016 ICML)

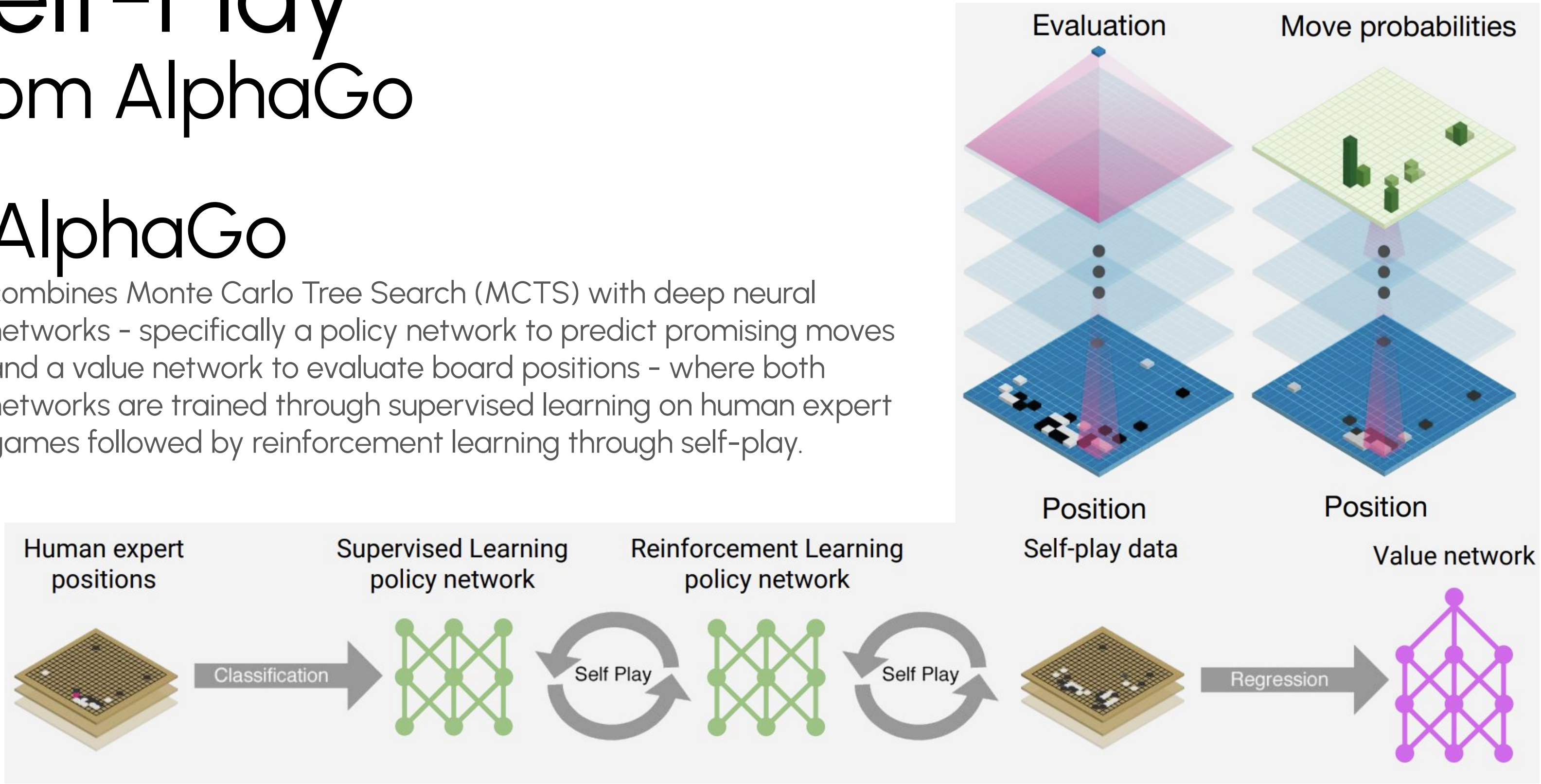
Self-Play
From AlphaGo



Self-Play
From AlphaGo

AlphaGo

combines Monte Carlo Tree Search (MCTS) with deep neural networks - specifically a policy network to predict promising moves and a value network to evaluate board positions - where both networks are trained through supervised learning on human expert games followed by reinforcement learning through self-play.



Source: AlphaGo-tutorial-slides (2016 ICML)

Self-Play
From AlphaGo

Supervised learning

Policy network: 12 layer convoluti

Training data: 30M positions from

Training algorithm: maximise likel

$$\Delta\sigma \propto \frac{\partial \log}{}$$

Training time: 4 weeks on 50 GPU

Results: 57% accuracy on held out test data (state-of-the art was 44%)

Google DeepMind

Reinforcement learni

Policy network: 12 layer convolution

Training data: games of self-play be

Training algorithm: maximise wins z

$$\Delta\sigma \propto \frac{\partial \log p_{}}{\partial}$$

Training time: 1 week on 50 GPUs u

Results: 80% vs supervised learning.

Google DeepMind

Reinforcement learning of value networks

Value network: 12 layer convolutional neural network

Training data: 30 million games of self-play

Training algorithm: minimise MSE by stochastic gradient descent

$$\Delta\theta \propto \frac{\partial v_\theta(s)}{\partial\theta}(z - v_\theta(s))$$

Training time: 1 week on 50 GPUs using Google Cloud

Results: First strong position evaluation function - previously thought impossible

Google DeepMind

Source: AlphaGo-tutorial-slides (2016 ICML)

Self-Play
Key Challenges

Challenge 1: Position Evaluation

- How do we know if the current state is good or bad?
- In Go: Who's winning in the middle of the game?
- No clear reward signal until the very end
- Traditional evaluation functions break down

Challenge 2: Curriculum Design

- What tasks should we practice next?
- How do we generate experiences that actually help learning?
- Too easy → no improvement
- Too hard → agent gets stuck or learns poorly
- Need the "Goldilocks zone" of difficulty

The Core Problem:

Self-play needs to solve both evaluation and curriculum generation simultaneously while the agent is still learning

Interactive Self-Play

Why Self-Play in Human/Machine Learning

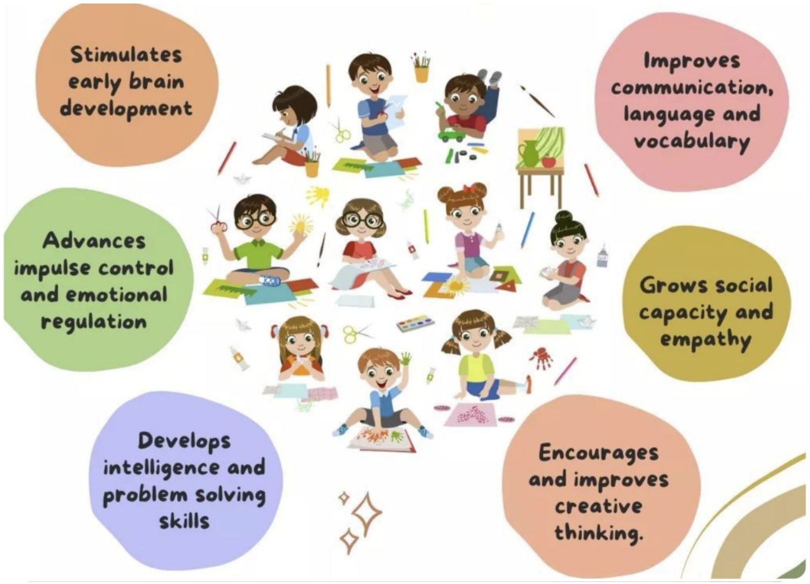
For human: foundation for learning

- Self-Play enables children to engage in **active hypothesis testing**, crucial for theory formation and causal learning
- Interactive play, especially with expert-designed curricula, scaffolds cognitive development
- Physical and social play enable **embodied learning**

For machines:

- Interactive agents that "play" can **autonomously generate learning examples**, refine its strategies, discover new behaviors, such providing rich, self-supervised learning signals that drive more effective learning.

Examples: Reinforcement Learning, Auto-Curriculum Learning



Source: Think Right

Interactive Self-Play

Why Self-Play in Human/Machine Learning

For human: Foundation for Learning

- Self-Play enables children to engage in **active hypothesis testing**, crucial for theory formation and causal learning
- Interactive play, especially with expert-designed curricula, scaffolds cognitive development
- Physical and social play enable **embodied learning**

For machines:

- Interactive agents that "play" can **autonomously generate learning examples**, refine its strategies, discover new behaviors, such providing rich, self-supervised learning signals that drive more effective learning.

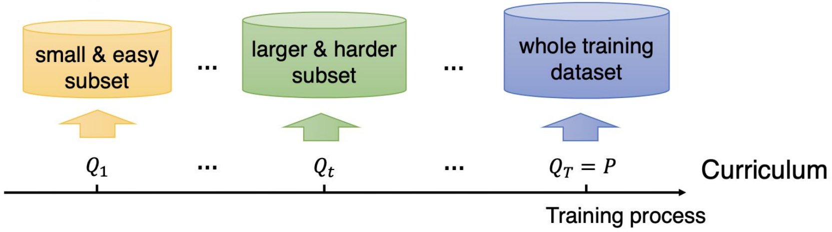
Examples: Reinforcement Learning, Auto-Curriculum Learning

Source: Think Right

Why Curriculum

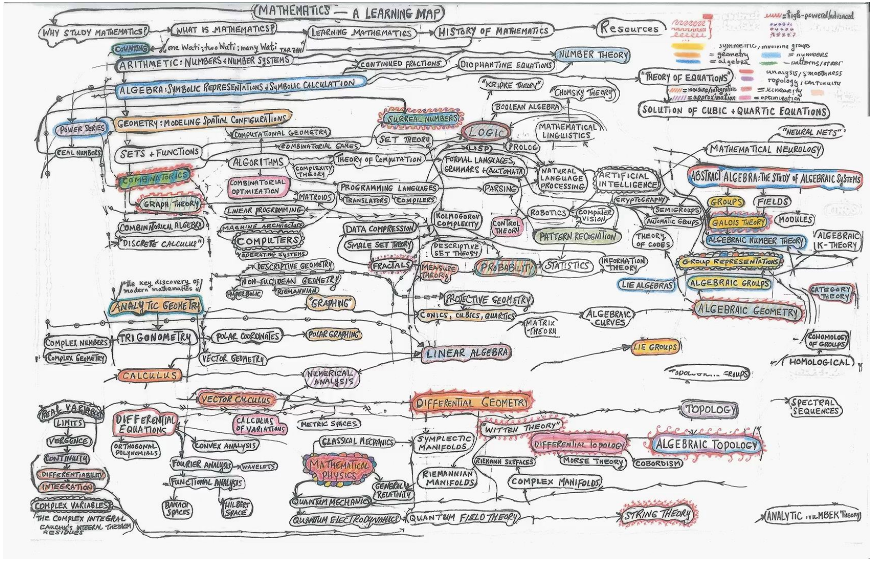
Learn Better when Experiences Build Progressively

- Humans and machines **learn faster** when experiences build progressively.
- Curricula help learners form **more abstract and transferable representations**, and gradual scaffolding **avoids cognitive overload** and keeps learners **motivated**
- Curriculum reduces the risk of **getting stuck in poor solutions (local minima)** by guiding learning through progressively harder tasks.



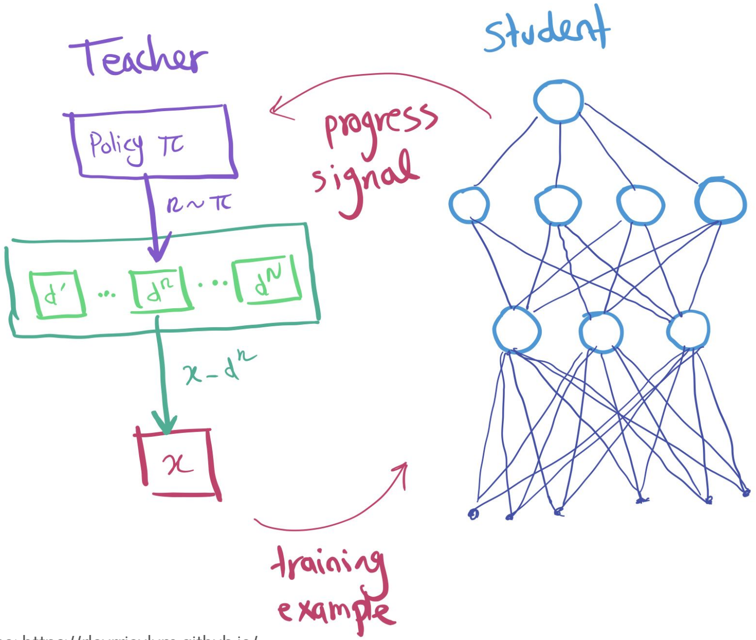
An example: Data Curriculum

Source: kitzorz



An example: You can never learn Math from Calculus

Auto-Curriculum
Curriculum Learning in Interactive Environments



Source: https://rlcurriculum.github.io/

An intuitive idea:
An expert defines a fixed curriculum for training

- Agents are not ideal students—they may learn at different paces, encounter unique challenges, and their capabilities can evolve unpredictably

Another intuitive idea:
Consider a teacher-student setup in which the teacher dynamically manages the learning process

- On what basis should the teacher evaluate and update the learning process?

Auto-Curriculum

Curriculum Learning in Interactive Environments

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

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

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

$$\hat{r}(t) = \frac{r(t)}{P(\text{pick task } k)}$$

Source: https://rlcurriculum.github.io/

A simple soluiton: Multi-armed bandit

An expert defines a fixed curriculum for training

- The idea is to have one "arm" per task, and try to find which tasks has the highest reward. In our case, reward really means "student progress".

Auto-Curriculum
Curriculum Learning in Interactive Environments

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

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

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

$$\hat{r}(t) = \frac{r(t)}{P(\text{pick task } k)}$$

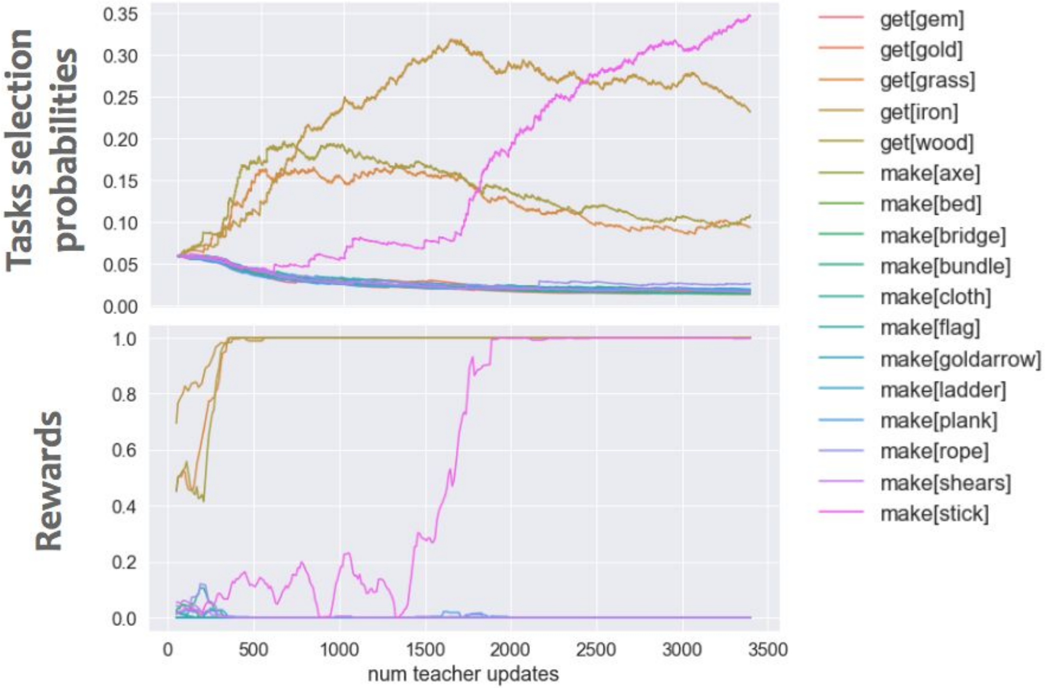
Source: https://rlcurriculum.github.io/

A simple solution: Multi-armed bandit
An expert defines a fixed curriculum for training

- The idea is to have one "arm" per task, and try to find which tasks has the highest reward. In this case, reward really means "student progress" (singnal).

Progress Signal	Definition
Prediction gain (PG)	$\nu_{PG} := L(x,\theta) - L(x,\theta')$
Gradient prediction gain (GPG)	$\nu_{GPG} := \lVert\nabla L(x,\theta)\rVert_2^2$
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)	$\nu_{GVCG} := [\nabla_{\phi,\psi} KL(P_\phi \Vert Q_\psi)]^\top \nabla_\phi \mathbb{E}_{\theta\sim P_\phi} L(x,\theta)$
L2 gain (L2G)	$L_{L2}(x,\theta) = L(x,\theta) + \frac{\alpha}{2}\lVert\theta\rVert_2^2$

Auto-Curriculum
Curriculum Learning in Interactive Environments

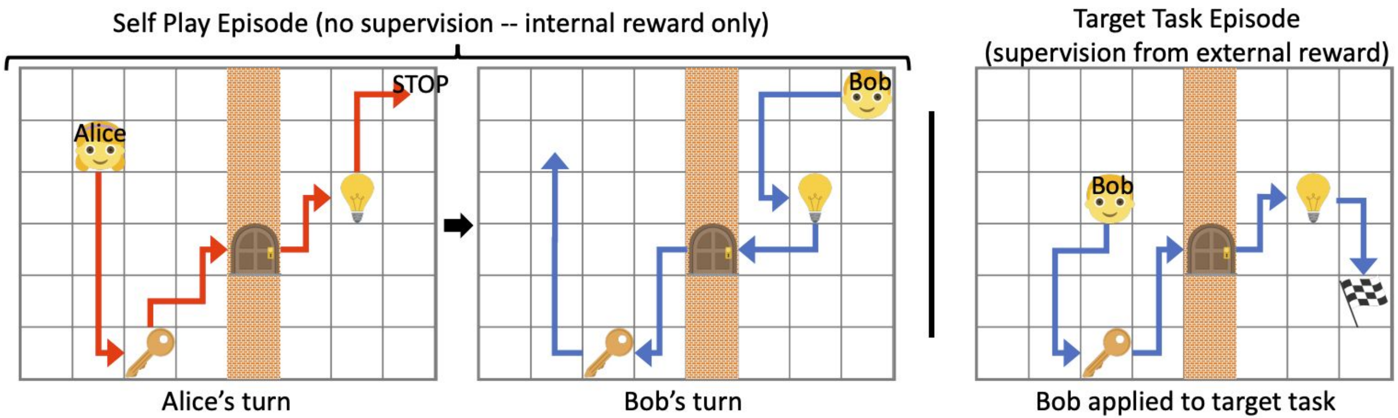


Source: https://rlcurriculum.github.io/

Auto-Curriculum
The Asymmetric Self-Play Idea

Another idea:
Two versions of the same agent:

- Alice will "propose" the task by doing a sequence of actions
- Bob must undo or repeat them, respectively.



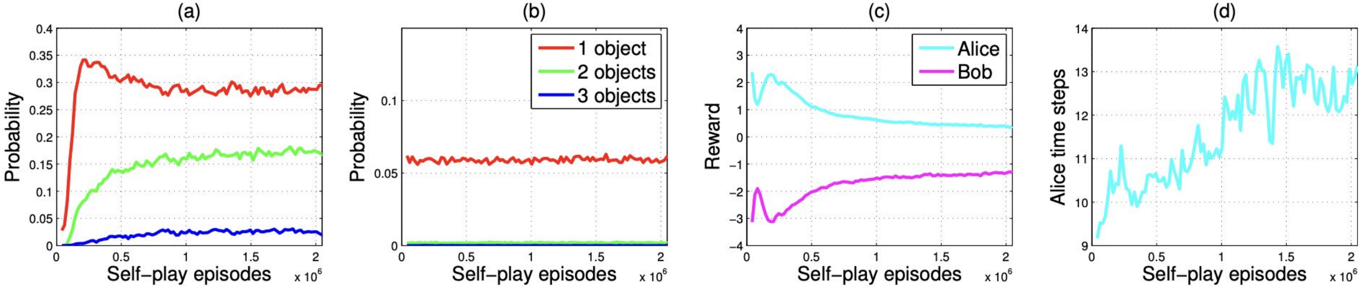
Source: Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play, ICLR.

Auto-Curriculum

Automatically Generating Curriculum

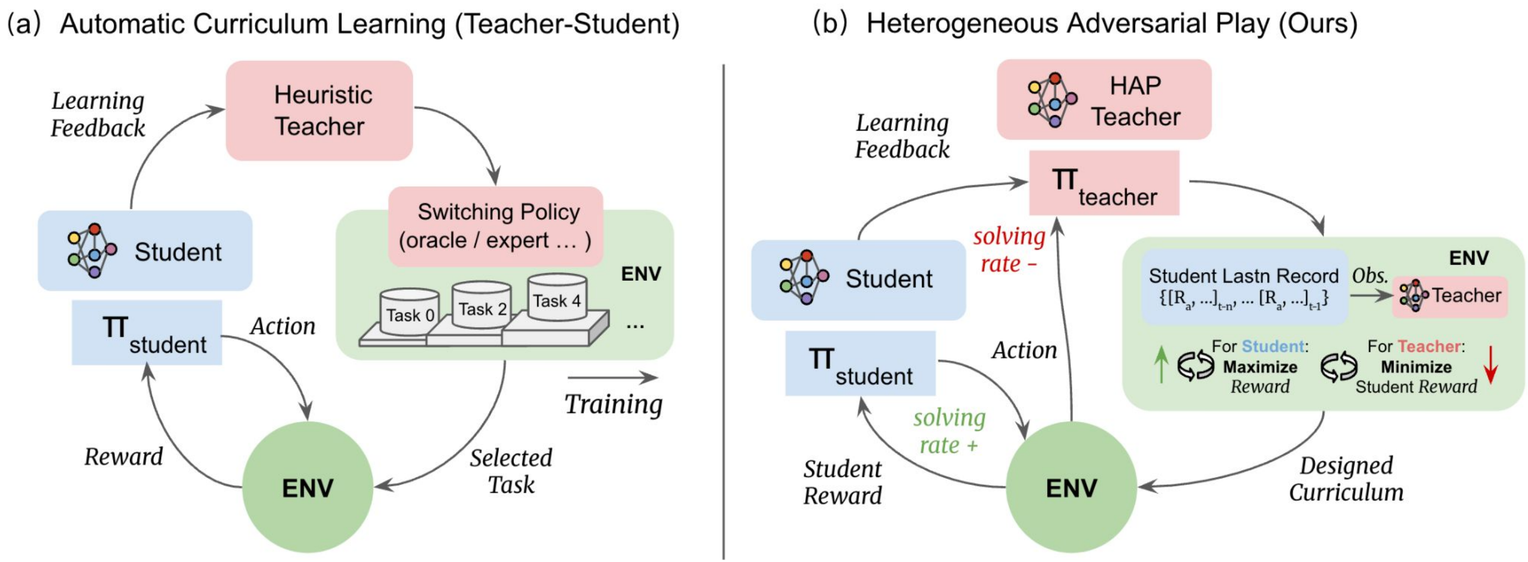
Algorithm 1 Pseudo code for training an agent on a self-play episode

function SELFPLAYEPISODE(REVERSE/REPEAT, $t_{\text{MAX}}, \theta_A, \theta_B$)
  $t_A \leftarrow 0$
  $s_0 \leftarrow$ env.observe()
  $s^* \leftarrow s_0$
  while True **do**
    # Alice's turn
    $t_A \leftarrow t_A + 1$
    $s \leftarrow$ env.observe()
    $a \leftarrow \pi_A(s, s_0) = f(s, s_0, \theta_A)$
    if $a =$ STOP **or** $t_A \geq t_{\text{Max}}$ **then**
      $s^* \leftarrow s$
      env.reset()
      break
    env.act(a)
  $t_B \leftarrow 0$
  while True **do**
    # Bob's turn
    $s \leftarrow$ env.observe()
    if $s = s^*$ **or** $t_A + t_B \geq t_{\text{Max}}$ **then**
      break
    $t_B \leftarrow t_B + 1$
    $a \leftarrow \pi_B(s, s^*) = f(s, s^*, \theta_B)$
    env.act(a)
  $R_A \leftarrow \gamma \max(0, t_B - t_A)$
  $R_B \leftarrow -\gamma t_B$
  policy.update(R_A, θ_A)
  policy.update(R_B, θ_B)
  return



Source: Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play, ICLR.

Heterogeneous Adversarial Play
Teacher-Student Interplay as Adversarial Game



At the simplest level: the task generator is rewarded if the problem solver **cannot** solve the problem, and the problem solver is so if the proposed challenge is **addressed**.

Heterogeneous Adversarial Play

Teacher-Student Interplay as Adversarial Game

Consider the task space $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$, where each C_i is a unique task.

The student:

- Student agent learns a solve policy $\pi(a|s, C; \theta)$ in environment $\mathcal{E}$:

$$\max_{\theta} \; J_{\text{student}}(\theta) = \mathbb{E}_{C \sim p_\phi(C)} \left[\mathbb{E}_{\tau \sim \pi(\cdot|C;\theta)} \left[R(\tau; C) \right] \right]$$

The teacher:

- The teacher agent evaluates the student's current learning state and strategically selects appropriate tasks.

$$\max_{\phi} \; J_{\text{teacher}}(\phi) = \mathbb{E}_{C \sim p_\phi(C)} \left[\mathbb{E}_{\tau \sim \pi(\cdot|C;\theta)} \left[-R(\tau; C) \right] \right]$$

Heterogeneous Adversarial Play

Adversarial Formulation

Algorithm 1: Heterogeneous Adversarial Play (HAP) Training Loop

Data: Initial θ, ϕ; learning rates α, β

while *not converged* **do**

  ; /* 1. Teacher's Adversarial Task Selection: */

  Generate task distribution: $p_\phi(C) \propto \exp(\phi)$;

  ; /* Minimization strategy: Sample task $C \sim p_\phi(C)$ to challenge current π */

  ; /* 2. Student's Policy Maximization: */

  Execute $\pi(a|s, C; \theta)$, collect trajectory τ;

  Compute reward signal: $R(\tau; C) = \sum_{t=0}^{H} \gamma^t r_t$;

  Update θ to *maximize* returns:;

    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathbb{E}_\tau [R(\tau; C)]$;

  ; /* 3. Teacher's Adversarial Update: */

  Update ϕ to *minimize* student success:;

    $\phi \leftarrow \phi - \beta \nabla_\phi \mathbb{E}_C \left[R(\tau; C)\right]$;

  where $\nabla_\phi J_{\text{teacher}} = -\mathbb{E}_C \left[\nabla_\phi \log p_\phi(C) \cdot R(\tau; C)\right]$;

end

Zero-sum game between the teacher and student can also be modeled as a **minimax game** for ease of implementation:

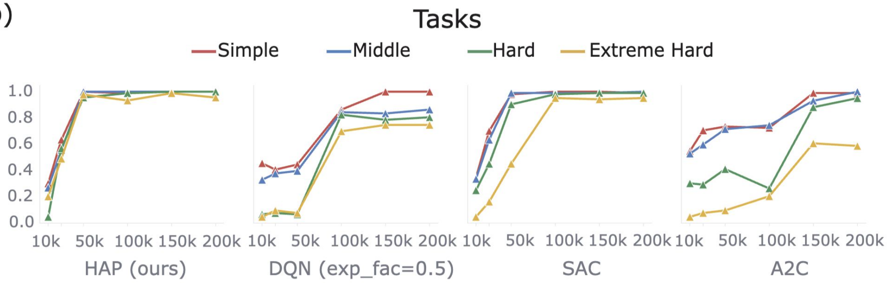
$$\min_\phi \max_\theta \; J(\theta, \phi)$$

Heterogeneous Adversarial Play
A Preliminary Study

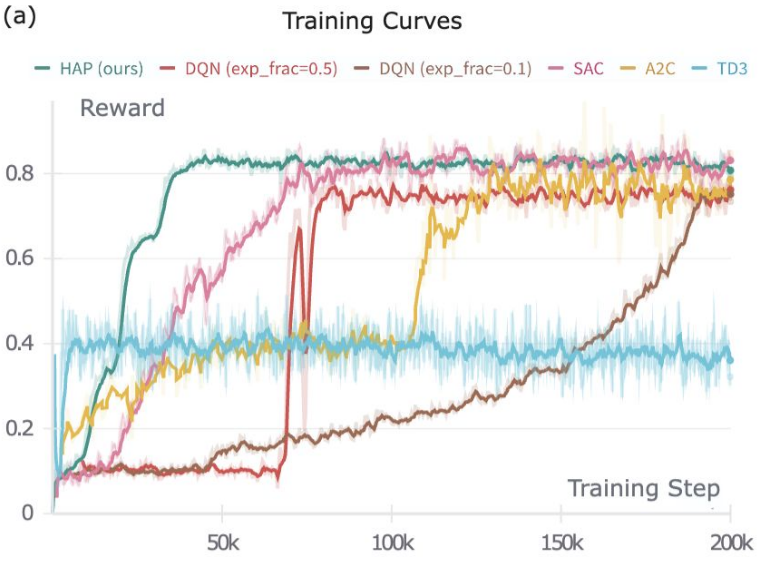
We discover the advantages of HAP through an intuitive navigation experiment.

- Simple: 2 grids
- Middle: 4 grids
- Hard: 8 grids
- Extreme Hard: 16 grids

(b)



(a)

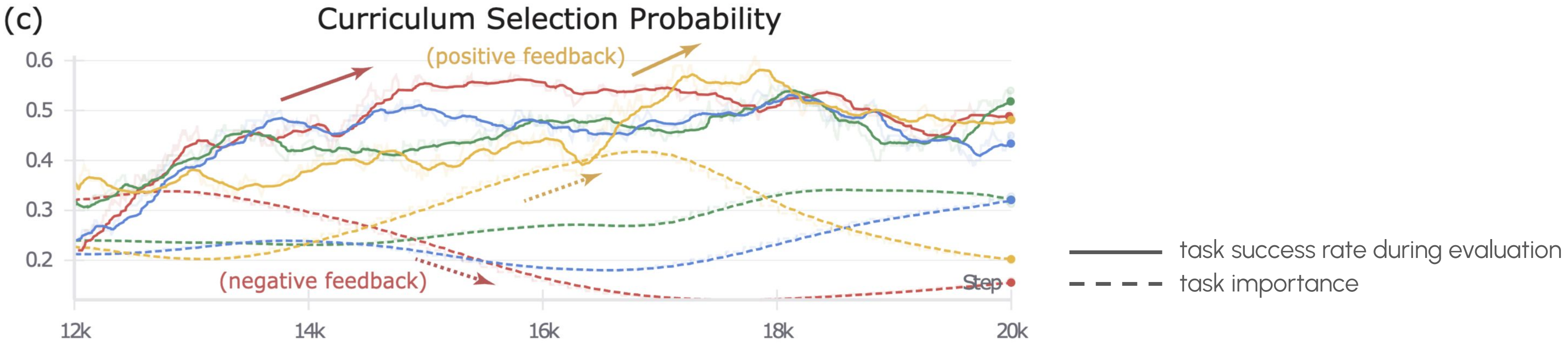


Heterogeneous Adversarial Play

A Preliminary Study

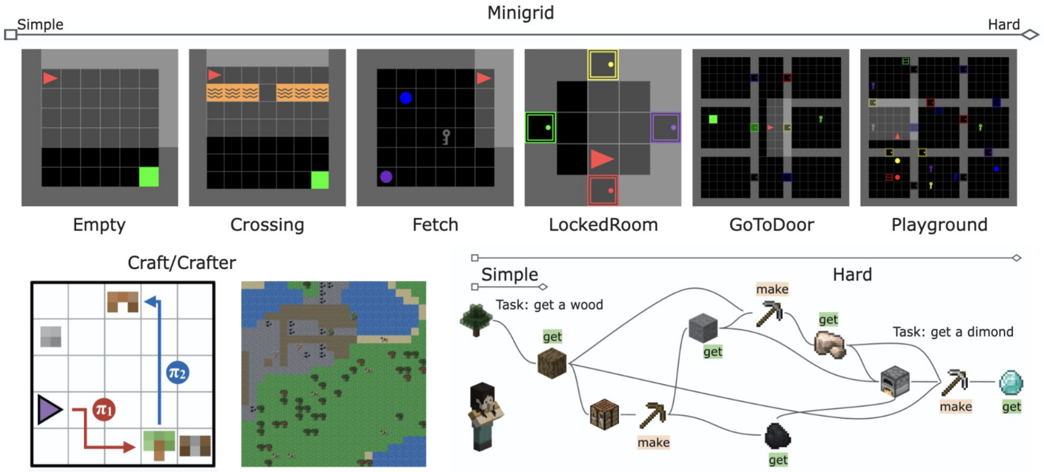
HAP benefits from two key design elements:

- (i) a positive feedback loop, where the teacher increases the sampling probability of a task the learner fails often, thereby accelerating the learner's acquisition in a specific skill
- (ii) a negative feedback mechanism, which lowers the sampling probability for tasks the learner has already mastered.



Heterogeneous Adversarial Play
Testing in Complex Multi-task Scenes

- Top: Minigrid environment with six selected tasks.
- Bottom Left: CRAFT and Crafter environments
- Bottom Right: A portion of the task dependency graph. Starting from the root node, any path defines a multi-step task that an agent must complete when interacting with the environment.



Heterogeneous Adversarial Play

Testing in Complex Multi-task Scenes

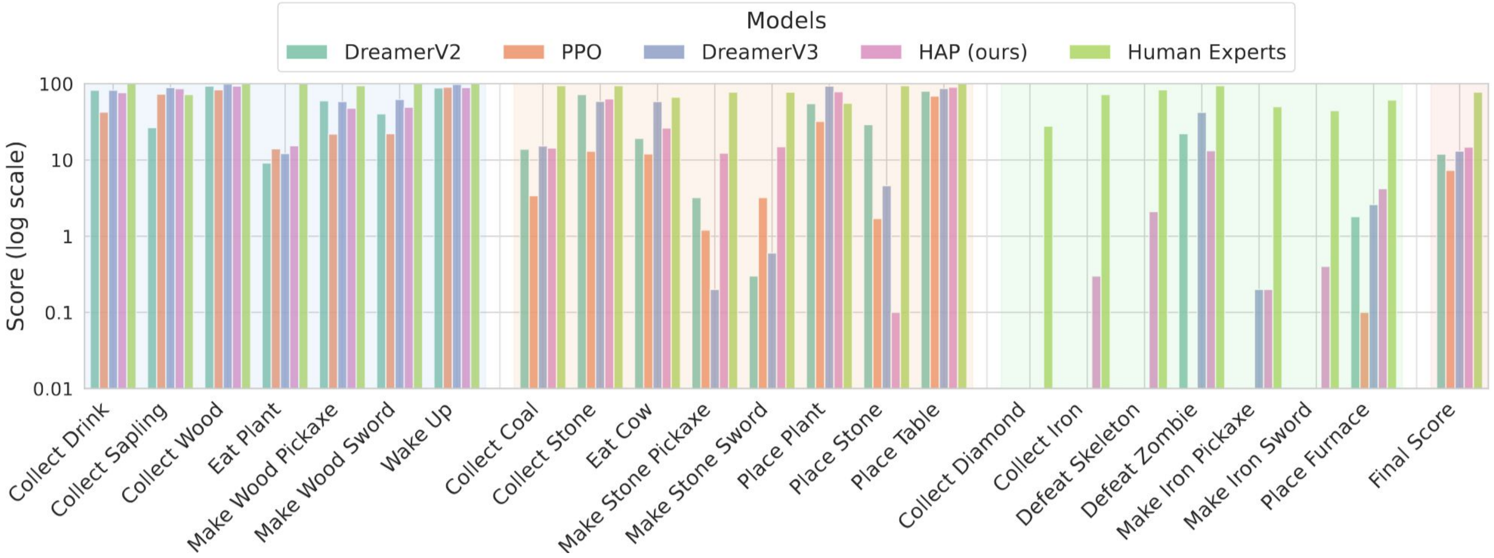
From Results: An active curriculum not only promotes **faster convergence** but also **bolsters stability** across tasks.

Env	DQN	A2C	PPO	SAC	TD3	DreamerV3	TSCL	EXP3	HAP	Human
	Algorithms									
Minigrid										
Easy	**0.98**	0.94	0.88	0.97	0.95	0.96	0.96	0.97	0.92	1.00
Middle	0.24	0.25	0.22	0.27	0.26	0.34	0.21	0.24	**0.46**	0.78
Hard	0.00	0.00	0.00	0.13	0.08	0.18	0.16	0.18	**0.20**	0.46
General	0.407	0.397	0.367	0.457	0.43	0.493	0.443	0.463	**0.527**	0.747
CRAFT										
Easy	0.78	0.84	0.87	0.87	0.86	0.89	**0.94**	0.91	0.88	0.94
Middle	0.26	0.45	0.48	0.42	0.42	0.55	0.24	0.56	**0.63**	0.86
Hard	0.02	0.14	0.12	0.15	0.14	0.27	0.03	0.24	**0.31**	0.66
General	0.278	0.415	0.426	0.413	0.407	0.516	0.307	0.513	0.562	0.802
Crafter										
Easy	0.61	0.79	**0.94**	0.91	0.84	0.91	0.82	0.87	0.91	0.99
Middle	0.28	0.37	0.67	0.47	0.39	0.66	0.45	0.58	**0.68**	0.82
Hard	0.00	0.00	0.47	0.22	0.29	0.52	0.00	0.02	**0.58**	0.74
General	0.297	0.387	0.693	0.533	0.507	0.697	0.423	0.49	**0.723**	0.85

Heterogeneous Adversarial Play

Qualitative Analysis on Crafter Scores

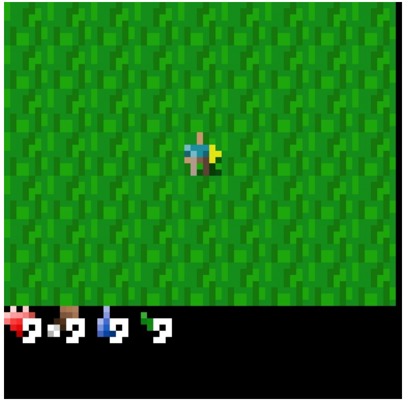
From Results: An active curriculum not only promotes **faster convergence** but also **bolsters stability** across tasks.



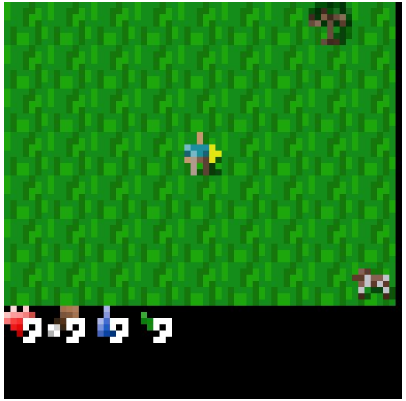
Heterogeneous Adversarial Play

Demos

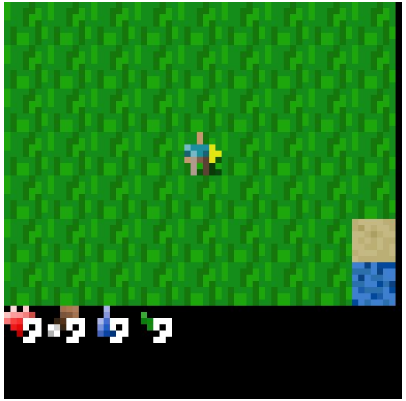
From Results: An active curriculum not only promotes **faster convergence** but also **bolsters stability** across tasks.



Reward (achievement) : 13



Reward: 15



Reward: 5

Reward: 3

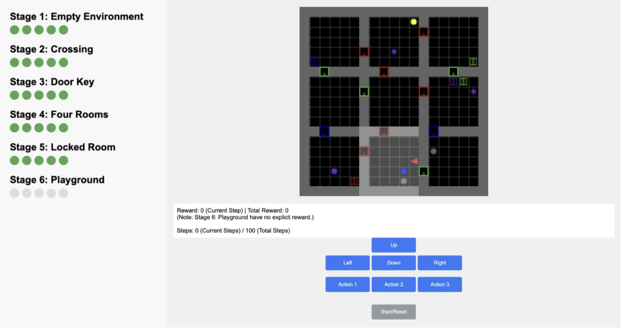
Good cases

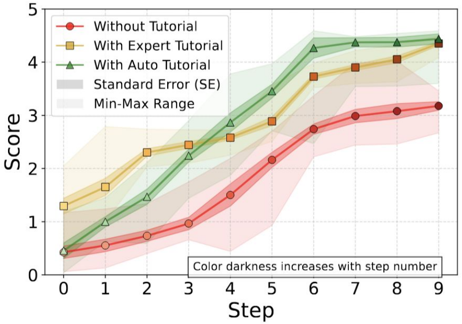
Bad cases

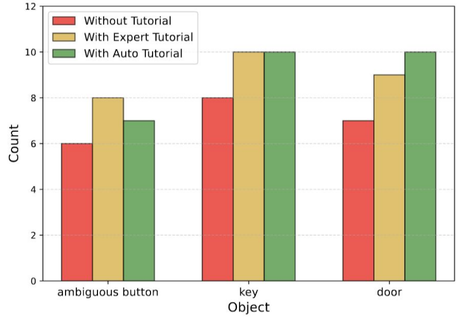
Heterogeneous Adversarial Play
Validation on Human Study

We conduct a human study to compare the curriculum proposed by the teacher model of HAP against expert-designed pedagogical strategies from human subjects. We find that:

- The presence of well-designed curriculum significantly accelerated early-stage mastery for both humans and models.
- While humans showed greater improvement within a single step when provided with expert step-by-step tutorials, the HAP framework offered more flexible, adaptive curricula tailored to individual behaviors.

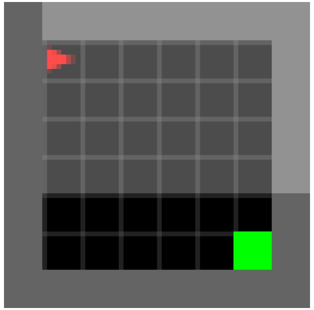


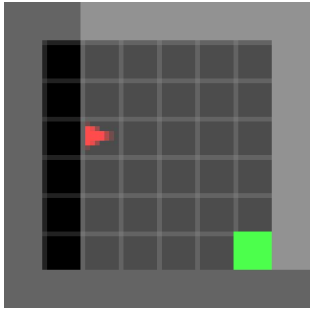




Heterogeneous Adversarial Play

Demos

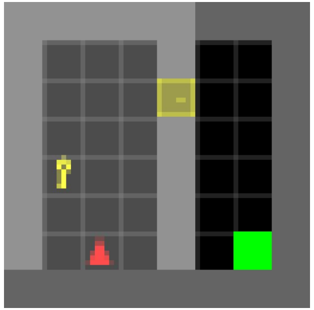




Bad cases

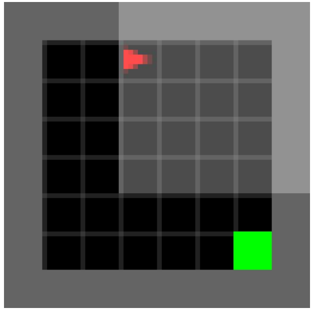
(Open the Door /

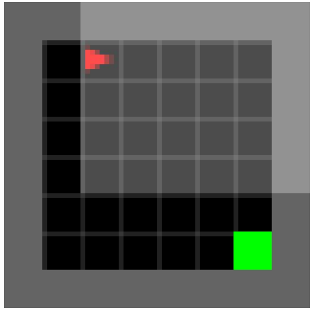
Use key)

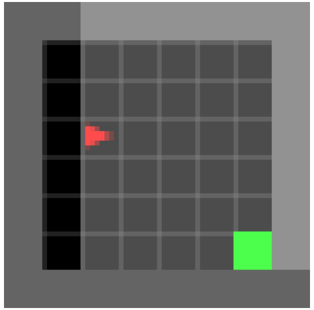


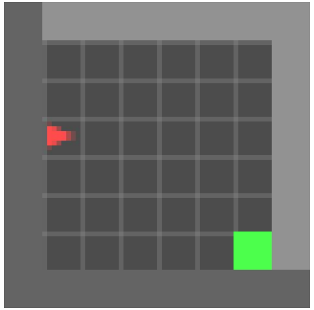
Curriculum

Generated by HAP





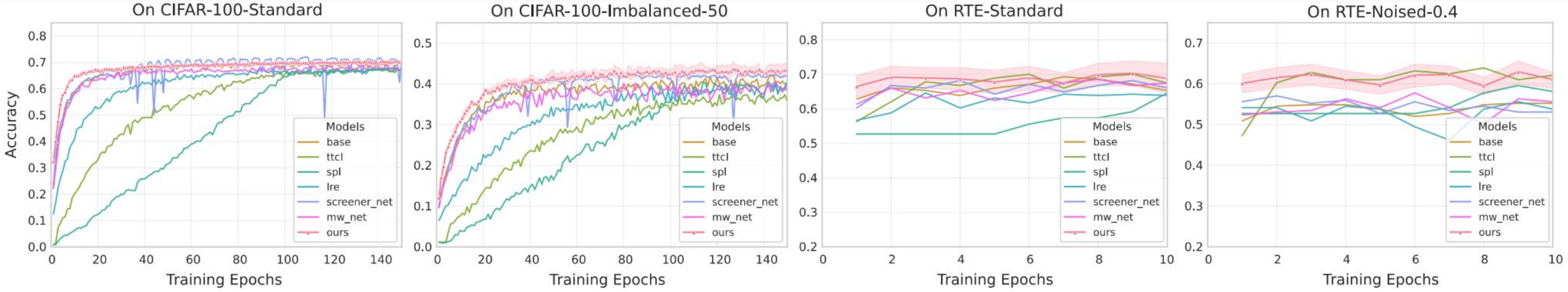




Good cases

Heterogeneous Adversarial Play
Further Study on Self-Supervised Learning

From Results: An active curriculum not only promotes **faster convergence** but also **bolsters stability** across tasks, **especially in harder settings**.



Heterogeneous Adversarial Play
Summary

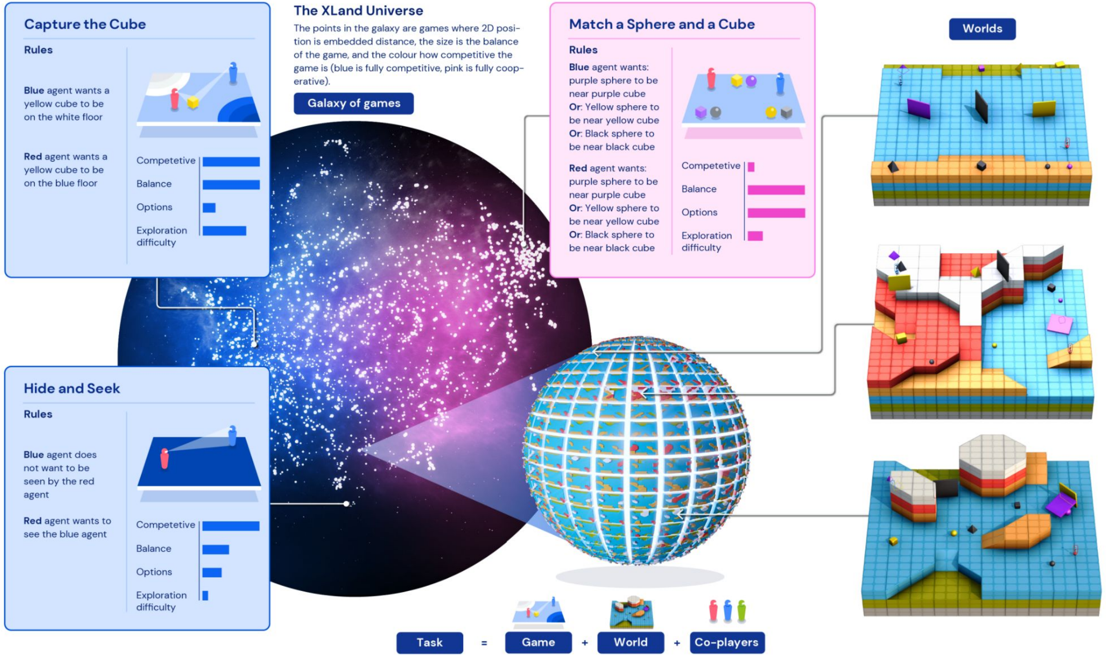
- We propose Heterogeneous Adversarial Play, a zero-sum game framework where distinct teacher and student networks co-adapt dynamically, enabling automated curriculum design without predefined task hierarchies or symmetric architectures.
- Our method achieves superior performance in complex, multi-task environments and supervised learning benchmarks, demonstrating faster convergence and robustness to noisy/imbalanced data compared to state-of-the-art baselines.
- HAP's emergent curricula mirror human pedagogical strategies—such as revisiting foundational skills during plateaus and scaling difficulty contextually—while offering personalized adaptation that surpasses fixed, expert-designed teaching approaches.

Heterogeneous Adversarial Play
Future Extension

- Open-ended Play
- Synthetic Data Generation via Self-play

Open-ended Play

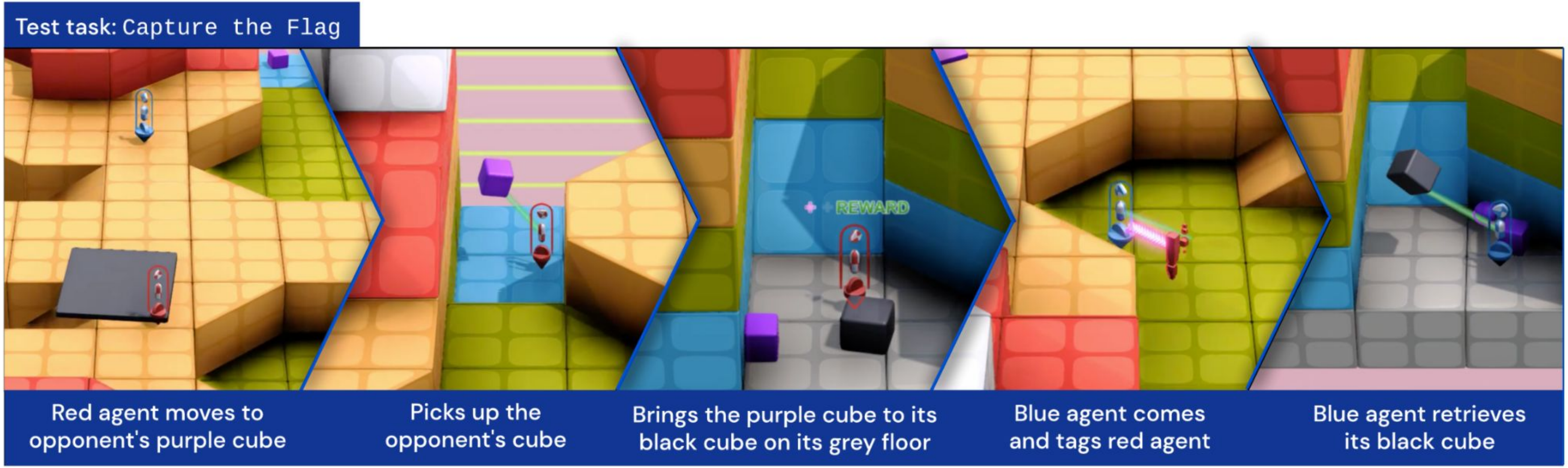
Generally capable agents emerge from open-ended play



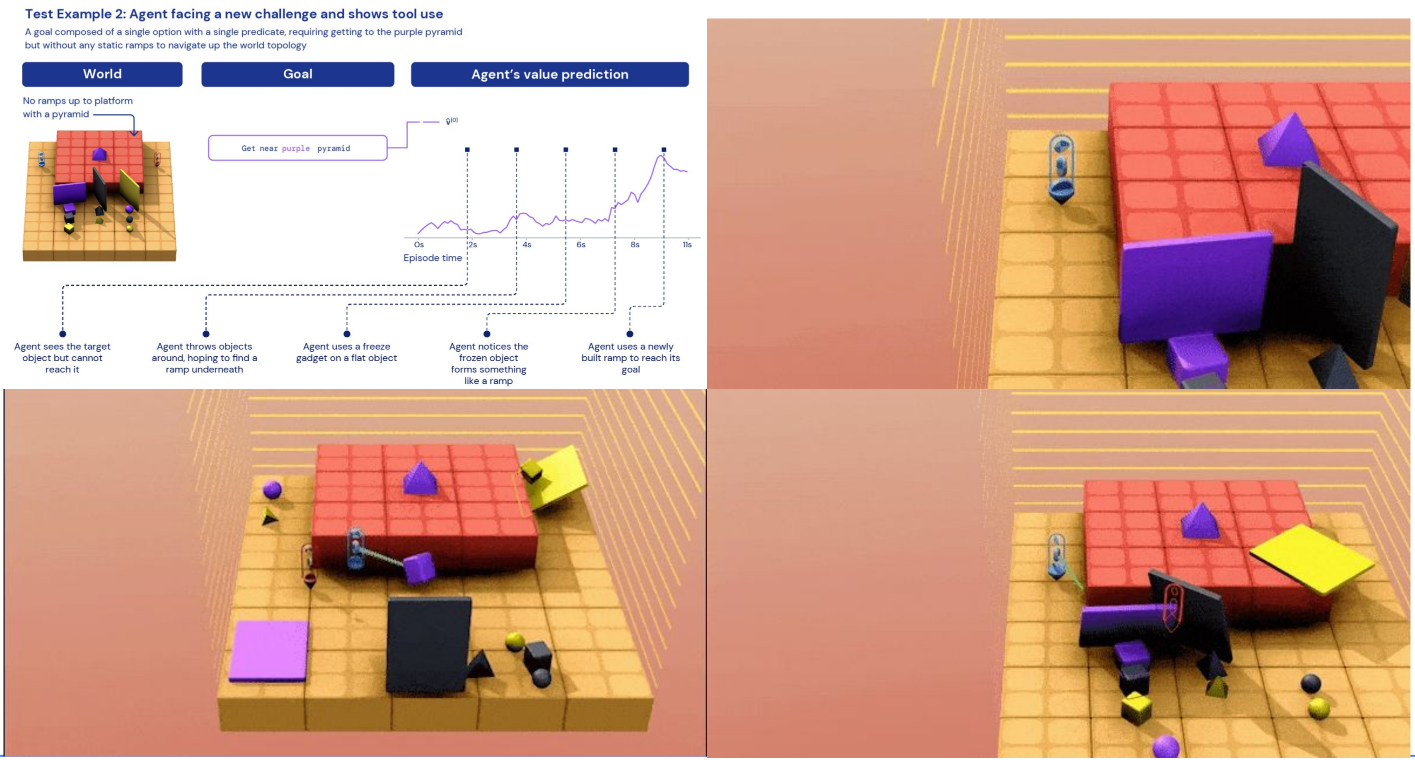
Open-ended neural agents, trained in a universe of ever-changing, automatically generated tasks and games, manifest highly generalizable problem-solving and social skills—pointing the way toward generally capable artificial intelligence systems.

Open-ended Play

Generally capable agents emerge from open-ended play



Open-ended Play



Open-ended Play

Emergent tool use from multi-agent interaction

Thank You for Watching !