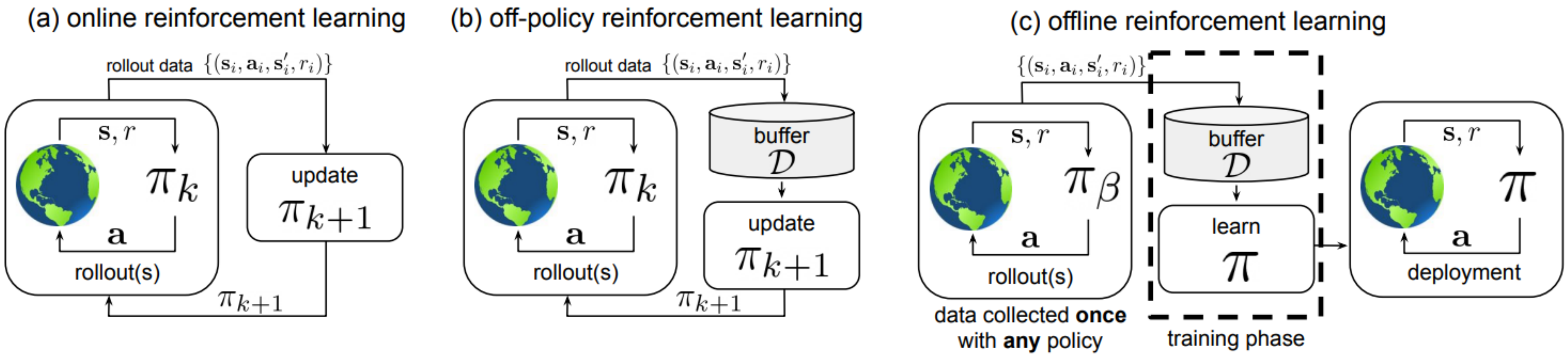
Uncertainty-Based Offline Reinforcement Learning with Diversified Q-Ensemble

ICML'21

Gaon An, Seungyong Moon, Jang-Hyun Kim, Hyun Oh Song

Seoul National University, Neural Processing Research Center, DeepMetrics

Offline RL



We aim to learn a policy π from the history of trajectories $\mathcal{D} = \{s_t, a_t, s'_t, r_t\}$ generated by behavioral policy π_β s.t. the performance $\pi \geq \pi_\beta$(Which means we want to train an agent's policy π only from the history of trajectories $\mathcal{D}$)

Challenge

- Extrapolation Error
  - The agent may overestimate unseen $Q^{\pi}(s,a)$, which gives a higher $Q^{\pi}(s,a)$ value than the optimal $Q^{\pi^*}(s,a)$ value .
  - In the offline setting, the policy cannot correct such over-optimistic Q-values.

- Solution
  - If we add a regularizer to the equation in order to make the agent **(1) underestimate the $Q^{\pi}(s, a)$ value of unseen action a given a state s** or **(2) choose the action that closes to the action already in the history trajectory $\mathcal{D}$ given a state(in practice, choose the action that are higher than the a threshold probability from the action distribution)**, we can avoid the crazy actions that the agent may do.
  - But a trade-off between **Optimality and Conservativeness**.

$$\hat{Q}^{\pi}_{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \left(r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{a}'\sim\pi_k(\mathbf{a}'|\mathbf{s}')}[\hat{Q}^{\pi}_{k}(\mathbf{s}',\mathbf{a}')]\right)\right)^2\right]$$

$$\pi_{k+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{s}\sim\mathcal{D}}\left[\mathbb{E}_{\mathbf{a}\sim\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^{\pi}_{k+1}(\mathbf{s},\mathbf{a})]\right] \text{ s.t. } D(\pi,\pi_\beta) \leq \epsilon.$$

Idea

- Penalize the Q-function with the most pessimistic Q-network of the ensemble Q-network.

Modify the following SAC objective

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

$$\max_{\theta} \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi_{\theta}(\cdot|\mathbf{s})}\left[Q_{\phi}(\mathbf{s},\mathbf{a})-\beta\log\pi_{\theta}(\mathbf{a}\mid\mathbf{s})\right]$$

Idea

To SAC-N

$$\min_{\phi_i} \mathbb{E}_{s,a,s'\sim\mathcal{D}} \left[\left(Q_{\phi_i}(s,a) - \left(r(s,a) + \gamma \mathbb{E}_{a'\sim\pi_\theta(\cdot|s')} \left[\min_{j=1,...,N} Q_{\phi'_j}(s',a') - \beta \log \pi_\theta(a' \mid s') \right] \right) \right)^2 \right]$$

$$\max_\theta \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_\theta(\cdot|s)} \left[\min_{j=1,...,N} Q_{\phi_j}(s,a) - \beta \log \pi_\theta(a \mid s) \right]$$

Where ϕ is the parameters of the Q-network Q_ϕ, θ is the parameters of policy network π_θ. The subscript j means the j-th network.

Idea

And the authors surprisingly found that **SAC-N will outperform than the SOTA offline-RL algorithm "CQL" when the number of ensemble is large enough.**

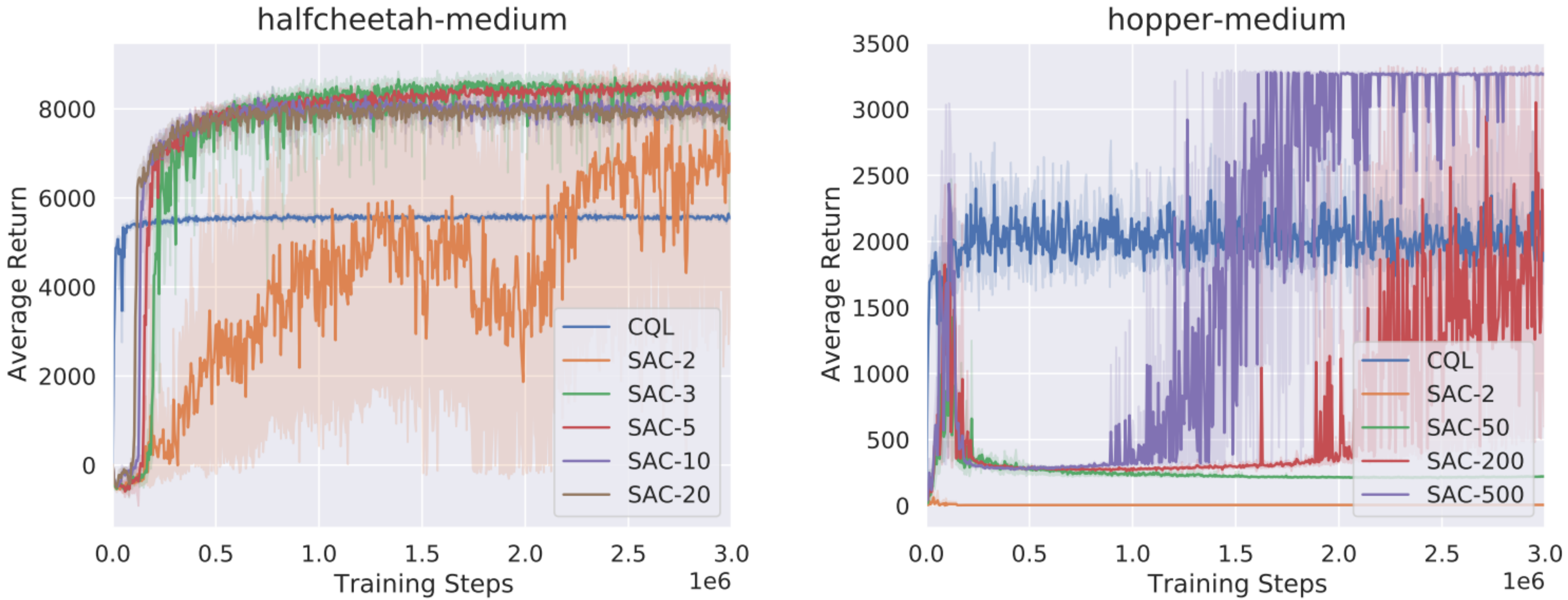


Figure 1: Performance of SAC-N on halfcheetah-medium and hopper-medium datasets while varying N, compared to CQL. 'Average Return' denotes the undiscounted return of each policies on evaluation. Results averaged over 4 seeds.

Idea

- Obviously, **the redundant Q-networks of SAC-N cost lots of computation**. The authors aim to **reduce the size of the ensemble Q-network while achieving the same performance.**
- The authors found that the **performance of SAC-N is negatively correlated with the degree to which the input gradients of Q-functions $\nabla_a Q_{\phi_j}(s,a)$ are aligned, which increases with** N.
- Note that **out-of-distribution state** means **the probability of the state that appears in the dataset is lower than a given threshold**. Similarly, **Out-of-distribution action means the action that appears in the dataset is lower than a given threshold and state.** In the other hand, **in-distribution action** means **higher than the threshold**.

Argument: Agent performance -> Variance of Q-value of OOD action -> Diversification of the gradients of the Q-network

Evidence 1

The Q-value predictions for the OOD actions have a higher variance.

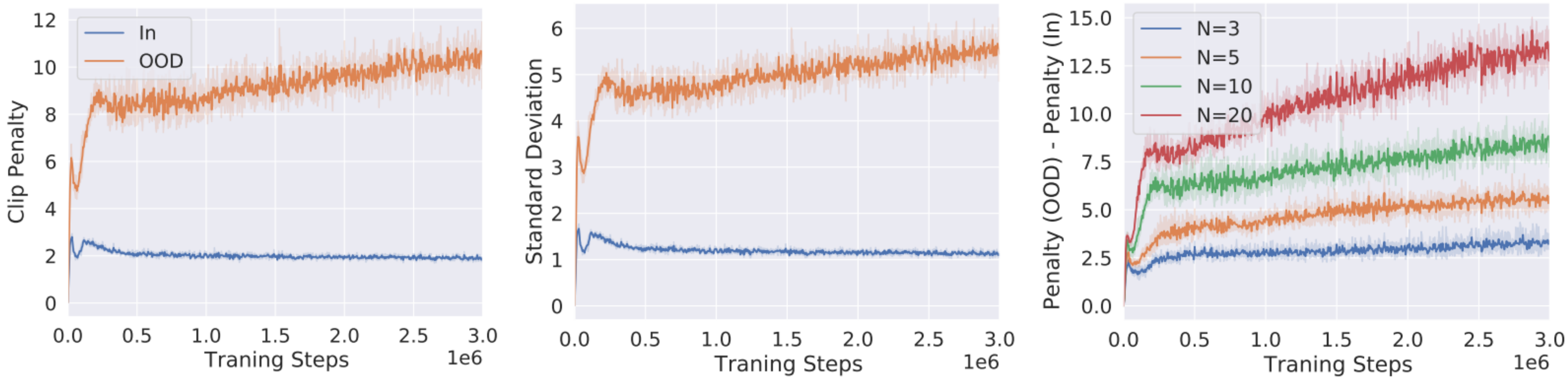
- Here we define the **penalty from the clipping as**

$$\mathbb{E}_{s\sim D,a\sim\pi(\cdot|s)}\left[\frac{1}{N}\sum_{j=1}^{N} Q_{\phi_j}(s,a)-\min_{j=1,\ldots,N} Q_{\phi_j}(s,a)\right]$$

- We also **approximate the expected minimum of the realizations** following the work of Royston

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

Where we suppose $Q(s,a)\sim\mathcal{N}(m(s,a),\sigma(s,a))$ and Φ is the CDF of the standard Gaussian distribution.



(a) Clip penalty (b) Standard deviation (c) Clip penalty gap

Figure 2: (a) and (b) each plots the size of the clip penalty and the standard deviation of the Q-value estimates for in-distribution (behavior) and OOD (random) actions while training SAC-10 on halfcheetah-medium dataset. (c) plots the gap of the clip penalty between the in-distribution and OOD actions while varying N. Results averaged over 4 seeds.

The Q-value predictions for the **OOD actions have a higher variance** and the **size of the penalty and the standard deviation are highly correlated**.

Evidence 2

The performance of the learned policy degrades significantly when the Q-functions share a similar local structure.

The minimum cosine similarity between the gradients of the Q-functions is

$$\min_{i \neq j} \langle \nabla_a Q_{\phi_i}(s,a), \nabla_a Q_{\phi_j}(s,a) \rangle$$

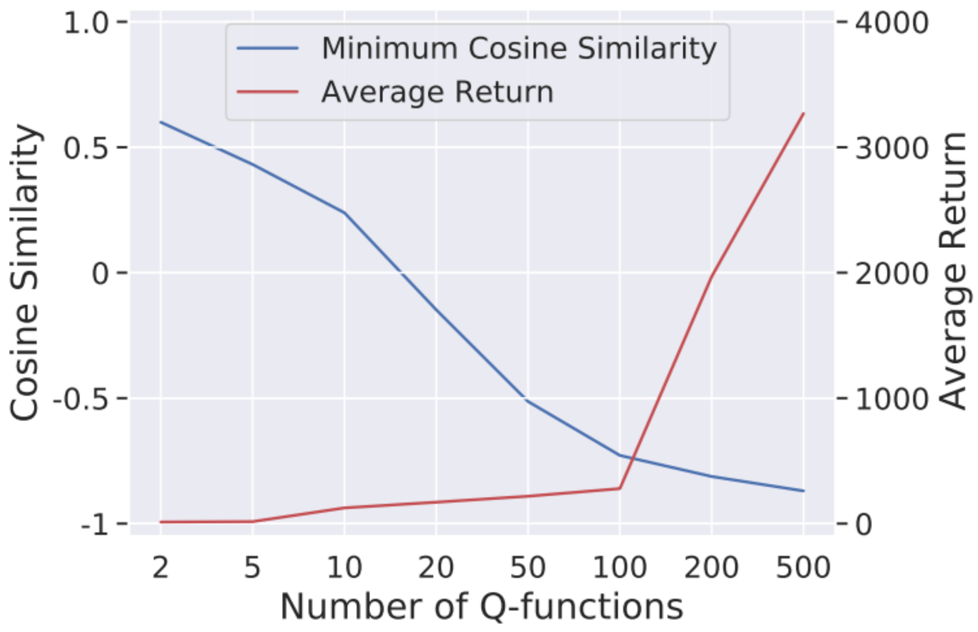


Figure 4: Plot of the minimum cosine similarity between the input gradients of Q-functions and the average return while varying the number of Q-functions.

We use Taylor expansion to expand the Q function $Q_{\phi_j}(s, a + kw_2)$

$$
\begin{aligned}
\mathrm{Var}(Q_{\phi_j}(s, a + kw_2)) &\approx \mathrm{Var}(Q_{\phi_j}(s, a) + k\langle w_2, \nabla_a Q_{\phi_j}(s, a)\rangle) \\
&= \mathrm{Var}(Q(s, a) + k\langle w_2, \nabla_a Q_{\phi_j}(s, a)\rangle) \\
&= k^2 \mathrm{Var}(\langle w_2, \nabla_a Q_{\phi_j}(s, a)\rangle) \\
&= k^2 w_2^\top \mathrm{Var}(\nabla_a Q_{\phi_j}(s, a)) w \\
&\geq k^2 w_{\min}^\top \mathrm{Var}(\nabla_a Q_{\phi_j}(s, a)) w_{\min} \\
&= k^2 \lambda_{min}
\end{aligned}
$$

Where $w_{\min}$ and $\lambda_{\min}$ are the smallest eigenvector and eigenvalue for the matrix $\mathrm{Var}(\nabla_a Q_{\phi_j}(s, a))$

However, It's hard to compute the smallest eigenvalue. Thus, the authors **decide to maximize the sum of the eigenvalues instead of the smallest eigenvalue.**

$$k^2\lambda_{\min} \leq \frac{k^2}{|\mathcal{A}|}\sum_{j=1}^{|\mathcal{A}|}\lambda_j$$

Since **the total variance is equal to the sum of all eigenvalues**(reference), derive

$$= \frac{k^2}{|\mathcal{A}|}\mathrm{tr}(\mathrm{Var}(\nabla_a Q_{\phi_j}(s,a)))$$

Lemma 1. *The total variance of the matrix* $\mathrm{Var}\left(\nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s},\mathbf{a})\right)$ *is equal to* $1 - \|\bar{q}\|_2^2$, *where* $\bar{q} = \frac{1}{N}\sum_{j=1}^{N}\nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s},\mathbf{a})$.

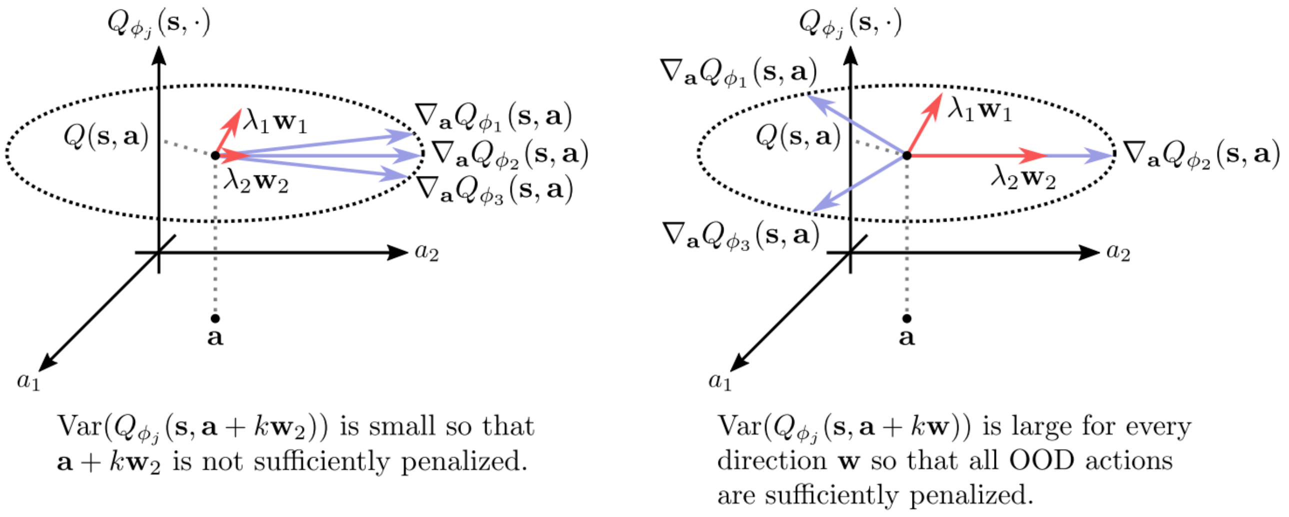
With Lemma 1, we can derive

$$= \frac{k^2}{|\mathcal{A}|}(1 - ||\bar{q}||_2^2) = \frac{k^2}{|\mathcal{A}|}(1 - \langle\frac{1}{N}\sum_{i=1}^{N}q_i, \frac{1}{N}\sum_{j=1}^{N}q_j\rangle)$$

Let $\min_{i \neq j} \langle \nabla_a Q_{\phi_j}(s,a), \nabla_a Q_{\phi_j}(s,a) \rangle = 1 - \epsilon$. With proposition 1, we can derive

$$\leq \frac{1}{|\mathcal{A}|} \frac{N-1}{N} k^2 \epsilon = \frac{1}{|\mathcal{A}|} \frac{N-1}{N} k^2 (1 - \min_{i \neq j} \langle \nabla_a Q_{\phi_j}(s,a), \nabla_a Q_{\phi_j}(s,a) \rangle)$$

If $\mathrm{Var}(Q_{\phi_j}(s, a + kw_2))$ is small, according to the approximation of the minimum Q-network ensemble is $\mathbb{E}\left[\min_{j=1,\ldots,N} Q_j(s,a)\right] \approx m(s,a) - \Phi^{-1}\left(\frac{N-\frac{\pi}{8}}{N-\frac{\pi}{4}+1}\right)\sigma(s,a)$, the action $a + kw_2$ is not sufficiently penalized.



(a) Without ensemble gradient diversification

(b) With ensemble gradient diversification

Figure 3: Illustration of the ensemble gradient diversification. The vector $\lambda_i \mathbf{w}_i$ represents the normalized eigenvector $\mathbf{w}_i$ of $\mathrm{Var}(\nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s}, \mathbf{a}))$ multiplied by its eigenvalue λ_i.

As a result, we now connect the inner product of the gradients of the Q-network(alignment) and the variance of the OOD action $\mathrm{Var}(Q_{\phi_j}(s, a + kw_2))$.

Thus, we aim to enlarge the penalty of OOD action. As a result, we aim to diversify the gradients of the Q-network ensemble $\nabla_a Q_{\phi_i}(s,a)$

$$\min_{\phi} J_{ES}(Q_\phi) := \mathbb{E}_{s,a\sim\mathcal{D}}\left[\langle \frac{1}{N}\sum_{i=1}^{N}\nabla_a Q_{\phi_i}(s,a), \frac{1}{N}\sum_{j=1}^{N}\nabla_a Q_{\phi_j}(s,a)\rangle\right]$$

Then, we can reformulate the equation

$$\min_{\phi} J_{ES}(Q_\phi) := \mathbb{E}_{s,a\sim\mathcal{D}}\left[\frac{1}{N-1}\sum_{1\le i\ne j\le N}\langle \nabla_a Q_{\phi_i}(s,a), \nabla_a Q_{\phi_j}(s,a)\rangle\right]$$

Algorithm

Algorithm 1 Ensemble-Diversified Actor Critic (EDAC)

1: Initialize policy parameters θ, Q-function parameters $\{\phi_j\}_{j=1}^N$, target Q-function parameters $\{\phi'_j\}_{j=1}^N$, and offline data replay buffer $\mathcal{D}$

2: **repeat**

3: Sample a mini-batch $B = \{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')\}$ from $\mathcal{D}$

4: Compute target Q-values (shared by all Q-functions):

$$y(r, \mathbf{s}') = r + \gamma \left(\min_{j=1,\ldots,N} Q_{\phi'_j}(\mathbf{s}', \mathbf{a}') - \beta \log \pi_\theta(\mathbf{a}' \mid \mathbf{s}') \right), \quad \mathbf{a}' \sim \pi_\theta(\cdot \mid \mathbf{s}')$$

5: Update each Q-function Q_{ϕ_i} with gradient descent using

$$\nabla_{\phi_i} \frac{1}{|B|} \sum_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\in B} \left(\left(Q_{\phi_i}(\mathbf{s}, \mathbf{a}) - y(r, \mathbf{s}') \right)^2 + \frac{\eta}{N-1} \sum_{1 \le i \ne j \le N} \mathrm{ES}_{\phi_i,\phi_j}(\mathbf{s}, \mathbf{a}) \right)$$

6: Update policy with gradient ascent using

$$\nabla_\theta \frac{1}{|B|} \sum_{\mathbf{s}\in B} \left(\min_{j=1,\ldots,N} Q_{\phi_j}(\mathbf{s}, \tilde{\mathbf{a}}_\theta(\mathbf{s})) - \beta \log \pi_\theta(\tilde{\mathbf{a}}_\theta(\mathbf{s}) \mid \mathbf{s}) \right),$$

where $\tilde{\mathbf{a}}_\theta(\mathbf{s})$ is a sample from $\pi_\theta(\cdot \mid \mathbf{s})$ which is differentiable w.r.t. θ via the reparametrization trick.

7: Update target networks with $\phi'_i \leftarrow \rho\phi'_i + (1-\rho)\phi_i$

Lemma 1. *The total variance of the matrix* $\mathrm{Var}\left(\nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s}, \mathbf{a})\right)$ *is equal to* $1 - \|\bar{q}\|_2^2$, *where* $\bar{q} = \frac{1}{N}\sum_{j=1}^{N} \nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s}, \mathbf{a})$.

Proposition 1. *Suppose* $Q_{\phi_j}(\mathbf{s}, \mathbf{a}) = Q(\mathbf{s}, \mathbf{a})$ *and* $Q_{\phi_j}(\mathbf{s}, \cdot)$ *is locally linear in the neighborhood of* $\mathbf{a}$ *for all* $j \in [N]$. *Let* $\lambda_{\min}$ *and* $\mathbf{w}_{\min}$ *be the smallest eigenvalue and the corresponding normalized eigenvector of the matrix* $\mathrm{Var}\left(\nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s}, \mathbf{a})\right)$ *and* $\epsilon > 0$ *be the value such that* $\min_{i \neq j} \left\langle \nabla_{\mathbf{a}} Q_{\phi_i}(\mathbf{s}, \mathbf{a}), \nabla_{\mathbf{a}} Q_{\phi_j}(\mathbf{s}, \mathbf{a}) \right\rangle = 1 - \epsilon$. *Then, the variance of the Q-values for an OOD action in the neighborhood along the direction of* $\mathbf{w}_{\min}$ *is upper-bounded as follows:*

$$\mathrm{Var}\left(Q_{\phi_j}(\mathbf{s}, \mathbf{a} + k\mathbf{w}_{\min})\right) \leq \frac{1}{|\mathcal{A}|} \frac{N-1}{N} k^2 \epsilon,$$

where $|\mathcal{A}|$ *is the action space dimension.*

Proof Sketch

Then, since **the smallest eigenvalue is hard to compute**, we compute the **sum of the all eigenvalues of the covariance matrix** $\mathrm{Var}(\nabla_a Q_{\phi_j}(s,a))$

$$\lambda_{\min} \leq \frac{1}{|\mathcal{A}|} \sum_{j=1}^{|\mathcal{A}|} \lambda_j$$

Since **the total variance is equal to the sum of all eigenvalues**, we can derive

$$= \frac{1}{|\mathcal{A}|} \mathrm{tr}(\mathrm{Var}(\nabla_a Q_{\phi_j}(s,a)))$$

Let $q_j = \nabla_a Q_{\phi_j}(s,a)$ be the normalized gradients of Q-network, and $\bar{q} = \frac{1}{N} \sum_j$

$$= \frac{1}{|\mathcal{A}|} \mathrm{tr}(\frac{1}{N} \sum_j (q_j - \bar{q})(q_j - \bar{q})^\top)$$

$$= \frac{1}{|\mathcal{A}|} \frac{1}{N} \sum_j \mathrm{tr}((q_j - \bar{q})(q_j - \bar{q})^\top)$$

$$= \frac{1}{|\mathcal{A}|} \frac{1}{N} \sum_j \mathrm{tr}((q_j - \bar{q})^\top (q_j - \bar{q}))$$

$$= \frac{1}{|\mathcal{A}|} \frac{1}{N} \sum_j \mathrm{tr}((q_j - \bar{q})^\top (q_j - \bar{q}))$$

$$= \frac{1}{|\mathcal{A}| N} \sum_j (q_j^\top q_j + \bar{q}^\top \bar{q} - 2 q_j^\top \bar{q})$$

$$= \frac{1}{|\mathcal{A}|} (1 + \bar{q}^\top \bar{q} - 2 (\frac{1}{N} \sum_j q_j^\top) \bar{q})$$

$$= \frac{1}{|\mathcal{A}|} (1 - ||\bar{q}||_2^2)$$

$$= \frac{1}{|\mathcal{A}|} (1 - \langle \frac{1}{N} \sum_{i=1}^{N} q_i, \frac{1}{N} \sum_{j=1}^{N} q_j \rangle)$$

$$= \frac{1}{|\mathcal{A}|}(1 - (\frac{1}{N^2}(\sum_{j=1}^{N}\langle q_j, q_j\rangle + \sum_{1\le i\neq j\le N}\langle q_i, q_j\rangle))))$$

$$= \frac{1}{|\mathcal{A}|}(1 - (\frac{1}{N^2}(\sum_{j=1}^{N}\langle q_j, q_j\rangle + \sum_{1\le i\neq j\le N}\langle q_i, q_j\rangle))))$$

Let $\min_{i\neq j}\langle \nabla_a Q_{\phi_j}(s,a), \nabla_a Q_{\phi_j}(s,a)\rangle = 1-\epsilon$

$$\le \frac{1}{|\mathcal{A}|}(1 - (N + N(N-1)(1-\epsilon)))$$

$$= \frac{1}{|\mathcal{A}|}\frac{N-1}{N}\epsilon$$

Thus,

$$\begin{aligned}
&\mathrm{Var}(\nabla_a Q_{\phi_j}(s, a + k w_{\min})) \\
&= k^2 w_{\min}^\top \mathrm{Var}(\nabla_a Q_{\phi_j}(s, a)) w_{\min} \\
&= k^2 \lambda_{\min} \leq \frac{1}{|\mathcal{A}|}(1 - \langle \frac{1}{N}\sum_{i=1}^{N} q_i, \frac{1}{N}\sum_{j=1}^{N} q_j \rangle) \\
&\leq \frac{1}{|\mathcal{A}|}\frac{N-1}{N} k^2 \epsilon = \frac{1}{|\mathcal{A}|}\frac{N-1}{N} k^2 (1 - \min_{i \neq j} \langle \nabla_a Q_{\phi_j}(s,a), \nabla_a Q_{\phi_j}(s,a) \rangle)
\end{aligned}$$

Thus, instead of maximize the smallest eigenvalue $\max_\phi k^2 \lambda_{\min}$, **minimizing the cosine similarity of the gradients of the Q-networks is cheaper**

$$\min_\phi \mathbb{E}_{s,a\sim\mathcal{D}} \left[\frac{1}{N-1} \sum_{1 \leq i \neq j \leq N} \langle \nabla_a Q_{\phi_j}(s,a), \nabla_a Q_{\phi_j}(s,a) \rangle \right]$$

Experiment - D4RL Dataset

- expert: a fully trained online expert
- medium: a suboptimal policy with approximately 1/3 the performance of the expert
- medium-expert: a mixture of medium and expert policies
- medium-replay: the replay buffer of a policy trained up to the performance of the medium agent
- full-replay: the final replay buffer of the expert policy
- 1M transitions

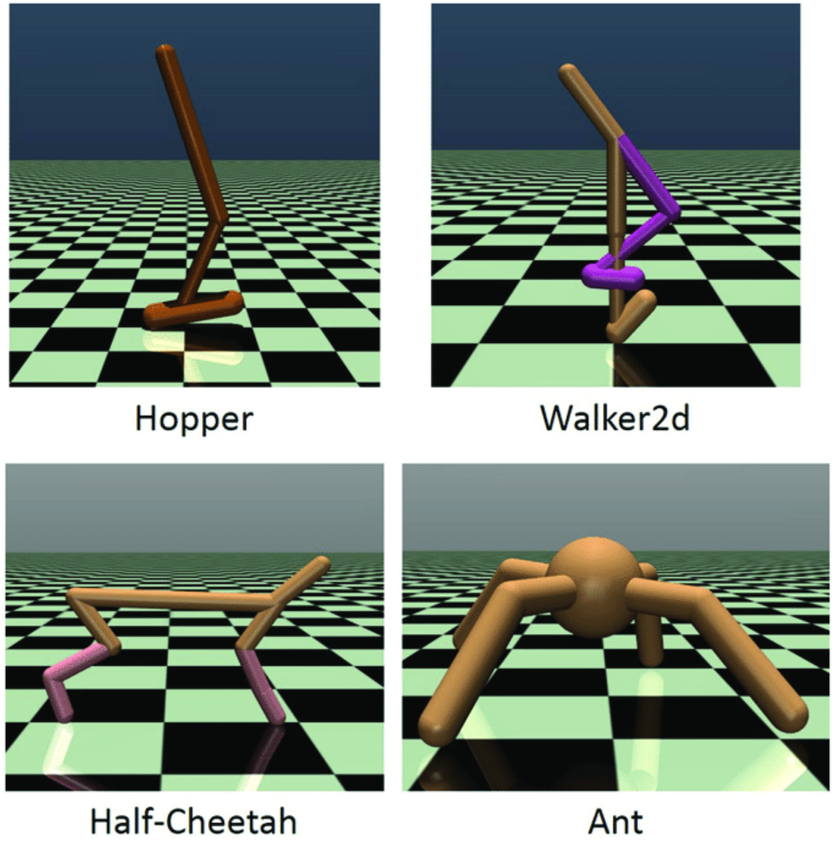


Table 1: Normalized average returns on D4RL Gym tasks, averaged over 4 random seeds. CQL (Paper) denotes the results reported in the original paper.

Task Name	**BC**	**SAC**	**REM**	**CQL (Paper)**	**CQL (Reproduced)**	**SAC-*N* (Ours)**	**EDAC (Ours)**
halfcheetah-random	2.2±0.0	29.7±1.4	-0.8±1.1	**35.4**	31.3±3.5	28.0±0.9	28.4±1.0
halfcheetah-medium	43.2±0.6	55.2±27.8	-0.8±1.3	44.4	46.9±0.4	**67.5±1.2**	**65.9±0.6**
halfcheetah-expert	91.8±1.5	-0.8±1.8	4.1±5.7	104.8	97.3±1.1	**105.2±2.6**	**106.8±3.4**
halfcheetah-medium-expert	44.0±1.6	28.4±19.4	0.7±3.7	62.4	95.0±1.4	**107.1±2.0**	**106.3±1.9**
halfcheetah-medium-replay	37.6±2.1	0.8±1.0	6.6±11.0	46.2	45.3±0.3	**63.9±0.8**	**61.3±1.9**
halfcheetah-full-replay	62.9±0.8	**86.8±1.0**	27.8±35.4	-	76.9±0.9	84.5±1.2	84.6±0.9
hopper-random	3.7±0.6	9.9±1.5	3.4±2.2	10.8	5.3±0.6	**31.3±0.0**	**25.3±10.4**
hopper-medium	54.1±3.8	0.8±0.0	0.7±0.0	86.6	61.9±6.4	**100.3±0.3**	**101.6±0.6**
hopper-expert	107.7±9.7	0.7±0.0	0.8±0.0	109.9	106.5±9.1	**110.3±0.3**	**110.1±0.1**
hopper-medium-expert	53.9±4.7	0.7±0.0	0.8±0.0	**111.0**	96.9±15.1	110.1±0.3	110.7±0.1
hopper-medium-replay	16.6±4.8	7.4±0.5	27.5±15.2	48.6	86.3±7.3	**101.8±0.5**	**101.0±0.5**
hopper-full-replay	19.9±12.9	41.1±17.9	19.7±24.6	-	101.9±0.6	**102.9±0.3**	**105.4±0.7**
walker2d-random	1.3±0.1	0.9±0.8	6.9±8.3	7.0	5.4±1.7	**21.7±0.0**	**16.6±7.0**
walker2d-medium	70.9±11.0	-0.3±0.2	0.2±0.7	74.5	79.5±3.2	**87.9±0.2**	**92.5±0.8**
walker2d-expert	108.7±0.2	0.7±0.3	1.0±2.3	**121.6**	109.3±0.1	107.4±2.4	115.1±1.9
walker2d-medium-expert	90.1±13.2	1.9±3.9	-0.1±0.0	98.7	109.1±0.2	**116.7±0.4**	**114.7±0.9**
walker2d-medium-replay	20.3±9.8	-0.4±0.3	12.5±6.2	32.6	76.8±10.0	**78.7±0.7**	**87.1±2.3**
walker2d-full-replay	68.8±17.7	27.9±47.3	-0.2±0.3	-	94.2±1.9	**94.6±0.5**	**99.8±0.7**
Average	49.9	16.2	6.2	-	73.7	**84.5**	**85.2**

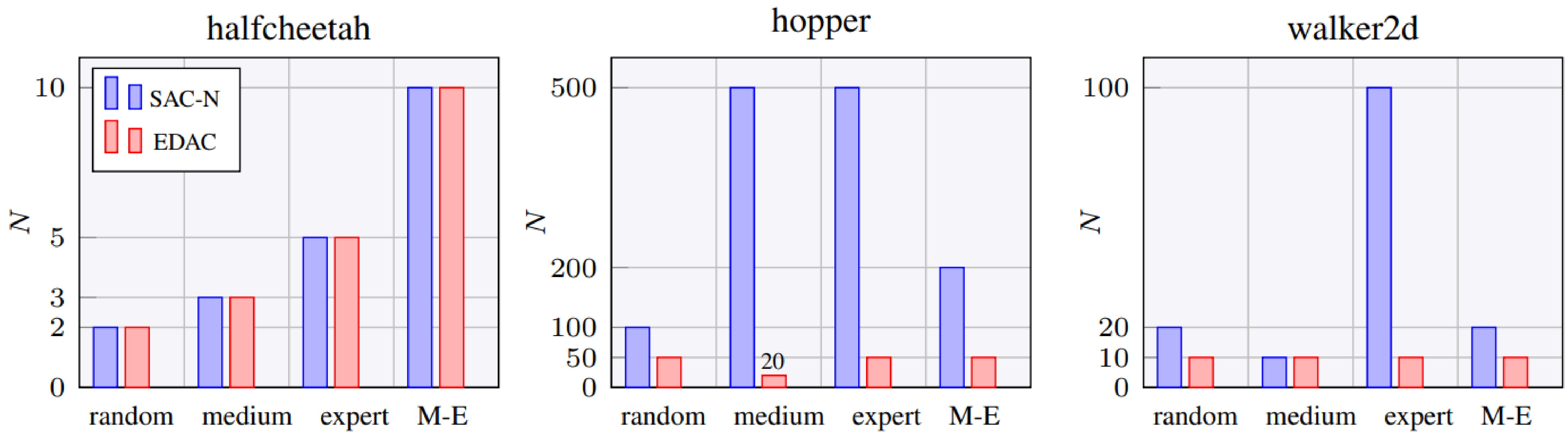


Figure 5: Minimum number of Q-ensembles (N) required to achieve the performance reported in Table 1. M-E denotes medium-expert. We omit the results of medium-replay and full-replay as SAC-N already works well with a small number of ensembles (less than or equal to 5). For more details of the experiment, please refer to Appendix C.

Table 3: Computational costs of each method.

	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

Table 2: Normalized average returns on D4RL Adroit tasks, averaged over 4 random seeds.

Task Name	**BC**	**SAC**	**REM**	**CQL (Paper)**	**CQL (Reproduced)**	**SAC-N (Ours)**	**EDAC (Ours)**
pen-human	25.8±8.8	4.3±3.8	5.4±4.3	**55.8**	35.2±6.6	9.5±1.1	52.1±8.6
hammer-human	3.1±3.2	0.2±0.0	0.3±0.0	2.1	0.6±0.5	0.3±0.0	0.8±0.4
door-human	2.8±0.7	-0.3±0.0	-0.3±0.0	9.1	1.2±1.8	-0.3±0.0	**10.7±6.8**
relocate-human	0.0±0.0	-0.3±0.0	-0.3±0.0	0.35	0.0±0.0	-0.1±0.1	0.1±0.1
pen-cloned	38.3±11.9	-0.8±3.2	-1.0±0.1	40.3	27.2±11.3	**64.1±8.7**	**68.2±7.3**
hammer-cloned	0.7±0.3	0.1±0.1	-0.3±0.0	5.7	1.4±2.1	0.2±0.2	0.3±0.0
door-cloned	0.0±0.0	-0.3±0.1	-0.3±0.0	3.5	2.4±2.4	-0.3±0.0	**9.6±8.3**
relocate-cloned	0.1±0.0	-0.1±0.1	-0.2±0.2	-0.1	0.0±0.0	0.0±0.0	0.0±0.0

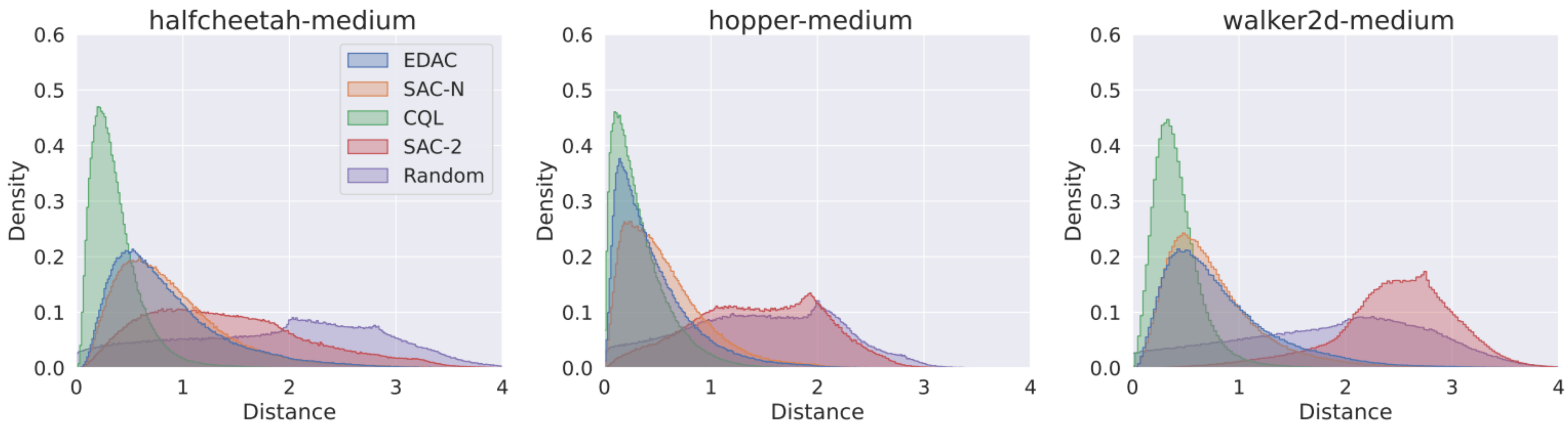


Figure 6: Histograms of the distances between the actions from each methods (EDAC, SAC-N, CQL, SAC-2, and a random policy) and the actions from the dataset. For more details of the experiment, please refer to Appendix C.

Conclusion

- SAC-N can be efficiently leveraged to construct an uncertainty-based offline RL method that outperforms previous methods on various datasets.
- we proposed Ensemble-Diversifying Actor-Critic (EDAC) that effectively reduces the required number of ensemble networks for quantifying and penalizing the epistemic uncertainty.