# Uncertainty-Based Offline Reinforcement Learning with Diversified Q-Ensemble

Advanced Topics in Reinforcement Learning
Winter Semester 2022/23

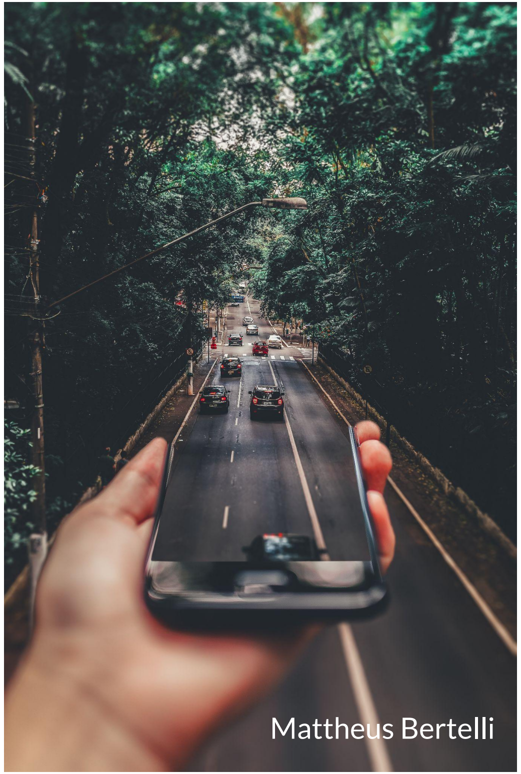
Technische Universität Berlin

Julian Dralle & Jonas Loos |

# Outline

1. Introduction - Why Offline RL?
2. Preliminaries
   a. D4RL: Benchmark for Offline RL
   b. Actor-Critic Method
   c. Conservative Q-Learning (CQL) - Baseline
3. SAC-N (Soft Actor-Critic)
4. EDAC (Ensemble-Diversifying Actor-Critic )
5. Comparison

# Why Offline RL?





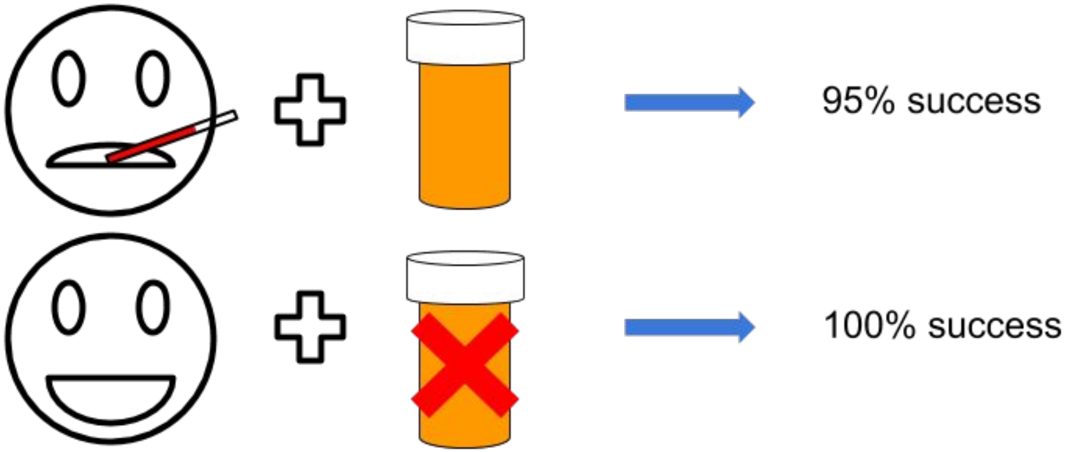
TRIAL & ERROR?

# Why Offline RL?

- **Online training** requires active **interaction** with the environment. Interaction with the real world can be **costly and dangerous**
- **Solution:** Offline / “batch” RL -
  Learn from large, previously collected datasets, **without interaction**
- **Problem:** Out-of-distribution (OOD) actions
  - erroneously high Q-values
  - no feedback from the environment

**How to manage uncertainty in offline RL?**

# Example: OOD actions




- Naive algorithm: treatment causes death?!
- We never see sick patients not treated!

# D4RL - Datasets for Deep Data-Driven RL

Datasets specifically designed for benchmarking Offline RL

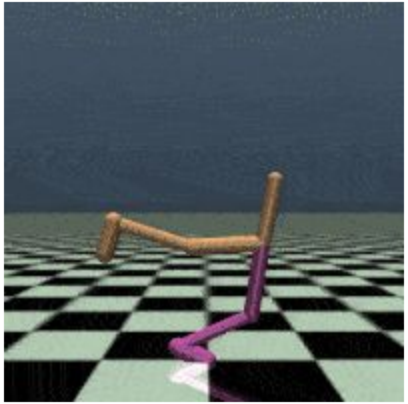
- Narrow and biased data distribution
- Undirected and multitask data
- Sparse rewards
- Suboptimal data
- Non-representable behavior policies
- Realistic domains

# D4RL - MuJoCo Gym environments

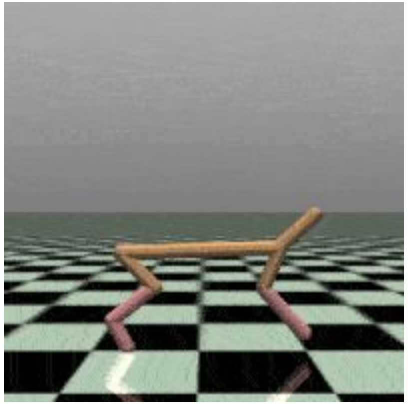
- Multi-Joint dynamics with Contact
- physics engine used in robotics, biomechanics, graphics and animation
- standardized in D4RL

Hopper

Walker2D



HalfCheetah



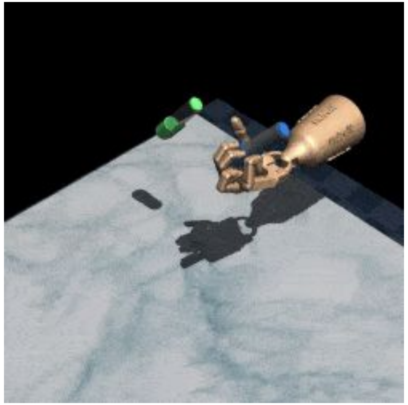
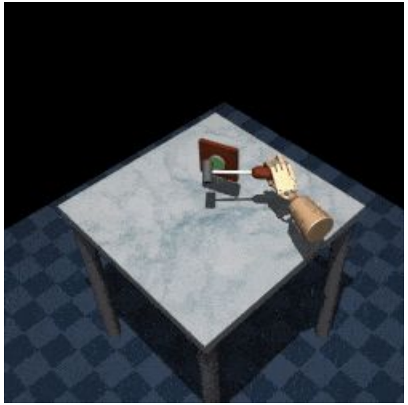
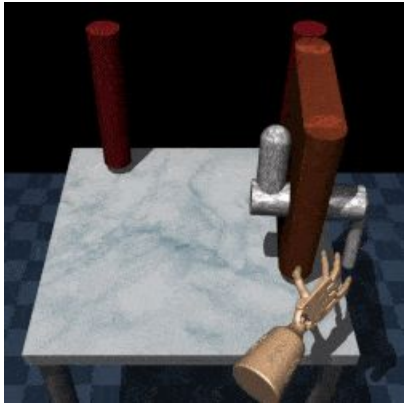
Source: [5]

# D4RL - MuJoCo Gym datasets

- expert: train a SAC algorithm online until the strategy reaches the expert performance level, using the expert strategy to collect 1 million samples of data
- medium: first train a SAC algorithm online, stop training in the middle, and then use this partially trained strategy to collect 1 million samples of data
- medium-expert: mix equal amounts of data collected by expert and medium strategies
- medium-replay：train a SAC algorithm online until the strategy reaches a moderate performance level, collecting all the samples placed in the buffer during training
- random：use a random initialization strategy to collect

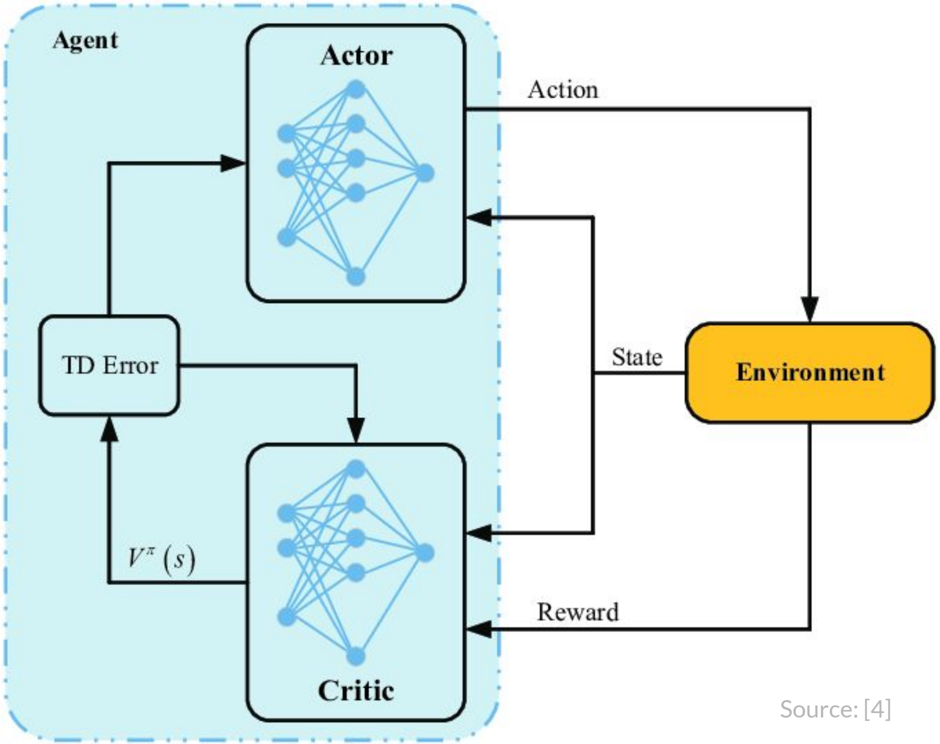
# D4RL - Adroit tasks

- Realistic robotic manipulation tasks, hand with 24-DoF
- human demonstrations recorded via motion capture
- enables studying human-generated data within a simulated robotic platform
- uses MuJoCo physics simulator

# Actor-Critic Method

policy-based actor




Source: [4]

value-based critic

# Actor-Critic in Offline RL

**Critic Network** minimizes

$$J_q(Q_\phi) := \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[\left(Q_\phi(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma\, \mathbb{E}_{\mathbf{a}'\sim\pi_\theta(\cdot|\mathbf{s}')}\left[Q_{\phi'}(\mathbf{s}',\mathbf{a}')\right]\right)\right)^2\right]$$

**Actor Network** maximizes

$$J_p(\pi_\theta) := \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi_\theta(\cdot|\mathbf{s})}\left[Q_\phi(\mathbf{s},\mathbf{a})\right]$$

updated in alternating fashion

# Conservative Q-Learning (CQL) - Baseline

As of 2021, “Conservative Q-Learning” (CQL) [2] is the state-of-the-art for offline RL. [1]

It uses a “simple Q-value regularizer” to prevent the overestimation of OOD actions.

$$\min_{\phi} \; J_q(Q_\phi) + \alpha\Big(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\cdot|\mathbf{s})}\left[Q_\phi\left(\mathbf{s},\mathbf{a}\right)\right] - \mathbb{E}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}}\left[Q_\phi\left(\mathbf{s},\mathbf{a}\right)\right]\Big)$$

We present two methods **beating the current state-of-the-art**:

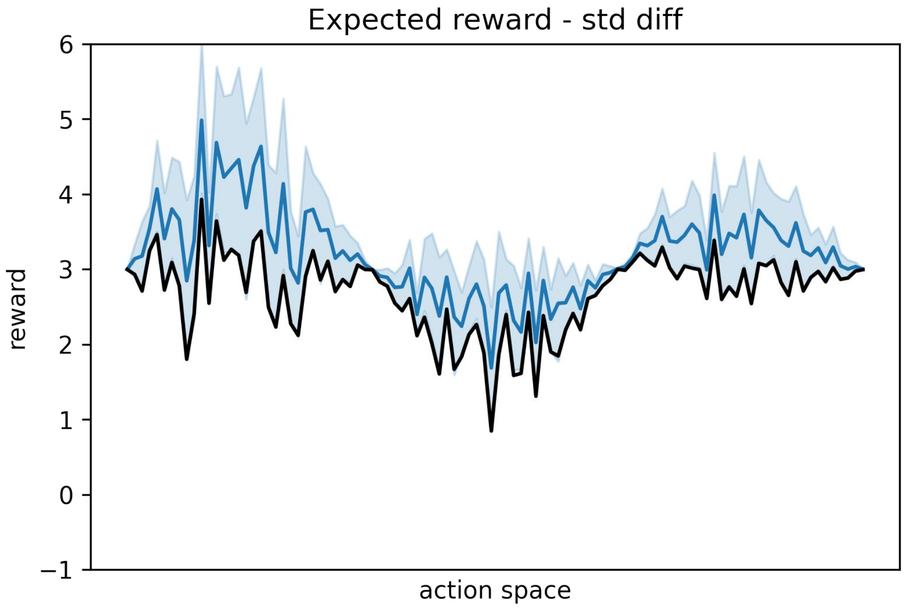
- SAC-N: beats CQL
- EDAC: improves runtime performance of SAC-N

# SAC-N (Soft Actor-Critic N)

- choose action based on expected reward

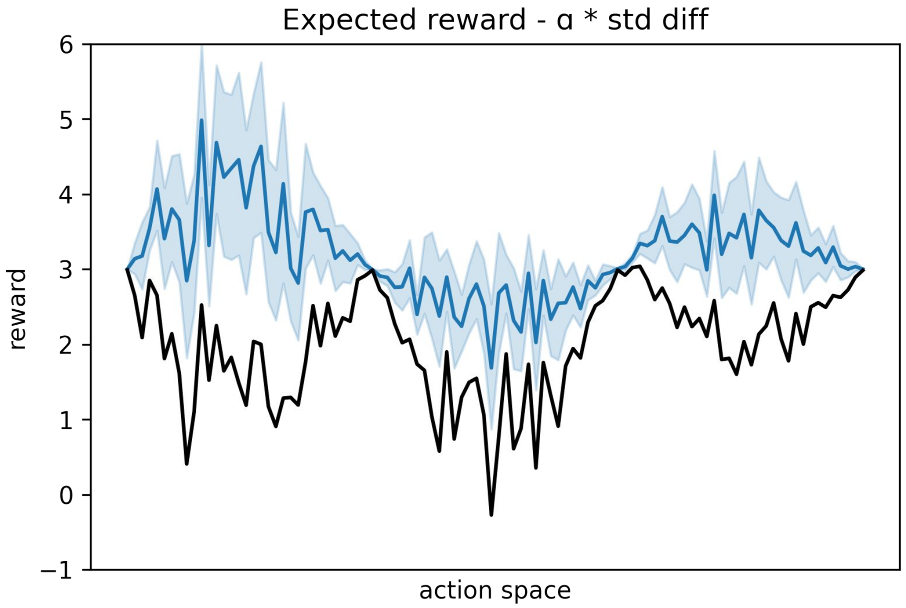
# SAC-N

- choose action based on expected reward
- penalize uncertainty
  - ➢ **use the standard deviation as a measure for uncertainty**



# SAC-N

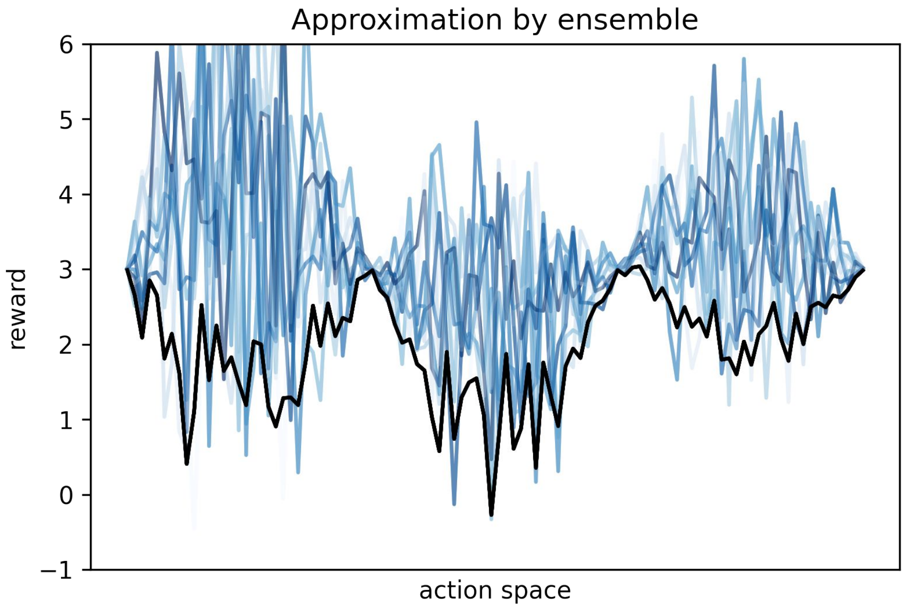
- choose action based on expected reward
- penalize uncertainty
  - ➢ **subtract a multiple of the standard deviation** from the expected reward



# SAC-N

- choose action based on expected reward
- penalize uncertainty
  - subtract a multiple of the standard deviation from the expected reward
  - **approximate uncertainty by ensemble of Q-networks**

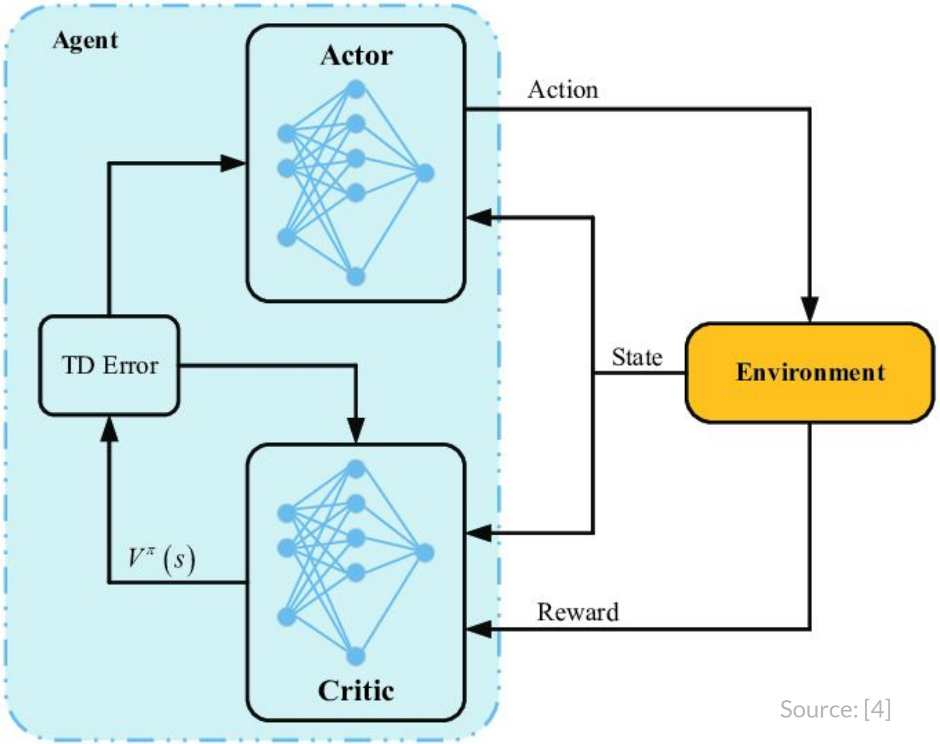
$$\mathbb{E}\left[\min_{j=1,\dots,N} Q_j(\mathbf{s},\mathbf{a})\right]$$
$$\approx m(\mathbf{s},\mathbf{a}) - \Phi^{-1}\left(\frac{N-\frac{\pi}{8}}{N-\frac{\pi}{4}+1}\right)\sigma(\mathbf{s},\mathbf{a})$$



# Before: Actor Critic Model

policy-based actor

value-based critic

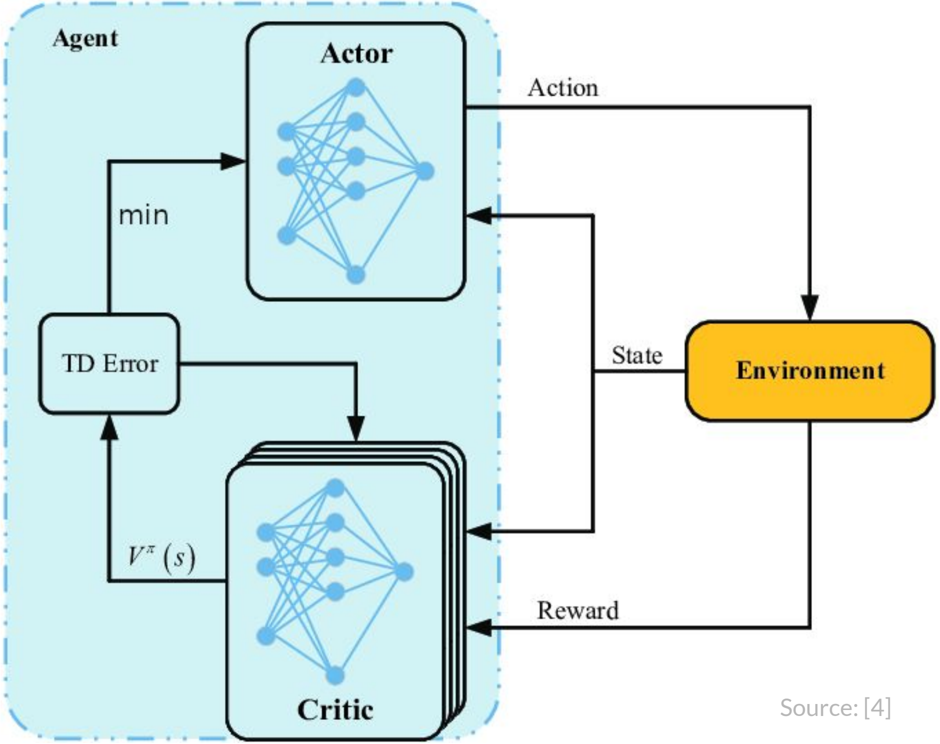



Source: [4]

# SAC-N

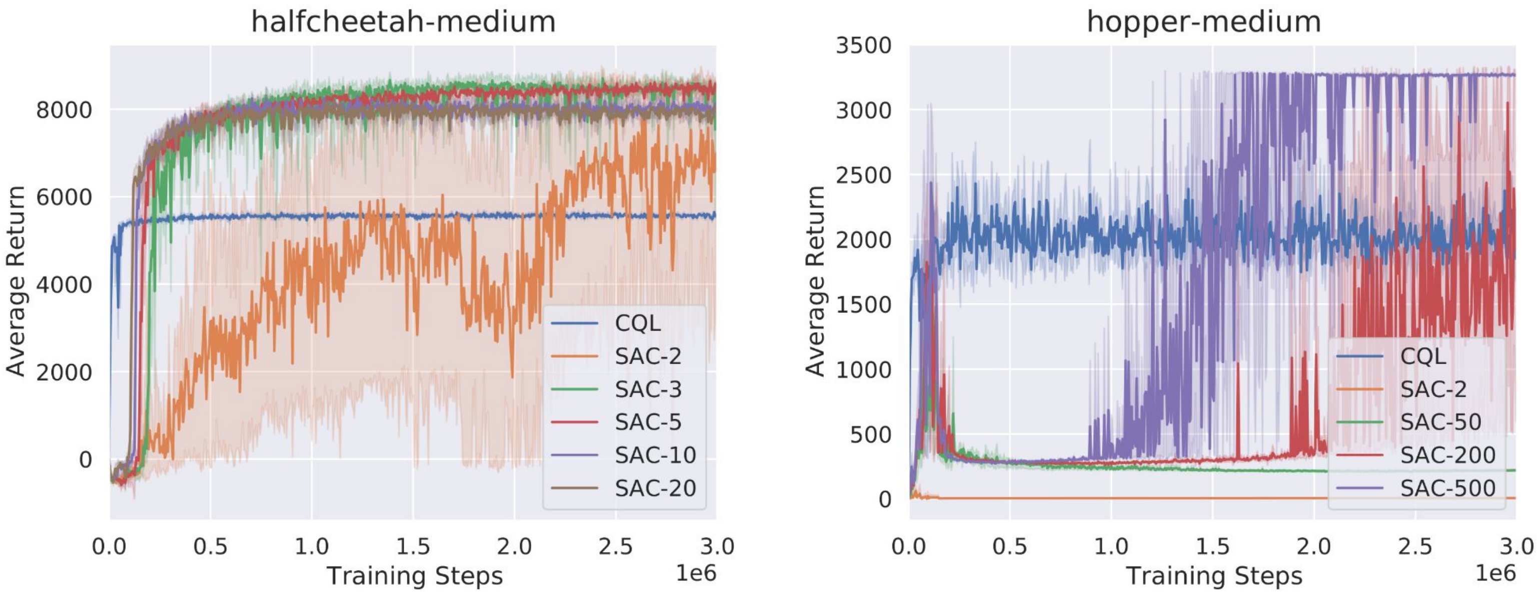
policy-based actor

ensemble of
value-based critics




Source: [4]

# SAC-N - Results




Source: [1]

# SAC-N → EDAC

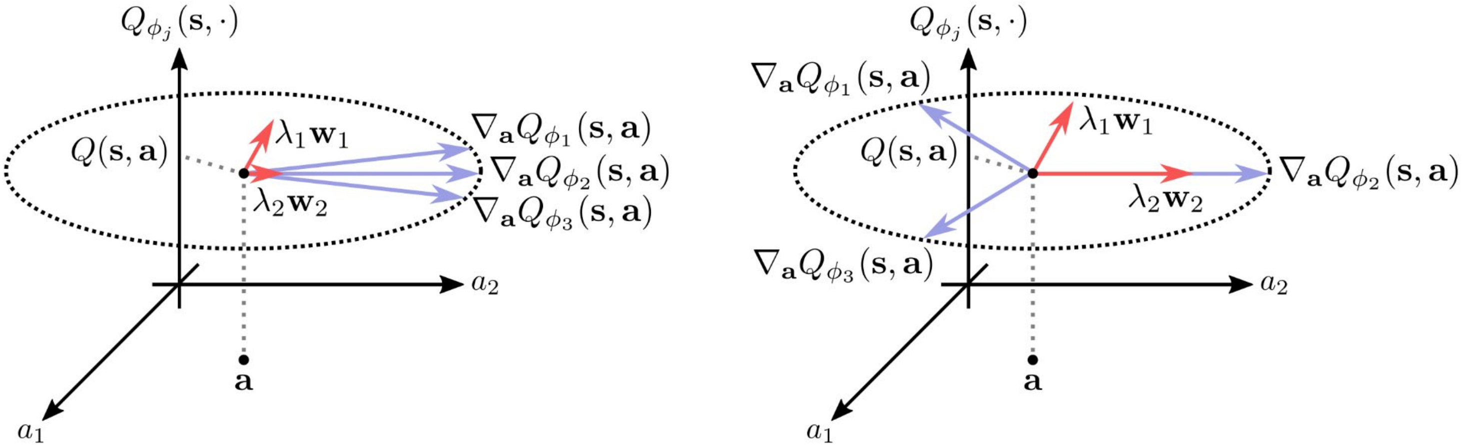
**Problem:**

A high number of ensemble networks is required, because often many are quite similar to each other.

**Idea:**

Make sure the ensemble networks are as diverse as possible for OOD actions. Then, fewer networks are required to penalize all possible OOD actions.

# EDAC (Ensemble-Diversified Actor Critic)




$\mathrm{Var}(Q_{\phi_j}(\mathbf{s}, \mathbf{a} + k\mathbf{w}_2))$ is small so that $\mathbf{a} + k\mathbf{w}_2$ is not sufficiently penalized.

$\mathrm{Var}(Q_{\phi_j}(\mathbf{s}, \mathbf{a} + k\mathbf{w}))$ is large for every direction $\mathbf{w}$ so that all OOD actions are sufficiently penalized.

(a) Without ensemble gradient diversification

(b) With ensemble gradient diversification

Source: [1]

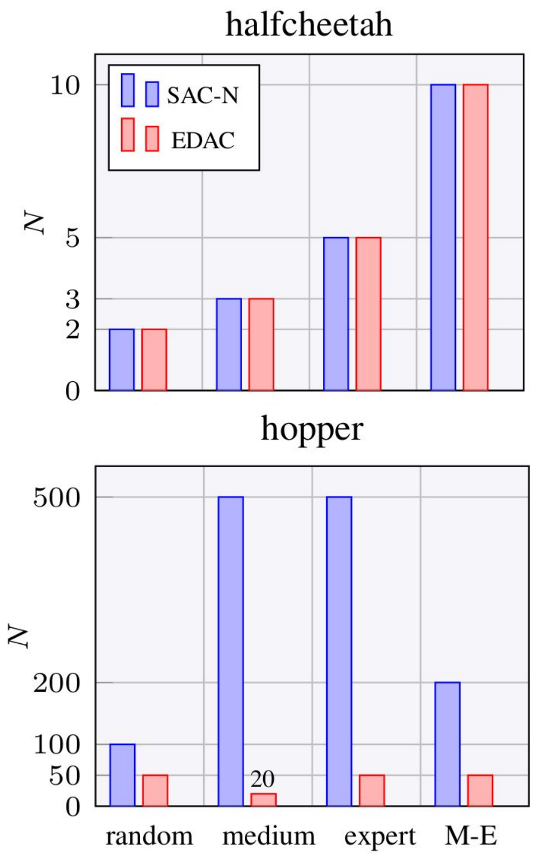
# EDAC

Maximize the smallest eigenvalue of the variance of the Q-values for near-distribution OOD actions

$$\underset{\phi}{\text{maximize}}\ \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[\lambda_{\min}\left(\mathrm{Var}\left(\nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s},\mathbf{a})\right)\right)\right]$$

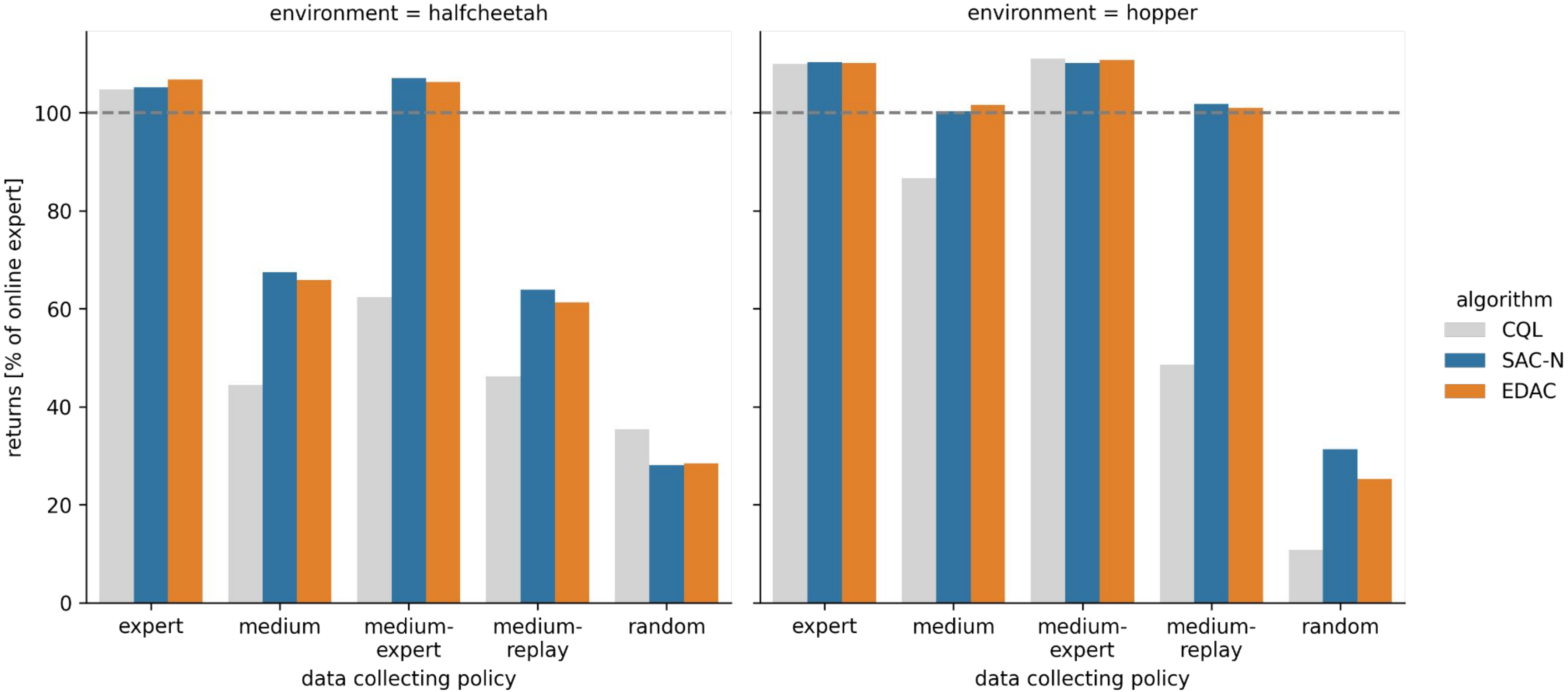
EDAC needs up 90% fewer ensemble networks, compared to SAC-N.

However, this highly varies from task to task.

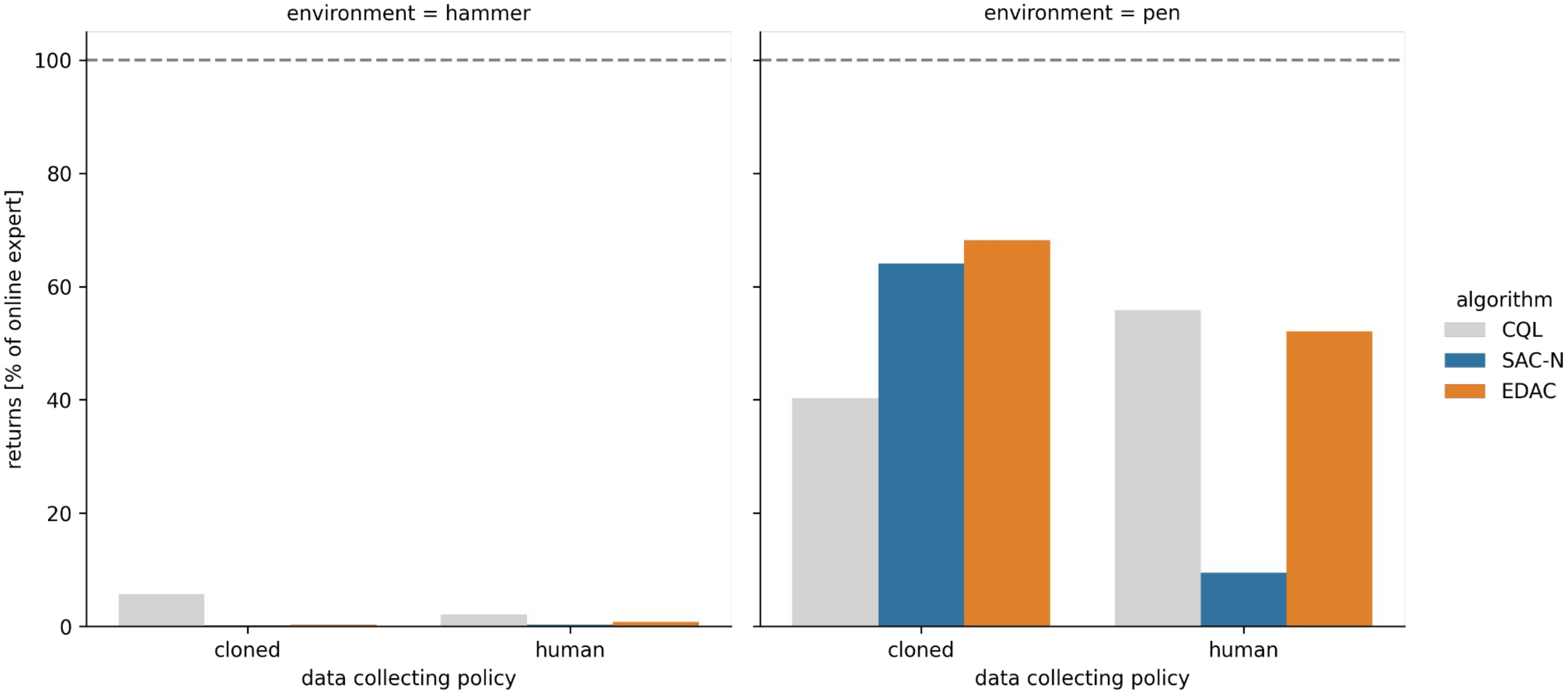



Source: [1]

# Comparison: MuJoCo Gym



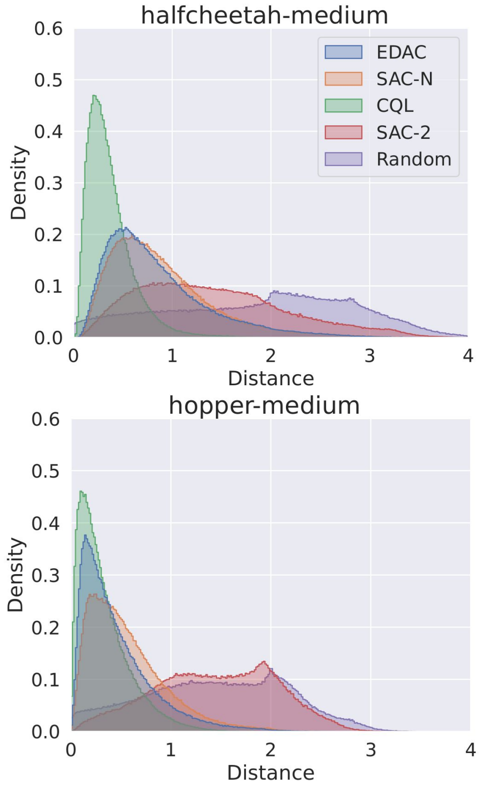
# Comparison: Adroit



# Comparison: dataset-similarity

SAC-N and EDAC are quite close to the actions in the dataset, but less so than CQL.

Due to the higher average returns, this could indicate a better tradeoff between staying safe, i.e. close to the dataset, and utilizing favorable OOD actions.




Source: [1]

# Comparison: Runtime

- EDAC is significantly faster and more memory efficient compared to SAC-500
- EDAC is faster than CQL, but has a higher memory footprint

| | **Runtime** (s/epoch) | **GPU Mem.** (GB) |
|---|---|---|
| **SAC** | 21.4 | 1.3 |
| **CQL** | 38.2 | 1.4 |
| **SAC-**500 | 44.1 | 5.1 |
| **EDAC** | 30.8 | 1.8 |

# Conclusion

- With offline RL, we can avoid trial and error, i.e. dangerous exploration, by using existing data
- A major difficulty are OOD actions, which are not part of the dataset
- We can estimate the uncertainty of the action values und thereby avoid OOD actions by using ensemble networks → SAC-N
- We can reduce the necessary ensemble size by diversification → EDAC
- EDAC outperforms the previous state-of-the-art method (CQL), while being faster (as of 2021)

# References


[1] An, G., Moon, S., Kim, J., & Song, H.O. (2021). Uncertainty-Based Offline Reinforcement Learning with Diversified Q-Ensemble. Neural Information Processing Systems.

[2] Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative Q-Learning for Offline Reinforcement Learning. ArXiv, abs/2006.04779.

[3]

[4] Giang, Hoang & Hoan, Tran & Thanh, Pham & Koo, Insoo. (2020). Hybrid NOMA/OMA-Based Dynamic Power Allocation Scheme Using Deep Reinforcement Learning in 5G Networks. Applied Sciences. 10. 4236. 10.3390/app10124236.

[5] Justin Fu (2020). D4RL: Building Better Benchmarks for Offline Reinforcement Learning. https://bair.berkeley.edu/blog/2020/06/25/D4RL/