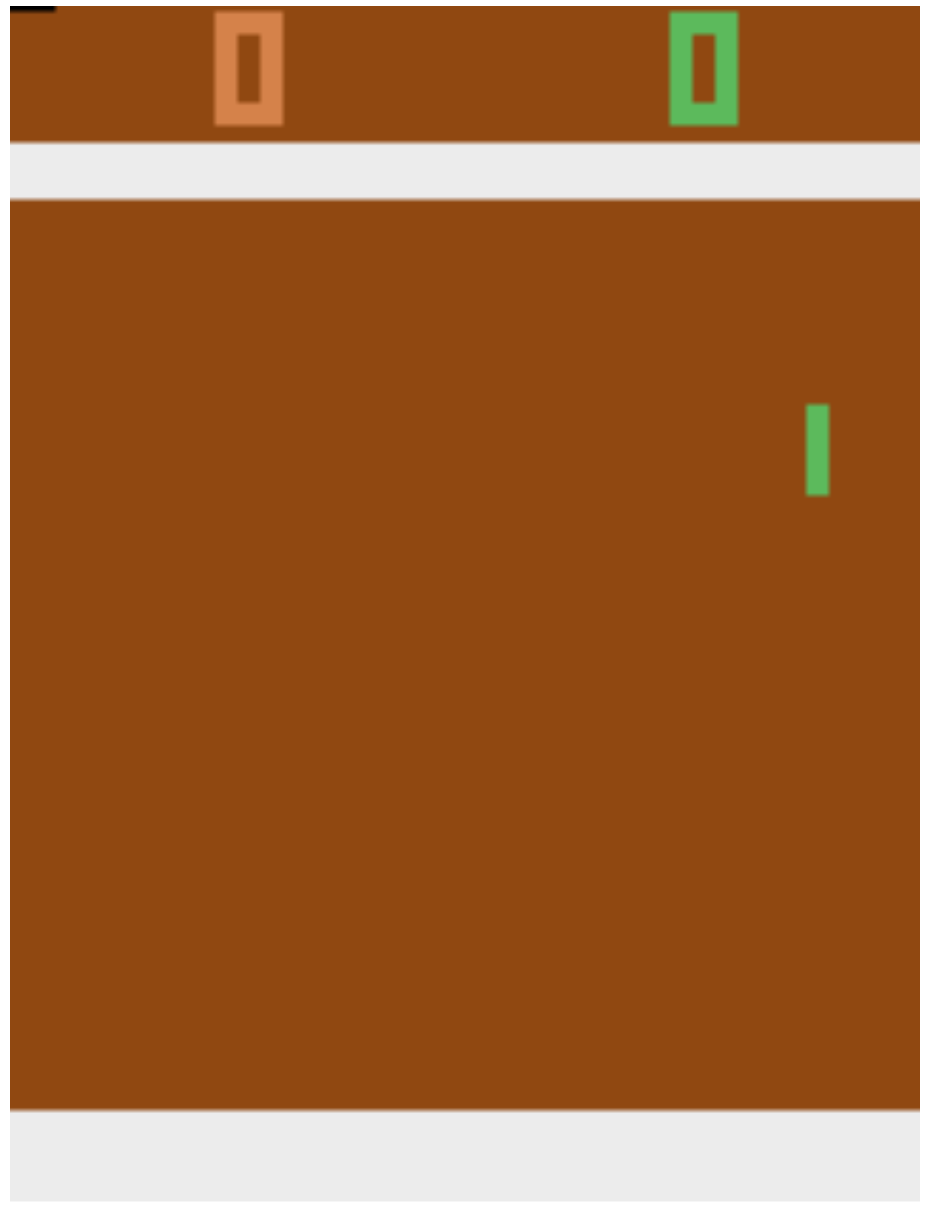
Robust Deep Reinforcement Learning through Bootstrapped Opportunistic Curriculum

Junlin Wu and Yevgeniy Vorobeychik

Background

Deep Reinforcement Learning

A Markov decision process (MDP) is defined as (S, A, R, p, γ)

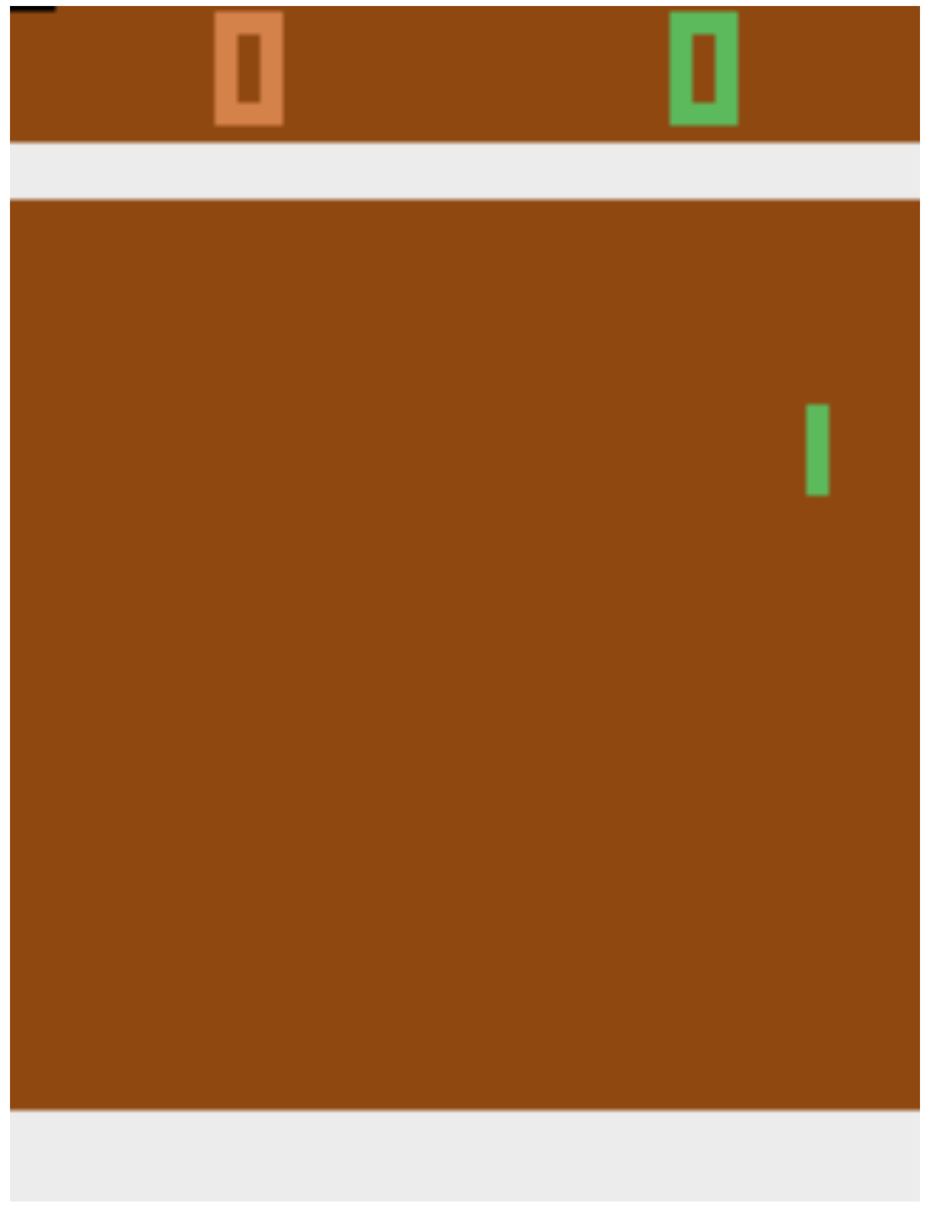


Background

Deep Reinforcement Learning

A Markov decision process (MDP) is defined as (S, A, R, p, γ)

- S is the state space

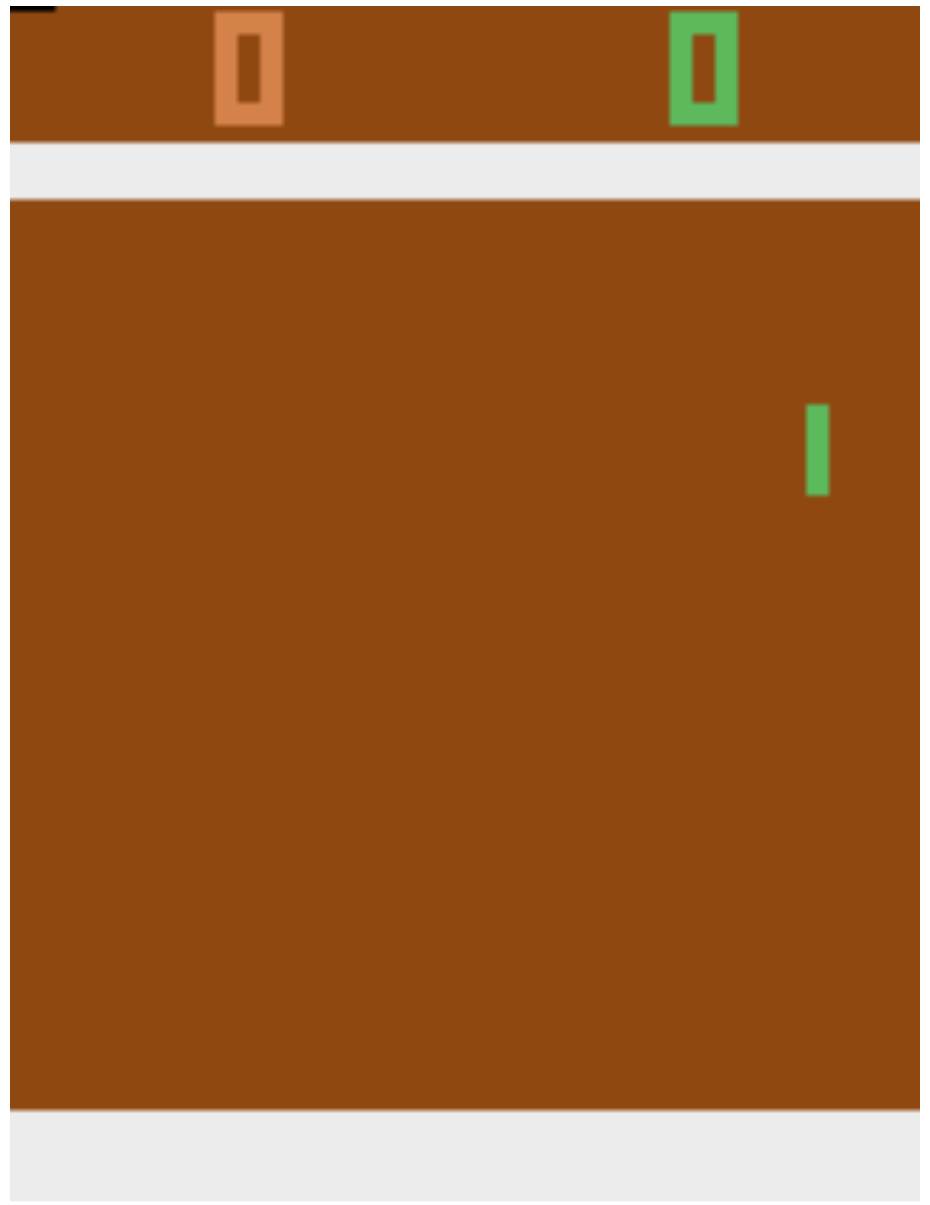


Background

Deep Reinforcement Learning

A Markov decision process (MDP) is defined as (S, A, R, p, γ)

- S is the state space
- A is the action space

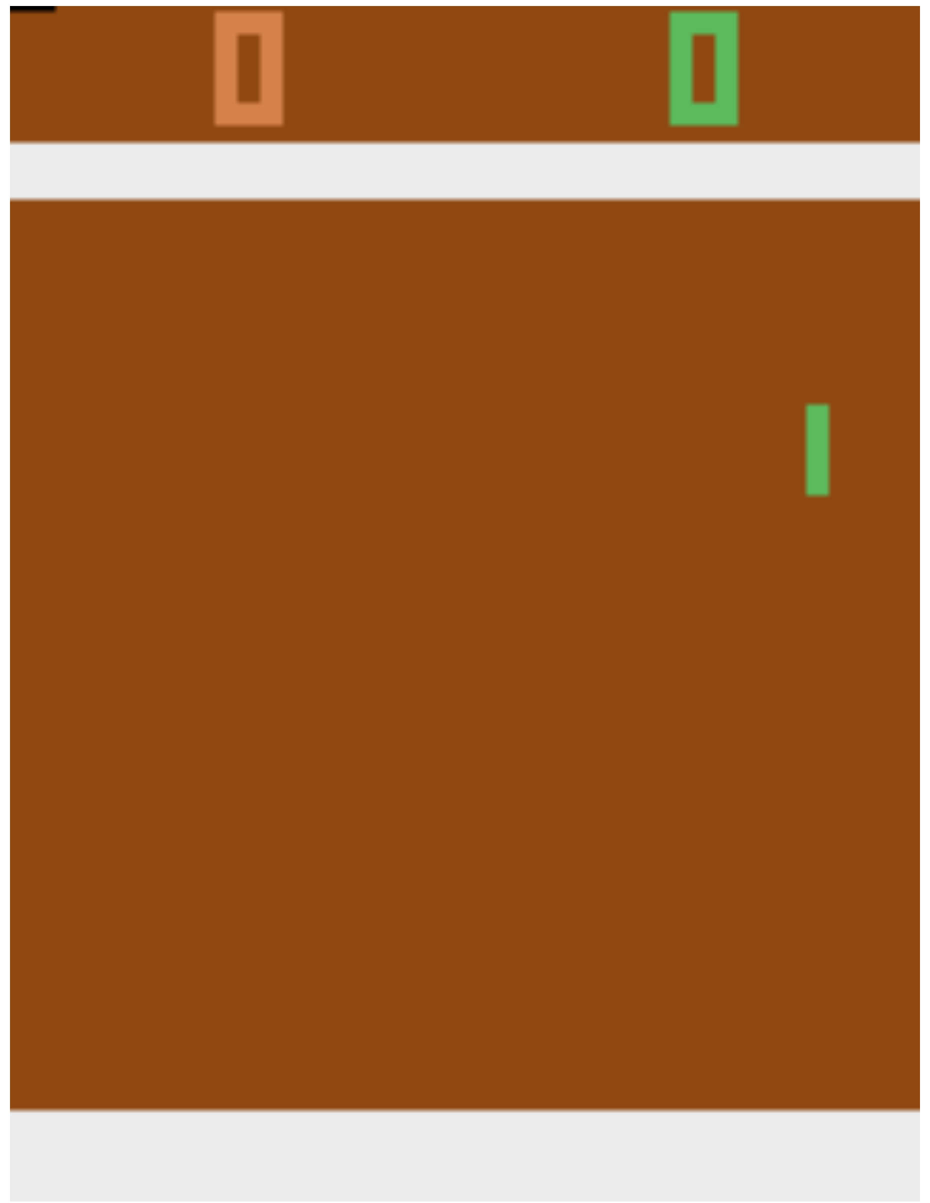


Background

Deep Reinforcement Learning

A Markov decision process (MDP) is defined as (S, A, R, p, γ)

- S is the state space
- A is the action space
- $p : S \times A \rightarrow P(S)$ is the transition probability of environment

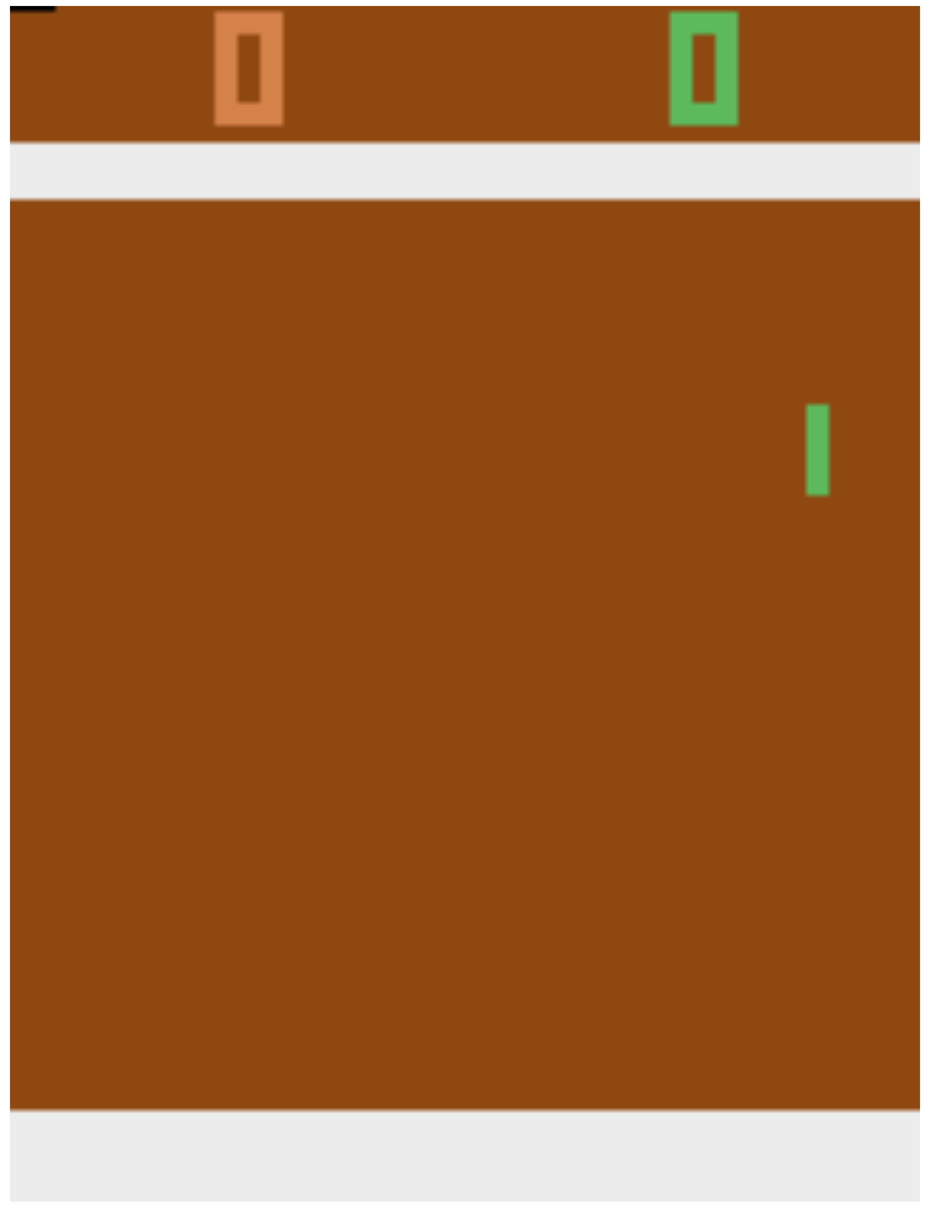


Background

Deep Reinforcement Learning

A Markov decision process (MDP) is defined as (S, A, R, p, γ)

- S is the state space
- A is the action space
- $p : S \times A \to P(S)$ is the transition probability of environment
- $R : S \times A \times S \to R$ is the reward function

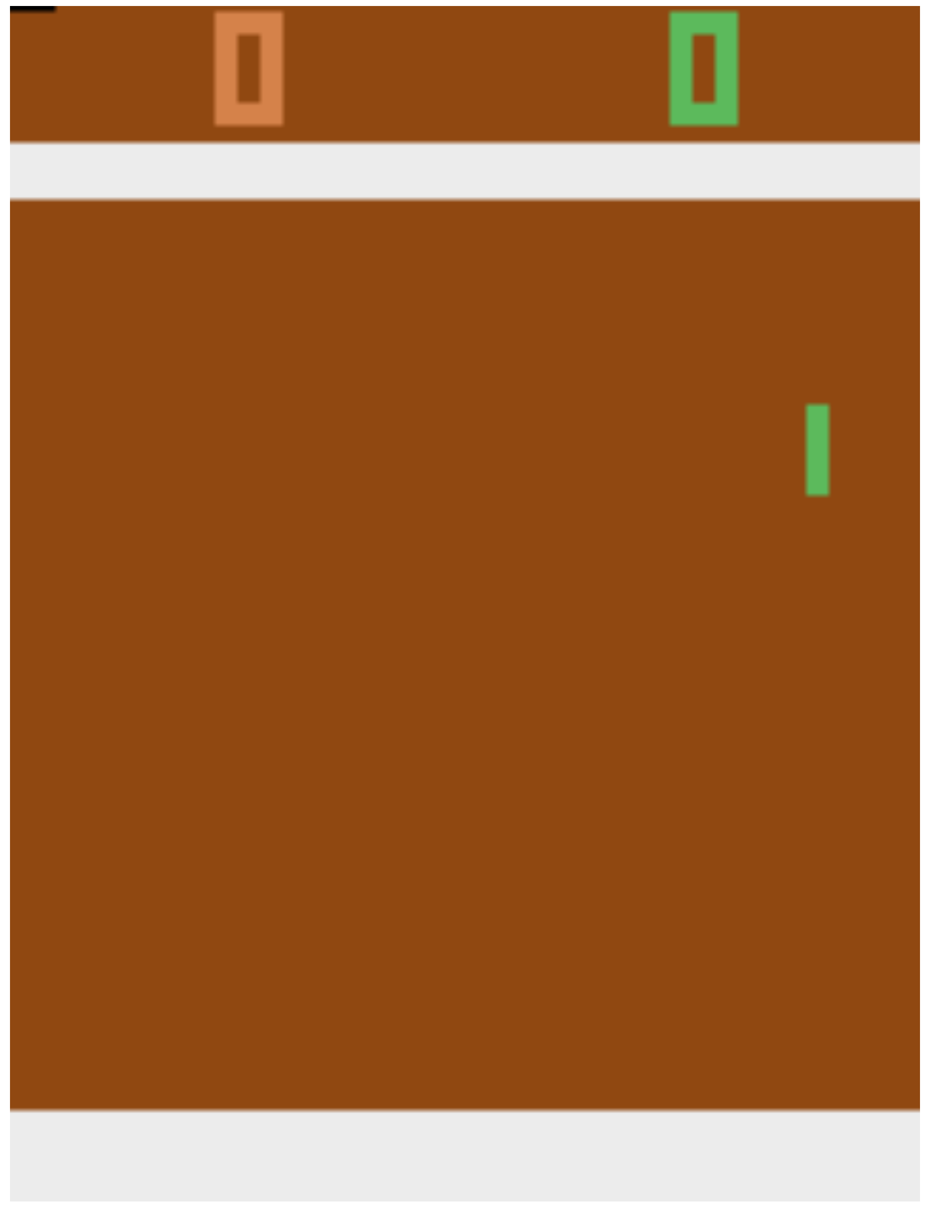


Background

Deep Reinforcement Learning

A Markov decision process (MDP) is defined as (S, A, R, p, γ)

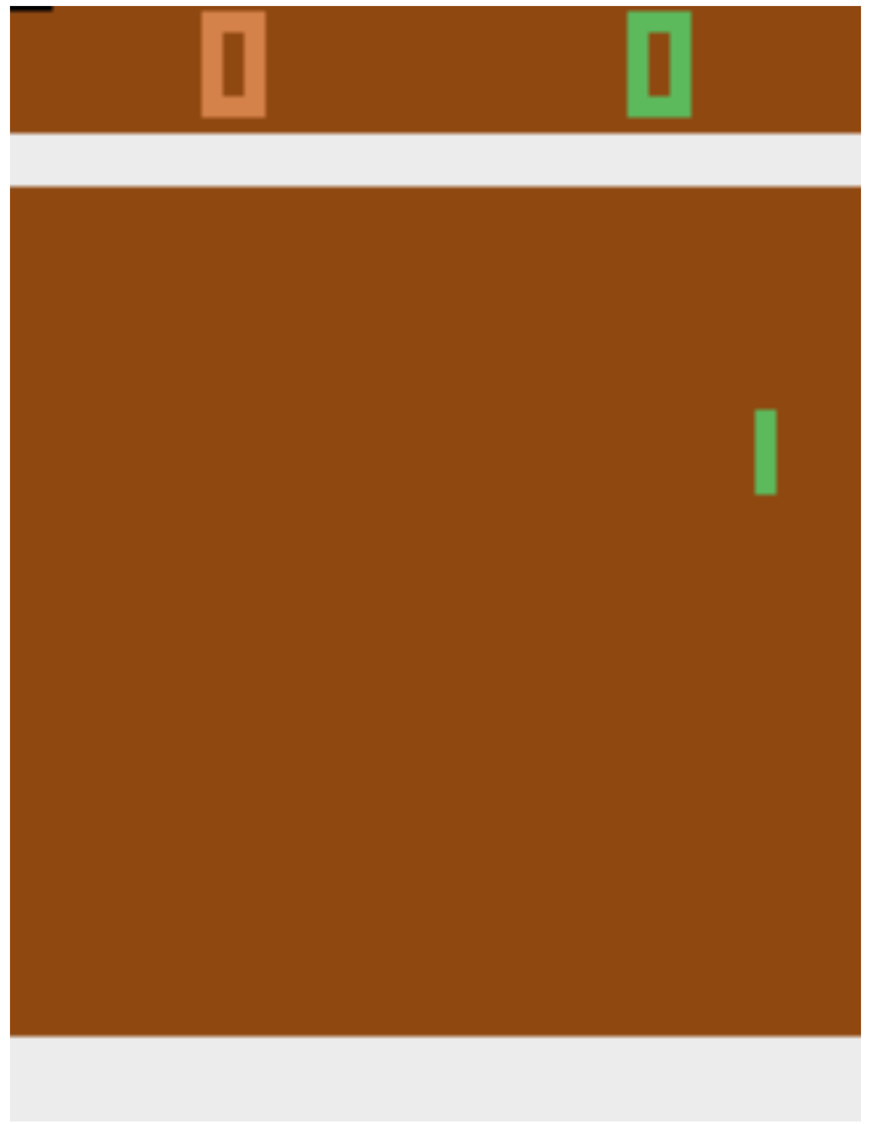
- S is the state space
- A is the action space
- $p : S \times A \rightarrow P(S)$ is the transition probability of environment
- $R : S \times A \times S \rightarrow R$ is the reward function
- γ is the discount factor



Background

Adversarial Deep Reinforcement Learning

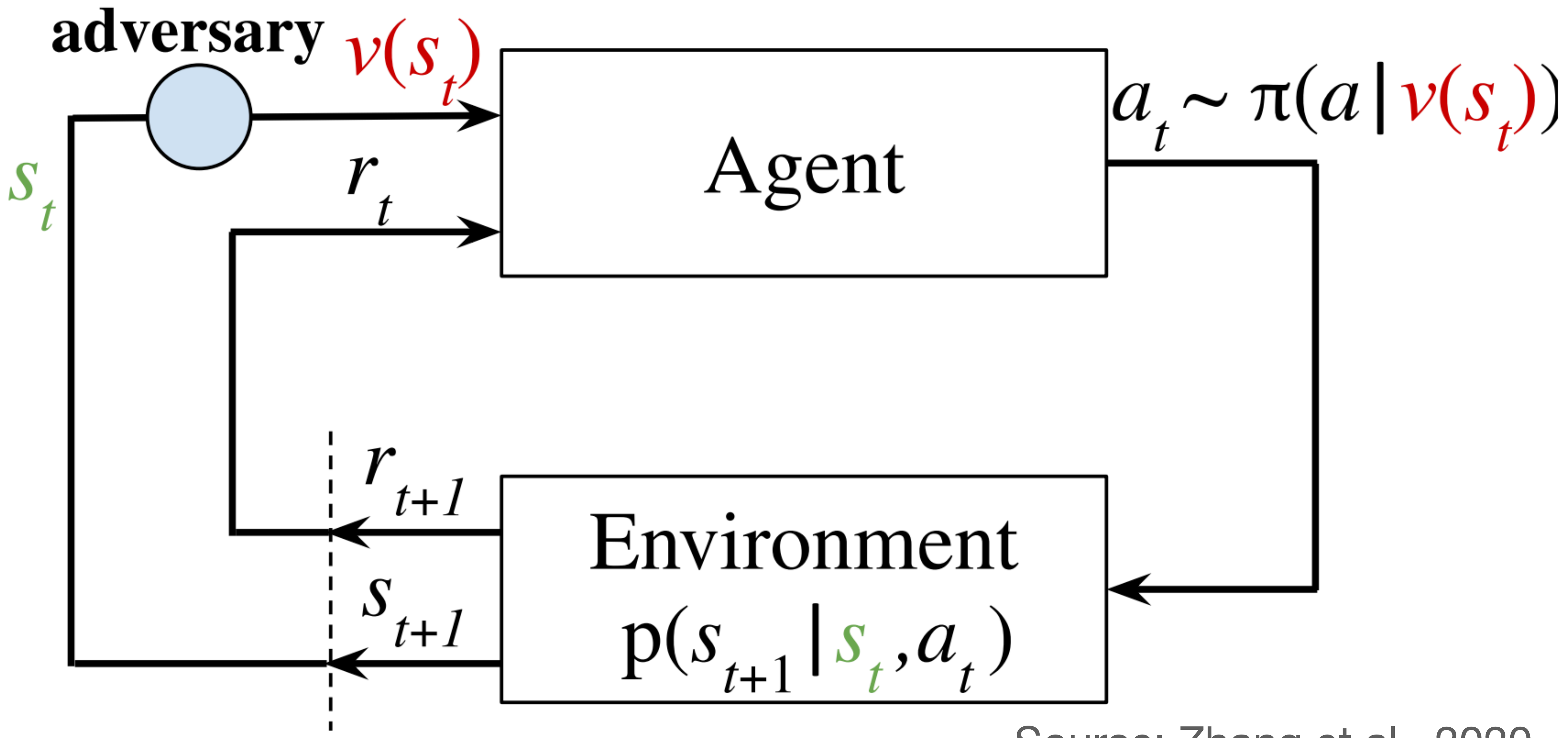
- The adversarial will add perturbation δ to the state (s) perceived by the agent



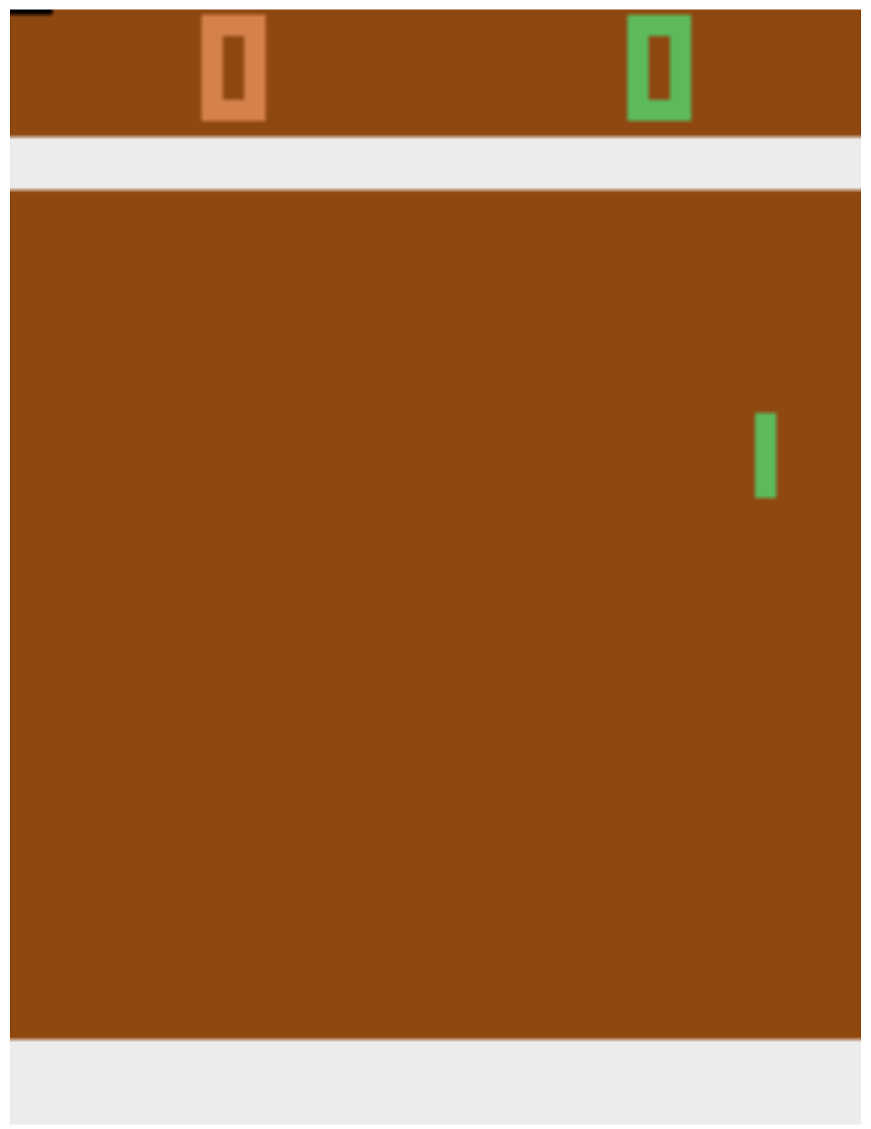
Background

Adversarial Deep Reinforcement Learning

- The adversarial will add perturbation δ to the state (s) perceived by the agent
- $\nu(s) = s + \delta,\ \|\delta\|_p \leq \epsilon$

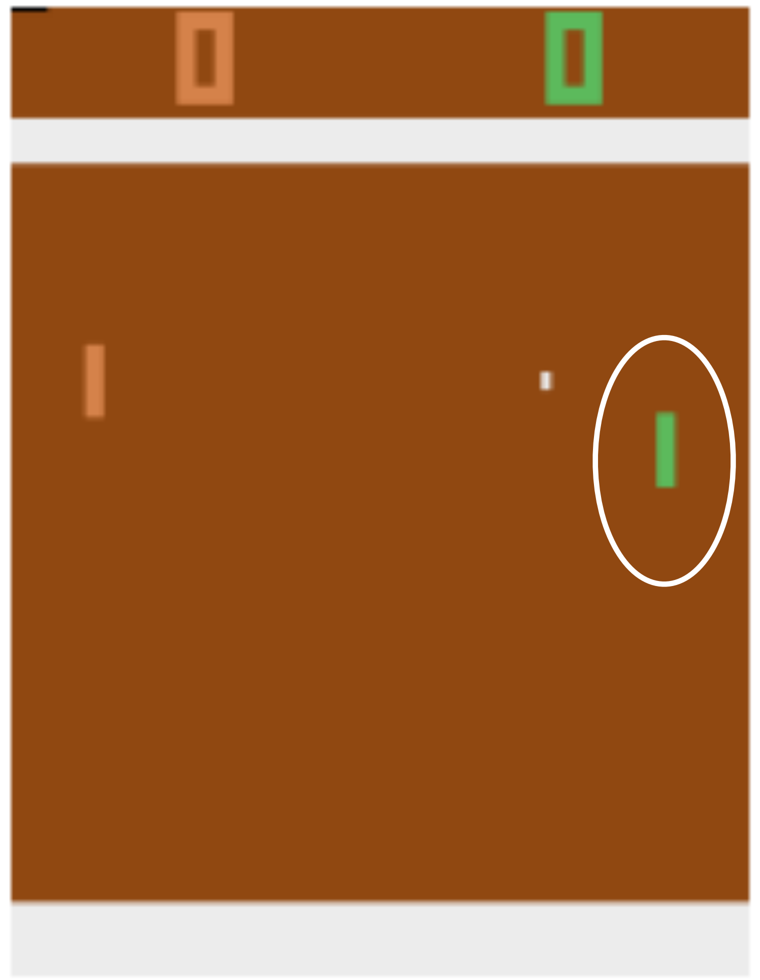


Source: Zhang et al., 2020



Background

Adversarial Deep Reinforcement Learning

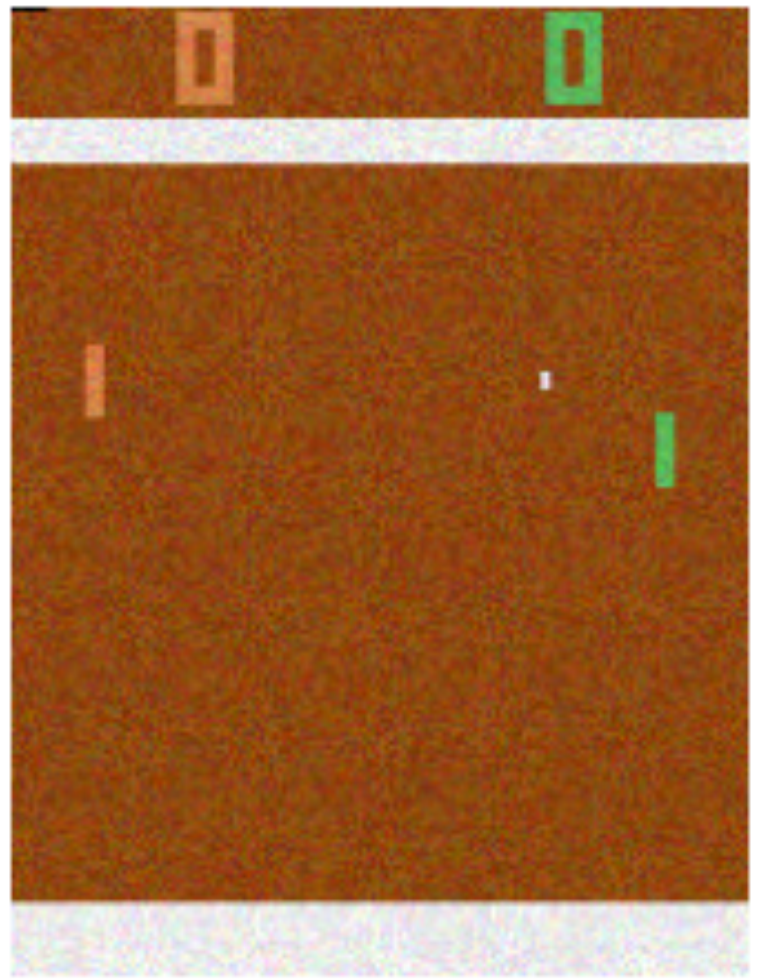


Original Image Input

NN

Action: Move up

Attacker



+ Adversarial Perturbation

NN

Action: Move Down

Background

Attacking Method

- A common attack on Deep Q-Network (DQN) aims maximize cross-entropy loss $\mathscr{L}(\mathrm{Softmax}(Q(s+\delta;\theta)), \pi(s))$ with respect to δ (adversarial perturbation), where $Q(s)$ is the vector of Q values over all actions in state s.

Background

Attacking Method

- A common attack on Deep Q-Network (DQN) aims maximize cross-entropy loss $\mathscr{L}(\text{Softmax}(Q(s+\delta;\theta)), \pi(s))$ with respect to δ (adversarial perturbation), where $Q(s)$ is the vector of Q values over all actions in state s.

- A **PGD** (projected gradient descent) attack updates δ iteratively:
$\delta_{k+1} \leftarrow \delta_k + \alpha \cdot \text{sign}(\nabla_\delta \mathscr{L}(Q(x+\delta_k;\theta), \pi(s)))$
over a fixed number of iterations with $||\delta||_\infty \leq \epsilon$.

Background

Attacking Method

- A common attack on Deep Q-Network (DQN) aims maximize cross-entropy loss $\mathscr{L}(\mathrm{Softmax}(Q(s+\delta;\theta)),\pi(s))$ with respect to δ (adversarial perturbation), where $Q(s)$ is the vector of Q values over all actions in state s.

- A **PGD** (projected gradient descent) attack updates δ iteratively:
$\delta_{k+1} \leftarrow \delta_k + \alpha \cdot \mathrm{sign}(\nabla_\delta \mathscr{L}(Q(x+\delta_k;\theta),\pi(s)))$
over a fixed number of iterations with $||\delta||_\infty \leq \epsilon$.

- A special class of PGD is **FGSM** (fast gradient sign method), where PGD is executed for only a single iteration and $\alpha = \epsilon$.

Prior Literature

Adversarial Deep Reinforcement Learning

- The goal is to train a robust RL agent (i.e., achieve a high reward when under adversarial attack $||\delta||_\infty \leq \epsilon$).

Prior Literature

Adversarial Deep Reinforcement Learning

- The goal is to train a robust RL agent (i.e., achieve a high reward when under adversarial attack $||\delta||_\infty \leq \epsilon$).
- The SOTA method is RADIAL [Oikarinen et al., 2021], where they leveraged interval bound propagation to increase the robustness of RL agent (e.g., robust up to 5/255 for Pong game).

Prior Literature

Adversarial Deep Reinforcement Learning

- The goal is to train a robust RL agent (i.e., achieve a high reward when under adversarial attack $||\delta||_\infty \leq \epsilon$).
- The SOTA method is RADIAL [Oikarinen et al., 2021], where they leveraged interval bound propagation to increase the robustness of RL agent (e.g., robust up to 5/255 for Pong game).
- Our goal is to increase the robustness of the RL agent further (robust against higher values of ϵ).

BCL Framework

Overview

- We propose ***B**ootstrapped Opportunistic Adversarial **C**urriculum **L**earning* (BCL), a novel flexible adversarial curriculum learning framework for robust reinforcement learning.

BCL Framework
Overview

- We propose ***B**ootstrapped Opportunistic Adversarial **C**urriculum **L**earning* (BCL), a novel flexible adversarial curriculum learning framework for robust reinforcement learning.

- Begins by creating a **baseline curriculum**: an increasing sequence of L attack budgets $\{\epsilon_i\}$, with $\epsilon_1 < \epsilon_2 < \cdots < \epsilon_L$, where $\epsilon_L = \epsilon$ is our target robustness level.

BCL Framework

Overview

- We propose ***B**ootstrapped Opportunistic Adversarial **C**urriculum **L**earning* (BCL), a novel flexible adversarial curriculum learning framework for robust reinforcement learning.
- Begins by creating a **baseline curriculum**: an increasing sequence of L attack budgets $\{\epsilon_i\}$, with $\epsilon_1 < \epsilon_2 < \cdots < \epsilon_L$, where $\epsilon_L = \epsilon$ is our target robustness level.
- In each curriculum phase, we run adversarial training (AT) **up to K times**, where each AT run is bootstrapped by the best model obtained thus far.

BCL Framework

Overview

- We propose ***B**ootstrapped Opportunistic Adversarial **C**urriculum **L**earning* (BCL), a novel flexible adversarial curriculum learning framework for robust reinforcement learning.
- Begins by creating a **baseline curriculum**: an increasing sequence of L attack budgets $\{\epsilon_i\}$, with $\epsilon_1 < \epsilon_2 < \cdots < \epsilon_L$, where $\epsilon_L = \epsilon$ is our target robustness level.
- In each curriculum phase, we run adversarial training (AT) **up to K times**, where each AT run is bootstrapped by the best model obtained thus far.
- For example, based on observed performance, we could speed up the training by
  - Performing fewer than K runs for each curriculum phases;
  - Skipping forward the curriculum phases.

Adversarial Loss Function

We experiment on two types of adversarial loss functions:

Adversarial Loss Function

We experiment on two types of adversarial loss functions:

- RADIAL method:

  Use **interval bound propagation** (RADIAL-DQN [Oikarinen et al., 2021]), which is to minimize the weighted overlapped IBP Q-values.

Adversarial Loss Function

We experiment on two types of adversarial loss functions:

- RADIAL method:

  Use **interval bound propagation** (RADIAL-DQN [Oikarinen et al., 2021]), which is to minimize the weighted overlapped IBP Q-values.

- AT method:

  Use FGSM-based method and leverage the structure of Double-DQN to generate **adversarial examples** efficiently during training time.

Adversarial Loss Function

We experiment on two types of adversarial loss functions:

- RADIAL method:

  Use **interval bound propagation** (RADIAL-DQN [Oikarinen et al., 2021]), which is to minimize the weighted overlapped IBP Q-values.

- AT method:

  Use FGSM-based method and leverage the structure of Double-DQN to generate **adversarial examples** efficiently during training time.

$$\min_{||\delta||_\infty \leq \epsilon} \mathrm{Softmax}(Q_{\mathrm{actor}}(s+\delta)) \odot Q_{\mathrm{target}}(s)$$

BCL Framework

Special Cases of BCL

- AT-DQN (Adversarial Training)
- NCL-AT/RADIAL-DQN (Naive Curriculum Learning)

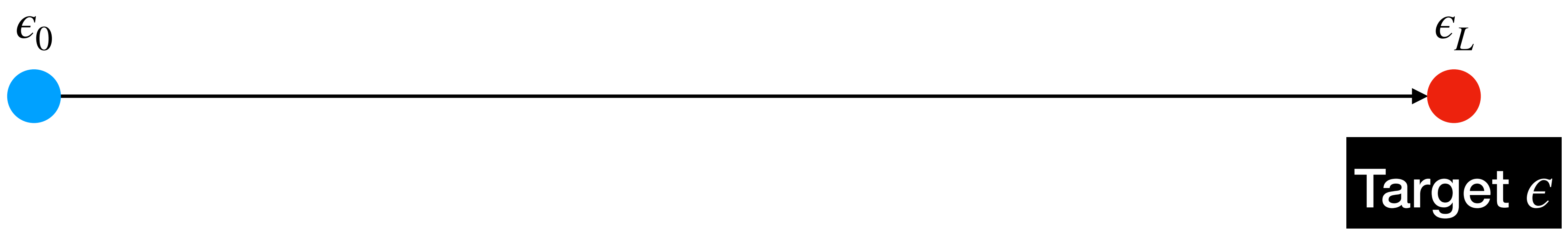
Benchmark Models

- BCL-C-AT-DQN (Conservatively Bootstrapped Curriculum Learning)
- BCL-MOS-AT-DQN (Maximum Opportunistic Skipping)
- BCL-RADIAL-DQN (BCL with RADIAL approach)
- BCL-RADIAL+AT-DQN (BCL-RADIAL-DQN + BCL-C-AT-DQN)

BCL Framework

Special Cases of BCL

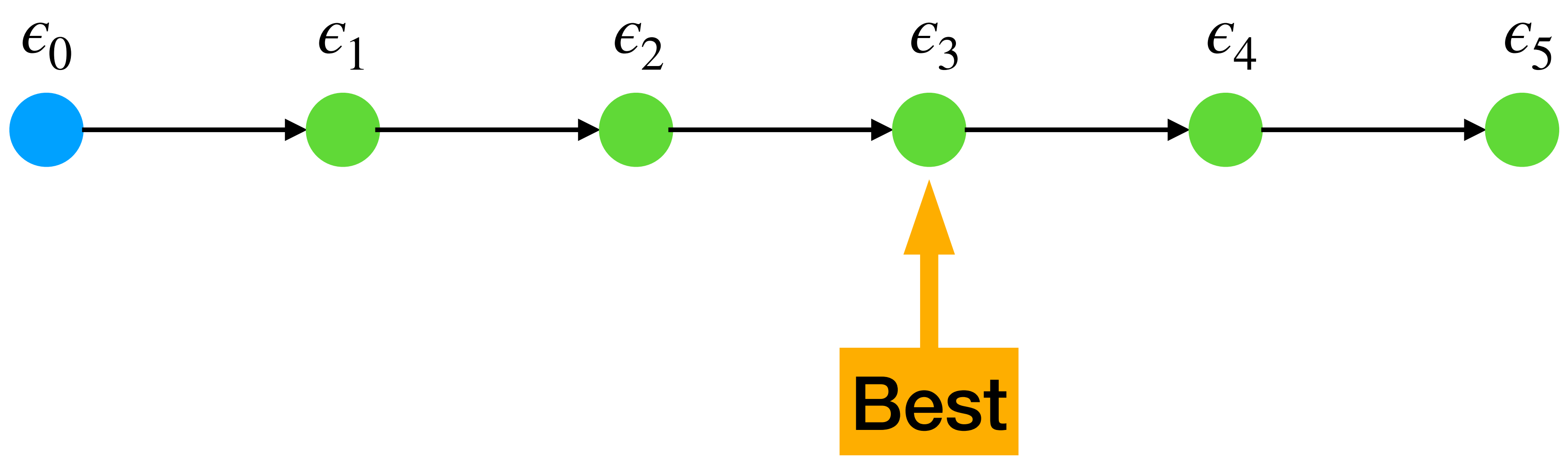
- AT-DQN (Adversarial Training)



BCL Framework

Special Cases of BCL

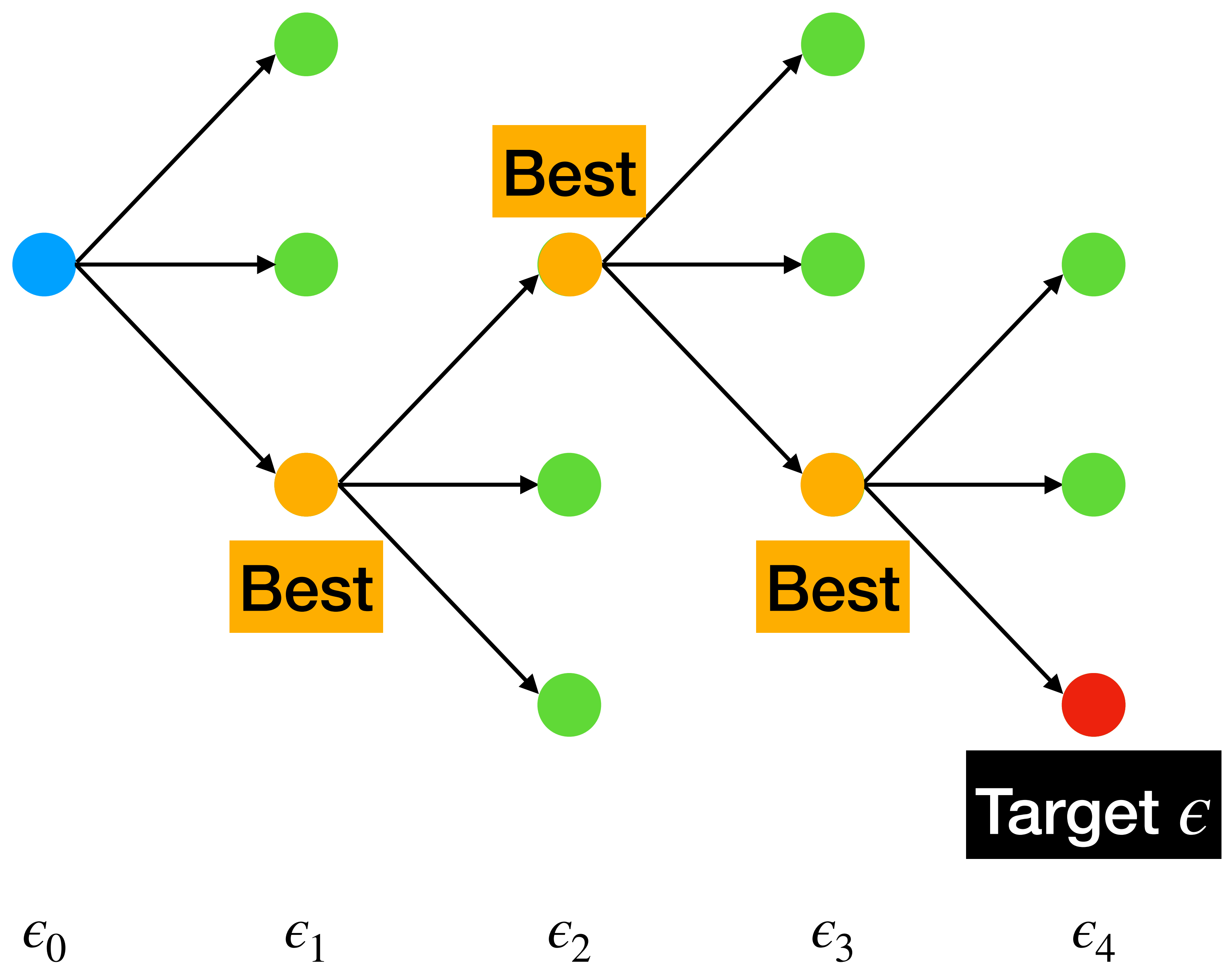
- AT-DQN (Adversarial Training)
- NCL-AT/RADIAL-DQN (Naive Curriculum Learning)



BCL Framework

Special Cases of BCL

- BCL-C-AT-DQN (Conservatively Bootstrapped Curriculum Learning)



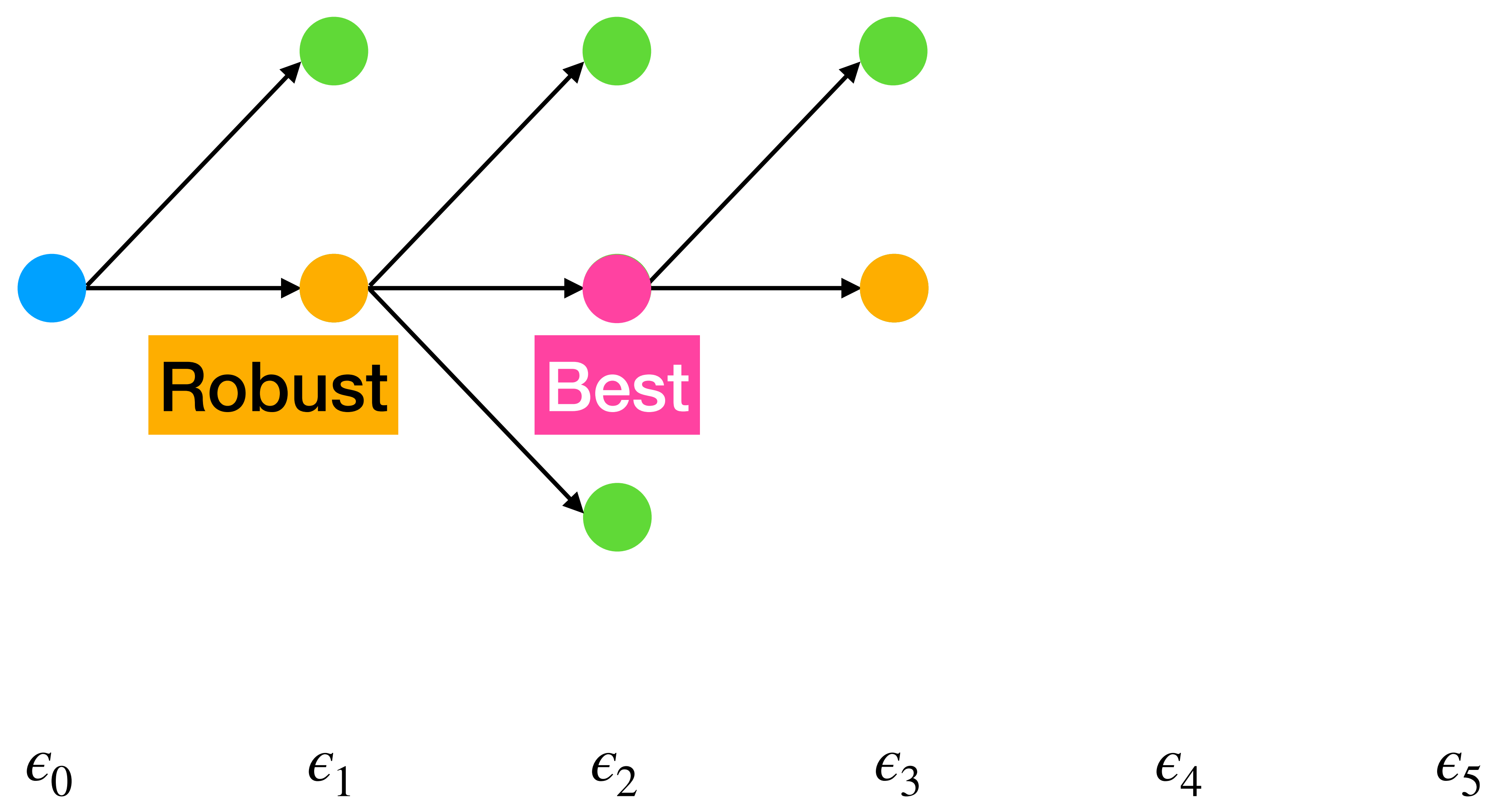
- Perform K runs for each phase
- Choose the best model among K results

BCL Framework

Special Cases of BCL

- BCL-C-AT-DQN (Conservatively Bootstrapped Curriculum Learning)
- BCL-MOS-AT-DQN (Maximum Opportunistic Skipping)

We use a threshold to decide whether a model is robust against ϵ_i



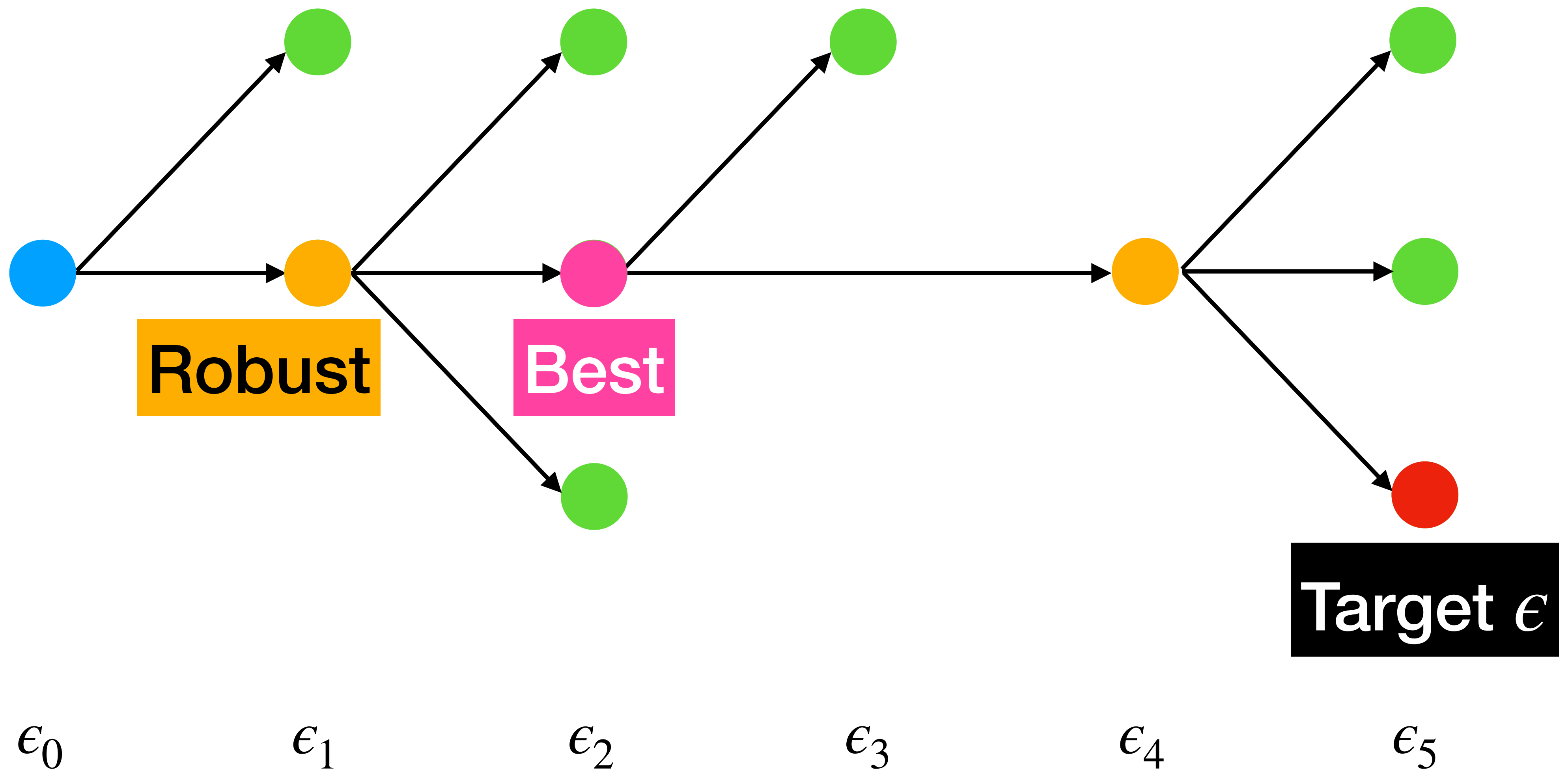
- Perform **up to** K runs for each phase
- If the model is robust against ϵ_{i+1}, skip forward the curriculum phase (train against ϵ_{i+2})

BCL Framework

Special Cases of BCL

- BCL-C-AT-DQN (Conservatively Bootstrapped Curriculum Learning)
- BCL-MOS-AT-DQN (Maximum Opportunistic Skipping)

We use a threshold to decide whether a model is robust against ϵ_i



- Perform **up to** K runs for each phase
- If the model is robust against ϵ_{i+1}, skip forward the curriculum phase (train against ϵ_{i+2})

BCL Framework

Special Cases of BCL

- BCL-C-AT-DQN (Conservatively Bootstrapped Curriculum Learning)
- BCL-MOS-AT-DQN (Maximum Opportunistic Skipping)
- BCL-RADIAL-DQN (BCL with RADIAL approach)

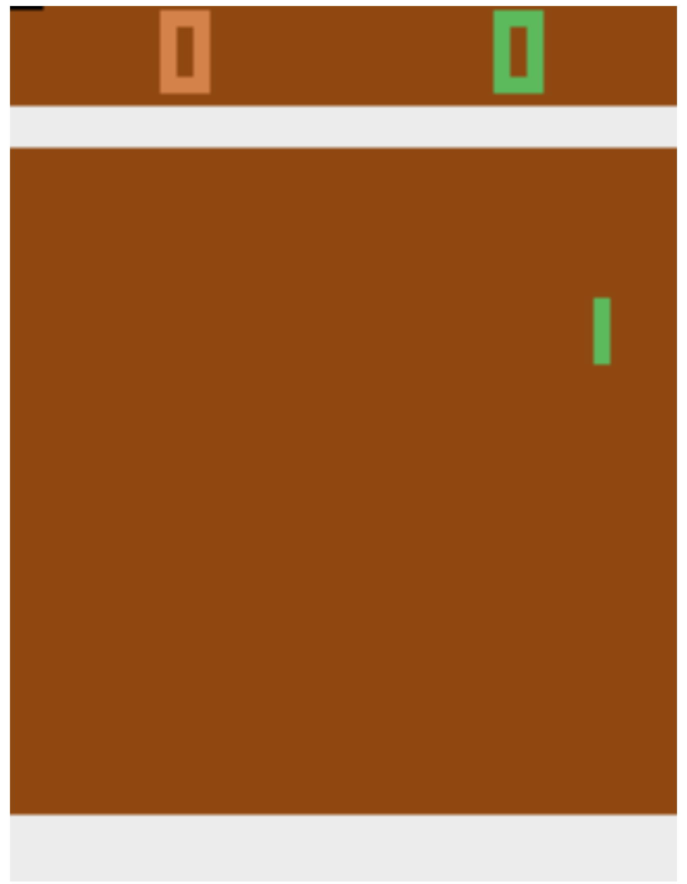
BCL Framework

Special Cases of BCL

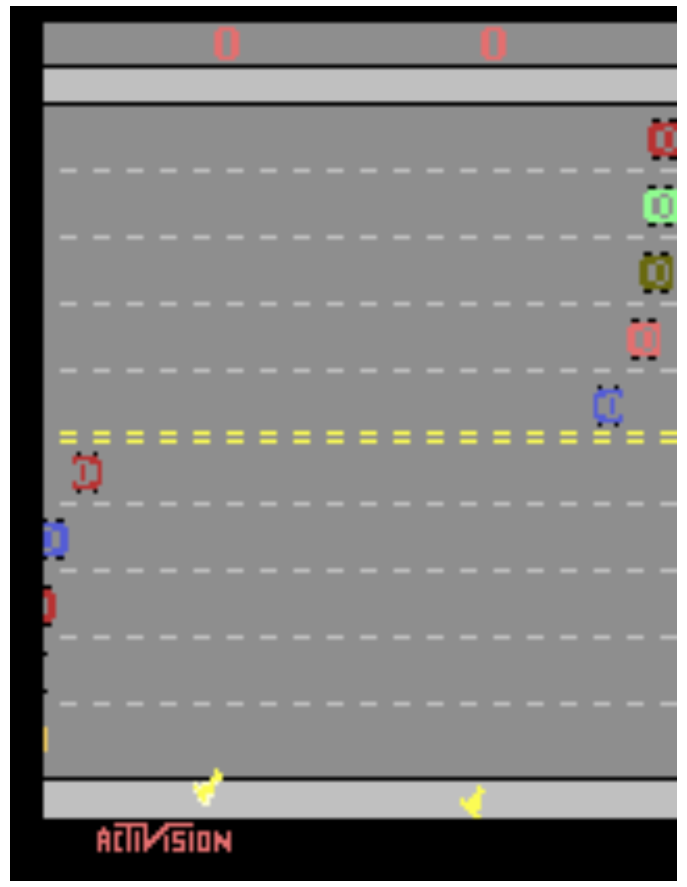
- BCL-C-AT-DQN (Conservatively Bootstrapped Curriculum Learning)
- BCL-MOS-AT-DQN (Maximum Opportunistic Skipping)
- BCL-RADIAL-DQN (BCL with RADIAL approach)
- BCL-RADIAL+AT-DQN (BCL-RADIAL-DQN + BCL-C-AT-DQN)

Experiments

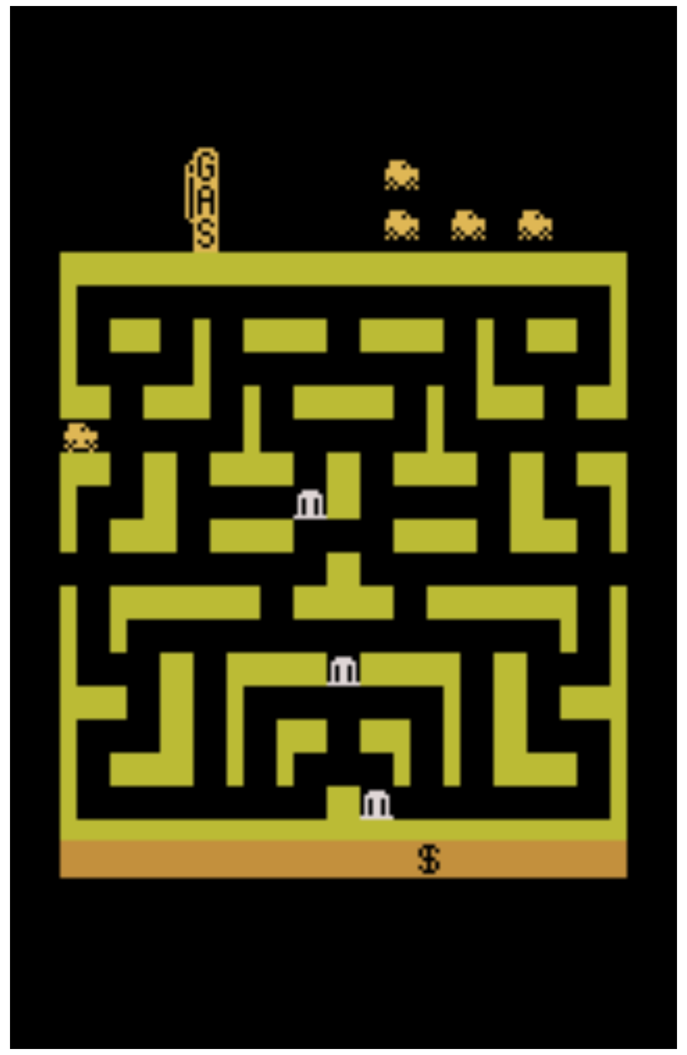
- We evaluate the proposed approach using four Atari-2600 games from the OpenAI Gym with discrete action space:



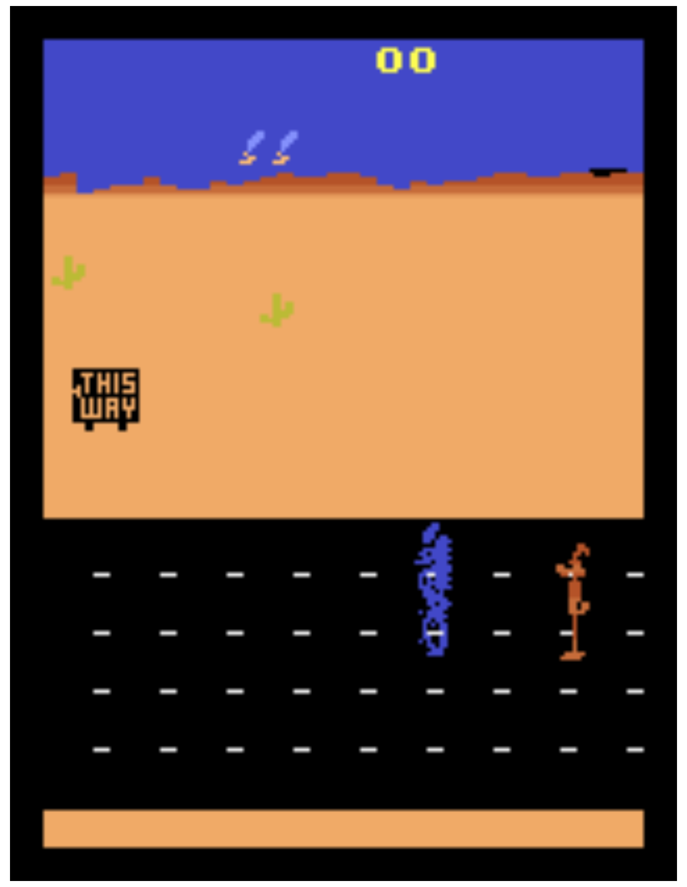
Pong



Freeway



BankHeist



RoadRunner

Experiments

Benchmark Models

- DQN (Vanilla)
- SA-DQN (Convex) [Zhang et al., 2020]
- RADIAL-DQN [Oikarinen et al., 2021]

- AT-DQN (standard adversarial training)
- NCL-AT-DQN (naive curriculum learning with adversarial examples)
- NCL-RADIAL-DQN (naive curriculum learning with RADIAL method)

Experiments

Results — Pong

- Our BCL models trained with adversarial examples (BCL-C/MOS-AT-DQN) significantly outperforms all benchmark models for higher values of ϵ.

Method/Metric	Pong			
	Nominal	30-step PGD/RI-FGSM Attack		
ϵ	0	10/255	20/255	25/255
DQN (Vanilla)	21.0	-21.0	-21.0	-21.0
SA-DQN (Convex)	21.0	-21.0	-21.0	-21.0
RADIAL-DQN	21.0	-21.0	-21.0	-21.0
AT-DQN	21.0	18.0	-0.8	-19.4
NCL-AT-DQN	21.0	20.4	-21.0	-21.0
NCL-RADIAL-DQN	21.0	-20.6	-21.0	-21.0
BCL-C-AT-DQN	21.0	21.0	21.0	21.0
BCL-MOS-AT-DQN	21.0	21.0	20.9	20.9
BCL-RADIAL-DQN	21.0	21.0	-20.9	-21.0

Experiments

Results — BankHeist

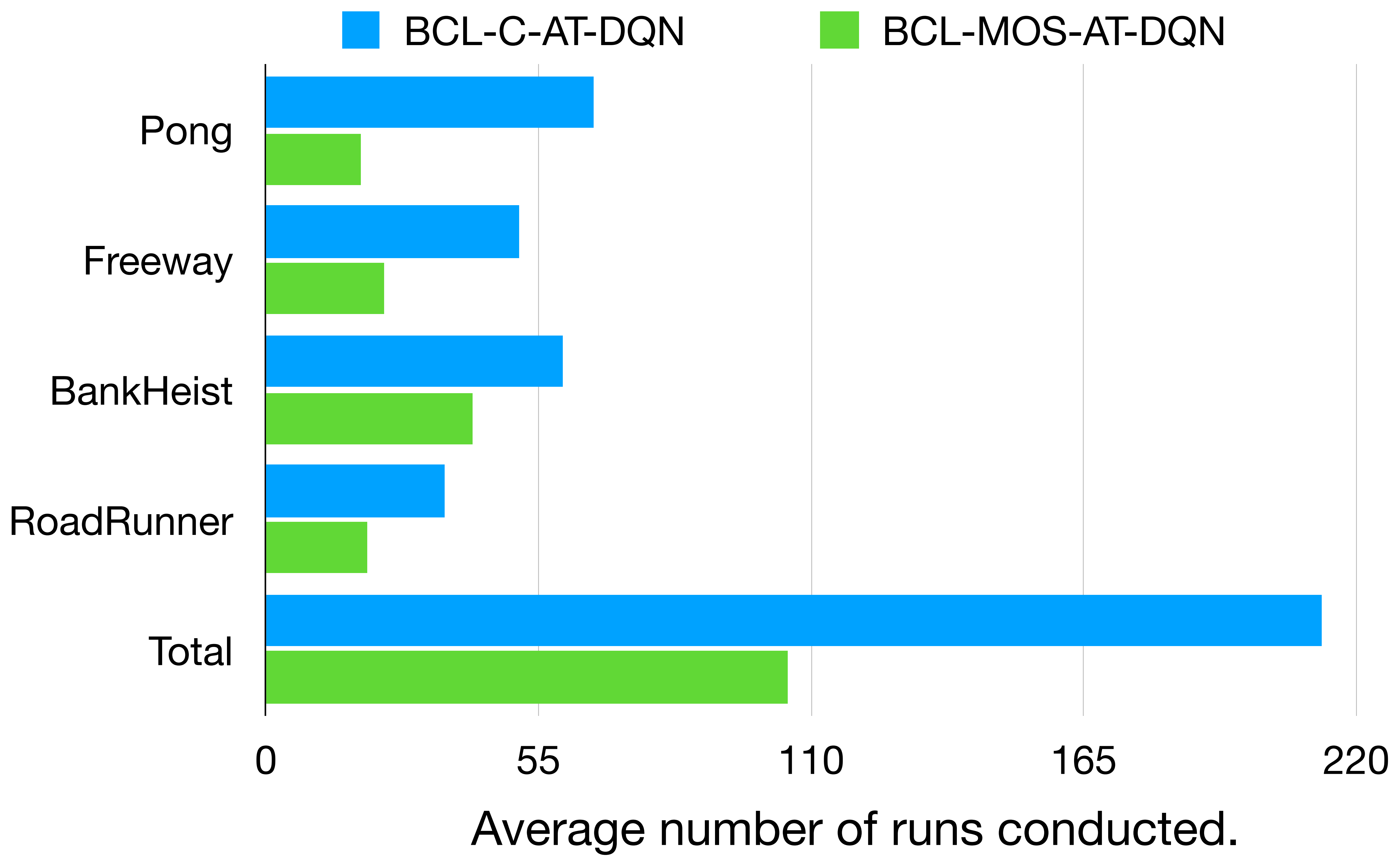
- Our BCL models outperform all benchmarks.
- BCL-RADIAL+AT-DQN models yield the most significant results.

BankHeist				
Method/Metric	Nominal	30-step PGD/RI-FGSM Attack		
ϵ	0	5/255	10/255	15/255
DQN (Vanilla)	1325.5	0.0	0.0	0.0
SA-DQN (Convex)	1237.5	1126.0	63.0	16.0
RADIAL-DQN	1349.5	581.5	0.0	0.0
AT-DQN	1271.0	129.0	5.5	0.0
NCL-AT-DQN	1311.0	245.0	1.0	0.0
NCL-RADIAL-DQN	1272.0	1168.0	59.5	9.0
BCL-C-AT-DQN	1285.5	1143.5	988.5	250.5
BCL-MOS-AT-DQN	1307.5	1095.5	664.0	586.5
BCL-RADIAL-DQN	1225.5	1225.5	1223.5	228.5
BCL-RADIAL+AT-DQN	1215.0	1093.0	1010.5	961.5

Maximum Opportunistic Skipping

BCL-C-AT-DQN vs BCL-MOS-AT-DQN

- BCL-MOS-AT-DQN significantly reduces training time (in terms of the number of training phases) and the performance is as good as BCL-C-AT-DQN.



Experiments

PPO-style

- We also experiment on two Procgen environments (FruitBot and Jumper) with PPO-style curriculum learning.

MODEL	DIST.	FRUITBOT NOMINAL	30-STEP PGD ATTACK		
		$\epsilon = 0$	$\epsilon = 10/255$	$\epsilon = 15/255$	$\epsilon = 20/255$
PPO (VANILLA)	TRAIN	30.20 ± 0.23	2.40 ± 0.21	0.73 ± 0.16	-0.72 ± 0.14
	EVAL	26.09 ± 0.33	1.70 ± 0.20	0.11 ± 0.14	-0.50 ± 0.13
RADIAL-PPO	TRAIN	28.03 ± 0.24	-0.90 ± 0.13	-1.28 ± 0.10	-1.64 ± 0.10
	EVAL	26.08 ± 0.29	-1.24 ± 0.13	-1.53 ± 0.11	-1.81 ± 0.11
AT-PPO	TRAIN	31.14 ± 0.19	28.69 ± 0.29	26.35 ± 0.32	$\mathbf{24.41 \pm 0.35}$
	EVAL	$\mathbf{28.26 \pm 0.29}$	26.47 ± 0.34	24.56 ± 0.36	20.44 ± 0.40
BCL-MOS(V)-AT-PPO	TRAIN	$\mathbf{32.11 \pm 0.17}$	29.98 ± 0.24	27.40 ± 0.31	$\mathbf{24.23 \pm 0.36}$
	EVAL	$\mathbf{28.81 \pm 0.28}$	$\mathbf{27.61 \pm 0.31}$	$\mathbf{25.52 \pm 0.35}$	$\mathbf{21.63 \pm 0.39}$
BCL-MOS(R)-AT-PPO	TRAIN	31.40 ± 0.20	$\mathbf{30.80 \pm 0.21}$	$\mathbf{28.22 \pm 0.30}$	20.18 ± 0.40
	EVAL	26.95 ± 0.34	26.28 ± 0.35	24.17 ± 0.37	17.87 ± 0.41

Experiments

PPO-style

- We also experiment on two Procgen environments (FruitBot and Jumper) with PPO-style curriculum learning.

Model	Dist.	Jumper Nominal		30-step PGD Attack	
		$\epsilon = 0$	$\epsilon = 10/255$	$\epsilon = 20/255$	$\epsilon = 40/255$
PPO (Vanilla)	Train	$\mathbf{8.69 \pm 0.11}$	3.42 ± 0.15	3.61 ± 0.15	2.94 ± 0.14
	Eval	4.22 ± 0.16	2.81 ± 0.14	2.62 ± 0.14	2.50 ± 0.14
RADIAL-PPO	Train	6.59 ± 0.15	5.43 ± 0.16	2.45 ± 0.14	1.44 ± 0.11
	Eval	3.85 ± 0.15	3.03 ± 0.14	2.04 ± 0.13	1.44 ± 0.11
AT-PPO	Train	7.57 ± 0.14	4.98 ± 0.16	4.35 ± 0.16	3.52 ± 0.15
	Eval	$\mathbf{4.55 \pm 0.16}$	3.81 ± 0.15	3.35 ± 0.15	2.51 ± 0.14
BCL-MOS(V)-AT-PPO	Train	$\mathbf{8.67 \pm 0.11}$	$\mathbf{8.15 \pm 0.12}$	$\mathbf{8.40 \pm 0.12}$	$\mathbf{7.84 \pm 0.13}$
	Eval	$\mathbf{4.57 \pm 0.16}$	$\mathbf{4.64 \pm 0.16}$	$\mathbf{4.65 \pm 0.16}$	$\mathbf{4.41 \pm 0.16}$
BCL-MOS(R)-AT-PPO	Train	8.09 ± 0.12	$\mathbf{8.29 \pm 0.12}$	$\mathbf{8.40 \pm 0.12}$	6.93 ± 0.15
	Eval	$\mathbf{4.39 \pm 0.16}$	4.29 ± 0.16	4.09 ± 0.16	3.85 ± 0.15

Conclusion

In summary, we make the following contributions:

- A novel flexible **adversarial curriculum learning framework for reinforcement learning** (BCL), in which bootstrapping each phase from multiple executions of previous phase plays a key role.
- A novel opportunistic adaptive generation variant that **opportunistically skips forward** in the curriculum.
- An approach that composes interval bound propagation and FGSM-based adversarial input generation as a part of adaptive curriculum generation.
- An extensive experimental evaluation using OpenAI Gym **Atari games (DQN-style)** and **Procgen (PPO-style, Appendix)** that demonstrates significant improvement in robustness due to the proposed BCL framework.