Online Decision Transformer

Qinqing Zheng, Amy Zhang, Aditya Grover

Introduction

Recent works such as Decision Transformer (DT, Chen et al. 2021) shows that offline RL problems can be casted as sequence modeling problems and solved by supervised learning methods.

The performance of offline RL however is bottlenecked by the dataset properties and often requires online finetuning for best results.

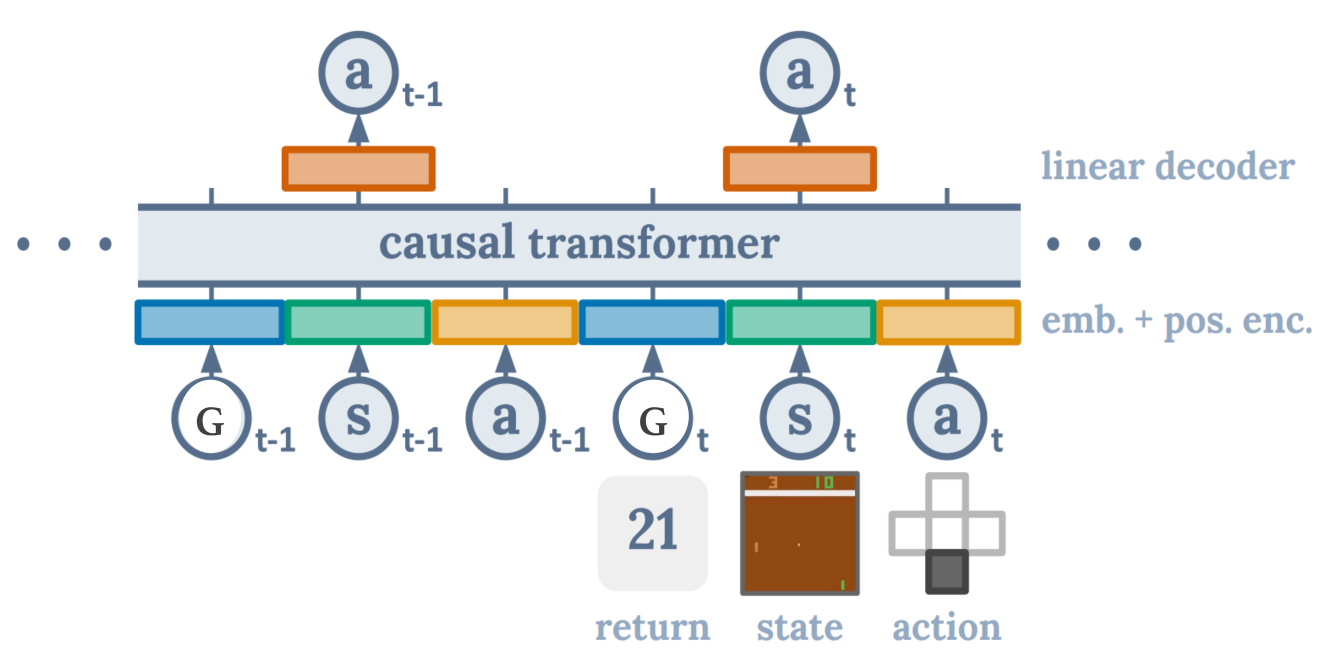
We propose Online Decision Transformers (ODT), an RL algorithm based on supervised sequence modeling that blends offline pretraining with online finetuning in a unified framework.

ODT enables stable online learning while retraining the simplicity of sequence modeling.

Base Model

Decision Transformer (Chen et al. 2021) models a trajectory τ as (RTG, state, action) sequences.

RTG (return-to-go) $g_t = \sum_{t'=t}^{|\tau|} r_{t'}$



DT architecture (Chen et al. 2021)

DT generates return-conditioned policies.

Rollout

1. Specify the desired return g_1 and an initial state s_1.
2. Generate a_1, execute it and then observe s_2 and r_1.
3. Compute $g_2 = g_1 - r_1$. Now we can generate a_2.
4. Repeat until the episode terminates.

Online Decision Transformer

Stochastic Policy

$$\pi_\theta(a_t|\mathbf{s}_{-K,t}, \mathbf{g}_{-K,t}) = \mathcal{N}(\mu_\theta(\mathbf{s}_{-K,t}, \mathbf{g}_{-K,t}), \Sigma_\theta(\mathbf{s}_{-K,t}, \mathbf{g}_{-K,t}))$$

Generate action based on recent K states and RTGs

Max-Ent Sequence Modeling

$$\min_\theta J(\theta) \quad \text{subject to} \quad H_\theta^{\mathcal{T}}[\mathbf{a}|\mathbf{s},\mathbf{g}] \geqslant \beta$$

- $J(\theta)$ negative log-likelihood of sequence data

$$J(\theta) = \frac{1}{K}\mathbb{E}_{(\mathbf{a},\mathbf{s},\mathbf{g})\sim\mathcal{T}}[-\log \pi_\theta(\mathbf{a}|\mathbf{s},\mathbf{g})]$$
$$= \frac{1}{K}\mathbb{E}_{(\mathbf{a},\mathbf{s},\mathbf{g})\sim\mathcal{T}}[-\textstyle\sum_{k=1}^{K} \log \pi_\theta(a_k|\mathbf{s}_{-K,k}, \mathbf{g}_{-K,k})]$$

simple supervised learning, no return optimization

- $H_\theta^{\mathcal{T}}[\mathbf{a}|\mathbf{s},\mathbf{g}]$ sequence-level policy entropy

$$H_\theta^{\mathcal{T}}[\mathbf{a}|\mathbf{s},\mathbf{g}] = \frac{1}{K}\mathbb{E}_{(\mathbf{s},\mathbf{g})\sim\mathcal{T}}\left[H[\pi_\theta(\mathbf{a}|\mathbf{s},\mathbf{g})]\right]$$
$$= \frac{1}{K}\mathbb{E}_{(\mathbf{s},\mathbf{g})\sim\mathcal{T}}\left[\textstyle\sum_{k=1}^{K} H[\pi_\theta(a_k|\mathbf{s}_{-K,k}, \mathbf{g}_{-K,k})]\right]$$

- β -dim(action)

Offline Pretraining + Online Finetuning

Algorithm 1: Online Decision Transformer

Input: offline data $\mathcal{T}_{\text{offline}}$, rounds R, exploration RTG g_{online}, buffer size N, gradient iterations I, pretrained policy π_θ
Intialization: Replay buffer $\mathcal{T}_{\text{replay}} \leftarrow$ top N trajectories in $\mathcal{T}_{\text{offline}}$.
for *round* $= 1, \dots, R$ **do**
  // use randomly sampled actions
  Trajectory $\tau \leftarrow$ Rollout using $\mathcal{M}$ and $\pi_\theta(\cdot|\mathbf{s}, \mathbf{g}(g_{\text{online}}))$.
  $\mathcal{T}_{\text{replay}} \leftarrow \{\mathcal{T}_{\text{replay}} \setminus \{\text{the oldest trajectory}\}\} \bigcup \{\tau\}$.
  $\pi_\theta \leftarrow$ Finetune ODT on $\mathcal{T}_{\text{replay}}$ for I iterations via Algorithm 2.

Algorithm 2: ODT Training

Input: model parameters θ, replay buffer $\mathcal{T}_{\text{replay}}$, training iterations I, context length K, batch size B
Compute the trajectory sampling probability $p(\tau) = |\tau| / \sum_{\tau\in\mathcal{T}} |\tau|$.
for $t = 1, \dots, I$ **do**
  Sample B trajectories out of $\mathcal{T}_{\text{replay}}$ according to p.
  for *each sampled trajectory* τ **do**
    // Hindsight Return Relabeling
    $\mathbf{g} \leftarrow$ the RTG sequence computed by the true rewards: $\mathbf{g}_t = \sum_{j=t}^{|\tau|} r_j,\ 1 \leqslant t \leqslant |\tau|$.
    $(\mathbf{a}, \mathbf{s}, \mathbf{g}) \leftarrow$ a length K sub-trajectory uniformly sampled from τ.
  $\theta \leftarrow$ one gradient update using the sampled $\{(\mathbf{a}, \mathbf{s}, \mathbf{g})\}$s.

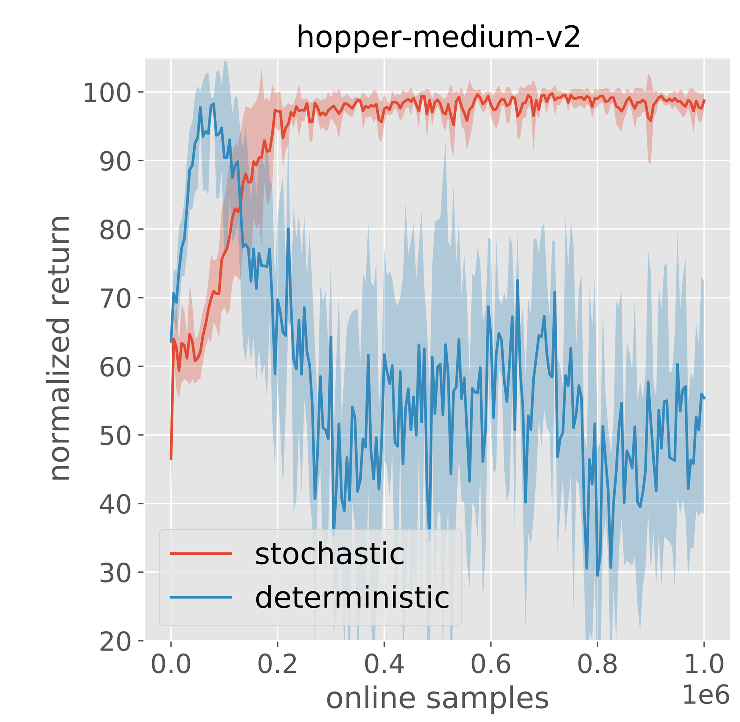
hindsight return relabeling - use observed return instead of target return

Benchmark Comparison

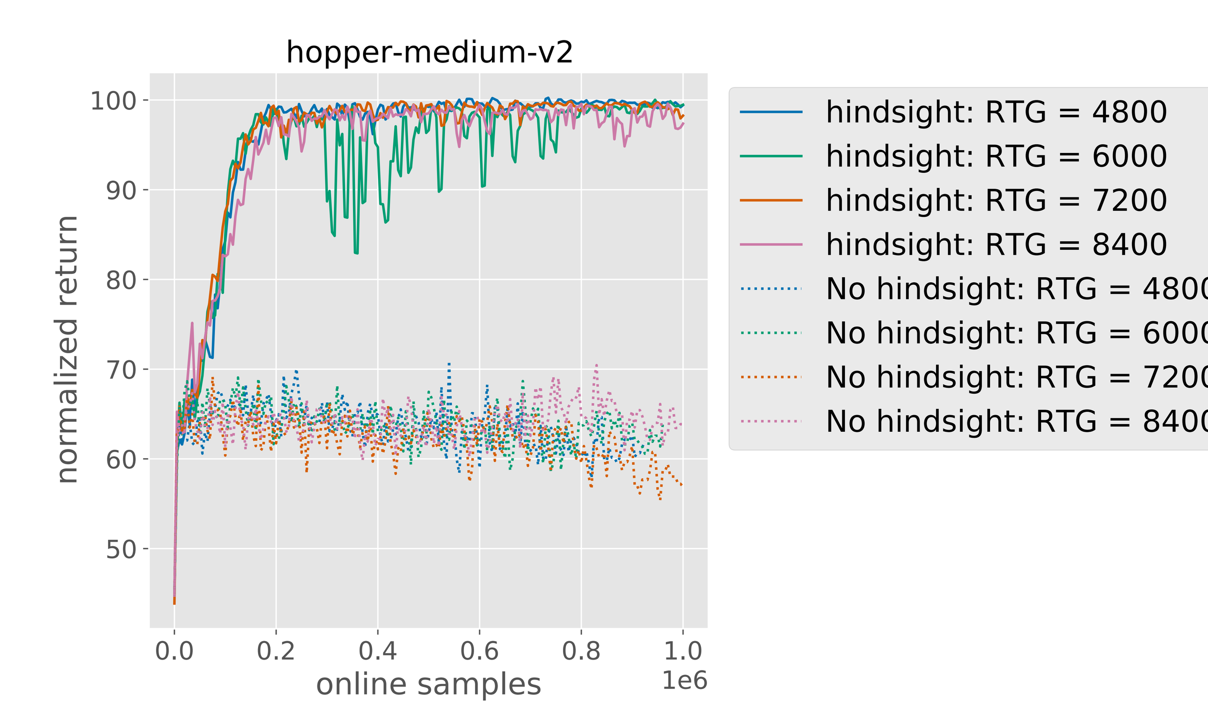
dataset	ODT (offline)	ODT (0.2m)	δ_{ODT}	IQL (offline)	IQL (0.2m)	δ_{IQL}
hopper-medium	66.95 ± 3.26	$\mathbf{97.54 \pm 2.10}$	**30.59**	63.81 ± 9.15	66.79 ± 4.07	2.98
hopper-medium-replay	86.64 ± 5.41	88.89 ± 6.33	2.25	92.13 ± 10.43	$\mathbf{96.23 \pm 4.35}$	**4.10**
walker2d-medium	72.19 ± 6.49	76.79 ± 2.30	**4.60**	79.89 ± 3.06	$\mathbf{80.33 \pm 2.33}$	0.44
walker2d-medium-replay	68.92 ± 4.79	$\mathbf{76.86 \pm 4.04}$	**7.94**	73.67 ± 6.37	70.55 ± 5.81	−3.12
halfcheetah-medium	42.72 ± 0.46	42.16 ± 1.48	−0.56	47.37 ± 0.29	47.41 ± 0.15	**0.04**
halfcheetah-medium-replay	39.99 ± 0.68	40.42 ± 1.61	**0.43**	44.10 ± 1.14	44.14 ± 0.3	0.04
ant-medium	91.33 ± 4.13	90.79 ± 5.80	−0.54	99.92 ± 5.86	$\mathbf{100.85 \pm 2.02}$	**0.93**
ant-medium-replay	86.56 ± 3.26	$\mathbf{91.57 \pm 2.73}$	**5.01**	91.21 ± 7.27	91.36 ± 1.47	0.15
sum		**605.02**	**49.72**		597.66	5.56
antmaze-umaze	53.10 ± 4.21	$\mathbf{88.5 \pm 5.88}$	**35.4**	87.1 ± 2.81	$\mathbf{89.5 \pm 5.43}$	2.4
antmaze-umaze-diverse	50.20 ± 6.69	$\mathbf{56.00 \pm 5.69}$	**7.99**	64.4 ± 8.95	$\mathbf{56.8 \pm 6.42}$	−7.6
sum		**144.5**	**43.39**		**146.3**	−5.2

Baseline: Implicit Q-Learning (IQL, Kostrikov 2021)
Absolute performance: ODT is comparable
Finetuning Gain: ODT is much better

Ablation Study



Stochasticity is important to enable stable performance improvement in online training



Hindsight return relabeling is critical for correcting bias in collected data