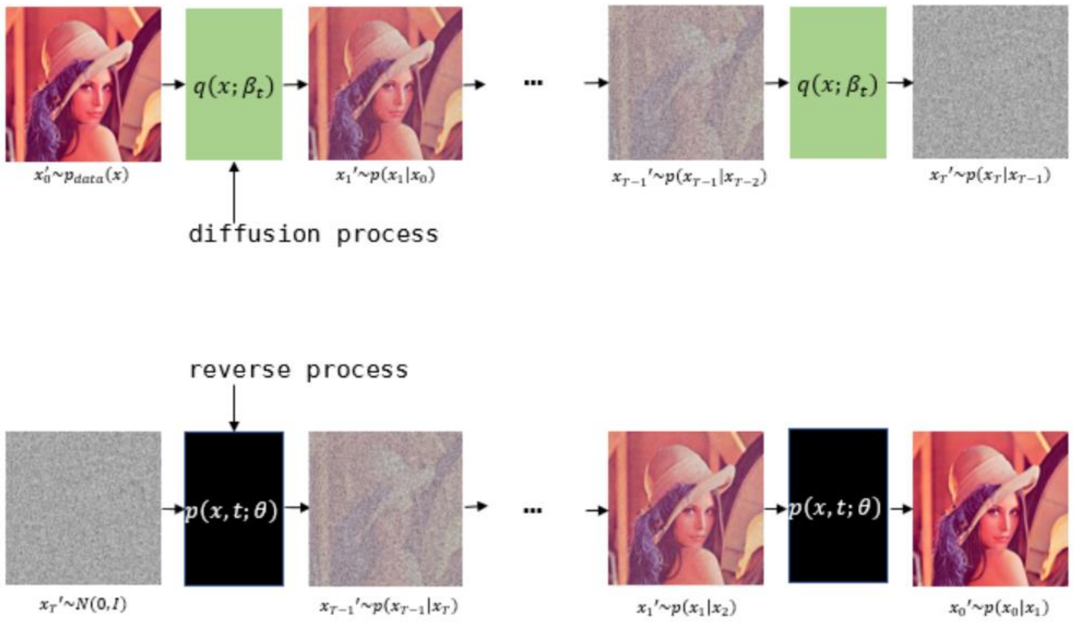
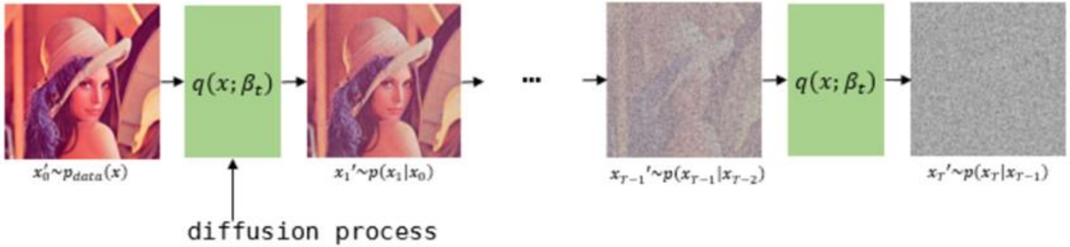
Diffusion Policies as an Expressive Policy Class for Offline Reinforcement Learning

Related works : Diffusion model(DDPM)



- Diffusion process에서는 원래 이미지에 noise를 조금씩, 반복적으로 추가하여 원래 이미지와 거의 independent한 noise인 latent variable을 생성
- 각 step은 Markov decision process라 가정
- Reverse process는 이 diffusion process의 역과정을 neural network를 통해 학습

Diffusion model : Diffusion process



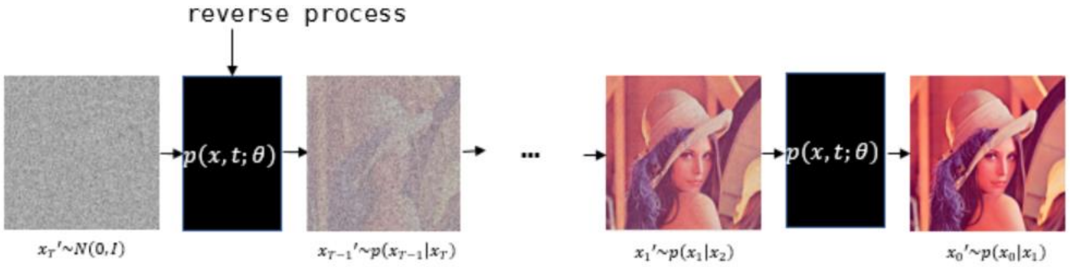
$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) = N(\sqrt{1-B_t}x_{t-1}, B_t I)$$

- q함수는 diffusion process이며 미세한 gaussian noise를 추가하는 과정
- Diffusion process는 trainable parameter가 없음(VDM에서 trainable로 변경)
- B_t는 noise scheduler

$$q(x_t|x_0) = N(x_t; \sqrt{\bar{\alpha}_t}x_{t-1}, (1-\bar{\alpha}_t)I)$$
$$= \sqrt{\bar{\alpha}_t}\, x_0 + \varepsilon\sqrt{1-\bar{\alpha}_t}, \varepsilon \sim N(0, I)$$
$$\text{with } \alpha_t = 1 - B_t \text{ and } \bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s$$

- 위의 수식으로t time에 있는 latent variable을 바로 계산할 수 있음

Diffusion model : Reverse process



$$q(x_t|x_{t-1}) \rightarrow q(x_{t-1}|x_t)$$

$$p_\theta(x_{0:T}) = p(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$
$$p_\theta(x_{t-1}|\, x_t) = N(\mu_\theta(x_t, t), \Sigma_\theta(x_t,t))$$

- mean에 해당하는 μ_θ와 covariance Σ_θ 학습이 목적
- NF에서는 change of variable theorem 으로 계산했지만 diffusion은 neural network를 사용
- 이때 diffusion process와 마찬가지로 reverse process 또한 Markov chain으로 가정함

Diffusion model (DDPM)

- DDPM의 핵심은 neural network로 표현되는 p 함수가 q 를 보고 noise를 걷어내는 과정을 학습하는 것

- loss는 VAE의 loss와 유사하게 negative log-likelihood로 전개됨

$$Loss_{Diffusion} = D_{KL}(q(z \mid x_0) \| P_\theta(x_0 \mid z)) - E_{z \sim q(z|x)}[\log P_\theta(z)]$$

$$= D_{KL}(q(z \mid x_0) \| P_\theta(z)) + \sum_{t=2} D_{kL}(q(x_{t-1} \mid x_t, x_0) \| P_\theta(x_{t-1} \mid x_t)) - E_q[\log P_\theta(x_0 \mid x_1)]$$

Regularizer on Encoder ***Denoising Process*** ***Reconstruction on Decoder***

***DDPM*에서는 *Loss*가 굉장히 간단한 식으로 정의됨**

$$\boxed{Loss_{DDPM} = \mathbb{E}_{x_0,\epsilon}\left[\left|\epsilon - \epsilon_\theta\left(\sqrt{\tilde{\alpha}_t} + \sqrt{1-\tilde{\alpha}_t}\epsilon, t\right)\right|^2\right]}$$

Related works : Offline RL

1. Constraining the learned value function to assign low values to OOD actions

2. Introducing model-based methods, which learn a model of the environment dynamics and perform pessimistic planning in the learned MDP

3. Treating offline RL as a problem of sequence prediction with return guidance(Offline Reinforcement Learning as One Big Sequence Modeling Problem,2021)

4. Regularizing how far the policy can deviate from the behavior policy

A method to perform policy regularization using diffusion models

Preliminaries

- MDP : $M = \{S, A, P, R, \gamma, d_0\}$
- In the offline setting, a static dataset $D \triangleq \{(s, a, r, s'\}$
- Diffusion process : $q(x_{1:T}|x_0) \coloneqq \prod_{t=1}^{T} q(x_t|x_{t-1})$
- Reverse diffusion chain, $p_\theta(x_{0:T}) \coloneqq N(x_T; 0, I) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$

Diffusion policy

- RL policy via the reverse process of a conditional diffusion model

$$\pi_\theta(a|s) = p_\theta(a^{0:N}|s) = N(a^N; 0, I) \prod_{t=1}^{T} p_\theta(a^{i-1}|a^i, s)$$

- We first sample $a^N \sim N(0, I)$ and them from the reverse diffusion chain as

$$p_\theta\left(a^{i-1}|a^i\right) = \frac{a^i}{\sqrt{\alpha_i}} - \frac{\beta_i}{\sqrt{1-\alpha_i}} \varepsilon_\theta(a^i, s^i, i) + \sqrt{\beta_i}\ \varepsilon$$

$$\beta_i = 1 - \alpha_i = 1 - e^{-\beta_{\min}\left(\frac{1}{N}\right) - 0.5(\beta_{\max} - \beta_{\min})\frac{2i-1}{N^2}},$$

- Objective function is proposed by DDPM,

$$\mathcal{L}_d(\theta) = \mathbb{E}_{i\sim\mathcal{U},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I}),(\boldsymbol{s},\boldsymbol{a})\sim\mathcal{D}}\left[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_i}\boldsymbol{a} + \sqrt{1-\bar{\alpha}_i}\boldsymbol{\epsilon}, \boldsymbol{s}, i)||^2\right]$$

Q-learning

- The policy-regularization loss $L_d(\theta)$ is a behavior-cloning term

- To improve the policy, we inject Q-value function guidance into the reverse diffusion chain in the training stage in order to learn to preferentially sample actions with high values

$$\pi = \underset{\pi_\theta}{\arg\min}\, \mathcal{L}(\theta) = \mathcal{L}_d(\theta) + \mathcal{L}_q(\theta) = \mathcal{L}_d(\theta) - \alpha \cdot \mathbb{E}_{\boldsymbol{s}\sim\mathcal{D},\boldsymbol{a}^0\sim\pi_\theta}\left[Q_\phi(\boldsymbol{s},\boldsymbol{a}^0)\right]$$

- We build two Q-networks, and target networks, optimize formula as

$$\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{s}_{t+1})\sim\mathcal{D},\boldsymbol{a}^0_{t+1}\sim\pi_{\theta'}}\left[\left|\left|\left(r(\boldsymbol{s}_t,\boldsymbol{a}_t)+\gamma\min_{i=1,2}Q_{\phi'_i}(\boldsymbol{s}_{t+1},\boldsymbol{a}^0_{t+1})\right)-Q_{\phi_i}(\boldsymbol{s}_t,\boldsymbol{a}_t)\right|\right|^2\right].$$

Algorithm

Algorithm 1 Diffusion Q-learning

Initialize policy network π_θ, critic networks Q_{ϕ_1} and Q_{ϕ_2}, and target networks $\pi_{\theta'}$, $Q_{\phi'_1}$ and $Q_{\phi'_2}$

for each iteration **do**

  Sample transition mini-batch $\mathcal{B} = \{(\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1})\} \sim \mathcal{D}$.

  # Q-value function learning

  Sample $\boldsymbol{a}^0_{t+1} \sim \pi_{\theta'}(\boldsymbol{a}_{t+1} \mid \boldsymbol{s}_{t+1})$ by Equation (1). $p_\theta(a^{i-1}|a^i) = \frac{a^i}{\sqrt{\alpha_i}} - \frac{\beta_i}{\sqrt{1-\alpha_i}}\varepsilon_\theta(a^i, s^i, i) + \sqrt{\beta_i}\,\varepsilon$

  Update Q_{ϕ_1} and Q_{ϕ_2} by Equation (4). (max Q backup by Kumar et al. (2020) could be added)

$$\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{s}_{t+1})\sim\mathcal{D},\boldsymbol{a}^0_{t+1}\sim\pi_{\theta'}}\left[\left|\left|\left(r(\boldsymbol{s}_t,\boldsymbol{a}_t) + \gamma \min_{i=1,2} Q_{\phi'_i}(\boldsymbol{s}_{t+1}, \boldsymbol{a}^0_{t+1})\right) - Q_{\phi_i}(\boldsymbol{s}_t,\boldsymbol{a}_t)\right|\right|^2\right].$$

  # Policy learning

  Sample $\boldsymbol{a}^0_t \sim \pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{s}_t)$ by Equation (1).

  Update policy by minimizing Equation (3). $\pi = \underset{\pi_\theta}{\arg\min}\, L(\theta) = L_d(\theta) + L_q(\theta)$

  # Update target networks

  $\theta' = \rho\theta' + (1-\rho)\theta$, $\phi'_i = \rho\phi'_i + (1-\rho)\phi_i$ for $i = \{1, 2\}$.

end for

Experiments

Gym Tasks	BC	AWAC	Diffuser	MoRel	Onestep RL	TD3+BC	DT	CQL	IQL	Diffusion-QL
halfcheetah-medium-v2	42.6	43.5	44.2	42.1	48.4	48.3	42.6	44.0	47.4	**51.1** ± 0.5
hopper-medium-v2	52.9	57.0	58.5	**95.4**	59.6	59.3	67.6	58.5	66.3	90.5 ± 4.6
walker2d-medium-v2	75.3	72.4	79.7	77.8	81.8	83.7	74.0	72.5	78.3	**87.0** ± 0.9
halfcheetah-medium-replay-v2	36.6	40.5	42.2	40.2	38.1	44.6	36.6	45.5	44.2	**47.8** ± 0.3
hopper-medium-replay-v2	18.1	37.2	96.8	93.6	97.5	60.9	82.7	95.0	94.7	**101.3** ± 0.6
walker2d-medium-replay-v2	26.0	27.0	61.2	49.8	49.5	81.8	66.6	77.2	73.9	**95.5** ± 1.5
halfcheetah-medium-expert-v2	55.2	42.8	79.8	53.3	93.4	90.7	86.8	91.6	86.7	**96.8** ± 0.3
hopper-medium-expert-v2	52.5	55.8	107.2	108.7	103.3	98.0	107.6	105.4	91.5	**111.1** ± 1.3
walker2d-medium-expert-v2	107.5	74.5	108.4	95.6	**113.0**	110.1	108.1	108.8	109.6	110.1 ± 0.3
Average	51.9	50.1	75.3	72.9	76.1	75.3	74.7	77.6	77.0	**88.0**

AntMaze Tasks	BC	AWAC	BCQ	BEAR	Onestep RL	TD3+BC	DT	CQL	IQL	Diffusion-QL
antmaze-umaze-v0	54.6	56.7	78.9	73.0	64.3	78.6	59.2	74.0	87.5	**93.4** ± 3.4
antmaze-umaze-diverse-v0	45.6	49.3	55.0	61.0	60.7	71.4	53.0	**84.0**	62.2	66.2 ± 8.6
antmaze-medium-play-v0	0.0	0.0	0.0	0.0	0.3	10.6	0.0	61.2	71.2	**76.6** ± 10.8
antmaze-medium-diverse-v0	0.0	0.7	0.0	8.0	0.0	3.0	0.0	53.7	70.0	**78.6** ± 10.3
antmaze-large-play-v0	0.0	0.0	6.7	0.0	0.0	0.2	0.0	15.8	39.6	**46.4** ± 8.3
antmaze-large-diverse-v0	0.0	1.0	2.2	0.0	0.0	0.0	0.0	14.9	47.5	**56.6** ± 7.6
Average	16.7	18.0	23.8	23.7	20.9	27.3	18.7	50.6	63.0	**69.6**

Adroit Tasks	BC	SAC	BCQ	BEAR	BRAC-p	BRAC-v	REM	CQL	IQL	Diffusion-QL
pen-human-v1	25.8	4.3	68.9	-1.0	8.1	0.6	5.4	35.2	71.5	**72.8** ± 9.6
pen-cloned-v1	38.3	-0.8	44.0	26.5	1.6	-2.5	-1.0	27.2	37.3	**57.3** ± 11.9
Average	32.1	1.8	56.5	12.8	4.9	-1.0	2.2	31.2	54.4	**65.1**

Kitchen Tasks	BC	SAC	BCQ	BEAR	BRAC-p	BRAC-v	AWR	CQL	IQL	Diffusion-QL
kitchen-complete-v0	33.8	15.0	8.1	0.0	0.0	0.0	0.0	43.8	62.5	**84.0** ± 7.4
kitchen-partial-v0	33.8	0.0	18.9	13.1	0.0	0.0	15.4	49.8	46.3	**60.5** ± 6.9
kitchen-mixed-v0	47.5	2.5	8.1	47.2	0.0	0.0	10.6	51.0	51.0	**62.6** ± 5.1
Average	38.4	5.8	11.7	20.1	0.0	0.0	8.7	48.2	53.3	**69.0**

Q&A