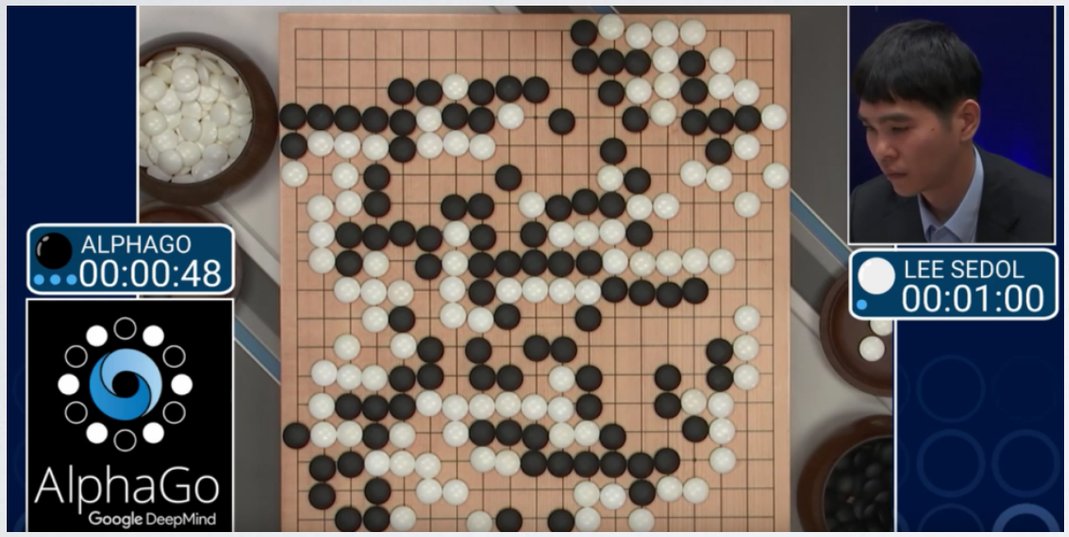
2017年臺大電機系大學部專精獎

強化式學習與卷積神經網絡於訓練視覺化電腦代理人之應用

電機四
劉廷緯 溫明浩

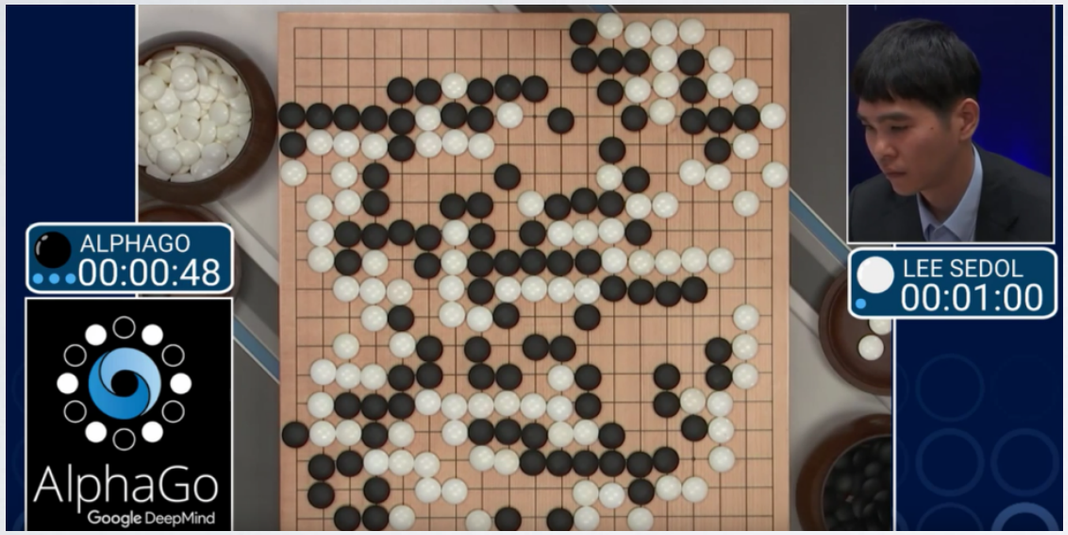
GAMES & AI

AlphaGo - Google DeepMind



GAMES & AI

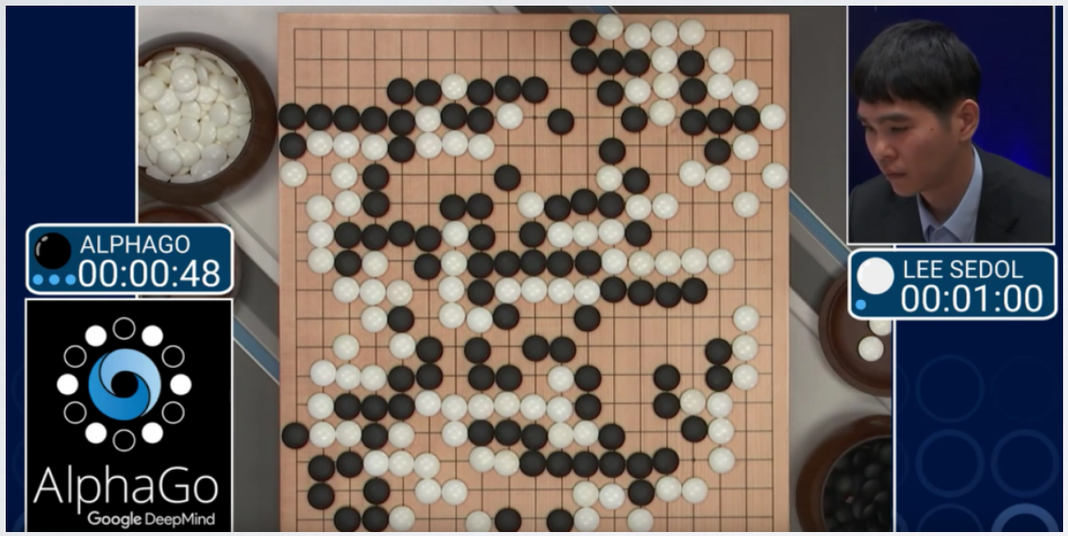
AlphaGo - Google DeepMind



StarCraft II - Google DeepMind

GAMES & AI

AlphaGo - Google DeepMind

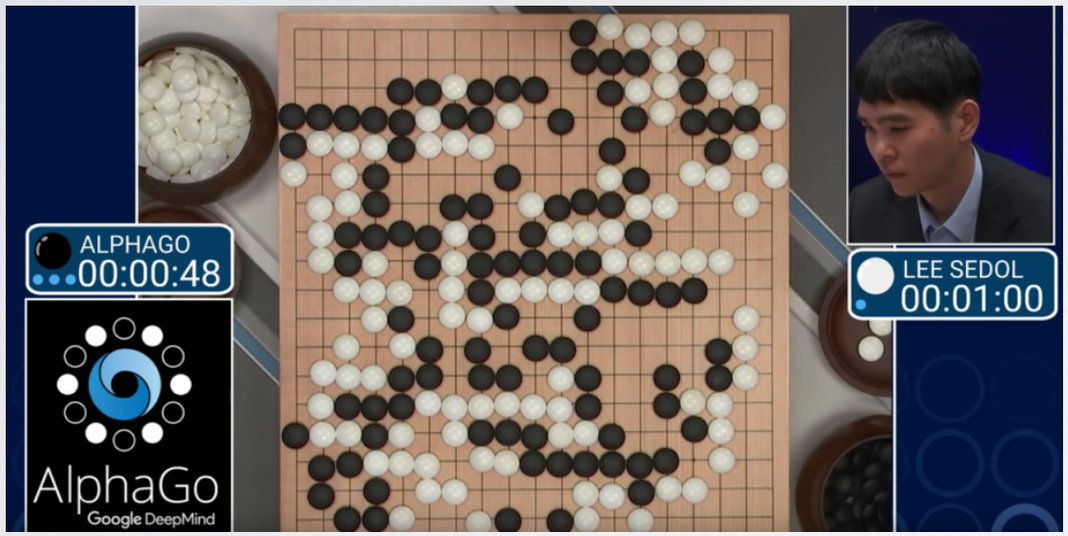


StarCraft II - Google DeepMind

GTA V - Harrison Kinsley

GAMES & AI

AlphaGo - Google DeepMind



StarCraft II - Google DeepMind

GTA V - Harrison Kinsley

Doom - Facebook Ai Research

Slither.io

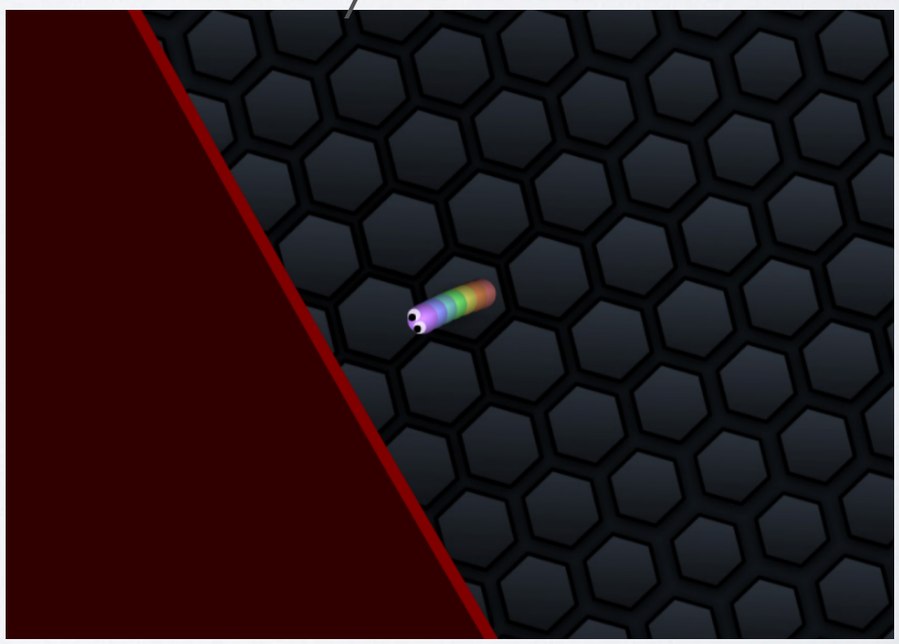
an online massively multiplayer browser game

Slither.io

an online massively multiplayer browser game

Circular map

Deadly Outer rim

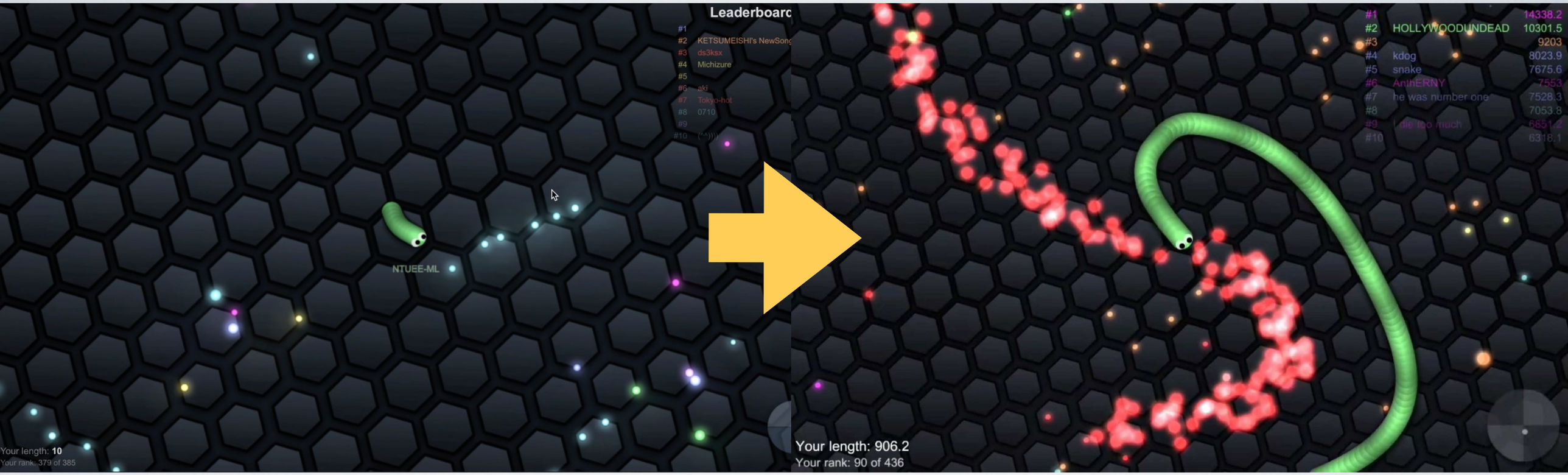


Spawn

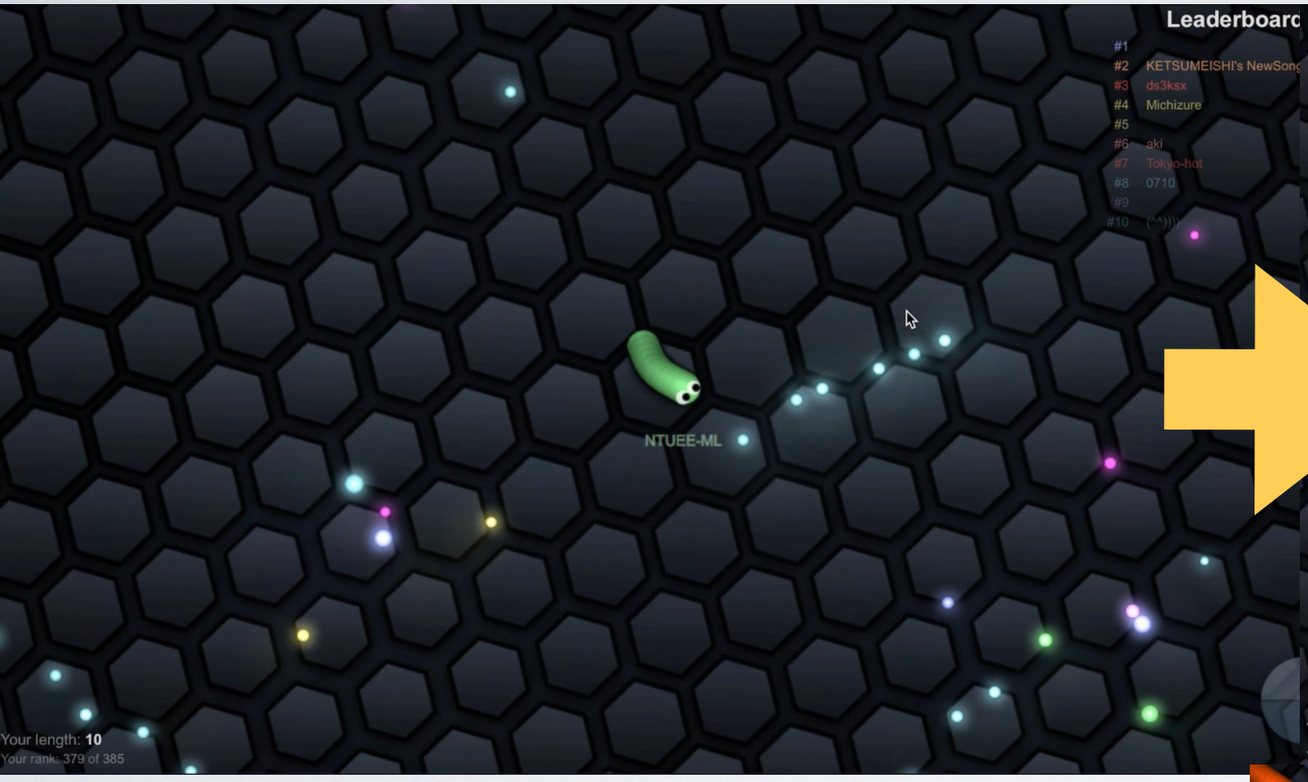

Leaderboard
#1 45336
#2 KETSUMEISHI's NewSong.. 22073
#3 ds3ksx 20262
#4 Michizure 18417
#5 14690
#6 akl 14682
#7 Tokyo-hot 12049
#8 0710 11105
#9 9073
#10 (^^)))) 8679
NTUEE-ML
Your length: 10
Your rank: 379 of 385

Spawn

Consume Pallets to grow



Spawn



Consume Pallets to grow

Grow the largest snake

Spawn

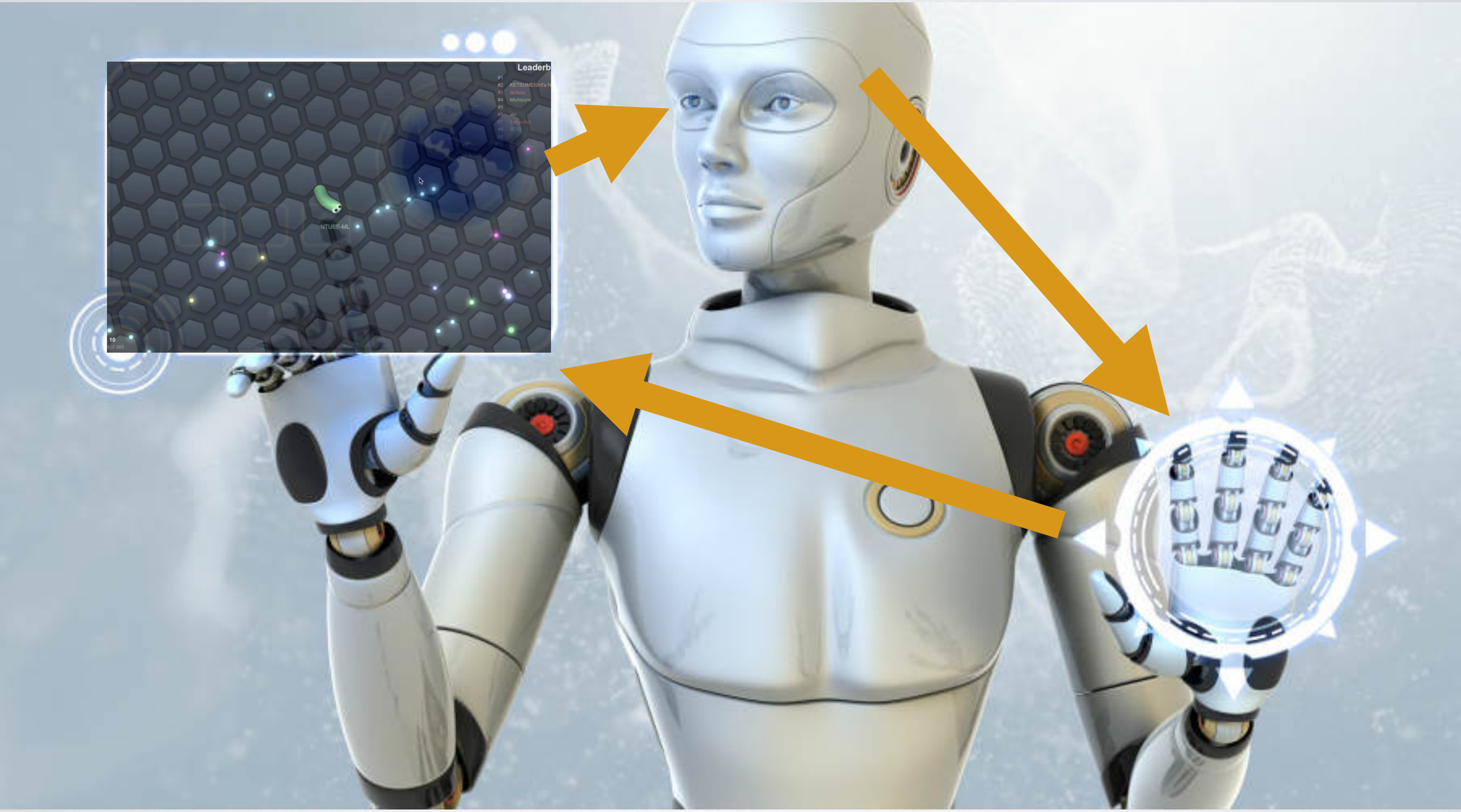
Consume Pallets to grow

Dies during head collision

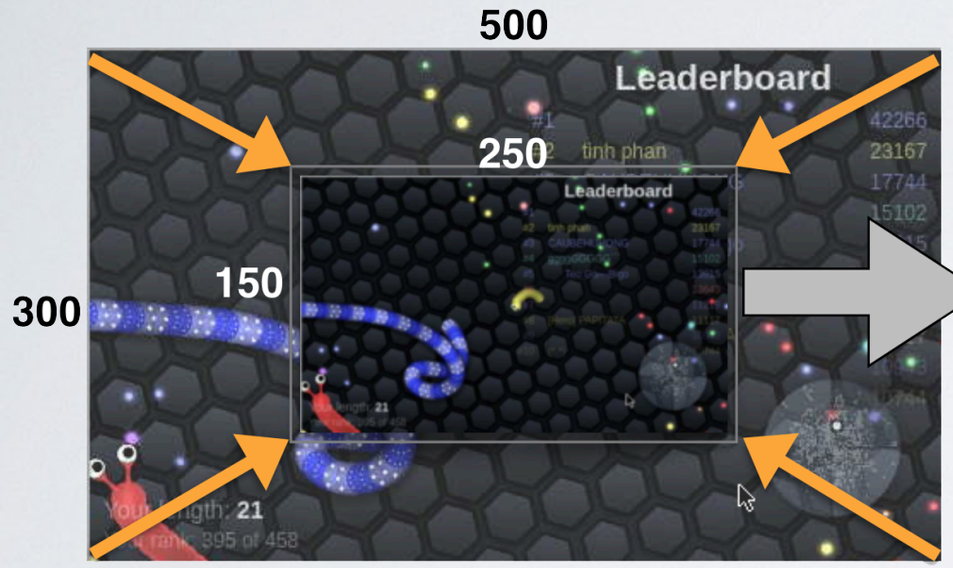
Grow the largest snake

VISION-BASED AGENT

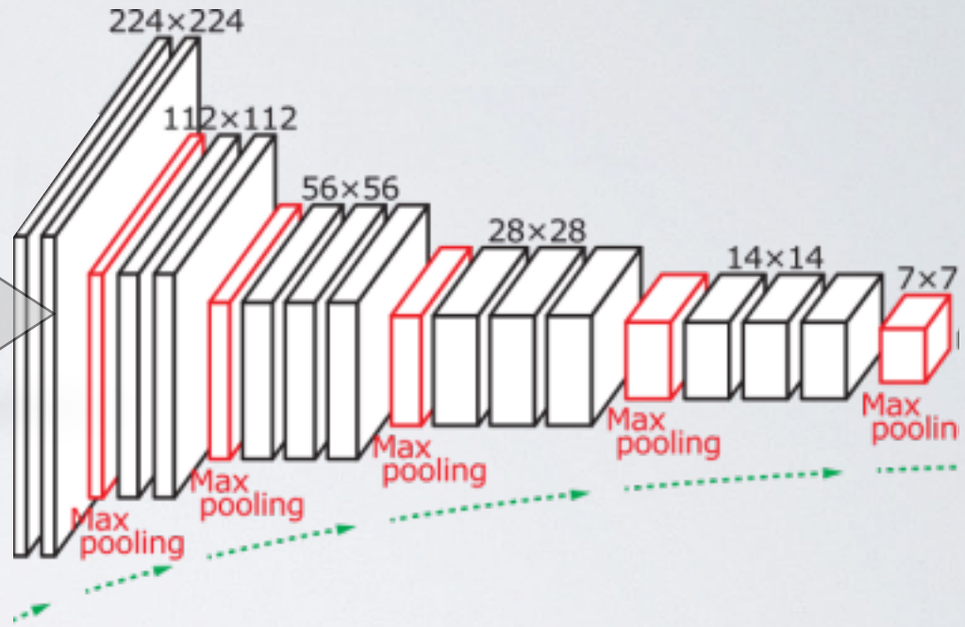
VISION-BASED AGENT



CNN



Screen Processing



Convolutional Neural Network

ACTOR CRITIC MODEL

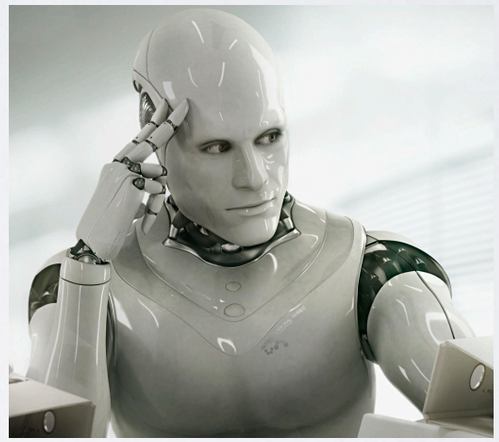
Goal: $\text{maximize}\, \mathbb{E}\left[\sum_{t=0}^{\infty} r_t\right]$

ACTOR CRITIC MODEL

Goal: $\text{maximize}\,\mathbb{E}\left[\sum_{t=0}^{\infty} r_t\right]$

Actor

Critic



Reinforces each other

Policy function

$\pi(a|s)$

Learns to choose the most rewardable action

Value function

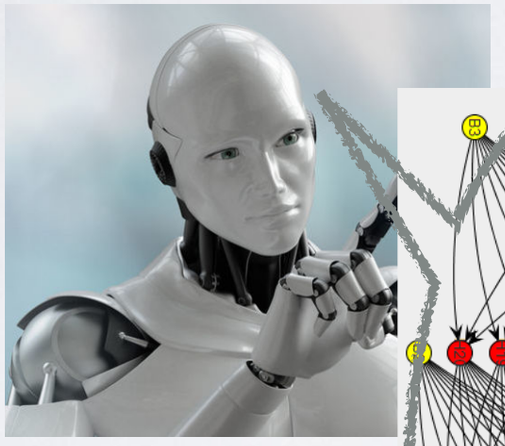
$V(s)$

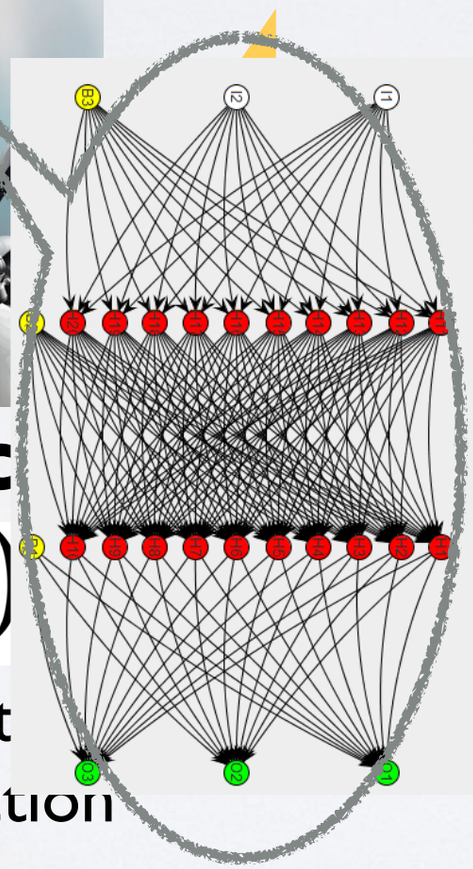
Learns the value of each state

ACTOR CRITIC MODEL

Goal: $\text{maximize}\,\mathbb{E}\left[\sum_{t=0}^{\infty} r_t\right]$

Actor





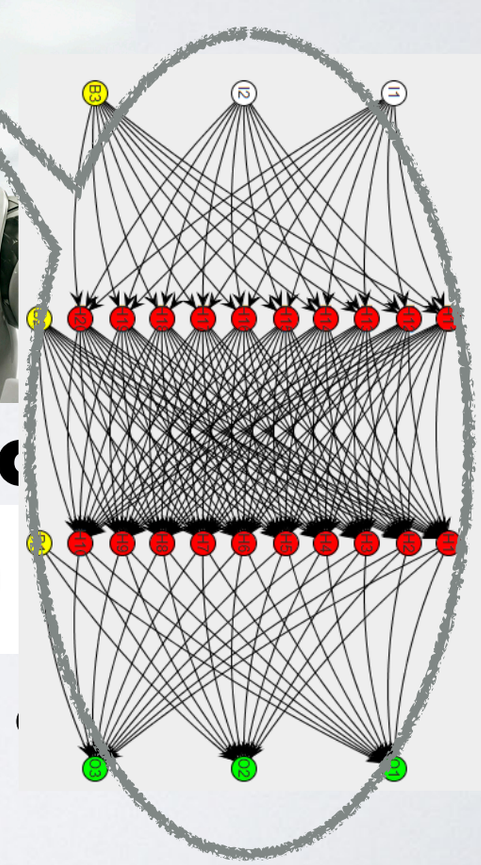
Policy func

$\pi(a|s)$

Learns to choose t
rewardable action

s each
er

Critic



Value func

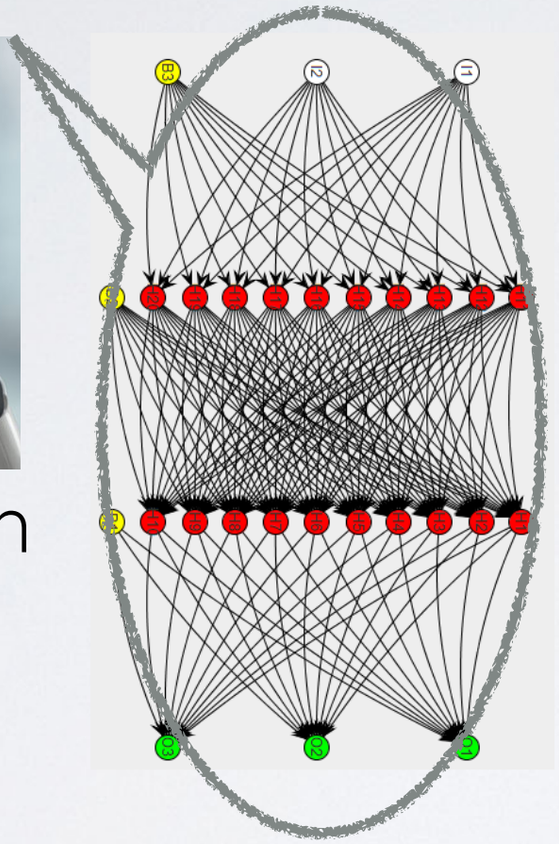
$V(s)$

Learns the value
state

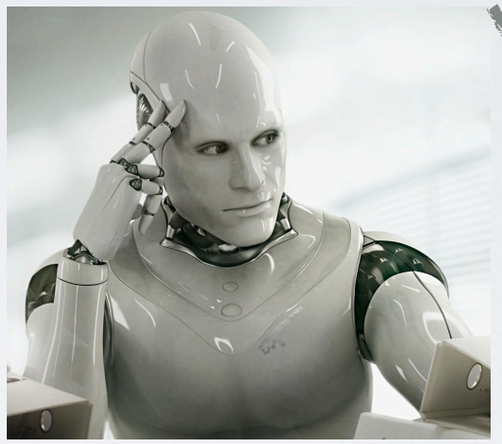
Actor

Policy function

$$\pi(a|s)$$

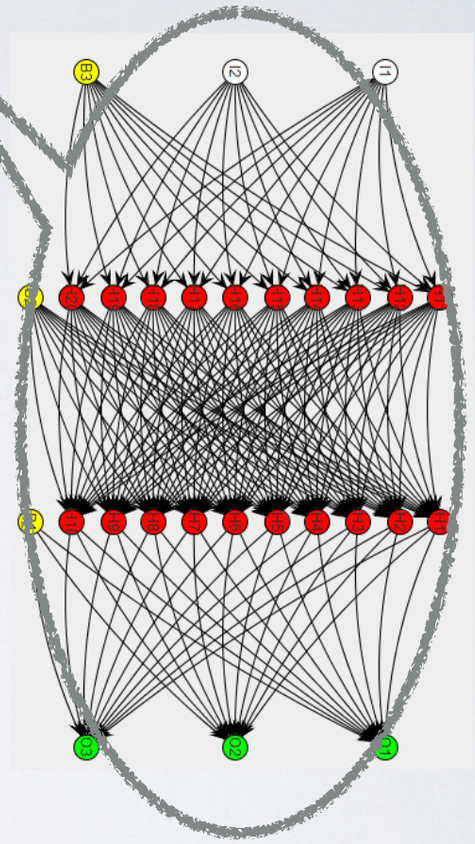


Critic

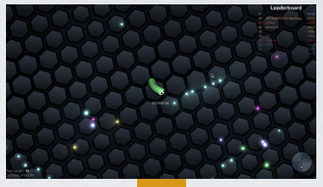


Value function

$$V(s)$$



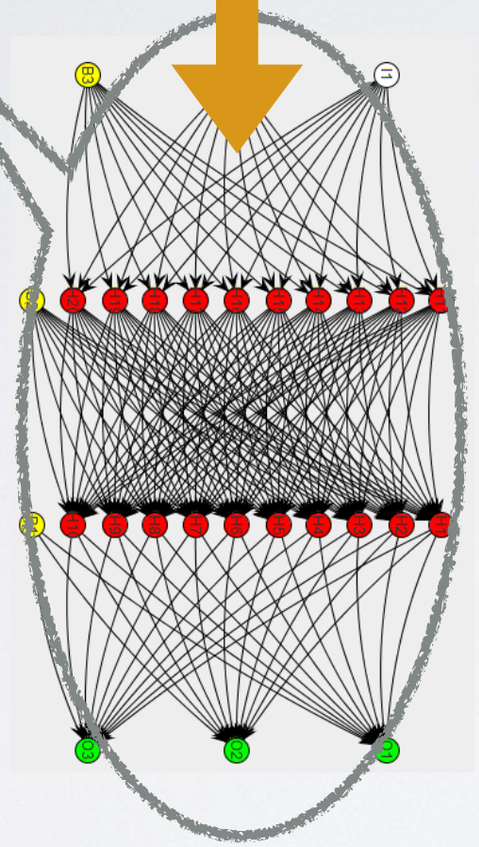
Game Frames as states



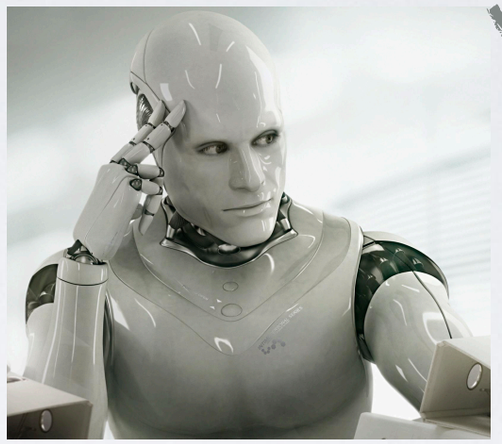
Actor

Policy function

$$\pi(a|s)$$

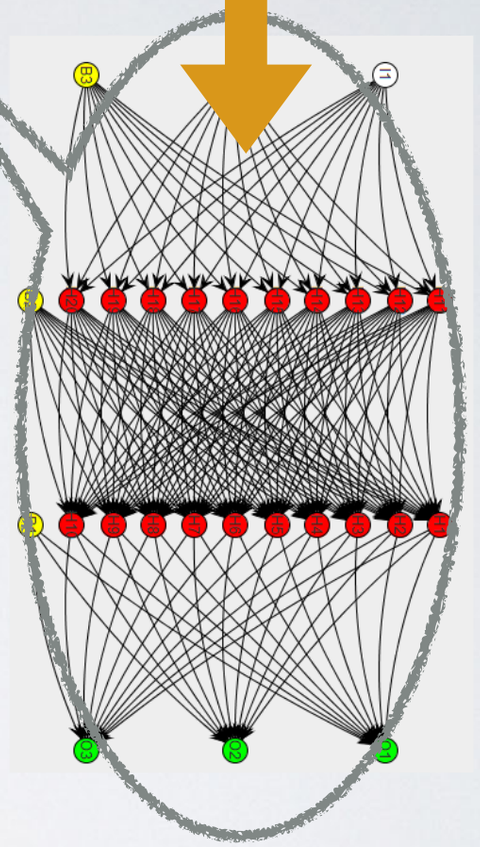


Critic

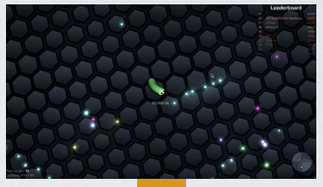


Value function

$$V(s)$$



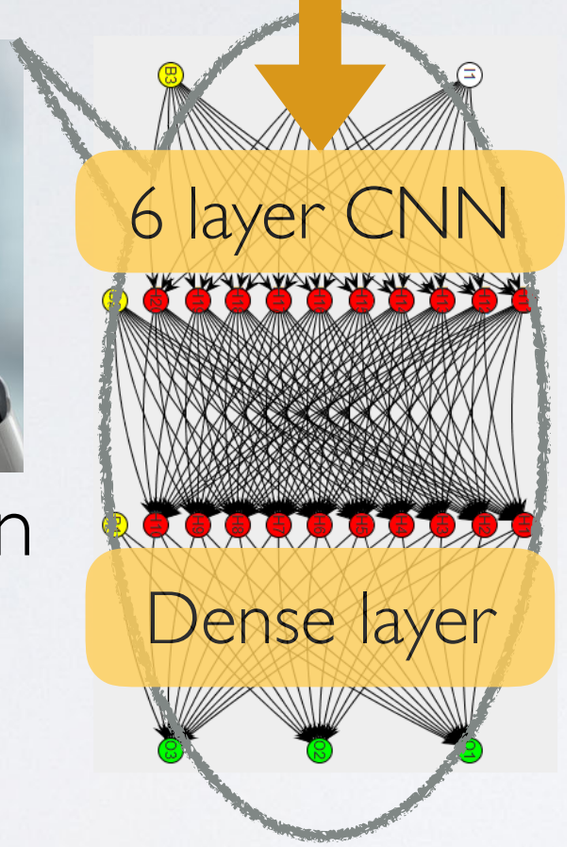
Game Frames as states



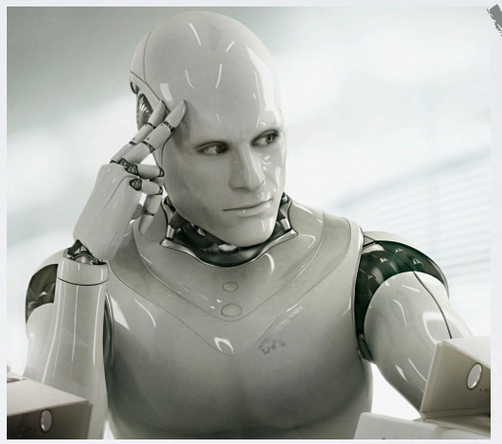
Actor

Policy function

$\pi(a|s)$

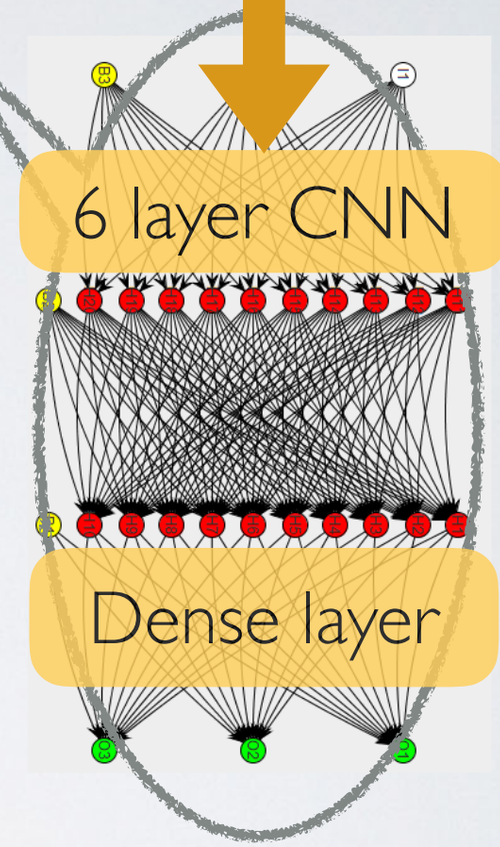


Critic

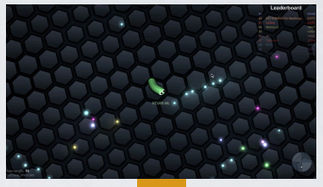


Value function

$V(s)$



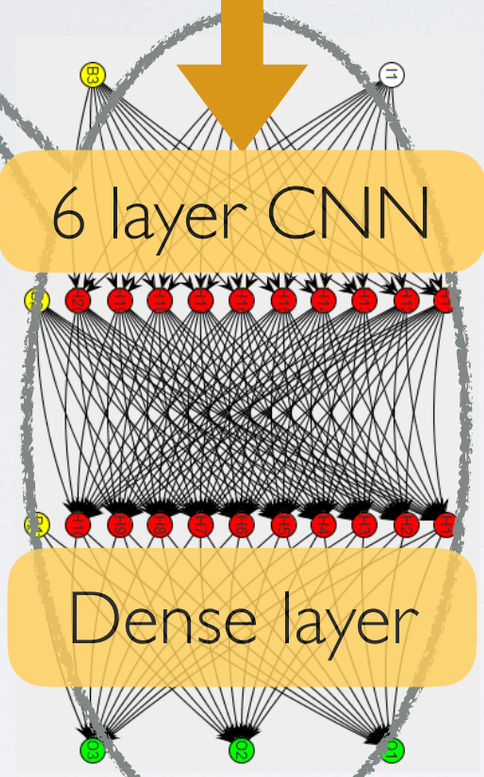
Game Frames as states

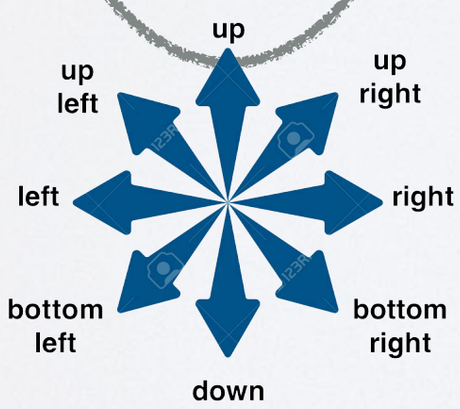


Actor

Policy function

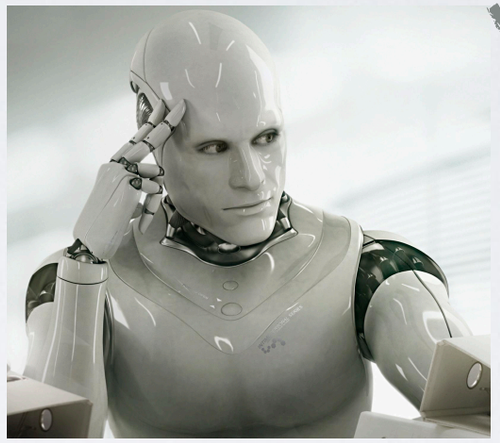
$$\pi(a|s)$$





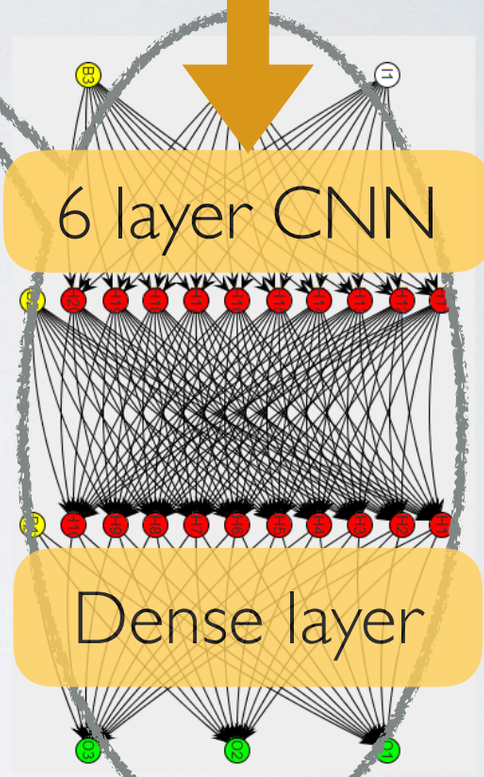
Softmax Output

Critic

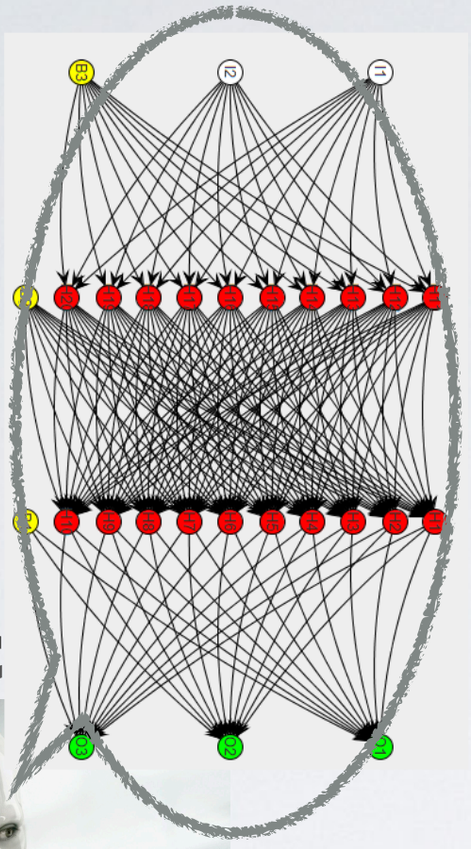


Value function

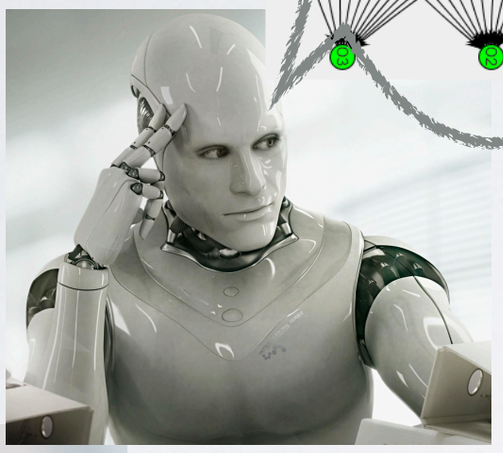
$$V(s)$$



Linear Regression Output



Critic



Actor

Value function

$$V(s)$$

Policy function

$$\pi(a|s)$$

UNIVERSE
OpenAI

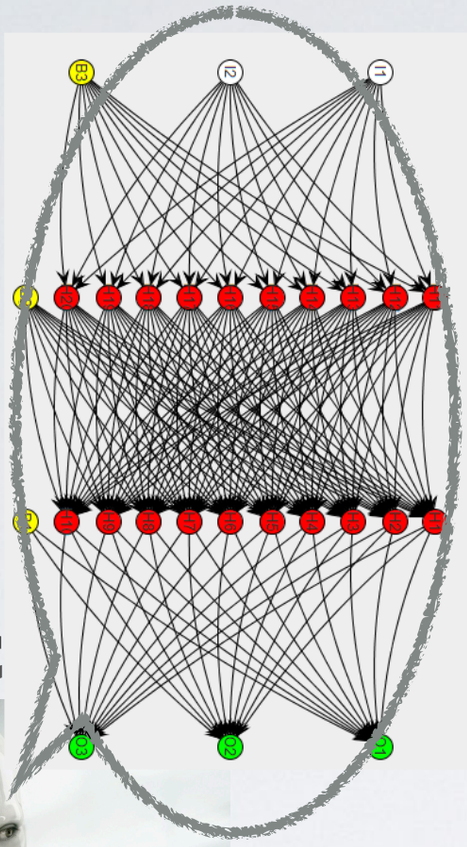
Setup Game Environment for Agent to interact

TensorFlow

Implement CNN network and Actor Critic Model

python™

Implement Training Algorithm



2D Convolution: filters=32, kernel=7*7 strides=2

2D Convolution: filters=64, kernel=7*7 strides=2

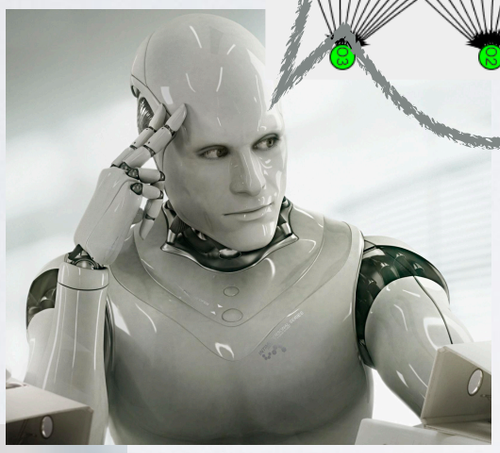
Max Pooling: pool=3*3 strides=2

2D Convolution: filters=128, kernel=3*3 strides=1

Max Pooling: pool=3*3 strides=2

2D Convolution: filters=192, kernel=3*3 strides=1

Critic



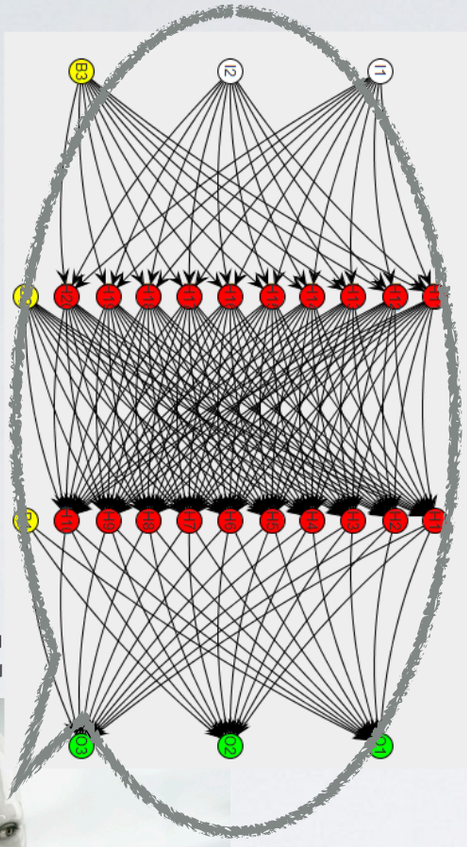
Value function

$$\mathbf{V}(s)$$

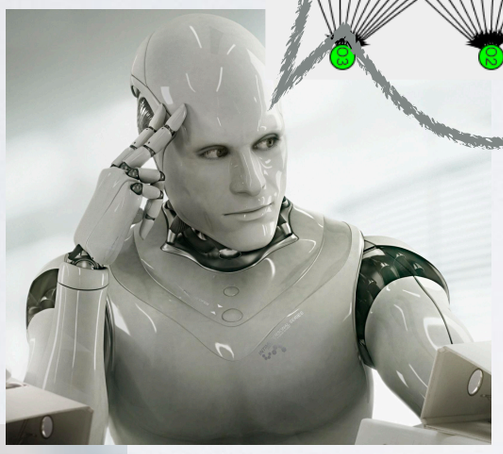
Actor

Policy function

$$\pi(a|s)$$



Critic



Actor

Value function

$$V(s)$$

Policy function

$$\pi(a|s)$$

2D Convolution: filters=32, kernel=7*7 strides=2

2D Convolution: filters=64, kernel=7*7 strides=2

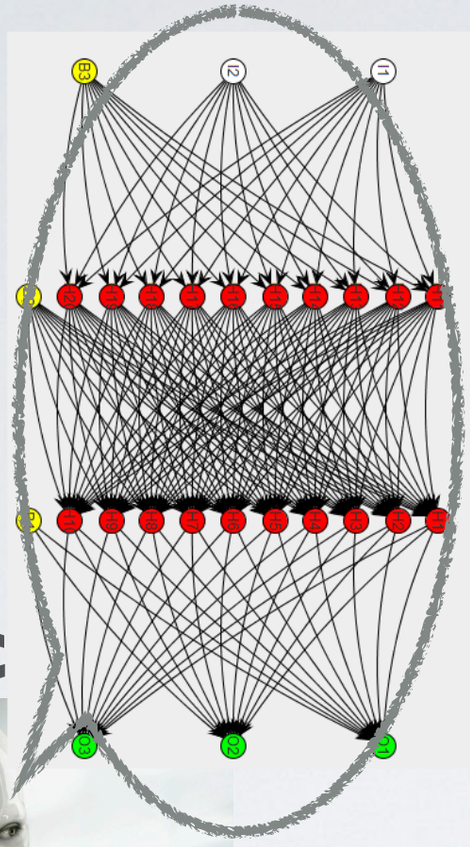
Max Pooling: pool=3*3 strides=2

2D Convolution: filters=128, kernel=3*3 strides=1

Max Pooling: pool=3*3 strides=2

2D Convolution: filters=192, kernel=3*3 strides=1

Dense Layer: units=1024



2D Convolution: filters=32, kernel=7*7 strides=2

2D Convolution: filters=64, kernel=7*7 strides=2

Max Pooling: pool=3*3 strides=2

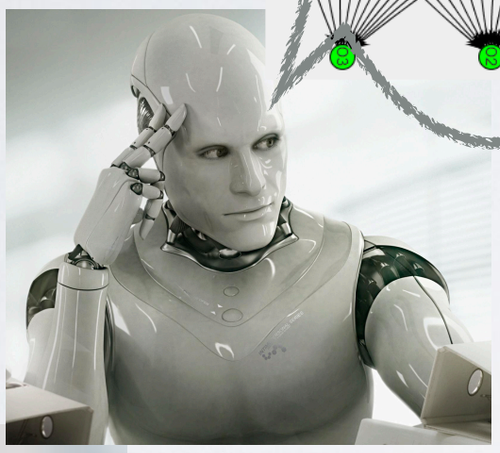
2D Convolution: filters=128, kernel=3*3 strides=1

Max Pooling: pool=3*3 strides=2

2D Convolution: filters=192, kernel=3*3 strides=1

Dense Layer: units=1024

Critic



Actor

Value function

$$V(s)$$

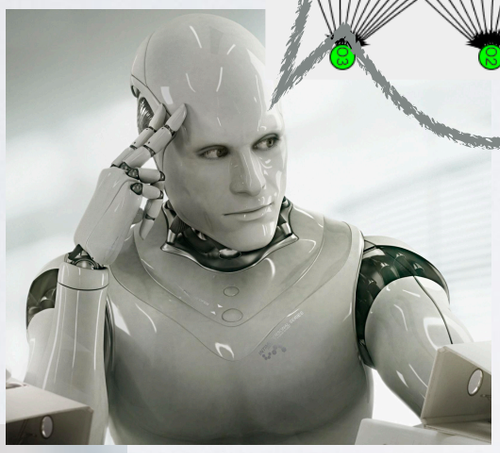
$$loss_V = (r + \gamma V(s_{t+1}) - V(s_t))^2$$

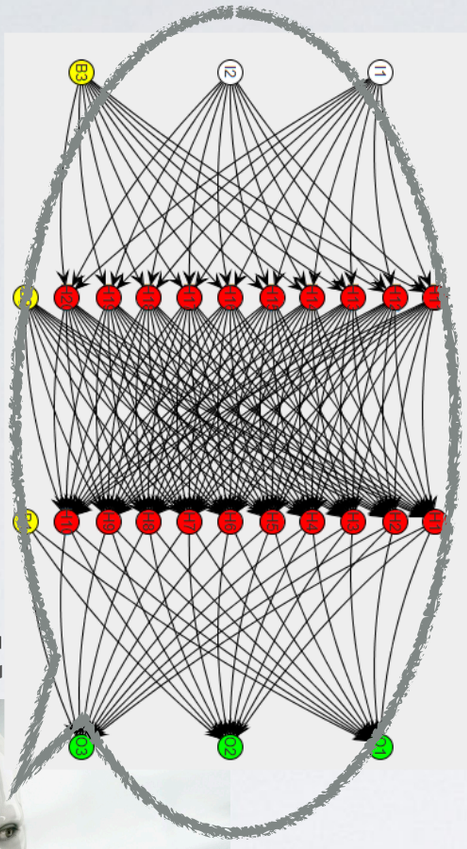
Policy function

$$\pi(a|s)$$

$$loss_\pi = -(r + \gamma V(s_{t+1}) - V(s_t)) \log \pi(a|s)$$

Critic





2D Convolution: filters=32, kernel=7*7 strides=2

2D Convolution: filters=64, kernel=7*7 strides=2

Max Pooling: pool=3*3 strides=2

2D Convolution: filters=128, kernel=3*3 strides=1

Max Pooling: pool=3*3 strides=2

2D Convolution: filters=192, kernel=3*3 strides=1

Dense Layer: units=1024

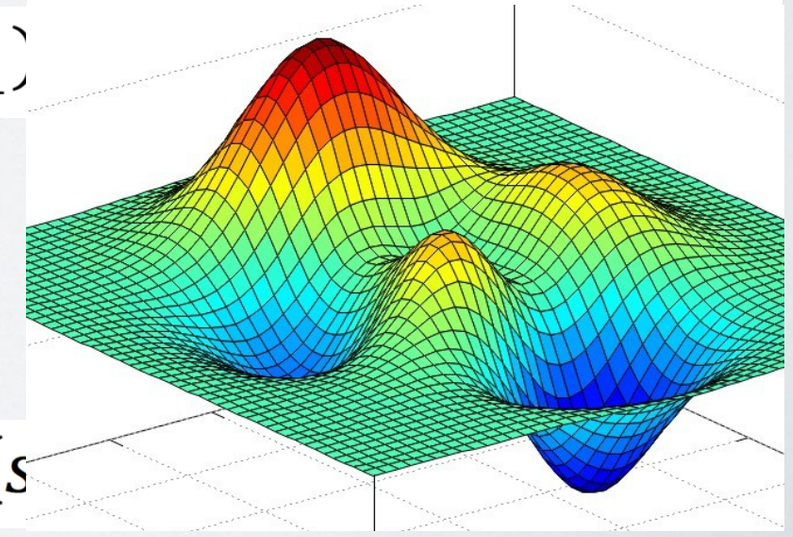
Actor

Value function

$$V(s)$$

$$loss_V = (r + \gamma V(s_{t+1})$$

Optimize with Gradient Descent

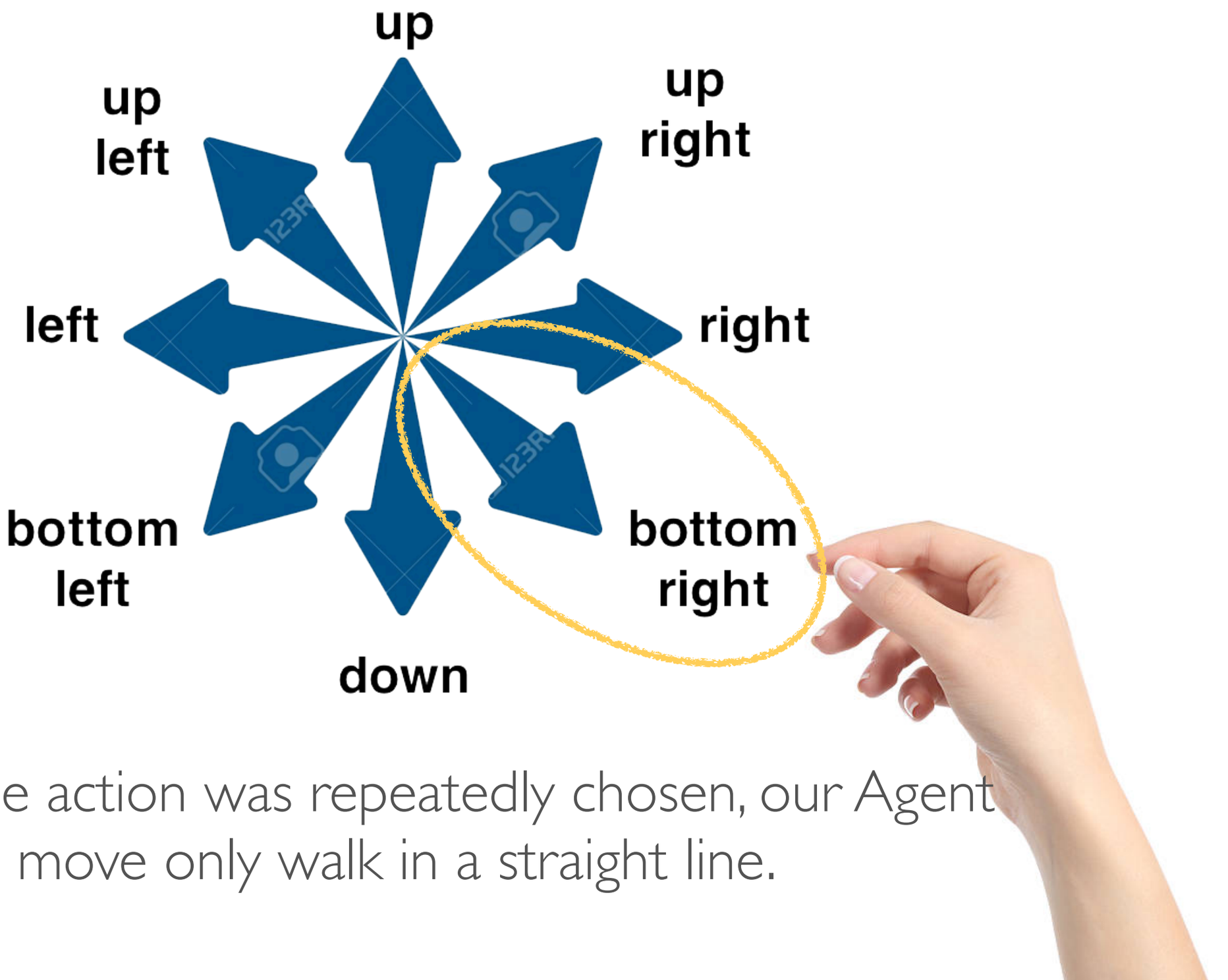


Policy function

$$\pi(a|s)$$

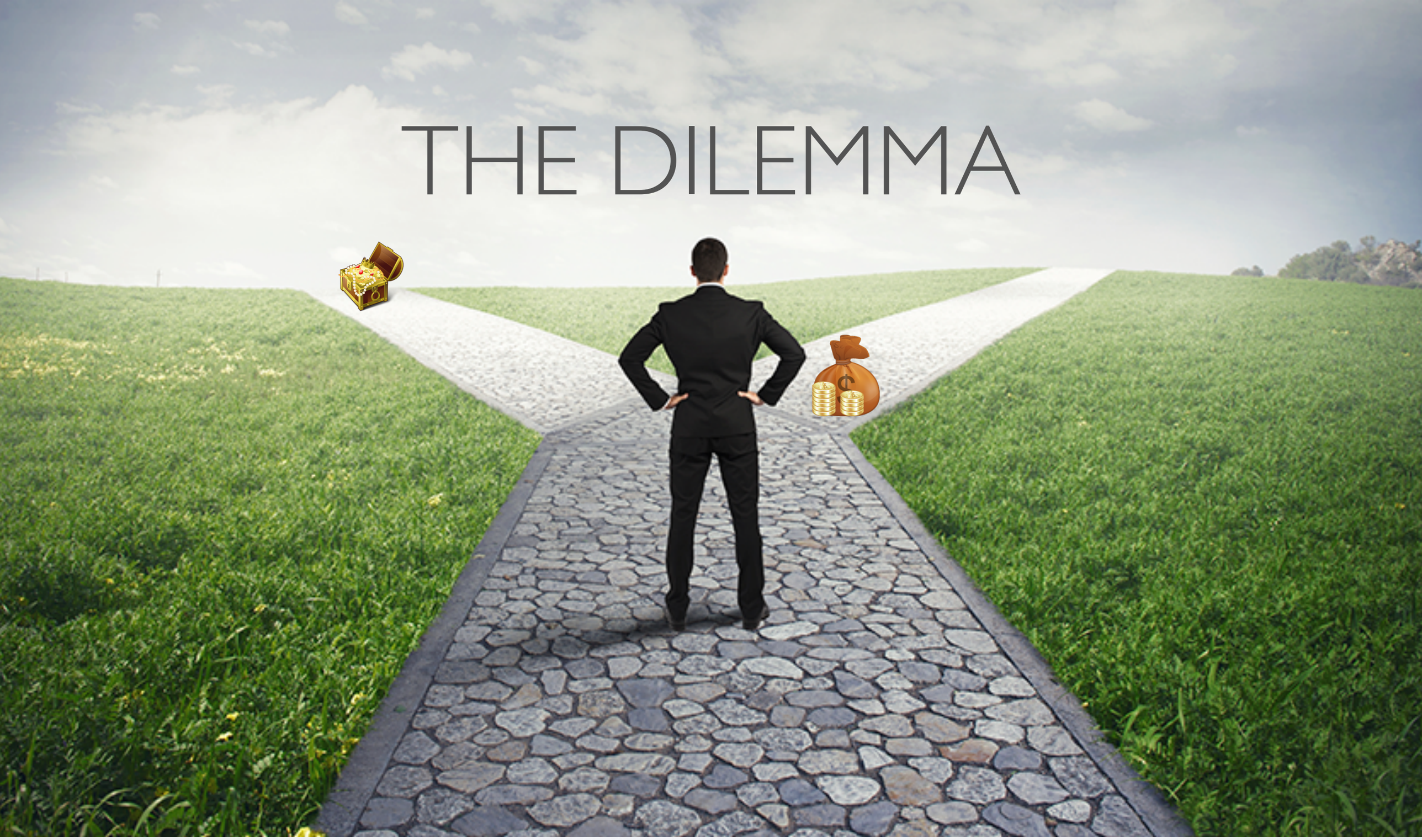
$$loss_\pi = -(r + \gamma V(s_{t+1}) - V(s$$

HIGHLY PEAKED POLICY FUNCTION DILEMMA



The same action was repeatedly chosen, our Agent move only walk in a straight line.

THE DILEMMA



Early convergence discourages exploration, if an action is over reinforced too early, we may be missing something that made another action better

EXPLORATION TUNING

Add an
Entropy Term

Adjust
Action Policy

Reward
Shaping

Add Exploration
Factor

Encourage exploration! Keep the search alive!
Prevent the agent from converging to a single action

Add an Entropy Term

Add an Entropy Term

Original Policy Network Loss Function:

$$loss_\pi = -[(r + \gamma V(s_{t+1}) - V(s_t)) \log \pi(a|s)$$

Add an Entropy Term

Uniform   Highly peaked

$$\mathrm{H}\big(\pi(\cdot\,|s)\big) = -Sum[\pi(\cdot\,|s)\log(\pi(\cdot\,|s) + \rho)]$$

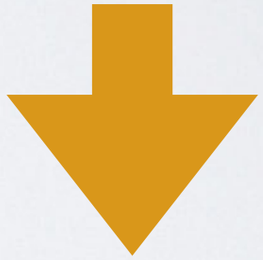
Original Policy Network Loss Function:

$$loss_\pi = -[(r + \gamma V(s_{t+1}) - V(s_t))\log\pi(a|s)$$

Add an Entropy Term

Uniform   Highly peaked

$$\mathrm{H}\big(\pi(\cdot\,|s)\big) = -Sum[\pi(\cdot\,|s)\log(\pi(\cdot\,|s)+\rho)]$$

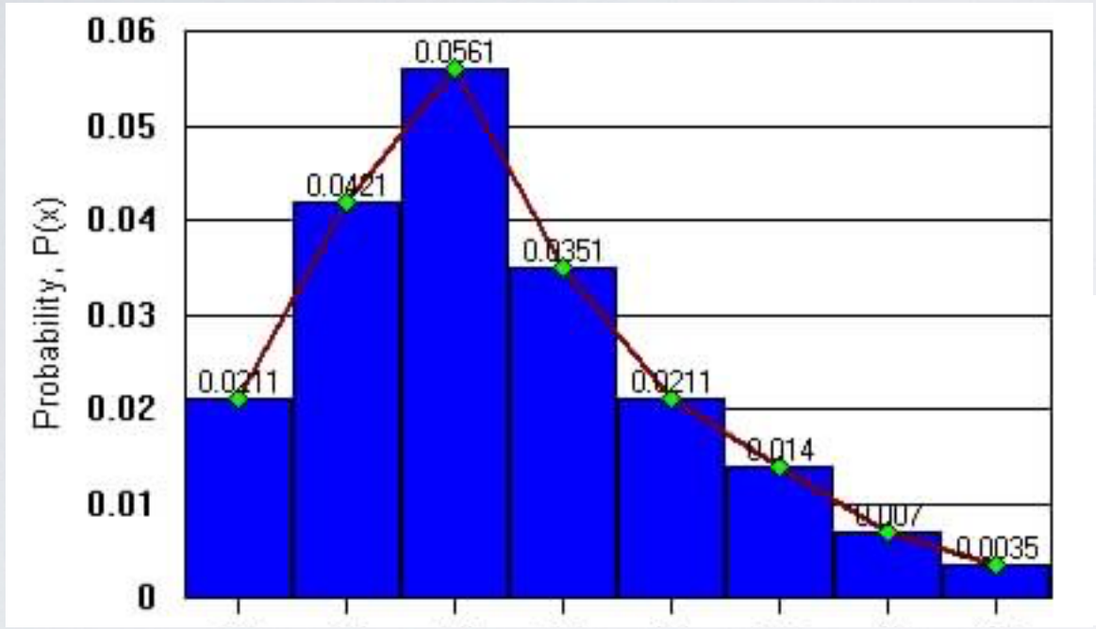


$$loss_\pi = -[(r+\gamma V(s_{t+1})-V(s_t))\log\pi(a|s)+\beta\mathrm{H}\big(\pi(\cdot\,|s)\big)]$$

maximizing the entropy term

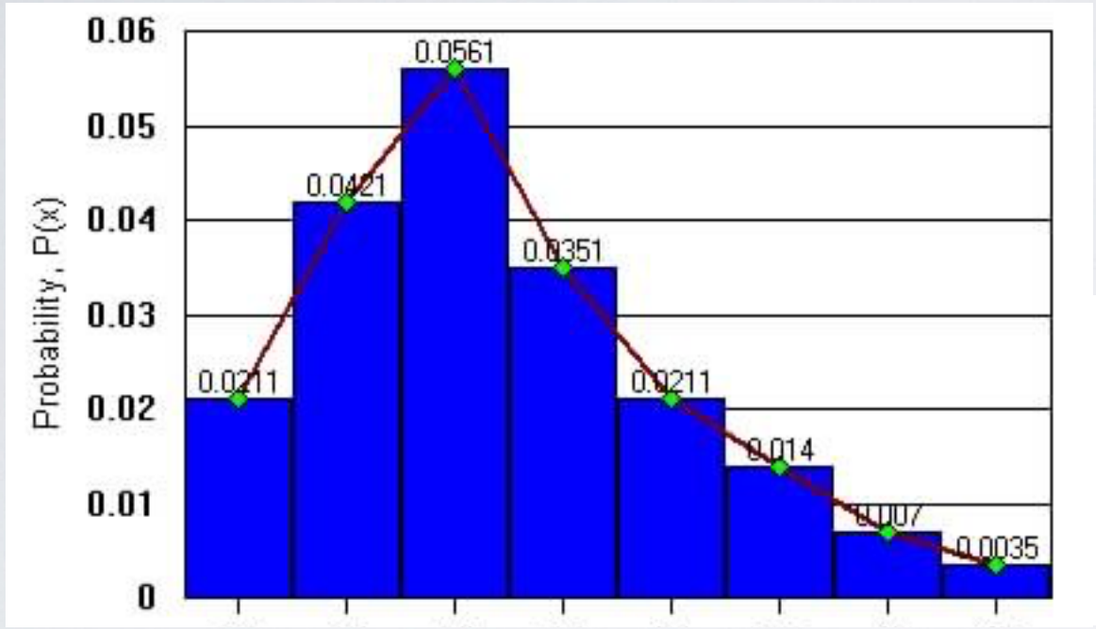
➡️ **encouraging uniform action probability**

Adjust
Action Policy



Adjust
Action Policy

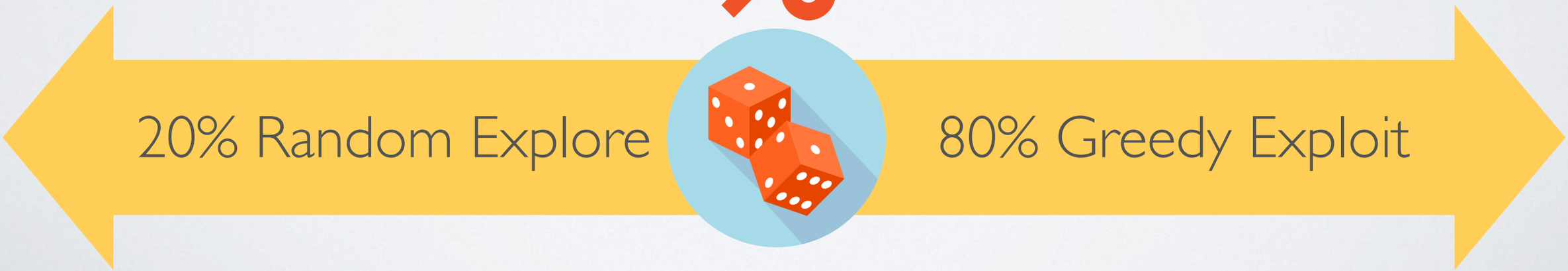
chosen randomly with respect to probability distribution



Adjust
Action Policy

chosen randomly with respect to probability distribution

Activate **Epsilon Greedy Policy During** Training



Reward
Shaping

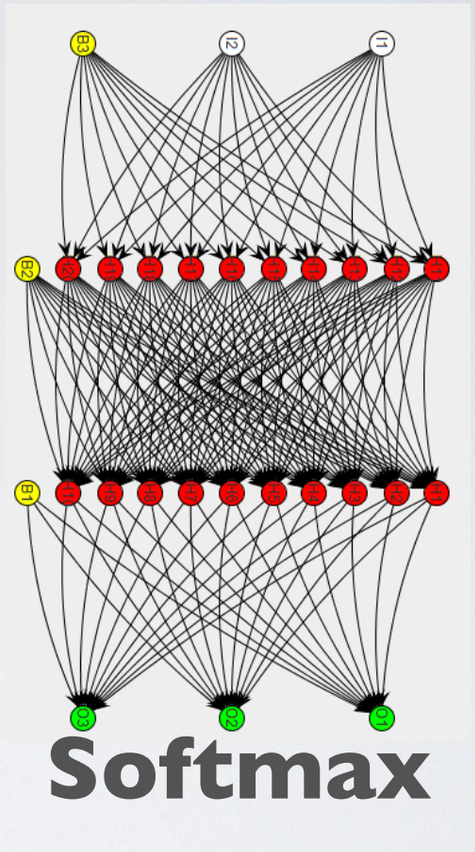
in addition to basic rewards, we apply immediate rewards:

Condition	**Reward, *r***
$r > 0$	$r \times = 2$
repeated action >3	$r -=$penalty, penalty+=0.25/repeated action
action != action’	$r += 1$
die	$r = -10$

Effective when rewards are sparse and delayed

Reward Shaping

Add Exploration Factor

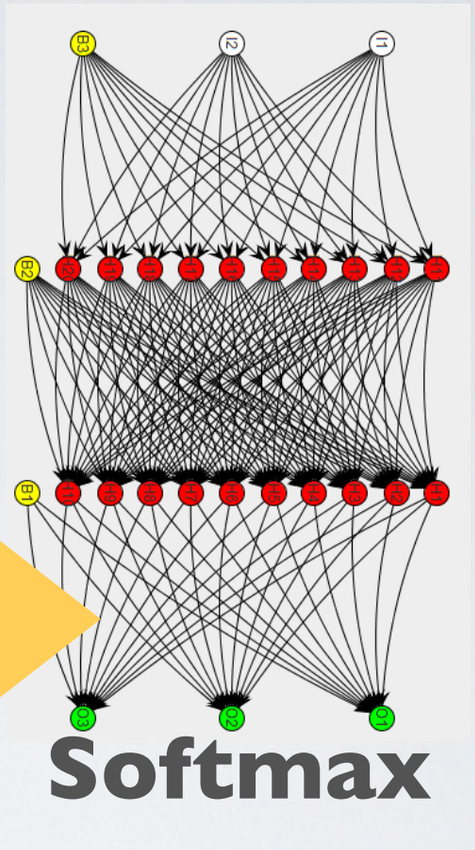


Softmax

Add Exploration Factor

0.2

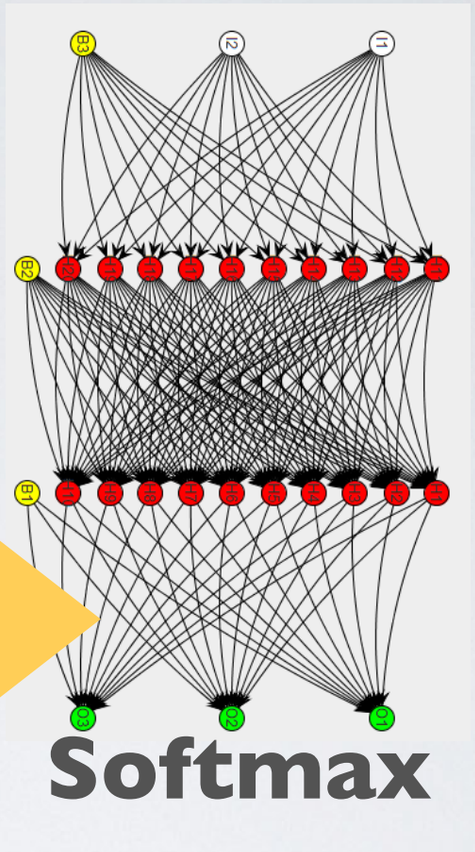
Exploration factor



Softmax

Add Exploration Factor

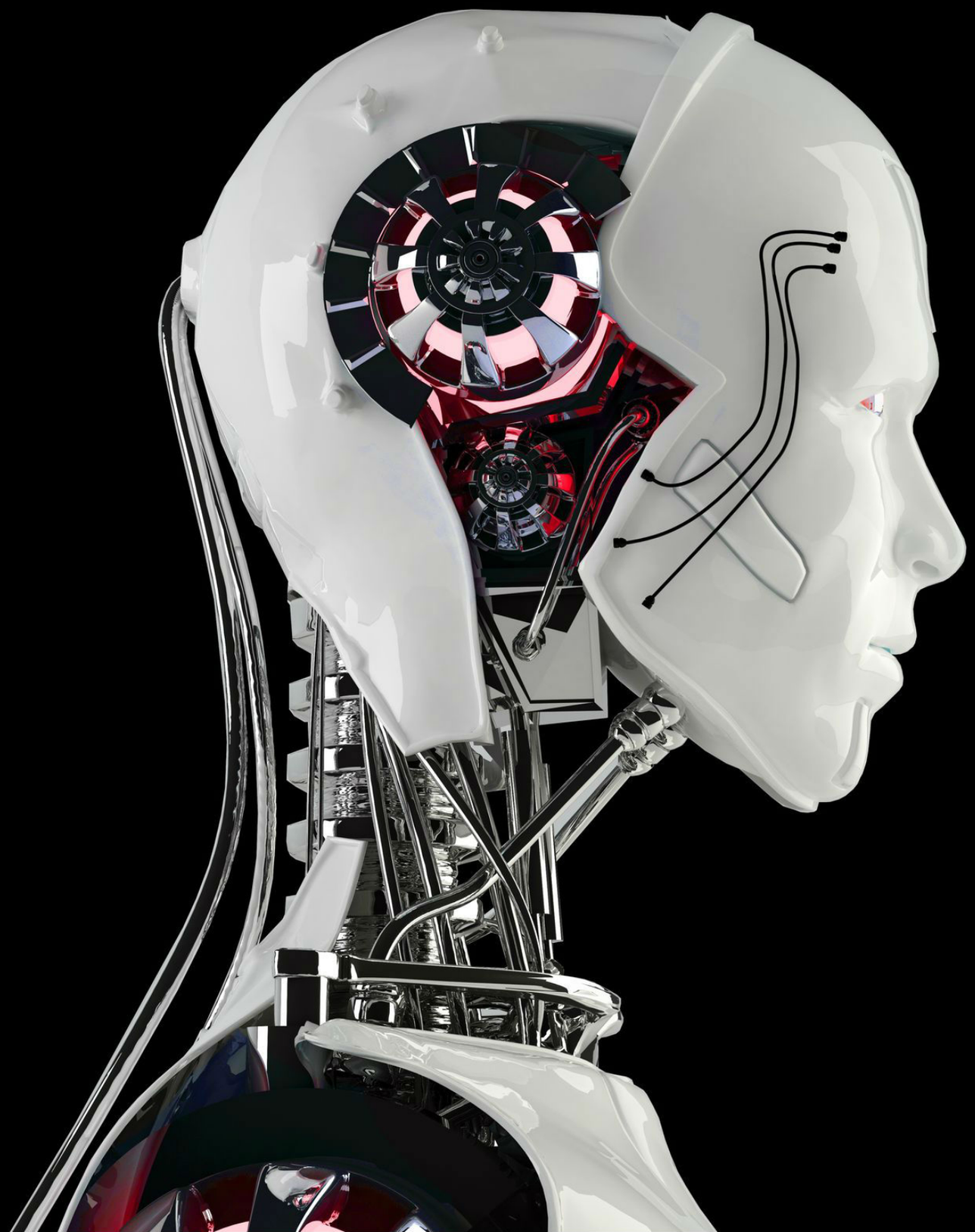
0.2 × Exploration factor



Softmax

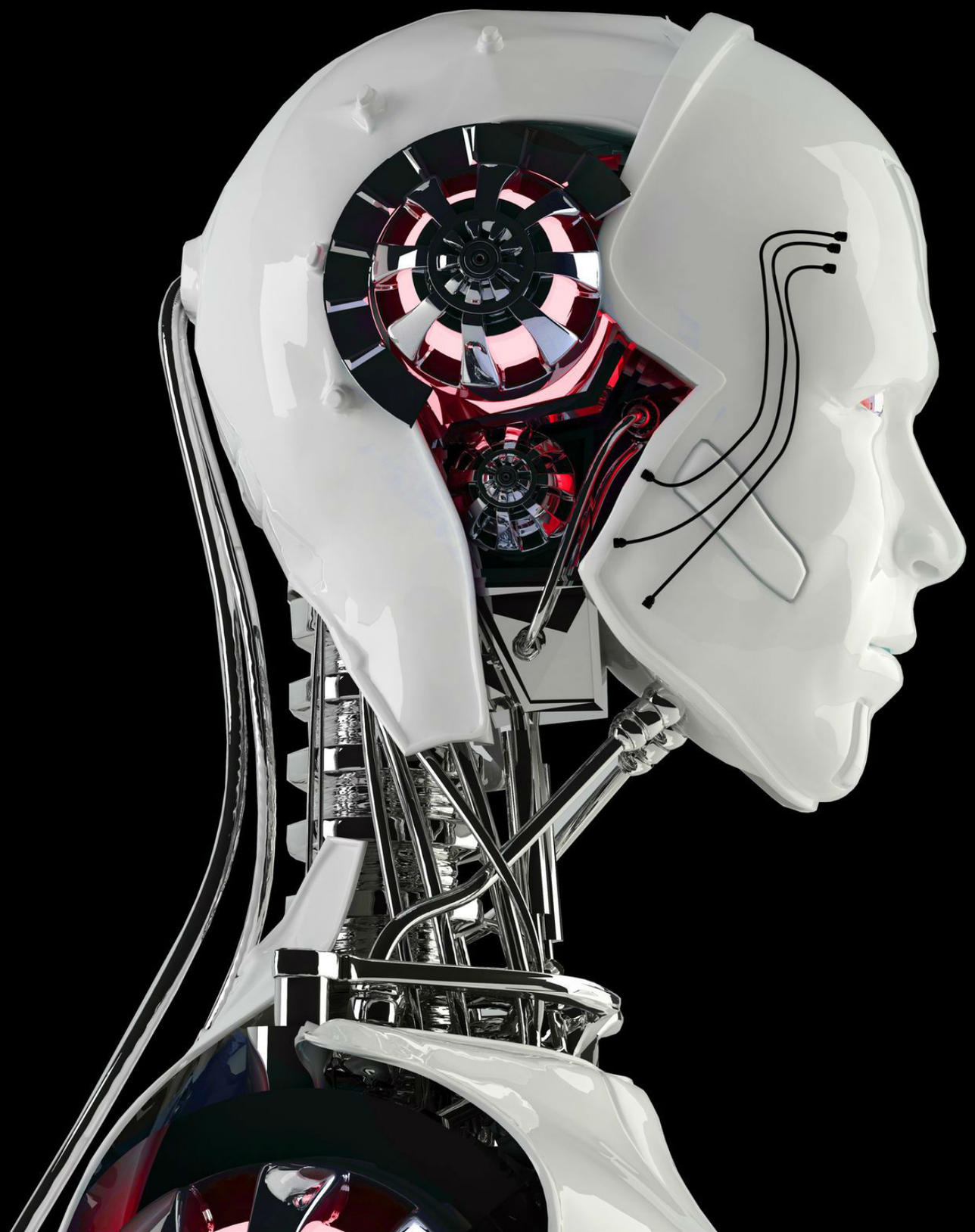
- scales down the affect of the highly-peaked policy function
- makes the output of the softmax more uniform

Add Exploration Factor



Well Trained Proficient Agent

Gif and video of the agents gameplay available



Well Trained Proficient Agent

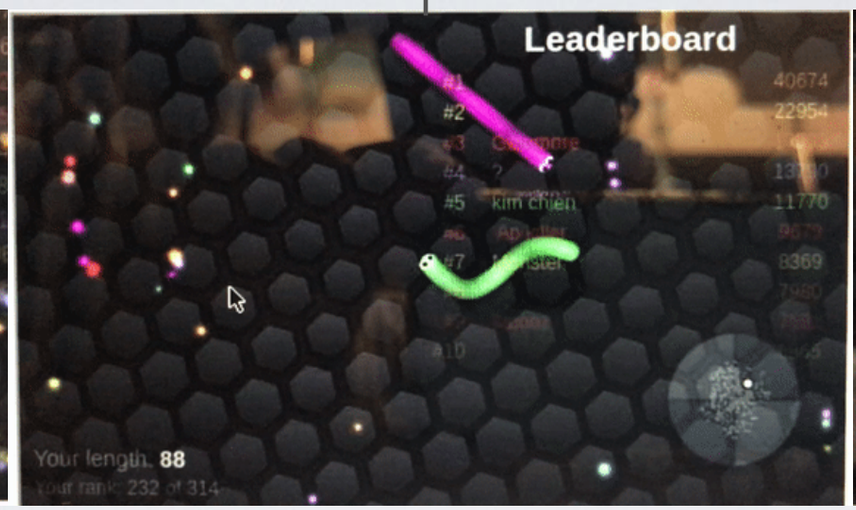
Gif and video of the agents gameplay available

Applications

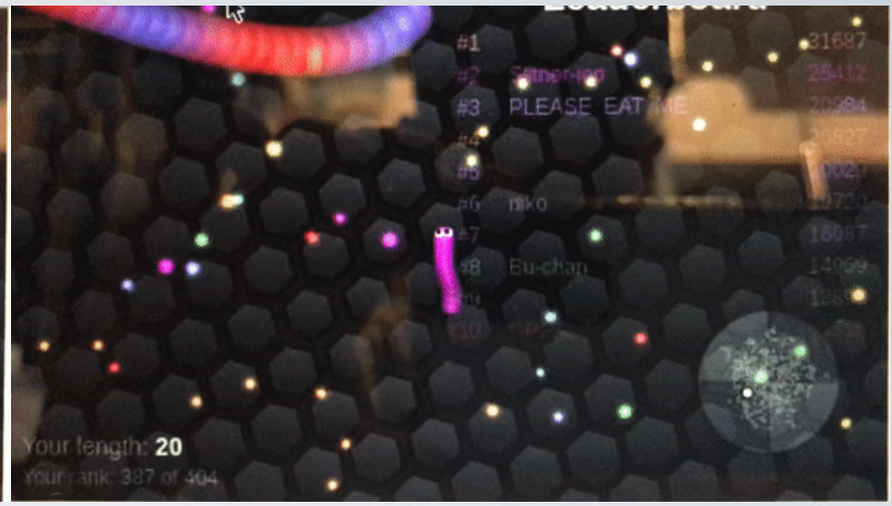
- RL used in NLP for training QA models
- One Day, agents will be able to play the largest game in the world - life!

Normal turn

Sharp Turn



Hard Twist

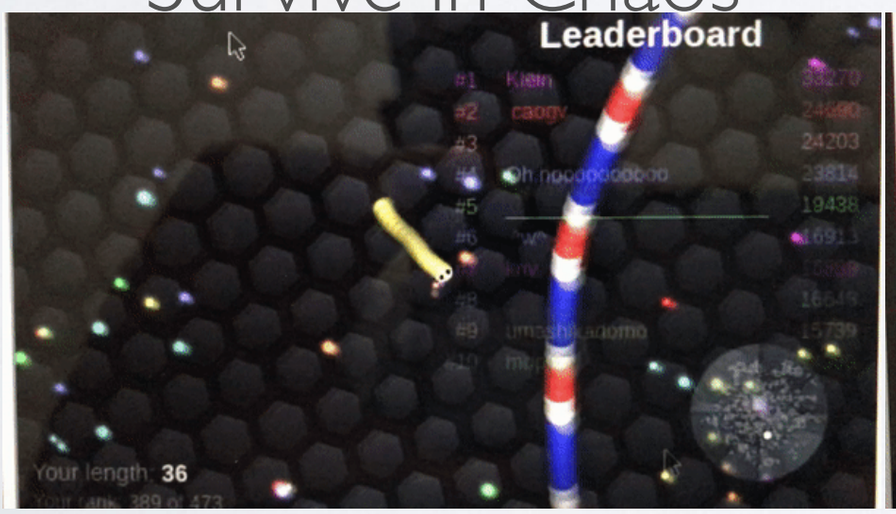


Snake Follow

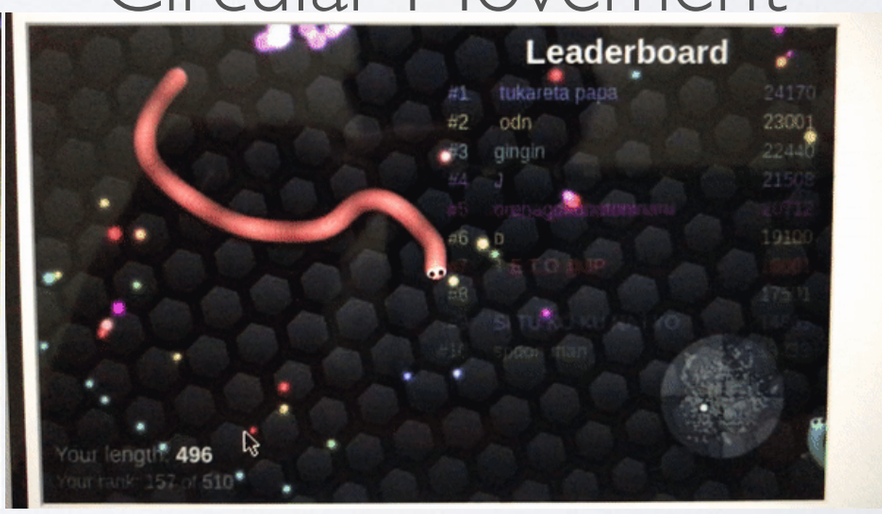
Kill Enemies

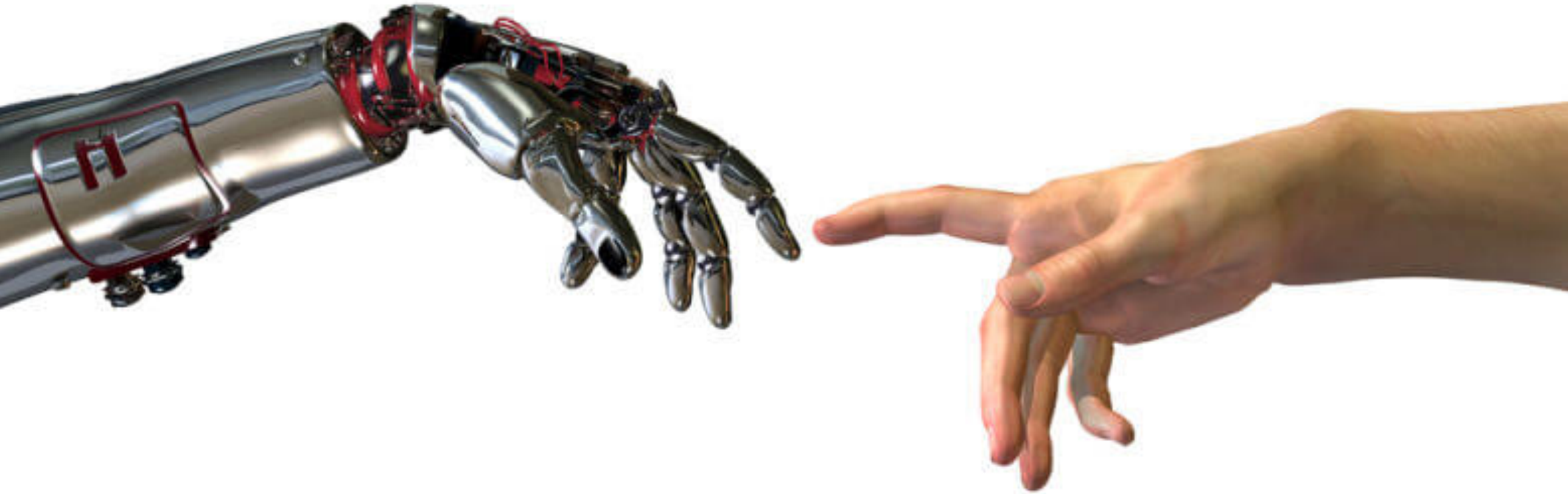
Bypass Enemies

Survive in Chaos



Circular Movement





THANK YOU

for your attention!