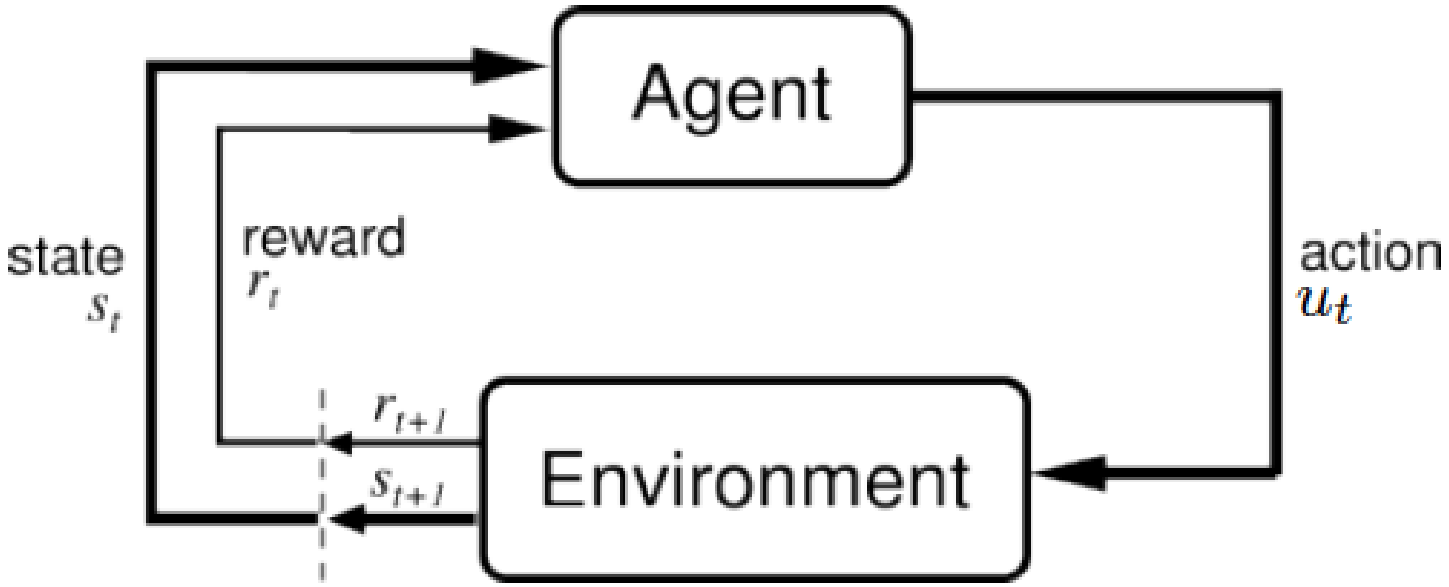
Introduction to Deep Reinforcement Learning

©Dr. Min Chi
mchi@ncsu.edu

AI Academy

Reinforcement Learning



Reinforcement Learning

Supervised Learning:

Memorization and not (yet) great at reasoning.

Reinforcement Learning:

Propagation of outcomes (reward) to knowledge about states and actions. This is a kind of "reasoning".

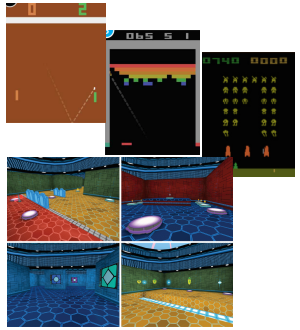
Some Reinforcement Learning Success Stories

Kohl and Stone, 2004

Ng et al, 2004

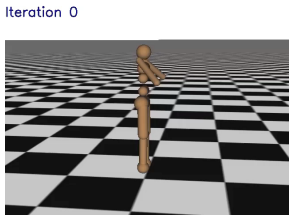
Tedrake et al, 2005

Kober and Peters, 2009

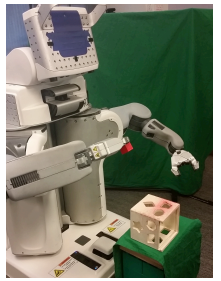


Mnih et al, 2015
(A3C)

Silver et al, 2014
(DPG)
Lillicrap et al, 2015
(DDPG)



Schulman et al,
2016 (TRPO + GAE)



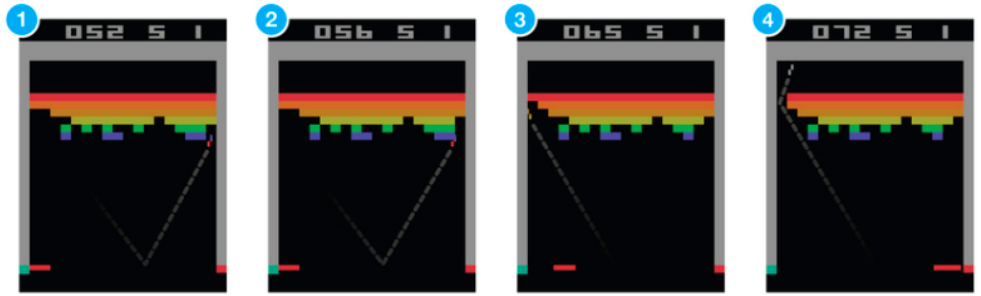
Levine*, Finn*, et
al, 2016
(GPS)

Silver*, Huang*, et
al, 2016
(AlphaGo)

Reinforcement Learning

Reinforcement learning is a general-purpose framework for decision-making:

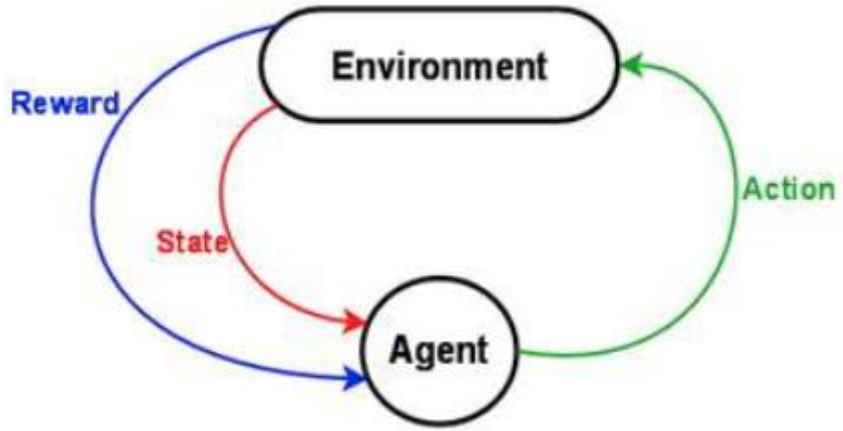
- An agent operates in an environment: **Atari Breakout**
- An agent has the capacity to **act**
- Each action influences the agent's **future state**
- Success is measured by a **reward** signal
- **Goal** is to select actions to maximize future reward



Agent and Environment

- At each step the agent:
  - Executes action
  - Receives observation (new state)
  - Receives reward

- The environment:
  - Receives action
  - Emits observation (new state)
  - Emits reward



Major Components of an RL Agent

An RL agent may include one or more of these components:

- **Policy:** agent's behavior function
- **Value function:** how good is each state and/or action
- **Model:** agent's representation of the environment

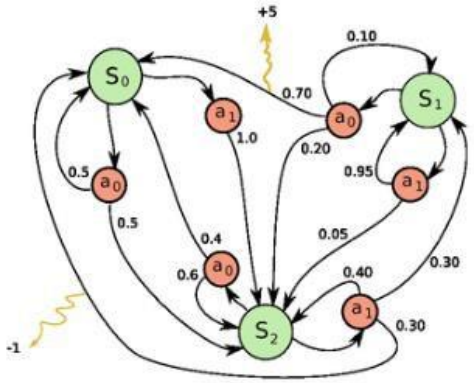
$$s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_{n-1}, a_{n-1}, r_n, s_n$$

state

action

reward

Terminal state



Value Function

- Future reward $R = r_1 + r_2 + r_3 + \cdots + r_n$

$$R_t = r_t + r_{t+1} + r_{t+2} + \cdots + r_n$$

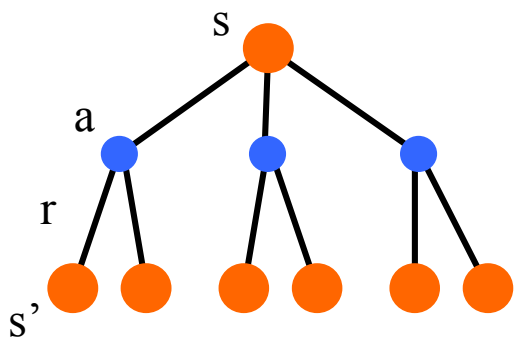
- Discounted future reward (environment is stochastic)

$$\begin{aligned} R_t &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{n-t} r_n \\ &= r_t + \gamma(r_{t+1} + \gamma(r_{t+2} + \cdots)) = \\ & r_t + \gamma R_{t+1} \end{aligned}$$

- A good strategy for an agent would be to always choose an action that maximizes the (discounted) future reward

Q-Learning

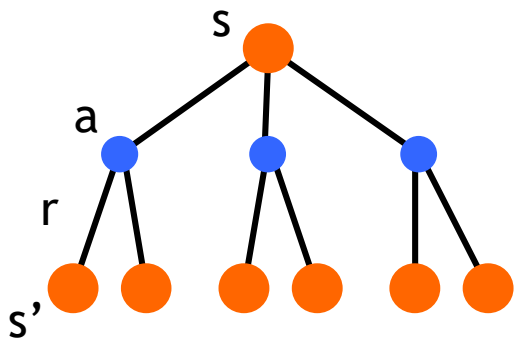
- State value function: $V^{\pi}(s)$
  - Expected return when starting in *s* and following π
- State-action value function: $Q^{\pi}(s,a)$
  - Expected return when starting in *s*, performing *a,* and following π
- Useful for finding the optimal policy
  - Can estimate from experience (Monte Carlo)
  - Pick the best action using $Q^{\pi}(s,a)$



- Q-learning: off-policy
  - Use any policy to estimate Q that maximizes future reward: $Q(s_t, a_t) = max\ R_{t+1}$
  - Q directly approximates Q* (Bellman optimality equation)
  - Independent of the policy being followed
  - Only requirement: keep updating each (s,a) pair

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha \left(R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t) \right)$$

Q-Learning



Learning Rate

Discount Factor

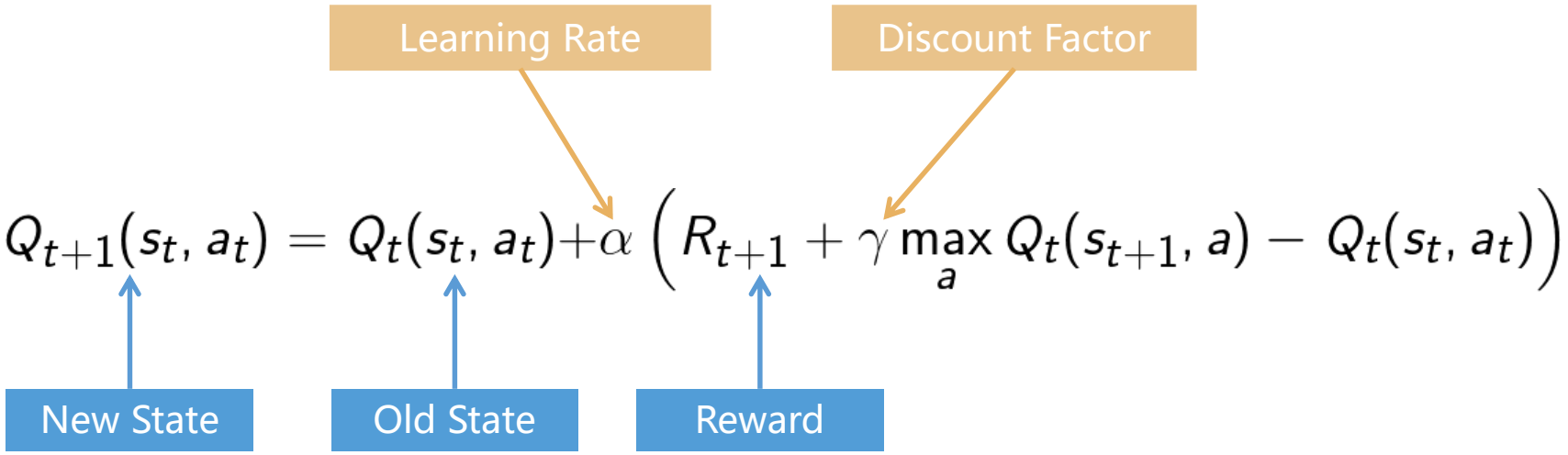
$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha \left(R_{t+1} + \gamma \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t) \right)$$

New State

Old State

Reward

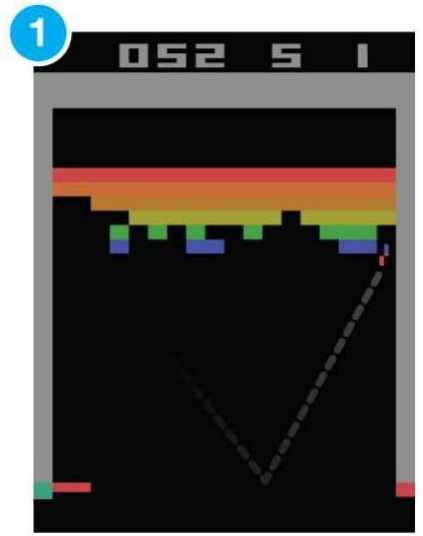
Q-Learning: Value Iteration




```
initialize Q[num_states,num_actions] arbitrarily
observe initial state s
repeat
      select and carry out an action a
      observe reward r and new state s'
      Q[s,a] = Q[s,a] + α(r + γ max_a' Q[s',a'] - Q[s,a])
      s = s'
until terminated
```

Q-Learning: Representation Matters

- In practice, Value Iteration is impractical
  - Very limited states/actions
  - Cannot generalize to unobserved states



- Think about the **Breakout** game
  - State: screen pixels
    - Image size: 84×84 (resized)
    - Consecutive 4 images
    - Grayscale with 256 gray levels

$256^{84\times84\times4}$ rows in the Q-table!

Philosophical Motivation for Deep Reinforcement Learning

Supervised Learning:

Memorization and not (yet) great at reasoning.

Reinforcement Learning:

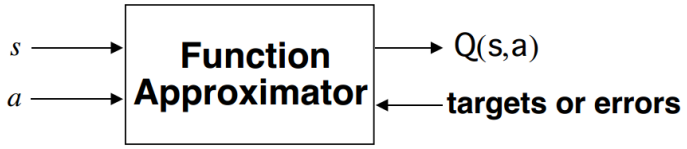
Propagation of outcomes (reward) to knowledge about states and actions. This is a kind of "reasoning".

Deep Learning + Reinforcement Learning:

General purpose artificial intelligence through efficient **generalizable learning** of the **optimal thing** to do given a formalized set of actions and states (possibly huge).

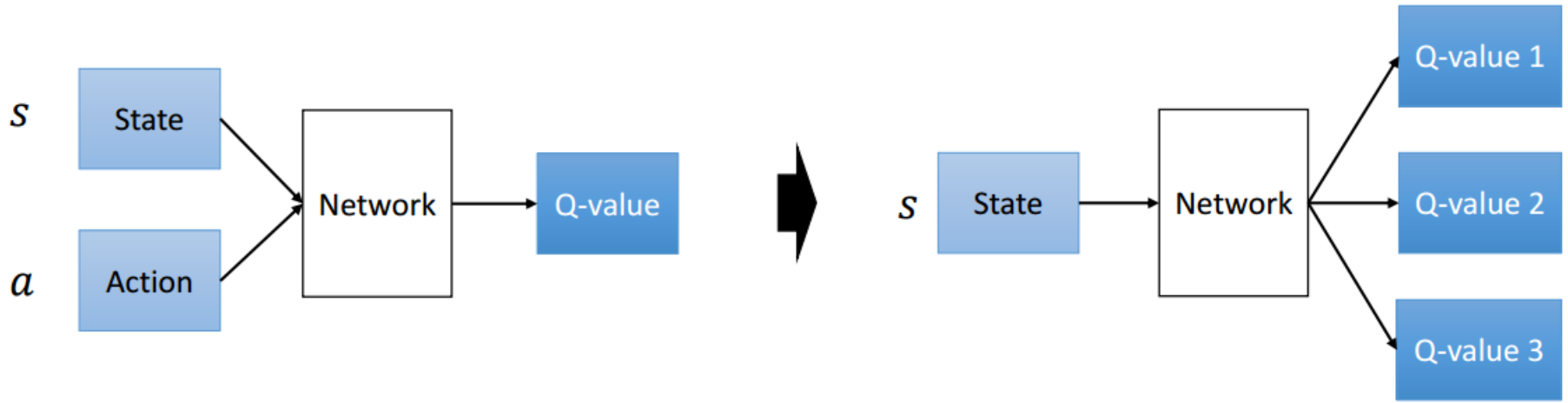
Deep Q-Learning

Use a function (with parameters) to approximate the Q-function

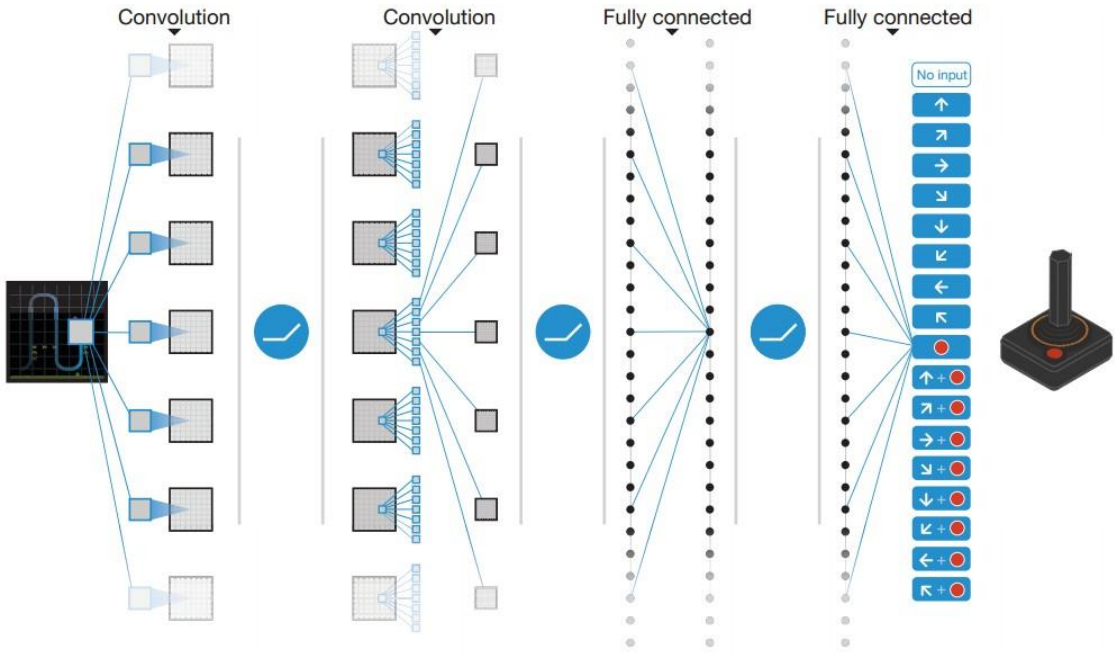


- Linear
- Non-linear: **Q-Network**

$$Q(s, a; \boldsymbol{\theta}) \approx Q^*(s, a)$$



Deep Q-Network: Atari



Layer	Input	Filter size	Stride	Num filters	Activation	Output
conv1	84x84x4	8x8	4	32	ReLU	20x20x32
conv2	20x20x32	4x4	2	64	ReLU	9x9x64
conv3	9x9x64	3x3	1	64	ReLU	7x7x64
fc4	7x7x64			512	ReLU	512
fc5	512			18	Linear	18

Mnih et al. "Playing atari with deep reinforcement learning." 2013.

Deep Q-Network Training

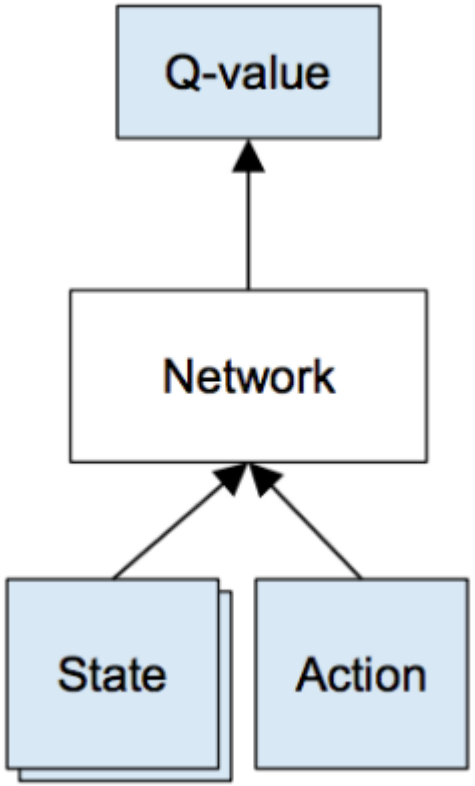
- Bellman Equation:

$$Q(s,a) = r + \gamma max_{a'} Q(s',a')$$

- Loss function (squared error):

$$L = \mathbb{E}[(\underbrace{\boldsymbol{r + \gamma max_{a'} Q(s',a')}}_{\textbf{target}} - Q(s,a))^2]$$

Deep Q-Network Training



Given a transition $< s, a, r, s' >$, the Q-table update rule in the previous algorithm must be replaced with the following:

- Do a feedforward pass for the current state s to get **predicted Q-values for all actions**
- Do a feedforward pass for the next state s' and calculate maximum overall network outputs $\boldsymbol{max_{a'} Q(s', a')}$
- Set Q-value target for action to $\boldsymbol{r + \gamma max_{a'} Q(s', a')}$ (use the max calculated in step 2).
  - For all other actions, set the Q-value target to the same as originally returned from step 1, making the error 0 for those outputs.
- Update the weights using backpropagation.

Exploration vs Exploitation

- Key ingredient of Reinforcement Learning

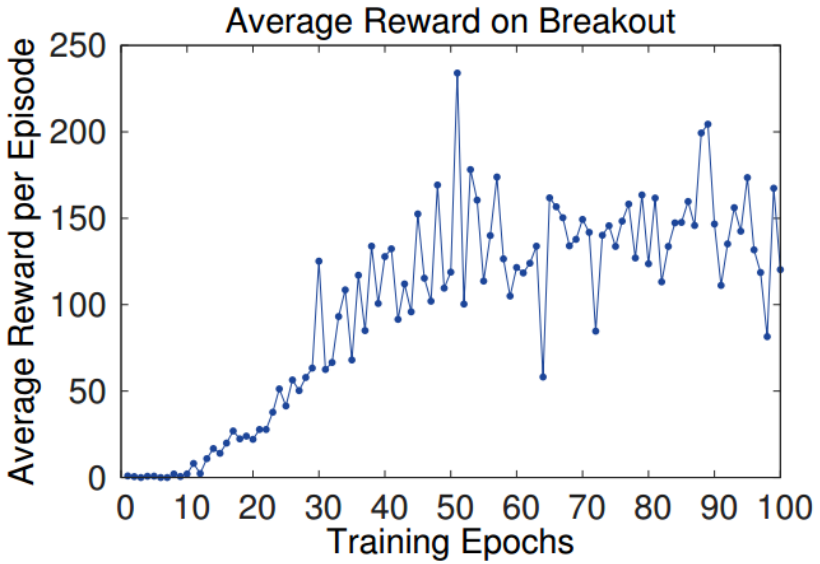
- Deterministic/greedy policy won't explore all actions
  - Don't know anything about the environment at the beginning
  - Need to try all actions to find the optimal one

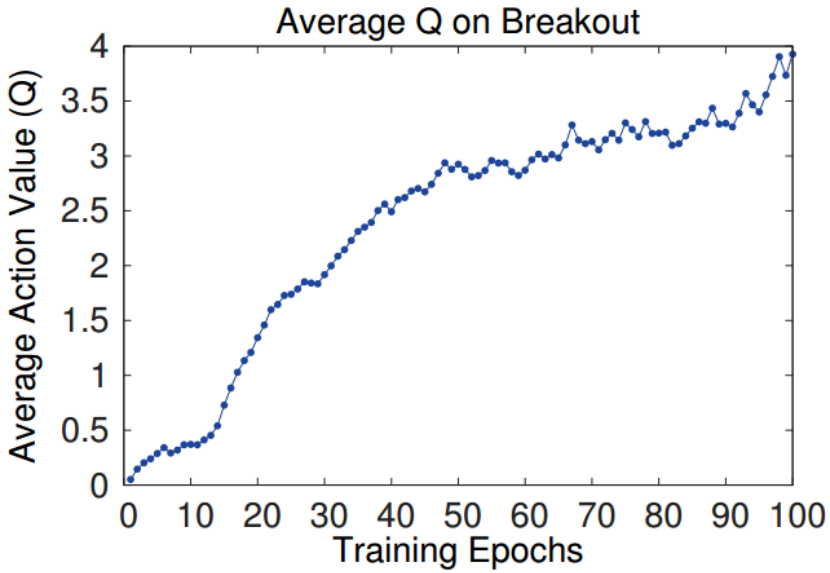
- Maintain exploration
  - Use *soft* policies instead: $\pi(s,a)>0$ (for all s,a)
- ε-greedy policy
  - With probability 1-ε perform the optimal/greedy action
  - With probability ε perform a random action

  - Will keep exploring the environment
  - Slowly move it towards greedy policy: ε -> 0

Atari Breakout

- A few tricks needed, most importantly:
  - Experience replay
  - Target Network





Deep Q-Learning Algorithm

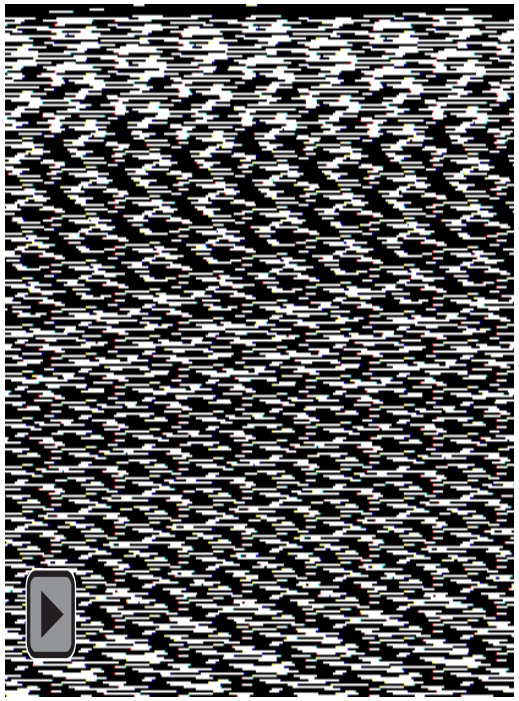
Initialize network Q
Initialize target network $\hat{Q}$
Initialize experience replay memory D
Initialize the *Agent* to interact with the Environment
while *not converged* **do**

  `/* Sample phase`
  $\epsilon \leftarrow$ setting new epsilon with ϵ-decay
  Choose an action a from state s using policy ϵ-greedy(Q)
  Agent takes action a, observe reward r, and next state s'
  Store transition $(s, a, r, s', done)$ in the experience replay memory D

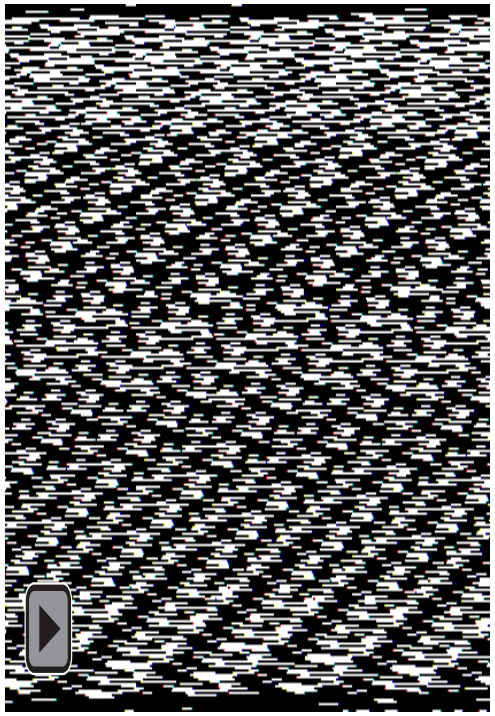
  if *enough experiences in* D **then**
    `/* Learn phase`
    Sample a random *minibatch* of N transitions from D
    for *every transition* $(s_i, a_i, r_i, s'_i, done_i)$ ***in*** *minibatch* **do**
      if $done_i$ **then**
        $y_i = r_i$
      else
        $y_i = r_i + \gamma \max_{a' \in \mathcal{A}} \hat{Q}(s'_i, a')$
      end
    end
    Calculate the loss $\mathcal{L}$ =1/N $\sum_{i=0}^{N-1}(Q(s_i, a_i) - y_i)^2$
    Update Q using the SGD algorithm by minimizing the loss $\mathcal{L}$
    Every C steps, copy weights from Q to $\hat{Q}$
  end
end

Atari Breakout

After
10 Minutes
of Training

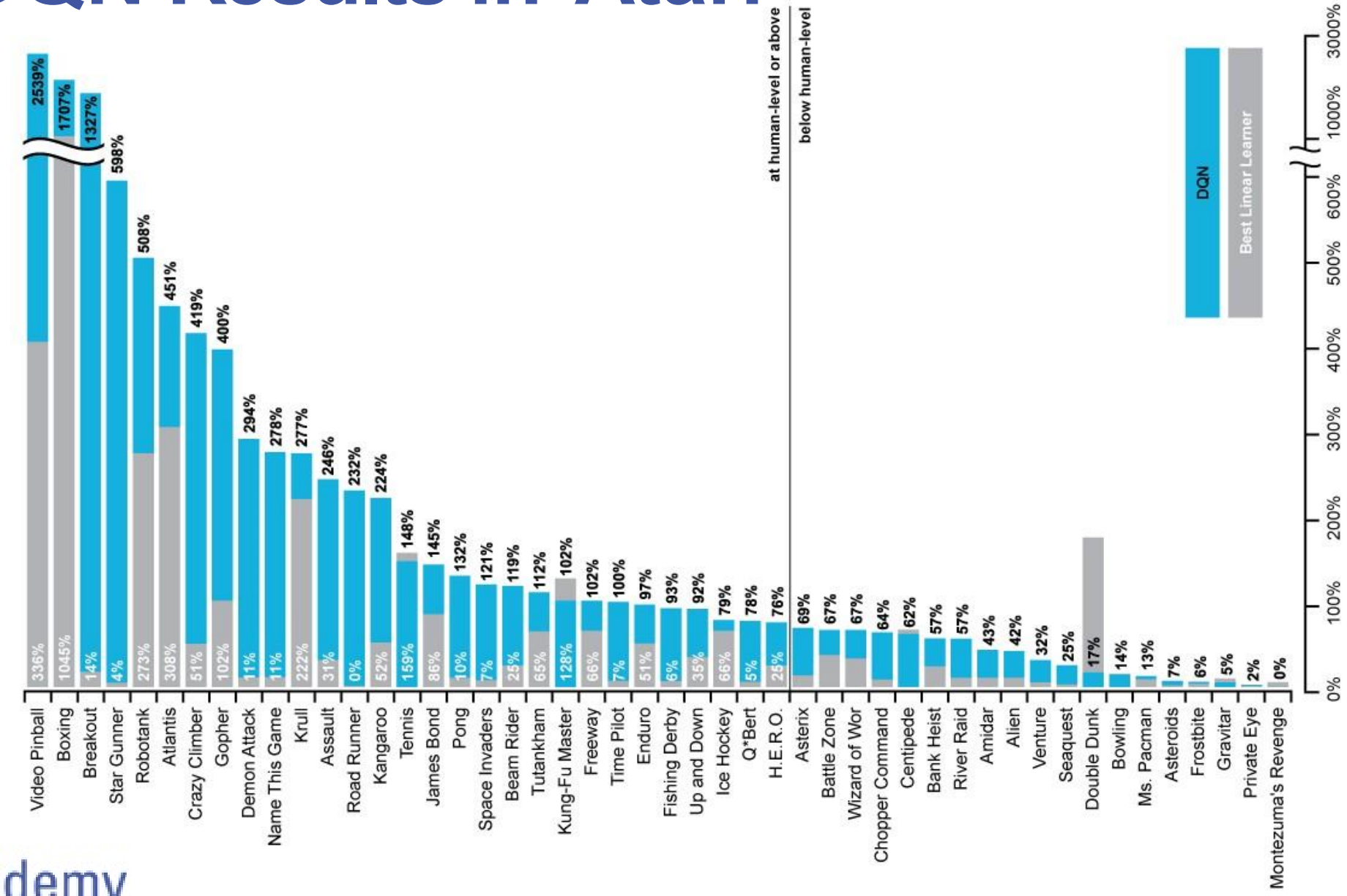


After
120 Minutes
of Training



After
240 Minutes
of Training

DQN Results in Atari

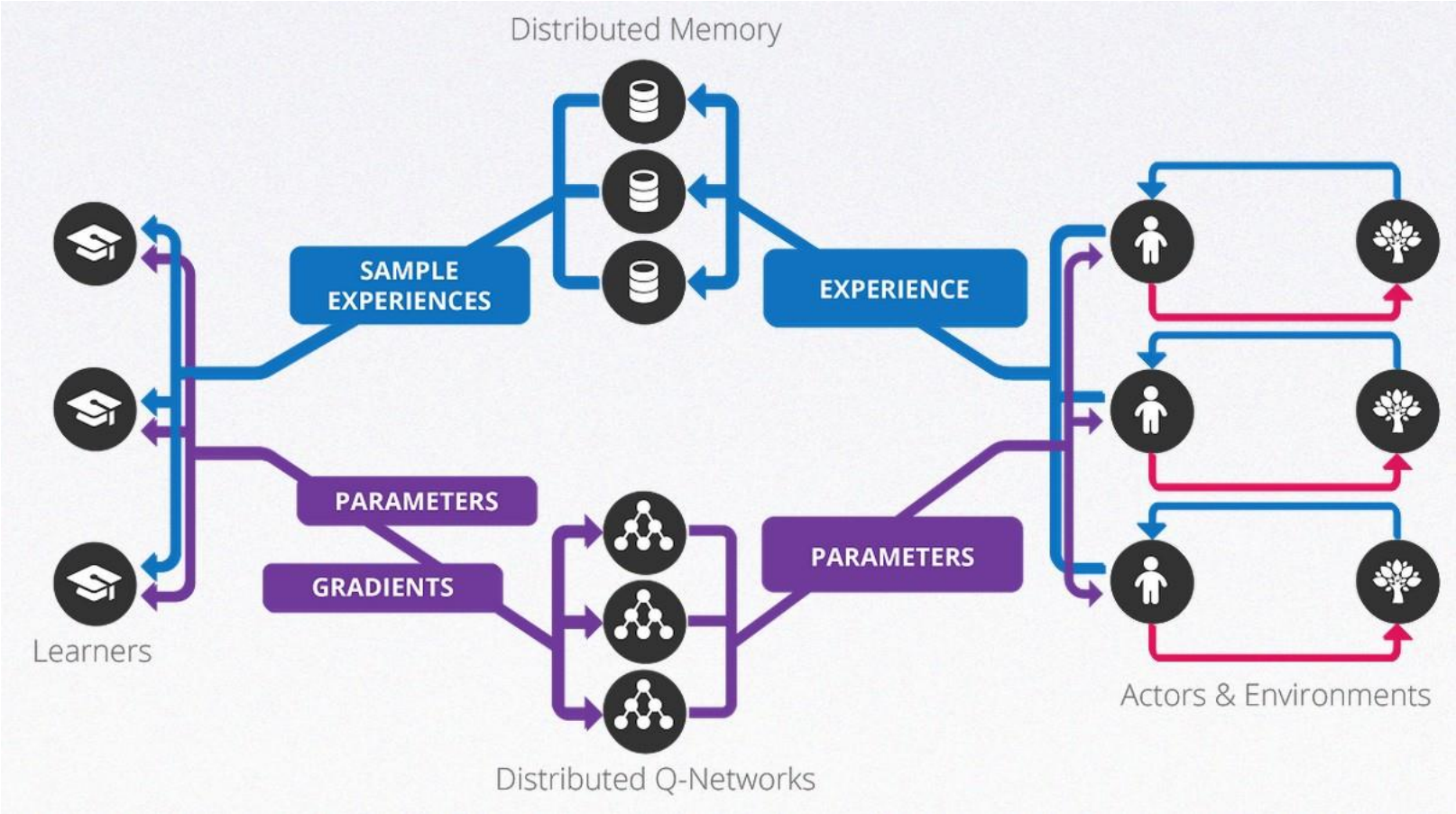


AI Academy

NC STATE

Gorila

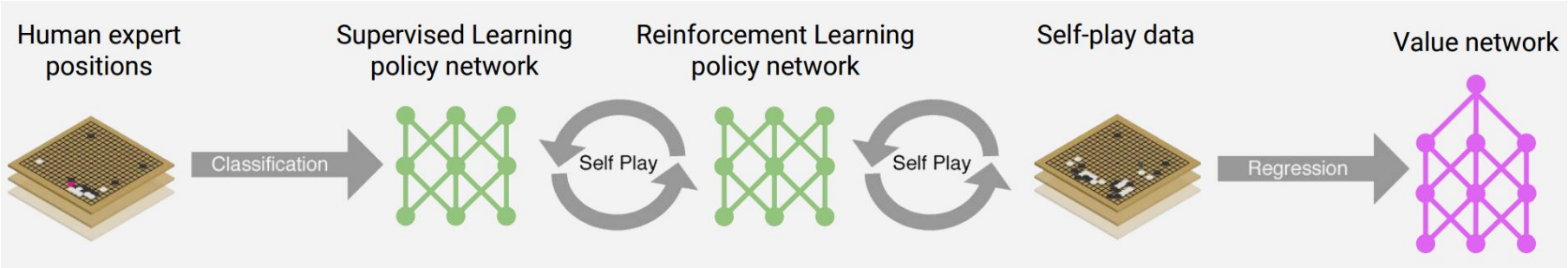
(General Reinforcement Learning Architecture)

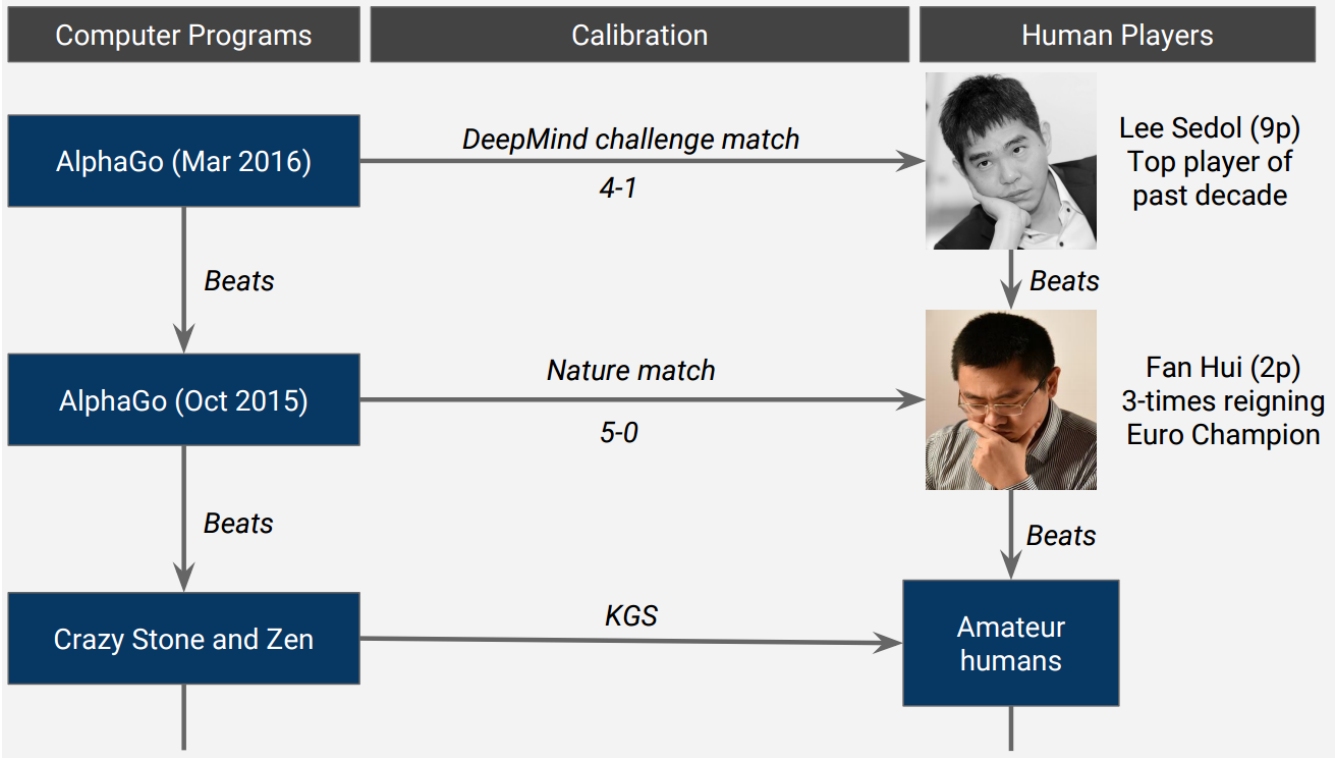


- 10x faster than Nature DQN on 38 out of 49 Atari games
- Applied to recommender systems within Google

Nair et al. "Massively parallel methods for deep reinforcement learning."

Human-in-the-Loop Reinforcement Learning





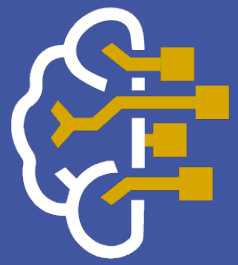
Reminder: Unexpected Local Pockets of High Reward

Stay Connected

Dr. Min Chi

Associate Professor
mchi@ncsu.edu
(919) 515-7825

AI Academy



AI Academy

go.ncsu.edu/aiacademy

NC STATE UNIVERSITY