PROJECT REPORT — Deep Q-Learning on ALE/Pong-v5

Harman S Kandola
CS 166 – Reinforcement Learning
Fall 2025

1. Introduction

This project explores Deep Q-Learning (DQN) applied to the Atari environment ALE/Pong-v5. Pong is a classic control problem with sparse, delayed rewards and long episodes, making it a solid benchmark for evaluating DQN stability and improvements.
The goal was to:

Implement a baseline DQN with experience replay and a target network.

Train a stronger variant using Double DQN + Dueling Networks.

Compare the learning behavior between the agents.

Produce training curves, videos, and documented tuning steps.

The focus is simple: understand why baseline DQN struggles early, what Double/Dueling fixes, and how each modification affects learning on a sparse-reward game.

2. Environment Description

Environment: ALE/Pong-v5
Reward structure:

+1 when the agent scores

–1 when the agent loses

Rewards are extremely sparse (long zero-reward stretches)

Observation Space:

After preprocessing: 4-frame stack, grayscale, resized 84×84

Final shape: (4, 84, 84), dtype uint8

Stacking captures ball motion and paddle velocity

Action Space:

Discrete (6)

Even though Pong really needs only UP/DOWN/NOOP, Atari enforces 6 actions

The agent must learn which ones actually matter

Episode Dynamics:

A game usually ends at -21 to +21

A random or untrained policy normally gets –21 every time

Episodes are long, which slows exploration and credit assignment

3. Methods

3.1 Baseline DQN Architecture

Standard convolutional head based on the original 2015 Atari DQN:

Conv layers:

32 filters (8×8, stride 4)

64 filters (4×4, stride 2)

64 filters (3×3, stride 1)

Flatten

Fully connected: 512 → |A|

Output: Q-values for all 6 actions

Optimizer: Adam

Loss: MSE

Discount $\gamma = 0.99$

3.2 Dueling Network Variant

Same conv encoder, but the head splits into:

Value stream: V(s)

Advantage stream: A(s,a)

Combined as:
$Q(s,a) = V(s) + (A(s,a) - \text{mean}(A))$

Purpose: separate state value from action-specific effects.

3.3 Double DQN Update

Instead of using the target network to both select and evaluate next states:

Online net selects action:
$a^* = \text{argmax}_a Q_{\text{online}}(s', a)$

Target net evaluates:
$Q_{\text{target}}(s', a^*)$

This reduces Q-value overestimation.

3.4 Replay Buffer + Epsilon-Greedy

Replay capacity: 50,000

Warm-up: 10,000 steps

Batch size: 32

Epsilon schedule:

start 1.0 → final 0.02

decay over 200,000 frames

3.5 Target Network

Synced every 1,000 frames (faster than typical DQN—Pong benefits from fresher targets).

4. Experimentation & Tuning

A total of 5 targeted changes were made during experimentation:

1. Lowered learning rate (2.5e-4 → 1e-4)

High LR produced unstable TD targets

Lower LR smoothed the early curve

2. Increased target sync rate (5000 → 1000 steps)

Pong was sensitive to stale targets

More frequent syncs improved stability

3. Slowed epsilon decay (100k → 200k frames)

Pong requires sustained exploration

More consistent behavior in the first 10–20 episodes

4. Increased replay warm-up (1,000 → 10,000 transitions)

Prevented early “garbage gradient” updates

5. Switched from baseline DQN → Double + Dueling

Reduced overestimation

More stable Q-values

Slightly more coherent policy behavior by mid-training

5. Results

5.1 Baseline DQN Learning Curve

Episode rewards remained around –21 to –18

Very slow improvement

High variance from episode to episode

Classic struggling baseline in a sparse environment

5.2 Double + Dueling Learning Curve

More stable updates

Smoother moving average over the last 100 episodes

Reduced oscillation in rewards

Better middle-stage policies (not strong, but visibly less random)

5.3 Videos

Two 10–30 second clips were recorded for each run:

Baseline Random Agent—ball missed immediately

Baseline Late Agent—still mostly random (expected)

Dueling+Double Early Agent—slightly better paddle control

Dueling+Double Later Agent—tracks the ball more consistently

Videos are included in the GitHub repository and linked in the README.

6. Reflection

I chose Pong because it's the clearest example of how sparse rewards can break a naïve DQN. Even though the game is simple for humans, the delayed feedback forces the agent to understand long-term credit assignment. Working through the baseline helped make that issue obvious.

The key improvement came from using Double DQN. Reducing overestimation bias immediately stabilized training—targets stopped jumping, and behavior became more consistent. Dueling also helped Q-values converge faster by separating the state value from actions that barely matter (most of the time, only UP/DOWN are meaningful).

The biggest challenge was how slowly learning begins. The agent lives in a large state space with delayed rewards, so exploration and replay warm-up mattered more than expected. Longer epsilon decay and a larger warm-up made the difference.

If I kept working on this, the next steps would be:

Try N-step returns to propagate rewards faster

Switch to Prioritized Replay

Add sticky actions (ALE default)

Increase total training frames to 2–5 million (full Atari scale)

Try reward shaping to accelerate early learning

Overall, this project showed exactly how sensitive DQN is on sparse Atari tasks and why modern variants fix core issues in the original algorithm.

7. References

Mnih et al. “Human-Level Control Through Deep Reinforcement Learning.” Nature, 2015.

Hasselt et al. “Deep Reinforcement Learning with Double Q-learning.” 2016.

Wang et al. “Dueling Network Architectures for Deep Reinforcement Learning.” 2016.

Gymnasium & ALE documentation