Fly! Bird, Fly!

Chaehyeon Kim (김채현)[1] Taekang Hwang (황태강)[1]

[1]Department of Aerospace engineering
KAIST

[1]Department of Industrial and Systems Engineering
KAIST

MDP/PRL final project

Table of contents

Flappy Bird Environment

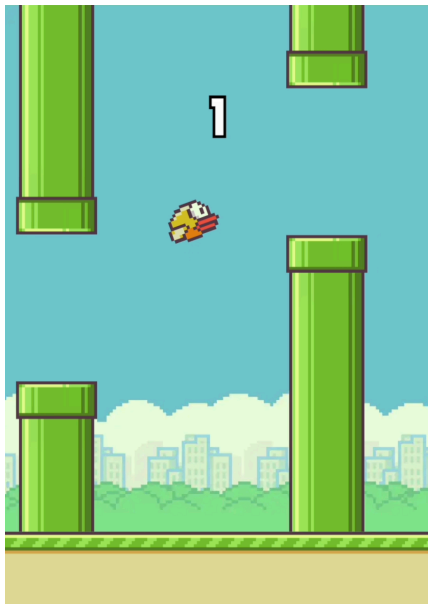


Figure: Description of the problem illustrated with the image.

- The goal of the game is to fly as long as possible avoiding crashed into randomly generated pipes.
- A continuous control problem.
- An episodic task.

$\mathcal{MDP}$ - State space($\mathcal{S}$)

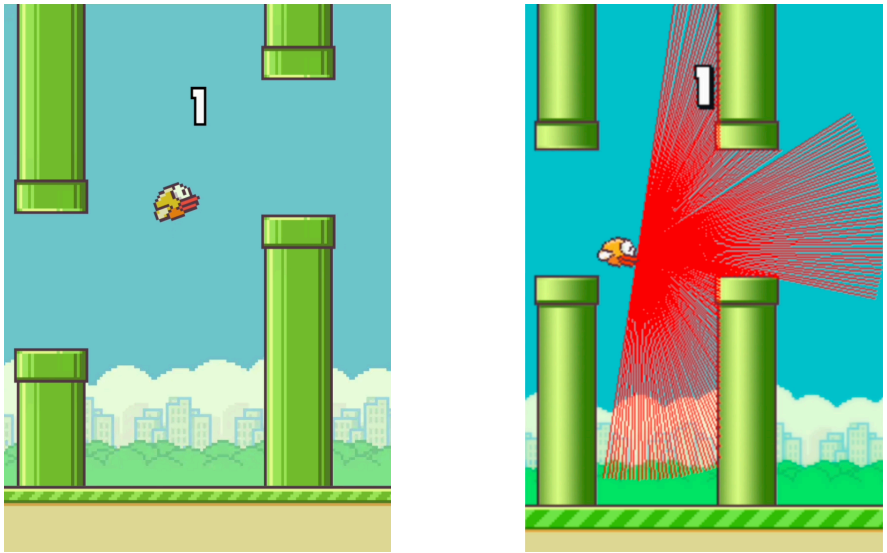


Figure: Description of the problem illustrated with the images.

- State representation using [1] *"LIDAR"* method .
- Lidar method is to cast 180 rays to calculate the distance between agent and the closest obstacle along the direction.
- It is efficient way to represent only the desirable states!

[1] "Playing Flappy Bird Based on Motion Recognition Using a Transformer Model and LIDAR Sensor"

$\mathcal{MDP}$ - Action space($\mathcal{A}$), Reward($\mathcal{R}$) and Dynamic($\mathcal{P}$)

1. Action space $(\mathcal{A})$

- 0 : IDLE
- 1 : FLAP

2. Reward $(\mathcal{R})$

- +0.1(per frame) : staying alive
- +1.0(per pipe) : passing a pipe
- -0.5 : touching the top of the screen
- -1.0 : crashing

3. Dynamics $(\mathcal{P})$

- Gravity : vertical deterministic dynamic
- Flap acceleration : vertical deterministic dynamic
- Pipe move : horizontal deterministic dynamic
- Pipe generation : random dynamic

Challenges of the $\mathcal{MDP}$

- **Continuous State Space:** $\mathcal{S} = \{s : s \in [0, {}^{2}d_{\text{max}}]^{180}\}$
- **Variance in Rewards:** Passing pipes trigger large reward signals.
- **Nonlinear & Random Dynamics:** Acceleration and random pipe generation.
- **Exploration & Exploitation:** Common challenge in most MDPs.

Function approximation and techniques to handle learning instability!

[2] The maximum possible distance.

DQN

- Q-learning + Deep Neural Networks
- Advantages :
  - Many successful works.
  - Q-learning (efficient in discrete action space) .
  - DNN (able to represent complex state spaces) .
- Challenges :
  - Overestimation bias, instability and poor exploration .

$\Rightarrow$ It seems to go well with out $\mathcal{MDP}$ setting !

Policy gradient method

- Direct optimization of the policy + Probabilistic policy learning
- Advantages :
  - Handles stochastic policies and continuous action spaces.
  - Balances exploration and exploitation effectively.
- Challenges :
  - High variance, unstable learning.

$\Rightarrow$ For our $\mathcal{MDP}$ setting, stochastic policies are inevitable !

$\Rightarrow$ This is compatible with DNN !

Improving and Implementing DQN - Dueling method

- Dueling method separates $Q(s, a)$ into two parts:

$$Q(s, a) \longleftarrow V(s) + \left(A(s, a) - \frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} A(s, a') \right)$$

- Dueling DQN architecture.

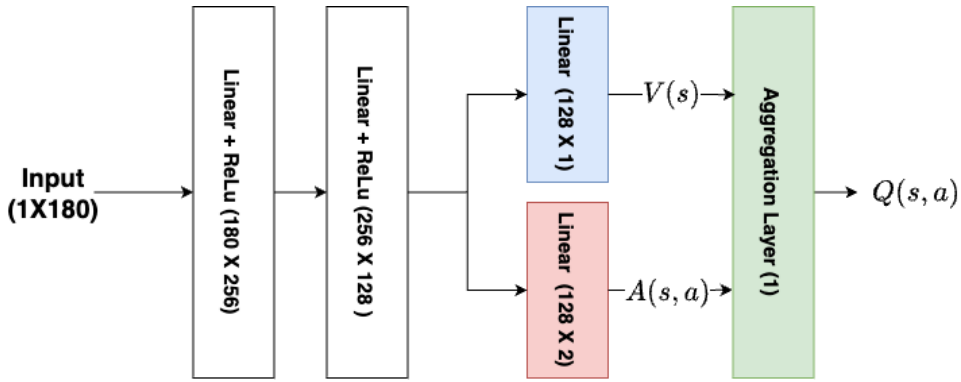


Figure: Dueling DQN architecture.

- **Benefits** : [3] Learning stability and Identifying valuable states.
- **Drawbacks** : Summation can be costly > but $|\mathcal{A}| = 2$!

[3] "Dueling Network Architectures for Deep Reinforcement Learning - Ziyu Wang et al."

Improving and Implementing PG - AC & GAE

- AC(actor-critic) net architecture.

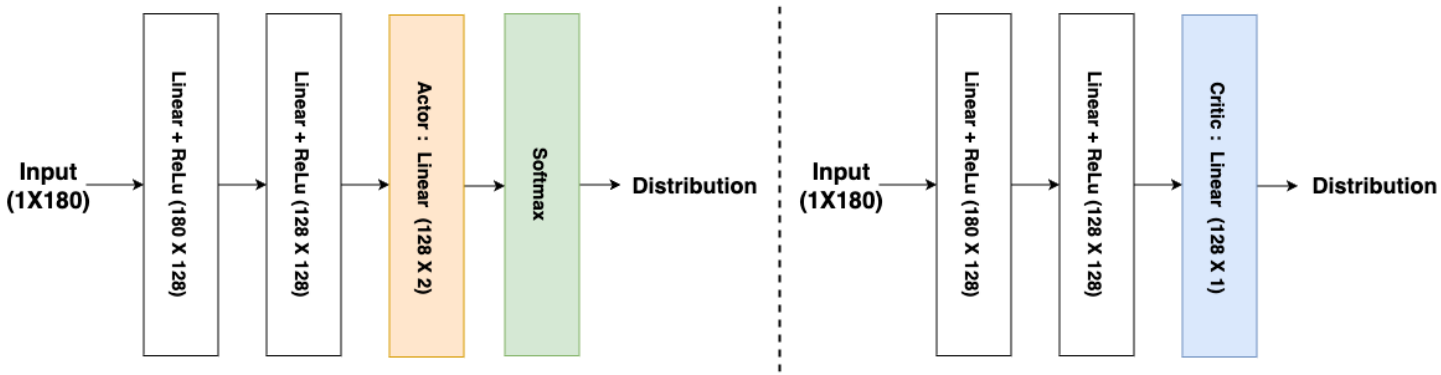


Figure: Actor critic architecture.

- **GAE** : Generalized Advantage Estimate
  - Definition : $\hat{A}_t^{\text{GAE}(\gamma,\lambda)} = \sum_{k=0}^{\infty} (\gamma\lambda)^k \delta_{t+k}^V$
  - Approximating advantage($\hat{A}$) with n-step TD-error.
  - **Benefits** : [4] Variance reduction and Improved efficiency.

[4] "HIGH-DIMENSIONAL CONTINUOUS CONTROL USING GENERALIZED ADVANTAGE ESTIMATION - John Schulman et al."

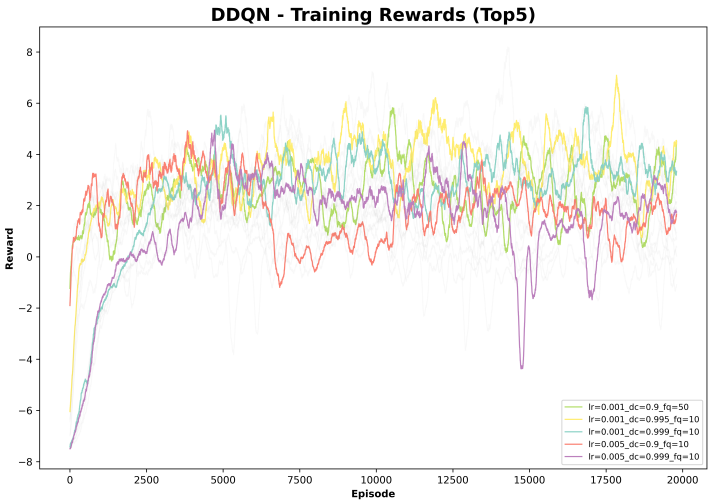
Improving and Implementing PG - PPO_{clip} & Others

- **PPO_{clip}** : Proximal Policy Optimization with Gradient Clipping
  - Definition :

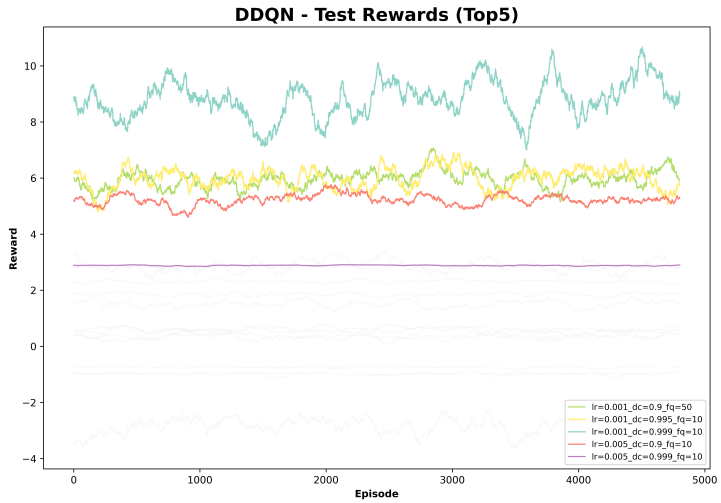
$$\mathcal{L}(s, a, \theta_k, \theta) = \min \left\{ \frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}(s,a)},\ \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1-\epsilon, 1+\epsilon \right) A^{\pi_{\theta_k}(s,a)} \right\}$$

  - Approximating advantage($\hat{A}$) with n-step TD-error.
  - Handles the high variance Reward of *Flappy Bird*.
- **Entropy Regularization** : Enhancing exploration.
- **Advantage Normalization** : Stability and Scalability.
- **Weighted Critic Loss** : $\mathcal{L}_{\text{total}} = \rho \cdot \mathcal{L}_{\text{critic}} + \mathcal{L}_{\text{actor}}$ > Stability.

Baseline selection - Dueling DQN



(a) DDQN - Training rewards

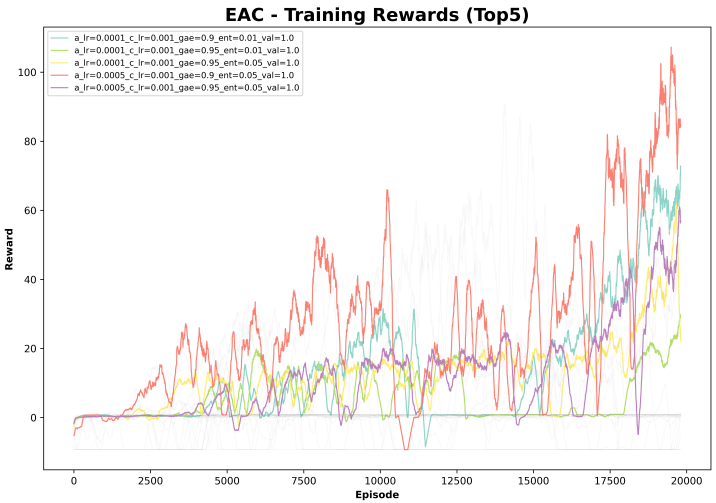


(b) DDQN - Test rewards

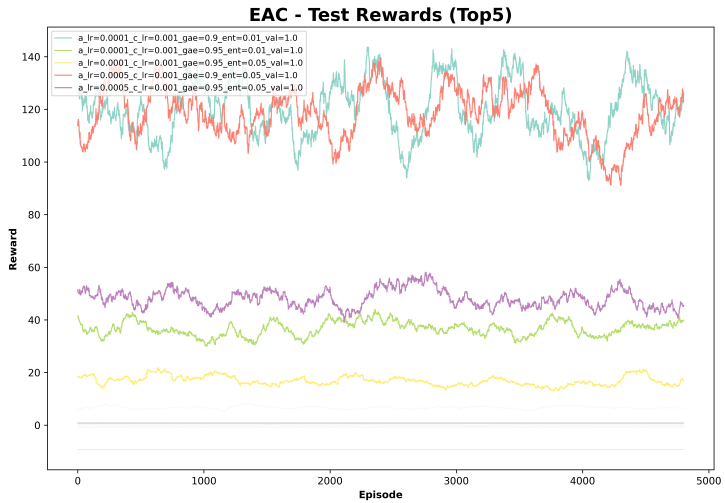
Figure: Dueling DQN experiment results

- Experiments on 18 combinations of hyper parameters.
- Top 5 combination is show - standard : $\text{mean}_{\text{test}} - 0.5 \cdot \text{std}_{\text{test}}$
- (Lr : 0.001 / ϵ-decay : 0.999 / Target update frequency : 10) is selected.

Baseline selection - EAC(Enhanced Actor Critic)



(a) EAC - Training rewards



(b) EAC - Test rewards

Figure: Enhanced Actor Critic experiment results

- Experiments on 32 combinations of hyper parameters.
- Top 5 combination is show - standard : $\text{mean}_{\text{test}} - 0.5 \cdot \text{std}_{\text{test}}$
- (Actor Lr : 0.005 / Critic Lr : 0.001 / λ : 0.95 / Entropy : 0.05 / Val coeff : 1.0) is selected.

Contributions - Further Improvements

We propose two novel and simple algorithm strategies.

1. **Oracle**
   - **Definition** : [5] $\mathcal{O}(s, a, \mathcal{E}) \equiv 1 + \mathbb{1}\{E(s, a) = T\} \pmod 2$
   - The action a is chosen at the state s but the next state is terminal state(T).
   - Then, the agent will be forced to choose the other action.
   - At the training process, [6] we can force agent to select the better action.

2. **Learning traces**
   - If ϵ meets the minimum ϵ, then algorithm start to trace if learning rewards keep increasing.
   - It it doesn't, then ϵ will be increased to ρ. > forcing agent to explore!

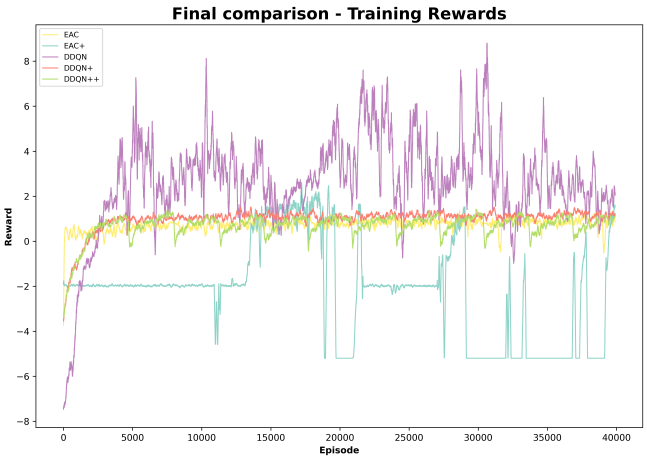
[5] $\mathcal{E}$ is an environment function, such that $\mathcal{E} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$.

[6] Since, with $\mathcal{E}$, we fully know the short-term future states of *Flappy Bird*.

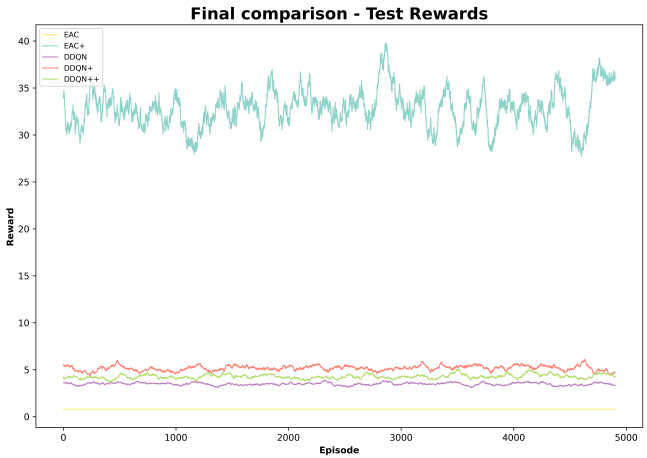
Contributions - Further Improvements

We've made **a little progress** with our own strategies :o

- EAC+ = EAC + Oracle
- DDQN+ = DDQN + Oracle
- DDQN++ = DDQN + Oracle + Learning traces



(a) Final training rewards



(b) Final test rewards

Figure: Final comparison results

Conclusion

- We've implemented and tuned two baseline RL algorithms to solve the *Flappy Bird* $\mathcal{MDP}$.
  - EAC(Enhanced Actor-Critic) : AC + GAE + PPO + Clip + Entropy reg. + etc.
  - DDQN(Dueling DQN) : DQN + Dueling-Q-Net
- We've proposed, for the $\mathcal{MDP}$, two novel and simple strategies expected to improve baseline algorithms.
  - EAC+ : EAC + Oracle
  - DDQN+ : DQN + Oracle
  - DDQN++ : DQN + Oracle + Learning traces
- The strategies we've proposed dominates baseline algorithms.

	Baseline		Improved		
	EAC	DDQN	EAC+	DDQN+	DDQN++
Learning	0.719(0.1)	**2.630(4.3)**	-2.187(2.4)	0.966(1.0)	0.677(1.0)
Test	0.799(6.0)	3.485(1.5)	**32.879(21.8)**	5.164(2.9)	4.276(2.2)

References

You can check out our code on :
https://github.com/take-001/fly_bird_fly/tree/main

- Playing Flappy Bird Based on Motion Recognition Using a Transformer Model and LIDAR Sensor - Iveta Dirgová Luptáková, Martin Kubovčík and Jiři Pospichal (2024)
- HIGH-DIMENSIONAL CONTINUOUS CONTROL USING GENERALIZED ADVANTAGE ESTIMATION - John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan and Pieter Abbeel(2016)
- Dueling Network Architectures for Deep Reinforcement Learning - Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot and Nando de Freitas(2016)
- https://github.com/markub3327/flappy-bird-gymnasium