Pre-training with Synthetic Data Helps Offline Reinforcement Learning

Zecheng Wang[1], Che Wang[2,4], Zixuan Dong[3,4], and Keith Ross[1]
[1]New York University Abu Dhabi, [2]New York University Shanghai
[3]SFSC of AI and DL, NYU Shanghai, [4]New York University

Overview

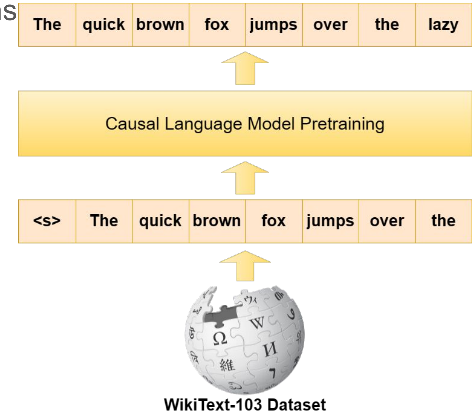
- Can language pre-training lead to **special** performance gains in offline RL?

Overview

- Can language pre-training lead to **special** performance gains in offline RL?
    - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.

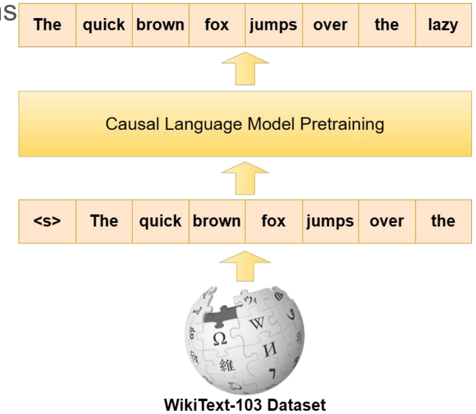
Overview

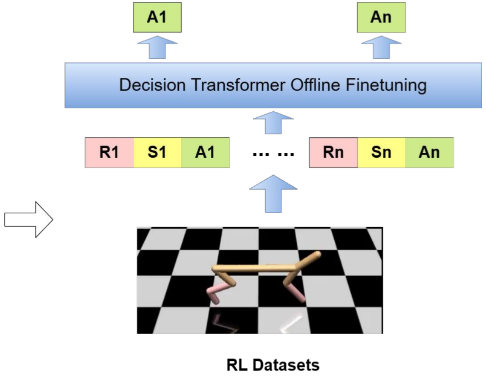
- Can language pre-training lead to **special** performance gains in offline RL?
  - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.



Overview

- Can language pre-training lead to **special** performance gains in offline RL?
    - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.





Overview

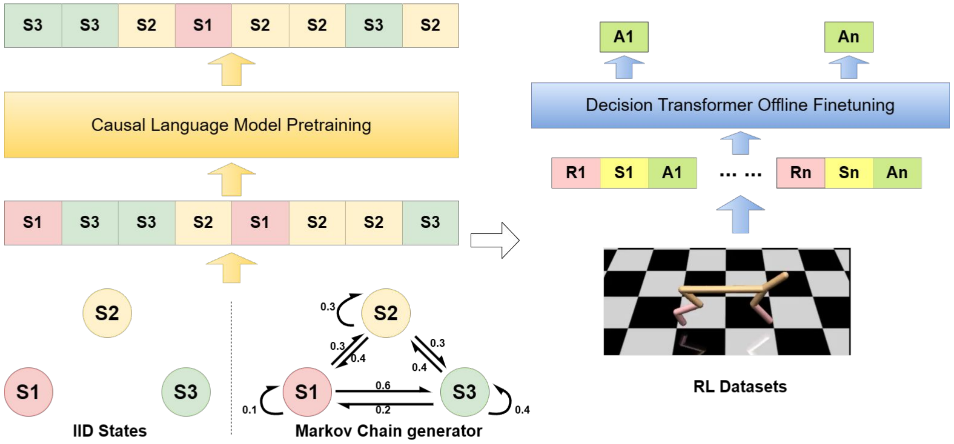
- Can language pre-training lead to **special** performance gains in offline RL?
    - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.
- How about simpler data without involving language?

Overview

- Can language pre-training lead to **special** performance gains in offline RL?
    - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.
- How about simpler data without involving language?
- We show that pre-training with simple synthetic data can provide even better performance.

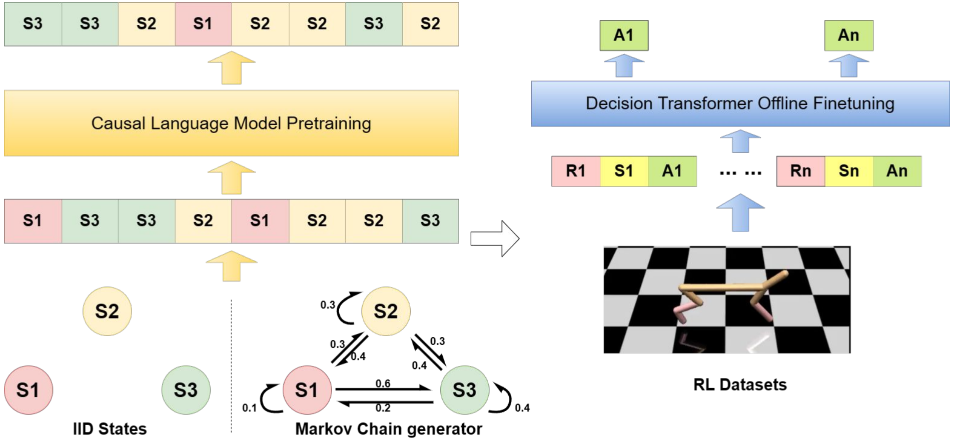
Overview

- Can language pre-training lead to **special** performance gains in offline RL?
  - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.
- How about simpler data without involving language?
- We show that pre-training with simple synthetic data can provide even better performance.
  - Randomized IID data/Markov Chain data;



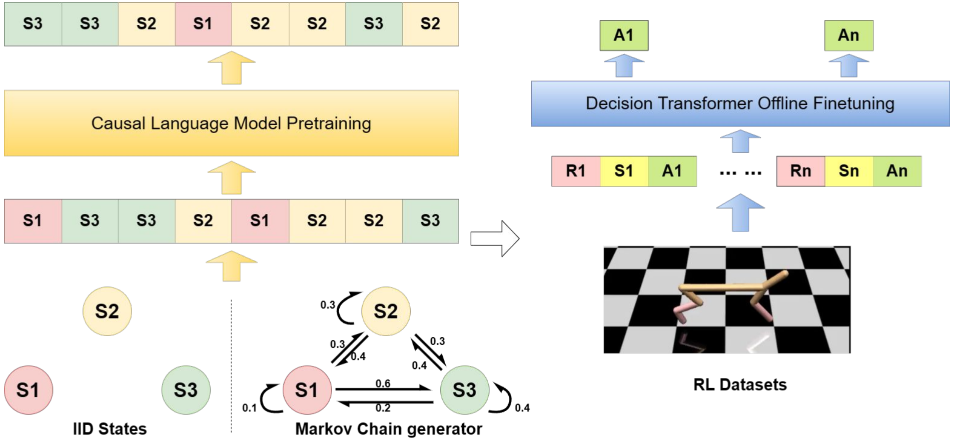
Overview

- Can language pre-training lead to **special** performance gains in offline RL?
    - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.
- How about simpler data without involving language?
- We show that pre-training with simple synthetic data can provide even better performance.
    - Randomized IID data/Markov Chain data;
    - Pre-training with smaller number of steps;



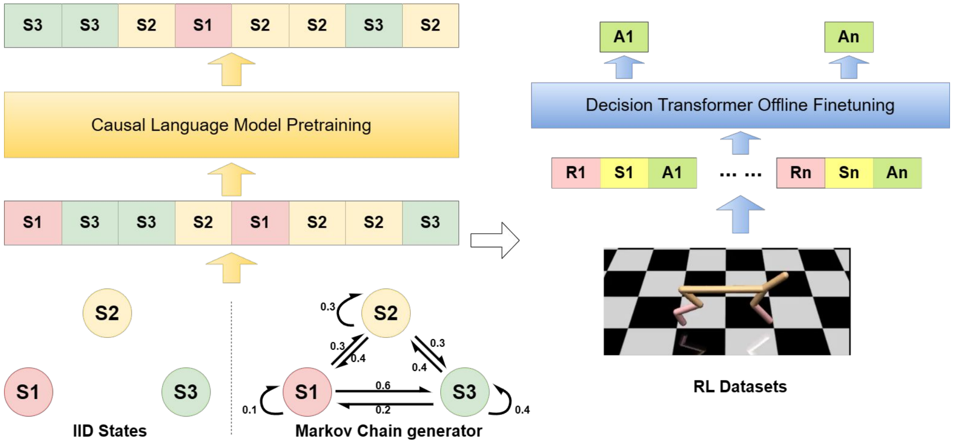
Overview

- Can language pre-training lead to **special** performance gains in offline RL?
  - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.
- How about simpler data without involving language?
- We show that pre-training with simple synthetic data can provide even better performance.
  - Randomized IID data/Markov Chain data;
  - Pre-training with smaller number of steps;
  - Applicable to both Transformer and MLP architectures.



Overview

- Can language pre-training lead to **special** performance gains in offline RL?
  - “Can Wikipedia Help Offline Reinforcement Learning?” (Reid et al., 2022) claimed that **it can**.
- How about simpler data without involving language?
- We show that pre-training with simple synthetic data can provide even better performance.
  - Randomized IID data/Markov Chain data;
  - Pre-training with smaller number of steps;
  - Applicable to both Transformer and MLP architectures.
- We therefore conclude that:
  - Language is **not essential** for improved performance;
  - Synthetic pre-training is **easy and effective** in improving offline RL.



Background: Pre-training

- Next State Prediction

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).
  - Given the previous states, predict the next state (Markov Chain with **discrete integer states**).
  - $\mathcal{L}(x_0, x_1, \ldots, x_T; \theta) = -\log P_\theta(x_0, x_1, \ldots, x_T) = -\Sigma_{t=1}^{T} \log P_\theta(x_t | x_0, x_1, \ldots, x_{t-1})$.

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).
  - Given the previous states, predict the next state (Markov Chain with **discrete integer states**).
  - $\mathcal{L}(x_0, x_1, \ldots, x_T; \theta) = -\log P_\theta(x_0, x_1, \ldots, x_T) = -\Sigma_{t=1}^{T} \log P_\theta(x_t | x_0, x_1, \ldots, x_{t-1})$.
  - Ideal for **Decision Transformer**

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).
  - Given the previous states, predict the next state (Markov Chain with **discrete integer states**).
  - $\mathcal{L}(x_0, x_1, \ldots, x_T; \theta) = -\log P_\theta(x_0, x_1, \ldots, x_T) = -\Sigma_{t=1}^{T} \log P_\theta(x_t | x_0, x_1, \ldots, x_{t-1}).$
  - Ideal for **Decision Transformer**
- Forward Dynamics Prediction (Janner et al., 2019, He et al., 2022)

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).
  - Given the previous states, predict the next state (Markov Chain with **discrete integer states**).
  - $\mathcal{L}(x_0, x_1, \ldots, x_T; \theta) = -\log P_\theta(x_0, x_1, \ldots, x_T) = -\Sigma_{t=1}^{T} \log P_\theta(x_t | x_0, x_1, \ldots, x_{t-1})$.
  - Ideal for **Decision Transformer**
- Forward Dynamics Prediction (Janner et al., 2019, He et al., 2022)
  - Predicting the next state s' given the current state s and action a (Markov Decision Process data).

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).
  - Given the previous states, predict the next state (Markov Chain with **discrete integer states**).
  - $\mathcal{L}(x_0, x_1, \ldots, x_T; \theta) = -\log P_\theta(x_0, x_1, \ldots, x_T) = -\Sigma_{t=1}^{T} \log P_\theta(x_t | x_0, x_1, \ldots, x_{t-1}).$
  - Ideal for **Decision Transformer**
- Forward Dynamics Prediction (Janner et al., 2019, He et al., 2022)
  - Predicting the next state s' given the current state s and action a (Markov Decision Process data).
  - Minimize the MSE loss between s' and the predicted next state ŝ': $(s' - \hat{s}')^2$.

Background: Pre-training

- Next State Prediction
  - Similar to autoregressive language modeling (Brown et al., 2020) (tokens as states).
  - Given the previous states, predict the next state (Markov Chain with **discrete integer states**).
  - $\mathcal{L}(x_0, x_1, \ldots, x_T; \theta) = -\log P_\theta(x_0, x_1, \ldots, x_T) = -\Sigma_{t=1}^{T} \log P_\theta(x_t | x_0, x_1, \ldots, x_{t-1})$.
  - Ideal for **Decision Transformer**
- Forward Dynamics Prediction (Janner et al., 2019, He et al., 2022)
  - Predicting the next state s' given the current state s and action a (Markov Decision Process data).
  - Minimize the MSE loss between s' and the predicted next state ŝ': $(s' - \hat{s}')^2$.
  - Ideal for **CQL**

ICLR International Conference On Learning Representations

NEW YORK UNIVERSITY جامعة نيويورك أبوظبي NYU ABU DHABI NYU 上海 SHANGHAI 纽约大学

Synthetic Data Generation: Markov Chain Generator

- Markov Chain Generator Setup
  - Define State Space (number of states **S**)
  - Initial State Distribution $\mathbf{P_0}$ over the state space
  - Number of steps to condition **N**
  - Transitional Distribution Matrices $\mathbf{P_N}$ for 1…N
- To generate distributions:
  - For each previous state(s), draw **S** IID values $\mathbf{z_{1...S}}$
  - Apply softmax given temperature **T:** $\frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$
  - **Distributions are fixed when generating data**
- Hyper-parameters
  - **N**-step conditioning
  - **S** number of states
  - **T** temperature

Synthetic Data Generation: Markov Chain Generator

P_0

S1	S2	S3
0.33	0.33	0.33

P_1

	S1	S2	S3
S1	0.33	0.33	0.33
S2	0.33	0.33	0.33
S3	0.33	0.33	0.33

Randomized IID

Synthetic Data Generation: Markov Chain Generator

P_0

S1	S2	S3
0.33	0.33	0.33

P_1

	S1	S2	S3
S1	0.33	0.33	0.33
S2	0.33	0.33	0.33
S3	0.33	0.33	0.33

Randomized IID

P_0

S1	S2	S3
0.1	0.4	0.5

P_1

	S1	S2	S3
S1	0.1	0.3	0.6
S2	0.4	0.3	0.3
S3	0.2	0.4	0.4

1-step MC

Synthetic Data Generation: Markov Chain Generator

Randomized IID

P_0

S1	S2	S3
0.33	0.33	0.33

P_1

	S1	S2	S3
S1	0.33	0.33	0.33
S2	0.33	0.33	0.33
S3	0.33	0.33	0.33

1-step MC

P_0

S1	S2	S3
0.1	0.4	0.5

P_1

	S1	S2	S3
S1	0.1	0.3	0.6
S2	0.4	0.3	0.3
S3	0.2	0.4	0.4

N-step MC

P_0

S1	S2	S3
0.1	0.4	0.5

$P_{1...N}$

	S1	S2	S3
S1,...,S1	0.2	0.3	0.5
...	...	...	...
S3,...,S3	0.7	0.1	0.2

xN

Synthetic Data Generation: Markov Decision Data Generator

- Similar to 1-step MC data, generate MDP data in the following way:
    - Apart from a discrete state space, define a discrete action space A
    - Define a policy distribution π over A given a state
    - Define Transition Matrices over the state space given previous state s **and action a**
    - To obtain distributions, draw IID values and pass through softmax as before
    - To generate states/actions, map the discrete states/actions to multi-dimensional vectors (dimensions should agree with downstream task)

Experiments: Decision Transformer

- Benchmark: D4RL datasets
  - Four MuJoCo environments (HalfCheetah, Hopper, Walker, Ant)
  - Three datasets for each environment (Medium, Medium-Expert, Medium-Replay)

Experiments: Decision Transformer

- Benchmark: D4RL datasets
  - Four MuJoCo environments (HalfCheetah, Hopper, Walker, Ant)
  - Three datasets for each environment (Medium, Medium-Expert, Medium-Replay)
- Training Hyper-parameters
  - We follow the settings from DT (Chen et al., 2021) and DT+Wiki (Reid et al., 2022)
  - **Pre-training:** Instead of 80K steps of language pre-training as in DT+Wiki, we pre-train with synthetic data with only 20K steps
  - **Evaluation:** Evaluating every 5K steps, we average returns over the **last 20K** steps out of a total of 100K fine-tuning steps
  - We run each experiment over **20** seeds

Experiments: Decision Transformer

- Benchmark: D4RL datasets
  - Four MuJoCo environments (HalfCheetah, Hopper, Walker, Ant)
  - Three datasets for each environment (Medium, Medium-Expert, Medium-Replay)
- Training Hyper-parameters
  - We follow the settings from DT (Chen et al., 2021) and DT+Wiki (Reid et al., 2022)
  - **Pre-training:** Instead of 80K steps of language pre-training as in DT+Wiki, we pre-train with synthetic data with only 20K steps
  - **Evaluation:** Evaluating every 5K steps, we average returns over the **last 20K** steps out of a total of 100K fine-tuning steps
  - We run each experiment over **20** seeds
- MC Data
  - By default, data are generated with 100 states, 1-step MC with a temperature of 1
  - The size of synthetic data are made to be similar to Wikitext-103 (Merity et al., 2016)

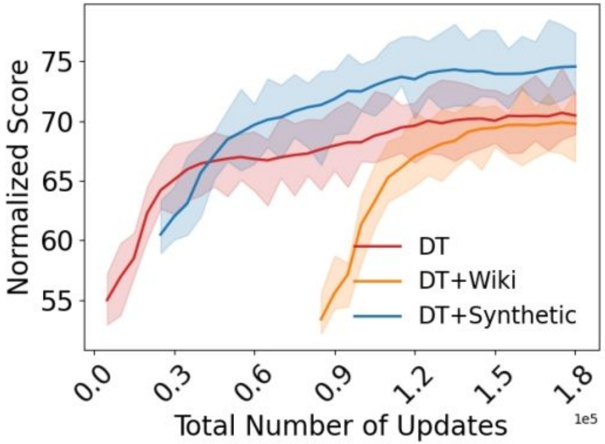
Experiments: Decision Transformer

- Main Results
  - The default synthetic data setting gives most consistent results
  - Our approach (DT+Synthetic) outperforms DT by **10%**
  - DT+Synthetic outperforms DT+Wiki by **5%**
  - Evidence that complex token dependencies and semantic meaning of the language **is not essential**

Average Last Four	DT	DT+Wiki	DT+Synthetic
halfcheetah-medium-expert	44.9 ± 3.4	43.9 ± 2.7	$\mathbf{49.5} \pm 9.9$
hopper-medium-expert	81.0 ± 11.8	94.0 ± 8.9	$\mathbf{99.6} \pm 6.5$
walker2d-medium-expert	105.0 ± 3.5	102.7 ± 6.4	$\mathbf{107.4} \pm 0.8$
ant-medium-expert	107.0 ± 8.7	113.9 ± 10.5	$\mathbf{117.9} \pm 8.7$
halfcheetah-medium-replay	37.5 ± 1.3	$\mathbf{39.1} \pm 1.6$	$\mathbf{39.3} \pm 1.1$
hopper-medium-replay	46.7 ± 10.6	51.4 ± 13.6	$\mathbf{61.8} \pm 13.9$
walker2d-medium-replay	49.2 ± 10.1	55.2 ± 7.7	$\mathbf{56.8} \pm 5.1$
ant-medium-replay	80.9 ± 3.9	78.1 ± 5.3	$\mathbf{88.4} \pm 2.7$
halfcheetah-medium	$\mathbf{42.4} \pm 0.5$	$\mathbf{42.6} \pm 0.2$	$\mathbf{42.5} \pm 0.2$
hopper-medium	58.2 ± 3.2	58.4 ± 3.3	$\mathbf{60.2} \pm 2.1$
walker2d-medium	70.4 ± 2.9	$\mathbf{70.8} \pm 3.0$	$\mathbf{71.5} \pm 4.1$
ant-medium	$\mathbf{89.0} \pm 4.7$	$\mathbf{88.5} \pm 4.2$	87.8 ± 4.2
Average over datasets	67.7 ± 5.4	69.9 ± 5.6	$\mathbf{73.6} \pm 4.9$

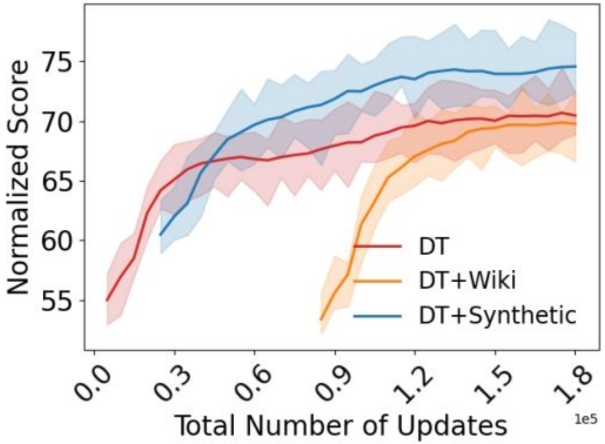
Experiments: Decision Transformer

- Computational Efficiency
  - **Pre-training:** DT+Synthetic consumes **3%** of the computation resources (Time x GPUs) needed for DT+Wiki
  - **Fine-tuning:** DT+Synthetic takes **67%** of the computation time needed for DT+Wiki under the same hardware setting (due to no auxiliary loss)



Experiments: Decision Transformer

- Computational Efficiency
  - **Pre-training:** DT+Synthetic consumes **3%** of the computation resources (Time x GPUs) needed for DT+Wiki
  - **Fine-tuning:** DT+Synthetic takes **67%** of the computation time needed for DT+Wiki under the same hardware setting (due to no auxiliary loss)



Computation Time	DT	DT+Wiki	DT+Synthetic
halfcheetah-medium-expert	2 hrs 27 mins	3 hrs 50 mins	2 hrs 32 mins
hopper-medium-expert	1 hrs 55 mins	3 hrs 25 mins	2hrs 11 mins
walker2d-medium-expert	2 hrs 17 mins	3 hrs 45 mins	2 hrs 18 mins
ant-medium-expert	2 hrs 8 mins	3 hrs 52 mins	2 hrs 46 mins
Average over datasets	2 hrs 12 mins	3 hrs 43 mins	2 hrs 27 mins

Experiments: Decision Transformer

- Ablations
  - Longer token dependencies **does not** give better performance

Average Last Four	DT	1-MC	2-MC	5-MC
halfcheetah-medium-expert	44.9 ± 3.4	**49.5** ± 9.9	44.3 ± 4.0	43.8 ± 3.0
hopper-medium-expert	81.0 ± 11.8	**99.6** ± 6.5	**99.1** ± 6.5	98.2 ± 5.7
walker2d-medium-expert	105.0 ± 3.5	**107.4** ± 0.8	105.7 ± 3.1	105.9 ± 3.1
ant-medium-expert	107.0 ± 8.7	117.9 ± 8.7	**122.2** ± 5.3	108.9 ± 11.7
halfcheetah-medium-replay	37.5 ± 1.3	**39.3** ± 1.1	**39.5** ± 1.3	**39.4** ± 0.9
hopper-medium-replay	46.7 ± 10.6	**61.8** ± 13.9	59.8 ± 11.0	60.1 ± 11.4
walker2d-medium-replay	49.2 ± 10.1	56.8 ± 5.1	**59.3** ± 3.9	**58.8** ± 5.8
ant-medium-replay	80.9 ± 3.9	**88.4** ± 2.7	86.9 ± 4.0	86.1 ± 4.4
halfcheetah-medium	**42.4** ± 0.5	**42.5** ± 0.2	**42.6** ± 0.3	**42.5** ± 0.3
hopper-medium	58.2 ± 3.2	**60.2** ± 2.1	59.3 ± 3.3	59.6 ± 2.8
walker2d-medium	70.4 ± 2.9	**71.5** ± 4.1	70.7 ± 4.2	70.1 ± 4.0
ant-medium	**89.0** ± 4.7	87.8 ± 4.2	87.0 ± 3.7	**88.6** ± 4.1
Average over datasets	67.7 ± 5.4	**73.6** ± 4.9	**73.0** ± 4.2	71.8 ± 4.8

Experiments: Decision Transformer

- Ablations
  - Longer token dependencies **does not** give better performance
  - A larger state space (similar to LM vocabularies) **does not** give better performance

Average Last Four	DT	S10	S100	S1000	S10000	S100000
halfcheetah-medium-expert	44.9 ± 3.4	43.4 ± 2.6	**49.5** ± 9.9	45.4 ± 4.5	44.0 ± 2.2	43.6 ± 2.7
hopper-medium-expert	81.0 ± 11.8	98.8 ± 8.4	99.6 ± 6.5	**102.2** ± 5.7	99.8 ± 6.2	99.4 ± 6.7
walker2d-medium-expert	105.0 ± 3.5	105.4 ± 4.1	**107.4** ± 0.8	**107.1** ± 1.9	105.9 ± 3.1	103.9 ± 5.0
ant-medium-expert	107.0 ± 8.7	114.6 ± 9.7	117.9 ± 8.7	118.7 ± 6.7	116.0 ± 10.5	**123.2** ± 6.3
halfcheetah-medium-replay	37.5 ± 1.3	**40.0** ± 0.9	39.3 ± 1.1	**40.0** ± 0.8	**39.6** ± 1.2	**39.9** ± 0.9
hopper-medium-replay	46.7 ± 10.6	58.6 ± 13.2	61.8 ± 13.9	**65.0** ± 10.8	62.0 ± 9.6	53.3 ± 12.6
walker2d-medium-replay	49.2 ± 10.1	52.6 ± 10.1	56.8 ± 5.1	59.5 ± 6.2	**60.1** ± 5.6	58.8 ± 8.5
ant-medium-replay	80.9 ± 3.9	87.1 ± 4.4	**88.4** ± 2.7	**87.8** ± 3.3	84.5 ± 4.8	86.8 ± 3.6
halfcheetah-medium	**42.4** ± 0.5	**42.5** ± 0.4	**42.5** ± 0.2	**42.4** ± 0.3	**42.5** ± 0.3	**42.4** ± 0.4
hopper-medium	58.2 ± 3.2	59.6 ± 3.0	**60.2** ± 2.1	**60.4** ± 2.7	58.7 ± 3.8	57.3 ± 3.3
walker2d-medium	70.4 ± 2.9	71.5 ± 3.8	71.5 ± 4.1	**72.8** ± 2.2	**72.4** ± 3.6	**72.4** ± 2.7
ant-medium	**89.0** ± 4.7	**88.9** ± 3.7	87.8 ± 4.2	87.1 ± 2.8	**88.8** ± 4.2	**88.3** ± 3.2
Average over datasets	67.7 ± 5.4	71.9 ± 5.3	**73.6** ± 4.9	**74.0** ± 4.0	72.9 ± 4.6	72.4 ± 4.7

Experiments: Decision Transformer

- Ablations
  - Longer token dependencies **does not** give better performance
  - A larger state space (similar to LM vocabularies) **does not** give better performance
  - Even randomized IID (infinite temperature) data provides better performance than DT+Wiki

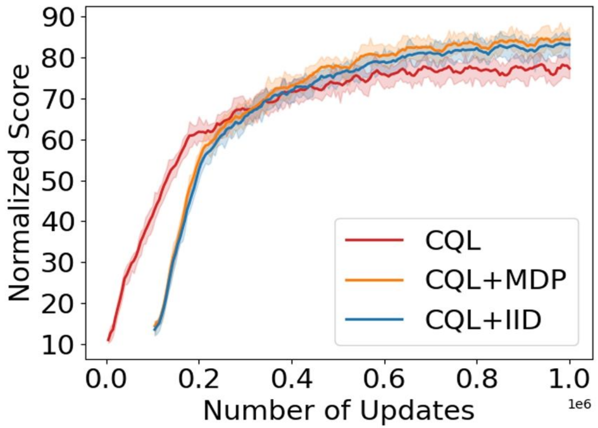
Average Last Four	DT	τ=0.01	τ=0.1	τ=1	τ=10	τ=100	IID uniform
halfcheetah-medium-expert	44.9 ± 3.4	46.6 ± 5.4	**52.6** ± 11.9	49.5 ± 9.9	43.3 ± 3.2	44.2 ± 3.3	44.5 ± 4.0
hopper-medium-expert	81.0 ± 11.8	95.4 ± 8.1	95.2 ± 9.2	**99.6** ± 6.5	**99.9** ± 6.3	98.7 ± 5.5	98.7 ± 7.1
walker2d-medium-expert	105.0 ± 3.5	**106.4** ± 2.6	**106.6** ± 2.9	**107.4** ± 0.8	106.3 ± 3.6	105.1 ± 4.3	103.2 ± 4.2
ant-medium-expert	107.0 ± 8.7	114.9 ± 6.9	**121.7** ± 5.5	117.9 ± 8.7	118.6 ± 10.1	108.2 ± 9.6	105.8 ± 11.1
halfcheetah-medium-replay	37.5 ± 1.3	39.5 ± 1.1	**40.2** ± 0.9	39.3 ± 1.1	39.7 ± 0.8	**40.1** ± 0.5	39.3 ± 0.9
hopper-medium-replay	46.7 ± 10.6	52.5 ± 12.0	52.8 ± 14.4	**61.8** ± 13.9	60.2 ± 9.4	60.8 ± 9.3	**61.6** ± 10.8
walker2d-medium-replay	49.2 ± 10.1	**57.3** ± 6.6	**57.0** ± 6.6	**56.8** ± 5.1	55.1 ± 8.6	56.7 ± 6.3	**57.2** ± 5.2
ant-medium-replay	80.9 ± 3.9	86.7 ± 3.5	**88.2** ± 3.7	**88.4** ± 2.7	85.8 ± 3.6	87.2 ± 4.6	86.1 ± 3.6
halfcheetah-medium	**42.4** ± 0.5	**42.4** ± 0.3	**42.5** ± 0.2	**42.5** ± 0.2	**42.5** ± 0.3	**42.6** ± 0.3	**42.6** ± 0.2
hopper-medium	58.2 ± 3.2	59.1 ± 3.4	59.4 ± 3.5	**60.2** ± 2.1	57.9 ± 3.1	59.4 ± 3.7	59.1 ± 3.2
walker2d-medium	70.4 ± 2.9	**71.7** ± 2.8	**71.5** ± 3.1	**71.5** ± 4.1	70.7 ± 3.6	**71.7** ± 4.1	69.1 ± 5.4
ant-medium	**89.0** ± 4.7	88.0 ± 3.5	**89.2** ± 3.0	87.8 ± 4.2	**88.4** ± 4.0	**88.4** ± 4.6	88.1 ± 4.9
Average over datasets	67.7 ± 5.4	71.7 ± 4.7	**73.1** ± 5.4	**73.6** ± 4.9	72.4 ± 4.7	71.9 ± 4.7	71.3 ± 5.1

Experiments: Decision Transformer

- Ablations
  - Longer token dependencies **does not** give better performance
  - A larger state space (similar to LM vocabularies) **does not** give better performance
  - Even randomized IID (infinite temperature) data provides better performance than DT+Wiki
  - DT+Synthetic gives robust results over different number of states, MC steps, and temperature
  - Further evidence that complex token dependencies from language **is not essential**

Experiments: CQL

- Main results
  - **Pre-train** for 100K steps with MDP data
  - **Fine-tune** for 1M steps
  - S stands for number of states/actions
  - Temperature for all distributions are 1
  - Results consistent with DT
  - CQL pre-training only takes 5 mins with one GPU!



Average Last Four	CQL	S=10	S=100	S=1,000	S=10,000	S=100,000
halfcheetah-medium-expert	35.9 ± 5.2	52.9 ± 5.8	63.1 ± 7.2	**66.2** ± 7.3	**65.6** ± 9.1	63.7 ± 6.8
hopper-medium-expert	59.3 ± 21.4	**90.4** ± 15.5	**90.2** ± 13.2	88.1 ± 10.6	**89.8** ± 13.0	84.9 ± 20.2
walker2d-medium-expert	107.8 ± 3.8	**109.8** ± 0.3	**109.8** ± 0.3	**110.1** ± 0.4	**110.1** ± 0.4	**110.1** ± 0.3
ant-medium-expert	118.8 ± 5.2	124.0 ± 5.1	126.0 ± 5.4	**131.4** ± 4.1	128.4 ± 4.7	129.2 ± 4.3
halfcheetah-medium-replay	**46.6** ± 0.3	**46.5** ± 0.3	**46.8** ± 0.4	**46.5** ± 0.3	**46.6** ± 0.2	**46.5** ± 0.3
hopper-medium-replay	94.2 ± 2.2	96.3 ± 2.9	95.3 ± 3.2	96.9 ± 1.9	**98.0** ± 1.4	**97.1** ± 2.0
walker2d-medium-replay	80.0 ± 4.1	**83.9** ± 3.0	**83.9** ± 2.4	**83.8** ± 1.6	81.3 ± 3.4	82.9 ± 1.9
ant-medium-replay	96.7 ± 3.8	**101.7** ± 4.0	**102.0** ± 3.5	**102.3** ± 2.4	**101.9** ± 2.6	100.6 ± 3.8
halfcheetah-medium	**48.3** ± 0.2	**48.6** ± 0.2	**48.7** ± 0.2	**48.7** ± 0.2	**48.7** ± 0.2	**48.6** ± 0.2
hopper-medium	**68.2** ± 4.0	64.6 ± 2.6	66.9 ± 4.1	66.2 ± 2.8	65.5 ± 3.3	66.9 ± 3.3
walker2d-medium	82.1 ± 1.8	82.8 ± 2.3	**83.4** ± 1.1	**83.7** ± 0.6	**83.2** ± 1.1	**83.5** ± 1.3
ant-medium	98.7 ± 4.0	102.4 ± 3.6	**103.2** ± 3.3	**103.3** ± 3.8	**103.4** ± 2.9	101.2 ± 3.4
Average over datasets	78.0 ± 4.7	83.7 ± 3.8	**84.9** ± 3.7	**85.6** ± 3.0	**85.2** ± 3.5	84.6 ± 4.0

Conclusion

- We propose a **simple yet effective** synthetic pre-training scheme for both DT and CQL
- A **smaller** state space/**simpler** token dependency challenges the previous view that language pre-training can provide unique benefits for offline RL
- Our results are **robust** over various hyper-parameters (state/action space **size**, **peakedness** of distributions, history **dependence**)
- Our approach is **extremely efficient** (DT+Synthetic uses 3% the resources needed for DT+Wiki, faster fine-tuning; CQL pre-training only takes 5 mins!)

Thank you!

- Reference

  - Reid, M., Yamada, Y., & Gu, S. S. (2022). Can wikipedia help offline reinforcement learning?. arXiv preprint arXiv:2201.12122.
  - Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877-1901.
  - Janner, M., Fu, J., Zhang, M., & Levine, S. (2019). When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, *32*.
  - He, T., Zhang, Y., Ren, K., Liu, M., Wang, C., Zhang, W., ... & Li, D. (2022). Reinforcement learning with automated auxiliary loss search. *Advances in Neural Information Processing Systems*, *35*, 1820-1834.
  - Fu, J., Kumar, A., Nachum, O., Tucker, G., & Levine, S. (2020). D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.
  - Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., ... & Mordatch, I. (2021). Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, *34*, 15084-15097.
  - Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, *33*, 1179-1191.
  - Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018, July). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning* (pp. 1861-1870). PMLR.
  - Merity, S., Xiong, C., Bradbury, J., & Socher, R. (2016). Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.