

# A Comparative Study of Goal-Conditioned Reinforcement Learning

From Value-Based to Actor-Critic Methods with Hindsight Experience Replay

Melikşah Beşir & İlteber Konuralp

Middle East Technical University  
CENG 7822 - Reinforcement Learning

January 2026

# Outline

- 1 Introduction
- 2 Background
- 3 Method
- 4 Results
- 5 Discussion
- 6 Conclusion

# Goal-Conditioned Reinforcement Learning

**Traditional RL:** Learn to maximize a single fixed reward

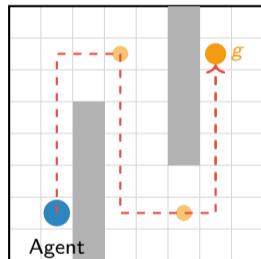
$$\pi^*(s) = \arg \max_{\pi} \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right]$$

**Goal-Conditioned RL:** Learn to achieve *any* goal  $g$

$$\pi^*(s, g) = \arg \max_{\pi} \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, g) \right]$$

## Why Goal-Conditioning?

- Warehouse robots  $\rightarrow$  different locations
- Manipulator arms  $\rightarrow$  varying positions
- Autonomous vehicles  $\rightarrow$  any destination



Same policy, different goals

# The Sparse Reward Challenge

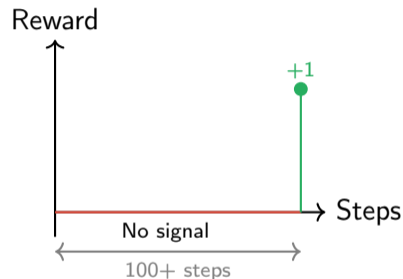
## Sparse Reward Structure:

$$r(s, g) = \begin{cases} 0 & \text{if } \|s - g\| < \epsilon \\ -1 & \text{otherwise} \end{cases}$$

## The Problem:

- No learning signal until goal is reached
- Random exploration rarely finds distant goals
- 100-step maze with 4 actions:  $4^{100}$  attempts needed!

**Credit Assignment:** Which of the 100 actions contributed to success?



Sparse rewards provide no gradient for learning until goal is reached

## This Study Addresses Three Fundamental Questions

- 1 **Exploration Mechanisms:** How do different exploration strategies interact with Hindsight Experience Replay (HER)?
- 2 **Sample Efficiency:** Which algorithms learn effectively within limited training budgets?
- 3 **Algorithmic Advances:** Do distributional critics (TQC) and hierarchical decomposition (HAC) provide benefits over simpler architectures?

**Experimental Design:** 9 algorithm configurations  $\times$  2 maze sizes

**Training Budget:** 50K steps (original 250K experiment crashed because of server fail from Google Colab after 24h 32min)

DQN (Dense), DQN $\pm$ HER, SAC $\pm$ HER, TQC $\pm$ HER, HAC $\pm$ HER

# Deep Q-Networks (DQN)

## Q-Learning with Neural Networks:

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

### Key Innovations:

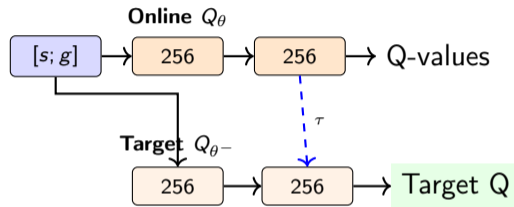
- **Experience Replay:** Break temporal correlations
- **Target Network:** Stabilize bootstrap targets

### Loss Function:

$$\mathcal{L}(\theta) = \mathbb{E} \left[ \left( r + \gamma \max_{a'} Q_{\theta-}(s', a') - Q_{\theta}(s, a) \right)^2 \right]$$

**Exploration:**  $\epsilon$ -greedy

$$a = \begin{cases} \text{random} & p = \epsilon \\ \arg \max_a Q(s, a) & p = 1 - \epsilon \end{cases}$$



**Limitation:**  $\epsilon$ -greedy is undirected—random actions rarely find distant goals

# Soft Actor-Critic (SAC)

## Maximum Entropy Objective:

$$J(\pi) = \sum_t \mathbb{E} [r_t + \alpha \mathcal{H}(\pi(\cdot|s_t))]$$

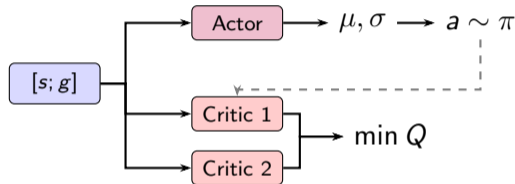
**Key Innovation:** Entropy bonus provides *intrinsic* exploration motivation

## Soft Q-Target:

$$y = r + \gamma \left( \min_{i=1,2} Q_{\theta_i}(s', a') - \alpha \log \pi(a'|s') \right)$$

## Advantages:

- Diverse trajectories even without reward
- Automatic temperature tuning
- Off-policy learning efficiency



**Key Insight:** Entropy maximization generates diverse trajectories that explore the state space

# Truncated Quantile Critics (TQC)

**Distributional RL:** Learn return *distribution*, not just expectation

$$Z(s, a) \sim \text{Distribution of } G_t$$

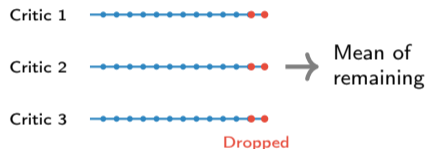
## Quantile Regression:

- 3 critics  $\times$  25 quantiles = 75 values
- Sort all quantiles
- **Truncate:** Drop top 2 per critic (6 total)
- Average remaining 69 quantiles

## Truncated Mean:

$$\bar{Q} = \frac{1}{69} \sum_{i=1}^{69} z_{(i)}$$

**Benefit:** Removes overestimated upper tail



**Advantage:** Richer gradient signal, faster convergence than SAC

# Hierarchical Actor-Critic (HAC)

## Two-Level Hierarchy:

### High-Level Policy (every $K$ steps):

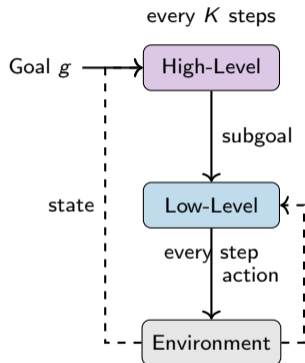
- Input: state  $s$ , final goal  $g$
- Output: subgoal  $g_{sub} \in \mathbb{R}^2$

### Low-Level Policy (every step):

- Input: state  $s$ , subgoal  $g_{sub}$
- Output: primitive action  $a \in \mathbb{R}^2$

## Three Key Mechanisms:

- 1 **HAT**: Relabel subgoal with achieved state
- 2 **Subgoal Testing**: Penalize unreachable subgoals
- 3 **HGT**: Relabel final goal (with HER)



**Benefit:** Reduces 100-step horizon to  $\sim 10$  high-level decisions

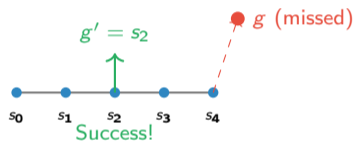
# Hindsight Experience Replay (HER)

**Key Insight:** Failed trajectories contain valuable information when reinterpreted with different goals

**Future Strategy ( $k = 4$ ):**

- 1 Store original:  $(s, a, r, s', g)$
- 2 Sample  $k$  future states:  $\{s_j\}$
- 3 For each  $s_j$ :
  - Relabel goal:  $g' = s_j$
  - Recompute:  $r' = 1[s' \approx g']$
  - Store:  $(s, a, r', s', g')$

**Effect:** Transforms every episode into useful training data



**Requirement:** Diverse trajectories that visit different states

## Discrete Grid Maze

Property	Small	Large
Grid Size	$10 \times 10$	$30 \times 30$
State Space	500	4500
Actions	4 (Cardinal)	4 (Cardinal)
Obstacles	30%	30%
Episode Limit	200	1800

### Features:

- Guaranteed solvability (BFS validation)
- Wall collision penalty:  $-0.5$
- Step cost:  $-0.1$

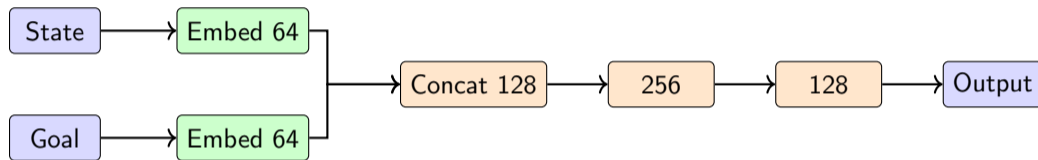
## PointMaze (Continuous)

Property	Small	Large
State Space	$\mathbb{R}^4$	$\mathbb{R}^4$
Action Space	$\mathbb{R}^2$	$\mathbb{R}^2$
Maze Type	U-shaped	Multi-corridor
Success Threshold	0.45	0.45
Episode Limit	300	700

### Features:

- Physics simulation (MuJoCo)
- Position + velocity state
- Continuous force control

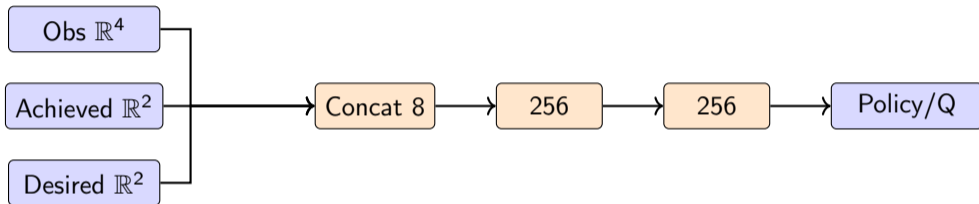
## Learned Embeddings Approach



### Why 64-dim embeddings?

- Smaller than state space (100)
- Encourages learning spatial structure
- Adjacent cells  $\rightarrow$  similar embeddings

## Direct Concatenation Approach



### MultInputPolicy (SB3):

- Direct concatenation of vectors
- Two hidden layers (256 units)
- ReLU activations

# Experimental Design: Continuous Environment

## Three-Tier Comparison

**Tier 1:** DQN with dense rewards (baseline—is environment learnable?)

**Tier 2:** Sparse rewards  $\pm$  HER (DQN, SAC, TQC ablations)

**Tier 3:** Hierarchical decomposition (HAC  $\pm$  HER)

## Configurations Tested:

- **DQN:** Dense, No HER, with HER
- **SAC:** No HER, with HER
- **TQC:** No HER, with HER
- **HAC:** No HER, with HER

## Evaluation Protocol:

- Evaluate every 5,000 steps
- 20 deterministic rollouts
- 3 random seeds

## Metrics:

- Success Rate
- Mean Steps to Goal
- Path Efficiency

## Three-Tier Comparison

**Tier 1:** Sparse rewards  $\pm$  HER (SAC, TQC ablations)

**Tier 2:** Hierarchical decomposition (HAC  $\pm$  HER)

### Configurations Tested:

- **SAC:** with HER
- **TQC:** with HER
- **HAC:** with HER

### Evaluation Protocol:

- Evaluate every 5,000 steps
- 20 deterministic rollouts
- 3 random seeds

### Metrics:

- Success Rate
- Subgoal Success Rate
- Mean Steps to Goal
- Path Efficiency

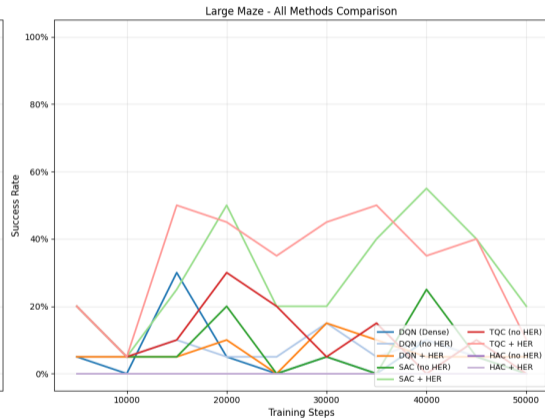
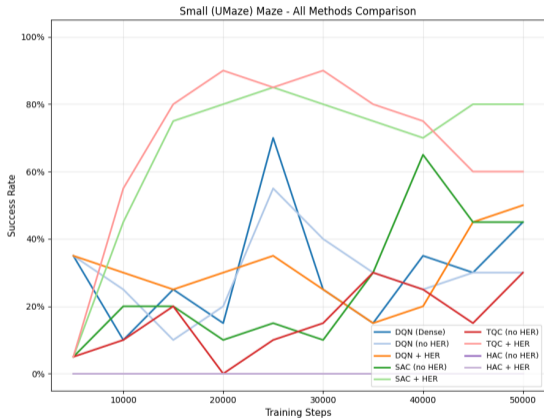
# Hyperparameters: Continuous Environment

Parameter	DQN	SAC/TQC	HAC
Learning Rate	$1 \times 10^{-3}$	$3 \times 10^{-4}$	$3 \times 10^{-4}$
Discount $\gamma$	0.99	0.99	0.99
Soft Update $\tau$	0.005	0.005	0.005
Batch Size	256	256	256
Buffer Size	100K	100K	100K
$\epsilon$ Start/End	1.0 / 0.05	—	—
Entropy Coefficient	—	auto	auto
Quantiles (TQC)	—	25	—
Critics (TQC)	—	3	—
Subgoal Horizon	—	—	10
HER Strategy	future	future	future
HER $k$	4	4	4

# Hyperparameters: Discrete Environment

Parameter	SAC/TQC	HAC
Learning Rate	$10^{-3}$	$10^{-3}$
Discount $\gamma$	0.99	0.99
Soft Update $\tau$	0.005	0.005
Batch Size	64	64
Buffer Size	$N \times N \times 5$	$N \times N \times 5$
Entropy Coefficient	auto	auto
Quantiles (TQC)	25	—
Critics (TQC)	5	—
Drop Top Quantiles	2	—
Subgoal Horizon	—	10
Subgoal Test Rate	—	0.3
HER Strategy	future	future
HER $k$	4	4

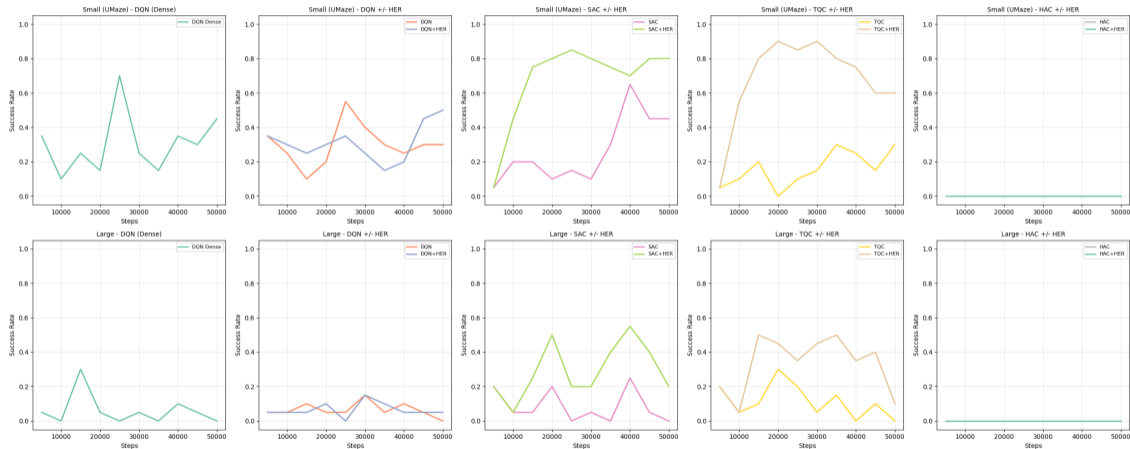
# Training Curves: All Methods Comparison in Continuous Environment



**Left:** Small (UMaze) - SAC+HER reaches 80%, TQC+HER peaks at 90% but drops

**Right:** Large Maze - Only HER methods achieve any success (SAC+HER: 20%)

# Individual Method Analysis in Continuous Environment



Each column: algorithm family comparing with/without HER. Note HAC's complete failure (rightmost).

# DQN Results: Discrete Actions on Continuous Environment

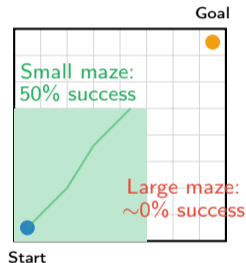
Table: \*

## DQN on PointMaze (50K steps)

Method	Small	Large
DQN (Dense)	45.0%	0.0%
DQN no HER	30.0%	0.0%
DQN + HER	50.0%	5.0%

### Key Observations:

- DQN works on **simple** mazes (45-50%)
- HER improves DQN: 30%  $\rightarrow$  50%
- **Fails on large maze** (too complex)
- $\epsilon$ -greedy can't explore long corridors



Complexity determines DQN viability

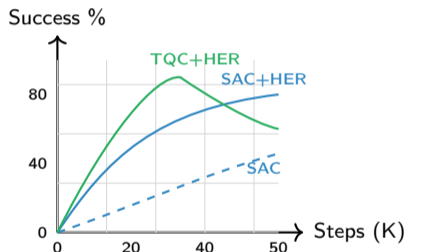
# Actor-Critic Results: SAC Dominates in Continuous Environment

**SAC+HER** achieves best performance: **80%**  
(small)

Method	Small	Large
SAC no HER	45.0%	0.0%
SAC + HER	<b>80.0%</b>	<b>20.0%</b>
TQC no HER	30.0%	0.0%
TQC + HER	60.0%	10.0%

## Surprising Finding:

- **TQC underperforms SAC** (60% vs 80%)
- Distributional critics need more samples
- 50K steps **insufficient** for TQC benefits
- TQC showed high variance (peaked at 90%, dropped)

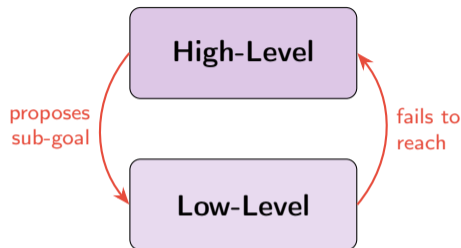


TQC peaks early but unstable

# HAC Results: Complete Failure (0% Success) in Continuous Environment

HAC achieves 0% on both mazes

Method	Small	Large
SAC + HER	80.0%	20.0%
TQC + HER	60.0%	10.0%
HAC no HER	0.0%	0.0%
HAC + HER	0.0%	0.0%



Both levels stuck at 0%

## Why HAC Failed:

- **Cold-start problem:** Neither level can learn without the other
- **Sample starvation:** High-level gets 1/10th updates
- **50K steps insufficient** for hierarchical convergence
- HER cannot rescue failing hierarchies



Chicken-and-egg problem:  
neither level bootstraps

# Complete Results Summary in Continuous Environments

Table: \*

Small (UMaze) - 50K Steps

Method	Success	Steps
DQN (Dense)	45.0%	337
DQN no HER	30.0%	366
DQN + HER	50.0%	281
SAC no HER	45.0%	369
SAC + HER	80.0%	181
TQC no HER	30.0%	419
TQC + HER	60.0%	233
HAC no HER	0.0%	477
HAC + HER	0.0%	338

Table: \*

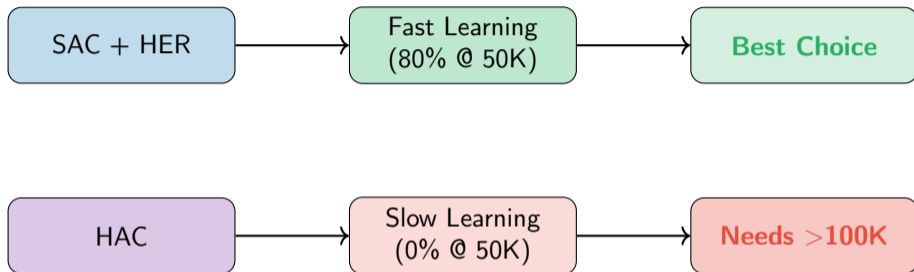
Large Maze - 50K Steps

Method	Success	Steps
DQN (Dense)	0.0%	5000
DQN no HER	0.0%	5000
DQN + HER	5.0%	4751
SAC no HER	0.0%	5000
SAC + HER	20.0%	4028
TQC no HER	0.0%	5000
TQC + HER	10.0%	4518
HAC no HER	0.0%	4518
HAC + HER	0.0%	4752

## Key Takeaway

**SAC+HER** is the clear winner: 80% (small), 20% (large). HER consistently helps all methods.

## Key Insight 1: Sample Efficiency Matters



### Conclusion

With limited training budget (50K steps), **SAC+HER** provides best results. HAC requires significantly more samples to bootstrap hierarchical learning.

## Key Insight 2: HER Consistently Improves Performance

### HER Improvement (Small Maze):

- DQN: 30%  $\rightarrow$  50% (+20%)
- SAC: 45%  $\rightarrow$  **80%** (+35%)
- TQC: 30%  $\rightarrow$  60% (+30%)
- HAC: 0%  $\rightarrow$  0% (can't help)

### HER Improvement (Large Maze):

- DQN: 0%  $\rightarrow$  5%
- SAC: 0%  $\rightarrow$  **20%**
- TQC: 0%  $\rightarrow$  10%
- HAC: 0%  $\rightarrow$  0%

### HER Cannot Help When:

- Both levels of hierarchy failing (HAC)
- Base exploration too weak for complex tasks
- Algorithm fundamentally stuck

### Best HER Combinations:

- 1 **SAC + HER** (best overall)
- 2 TQC + HER (high variance)
- 3 DQN + HER (simple tasks only)

**Key:** HER amplifies working algorithms

## Key Insight 3: TQC vs HAC - Both Disappointing

### TQC (Distributional Critics)

**Expected:** Faster convergence than SAC

**Actual Result:**

- 60% vs SAC's 80% (small maze)
- High variance, unstable training
- Peaked at 90%, then dropped
- Needs more samples for distributional estimates

**Verdict:** **Not recommended** for limited budgets

### HAC (Hierarchical)

**Expected:** Better on long horizons

**Actual Result:**

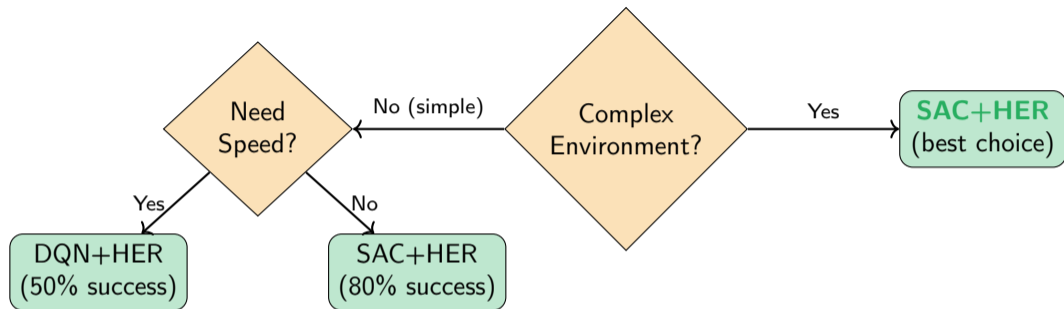
- 0% success everywhere
- Cold-start problem fatal
- HER couldn't rescue it
- Needs >>50K steps

**Verdict:** **Avoid** unless budget >100K steps

### Lesson Learned

Theoretical advantages don't always translate to practical benefits with limited training.

# Algorithm Selection Guide (Based on Our Results)



**Avoid:** HAC, TQC  
(with 50K budget)

**Simple Rule:** **SAC+HER** for everything unless environment is trivially simple

# Summary of Findings in Continuous Environment

## ① SAC+HER is the Clear Winner

- 80% success (small), 20% success (large)
- Best sample efficiency within 50K steps
- Most stable training dynamics

## ② HER Consistently Improves All Methods

- DQN: 30%  $\rightarrow$  50%, SAC: 45%  $\rightarrow$  80%
- Only exception: HAC (can't help failing hierarchy)

## ③ TQC Underperforms with Limited Training

- 60% vs SAC's 80% on small maze
- High variance, needs more samples

## ④ HAC Completely Failed (0% Success)

- Cold-start problem: neither level bootstraps
- Needs  $\gg 50K$  steps to converge
- **Not recommended** for limited budgets

# Practical Recommendations for Continuous Environment

## For Practitioners (Based on 50K Step Budget)

- 1 **Default:** Use **SAC+HER** for goal-conditioned control
- 2 **Simple environments:** DQN+HER can work (50% on small maze)
- 3 **Avoid TQC:** Unless you have  $>100K$  training budget
- 4 **Avoid HAC:** Unless you have  $>>100K$  training budget

## Key Takeaway

Sample efficiency matters more than theoretical advantages.

# Limitations and Future Work in Continuous Environment

## Limitations:

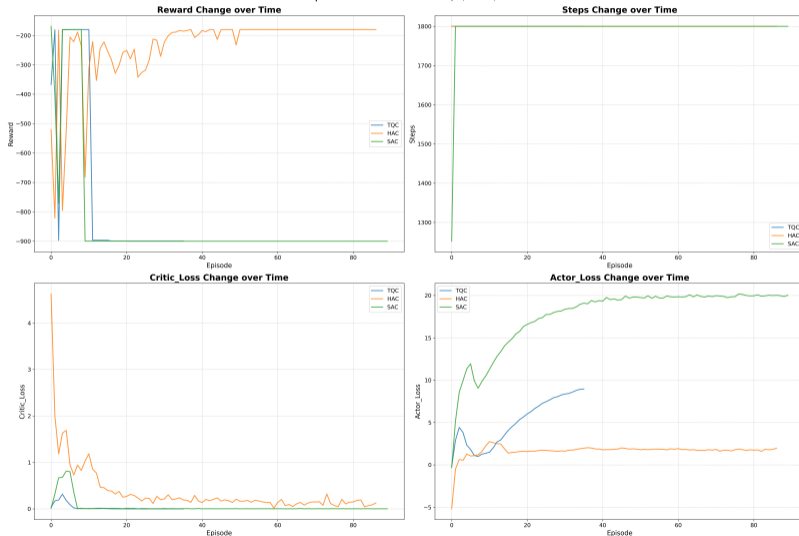
- **Training budget:** 50K steps (250K experiment crashed after 24h)
- Only PointMaze environment (2D navigation)
- Fixed hyperparameters across methods
- Results may change with longer training

## Future Directions:

- Complete 250K+ step experiments for asymptotic comparison
- Pre-train HAC low-level controller to address cold-start
- Test on manipulation domains (FetchReach, etc.)
- Investigate why TQC showed high variance

# Training Results: Three Methodologies Comparison in Discrete Environment

Comparison of RL Models: TQC, HAC, SAC



# Thank You!









Questions?

**Code:** [https://github.com/ilteberkonuralp/Term\\_Project\\_CENG7822](https://github.com/ilteberkonuralp/Term_Project_CENG7822)

**Contact:**

Melikşah Beşir & İlteber Konuralp  
Middle East Technical University

# References I

-  Andrychowicz, M., et al. (2017). Hindsight Experience Replay. *NeurIPS*, 5048–5058.
-  Haarnoja, T., et al. (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep RL. *ICML*, 1861–1870.
-  Kuznetsov, A., et al. (2020). Controlling Overestimation Bias with Truncated Quantile Critics. *ICML*, 5556–5566.
-  Levy, A., et al. (2019). Learning Multi-Level Hierarchies with Hindsight. *ICLR*.
-  Mnih, V., et al. (2015). Human-level Control through Deep Reinforcement Learning. *Nature*, 518, 529–533.
-  Schaul, T., et al. (2015). Universal Value Function Approximators. *ICML*, 1312–1320.
-  Bellemare, M.G., et al. (2017). A Distributional Perspective on Reinforcement Learning. *ICML*, 449–458.
-  Fujimoto, S., et al. (2018). Addressing Function Approximation Error in Actor-Critic Methods. *ICML*, 1587–1596.

---

## Algorithm 1 Goal-Conditioned DQN

---

```
1: Initialize  $Q_\theta$ , target  $Q_{\theta^-}$ , buffer  $\mathcal{D}$ 
2: for episode = 1, 2, ... do
3:   Sample goal  $g$ , initial state  $s_0$ 
4:   for  $t = 0, 1, \dots, T - 1$  do
5:      $a_t \leftarrow \epsilon\text{-greedy}(Q_\theta(s_t, \cdot, g))$ 
6:     Execute  $a_t$ , observe  $r_t, s_{t+1}$ 
7:     Store  $(s_t, a_t, r_t, s_{t+1}, g)$  in  $\mathcal{D}$ 
8:     Sample minibatch from  $\mathcal{D}$ 
9:      $y_i = r_i + \gamma \max_{a'} Q_{\theta^-}(s'_i, a', g_i)$ 
10:    Update  $\theta$ : minimize  $(y_i - Q_\theta(s_i, a_i, g_i))^2$ 
11:    Soft update:  $\theta^- \leftarrow \tau\theta + (1 - \tau)\theta^-$ 
12:   end for
13: end for
```

---

---

## Algorithm 2 SAC for Goal-Conditioned Tasks

---

```
1: Initialize actor  $\pi_\phi$ , critics  $Q_{\theta_1}, Q_{\theta_2}$ , targets,  $\alpha$ 
2: for each iteration do
3:   Sample  $a \sim \pi_\phi(\cdot|s, g)$ , observe  $r, s'$ 
4:   Store  $(s, a, r, s', g)$  in  $\mathcal{D}$ 
5:   for each gradient step do
6:      $a' \sim \pi_\phi(\cdot|s', g)$ 
7:      $y = r + \gamma(\min_i Q_{\bar{\theta}_i}(s', a', g) - \alpha \log \pi(a'|s', g))$ 
8:     Update critics:  $\nabla_{\theta_i} (Q_{\theta_i} - y)^2$ 
9:     Update actor:  $\nabla_\phi (\alpha \log \pi - \min_i Q_{\theta_i})$ 
10:    Update  $\alpha$ :  $\nabla_\alpha (-\alpha(\log \pi + \bar{\mathcal{H}}))$ 
11:    Soft update targets
12:  end for
13: end for
```

---

---

## Algorithm 3 HER Future Strategy

---

```
1: Input: Episode  $\{(s_t, a_t, r_t, s_{t+1})\}_{t=0}^T$ , original goal  $g$ ,  $k$  samples
2: for  $t = 0, 1, \dots, T - 1$  do
3:   Store original:  $(s_t, a_t, r_t, s_{t+1}, g)$ 
4:   Sample  $k$  indices from  $\{t + 1, \dots, T\}$ 
5:   for each sampled index  $j$  do
6:     Hindsight goal:  $g' \leftarrow s_j$  (achieved state)
7:     Recompute reward:  $r' \leftarrow 1[\|s_{t+1} - g'\| < \epsilon]$ 
8:     Store hindsight:  $(s_t, a_t, r', s_{t+1}, g')$ 
9:   end for
10: end for
```

---

**Effect:** Generates  $(k + 1) \times$  more training samples per episode

## Backup: Discrete vs Continuous Comparison

Aspect	Discrete Grid	Continuous PointMaze
State	Grid index	Position + velocity
Actions	4 cardinal	Continuous force
Dynamics	Deterministic	Physics simulation
Computation	Low	High (MuJoCo)
Episode Length	50-200	100-1000
Training Time	Minutes	Hours
Real-World Use	Limited	Robotics