TEXAS A&M UNIVERSITY
Engineering

Decision Transformer: Reinforcement Learning via Sequence Modeling (Atari Game)

Team Members:
Jianglong Yu, CS, Master
Xin Yi, CS, Master
Michael Xu, ECEN, Master
Lipai Huang, CVEN, PhD

GitHub Link

Tasks Assign

TEXAS A&M UNIVERSITY
Engineering

Jianglong Yu	Lead Decision Transformer Reproduction & Model Tuning
Xin Yi	Model Improvement
Michael Xu	Decision Transformer Reproduction
Lipai Huang	Pre-Trained Data Collection & Code Testing

Motivation

TEXAS A&M UNIVERSITY Engineering

- Power of Transformer [1]
- Traditional RL challenges
  - Complexity in Learning Algorithms, Temporal Credit Assignment, Reward Sparsity
- Why Decision Transformers [2]
  - Sequence Modeling Approach
  - Direct Learning from Trajectories
  - Handling Long Sequences

[1] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).
[2] Chen, Lili, et al. "Decision transformer: Reinforcement learning via sequence modeling." *Advances in neural information processing systems* 34 (2021): 15084-15097.

Introduction

- DT Architecture
    - Embeddings
    - Causal Self-Attention
- Atari Game
    - Pong
- DT to Atari Game
    - Strategy Optimization
    - Model Pruning

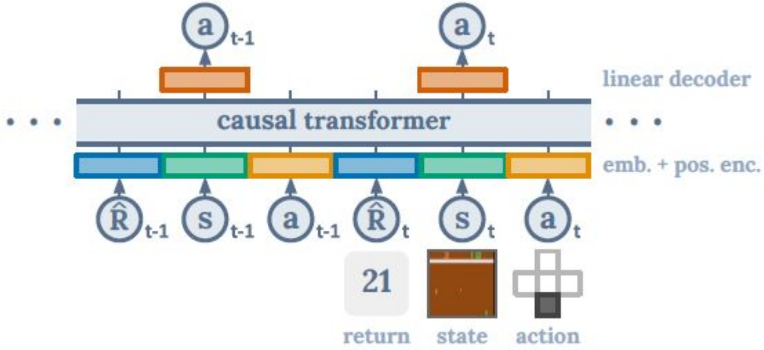


Figure 1: Decision Transformer architecture.

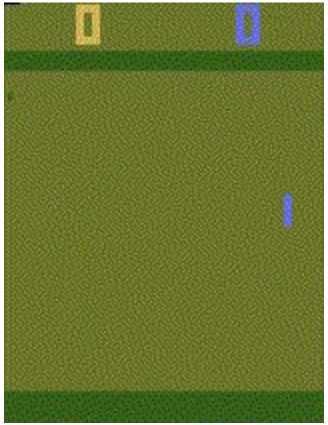


Figure 2: Atari Pong game Sample. URL: https://www.gymlibrary.dev/environments/atari/pong/

Markov Decision Process in Pong Game

Traditionally, an MDP is described by the tuple ***(S, A, P, R)***, which consists of states $s \in S$, actions $a \in A$, transition probability $P(s'|s,a)$, reward function $r = R(s,a)$.
The goal of learning is to learn a policy that maximize the $\mathbb{E}\left[\sum_{t=1}^{T} r_t\right]$.

- **Model-based RL**

  Predicts the next state and reward for each action taken in a given state.

  Eg, Dyna-Q, Monte Carlo tree search

- **Q-learning**

  Model-free RL, Value-based RL, the agent learns the value of each action for each possible state in the environment through the Q-function $Q(s,a)$

  Eg, DQN, Double DQN, Duel DQN

- **Policy Gradient Methods**

  Model-free RL, Policy-based RL, the agent directly learns a policy that dictates the probability of selecting an action in a given state by a policy function, $\pi\theta(a|s)$.

  Eg, REINFORCE, Actor-Critic Methods, Proximal Policy Optimization.

- **Decision Transformer**

  Different from traditional reinforcement learning. It employs Transformer to directly learn the actions that the agent should take using a sequence-to-sequence model, rather than focusing on learning strategies or value functions.

Pong Environment

- **Actions**
  Discrete action space.

Action	Behavior	Action	Behavior
0	No operation	3	Move left
1	Fire	4	Fire right
2	Move right	5	Fire left

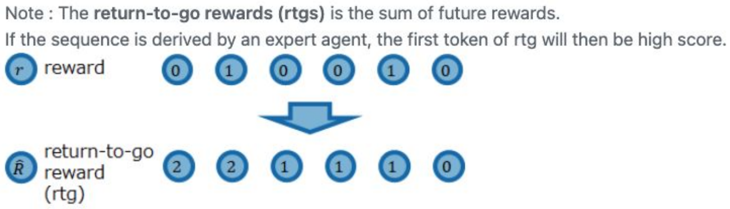
- **States**
  Fully observable, as the entire playing area is visible and can be completely accounted by the input image in grayscale.
  [[0 ... 0] ... [0 ... 0]], [[255 ... 255] ... [255 ... 255]], (84, 84), uin8

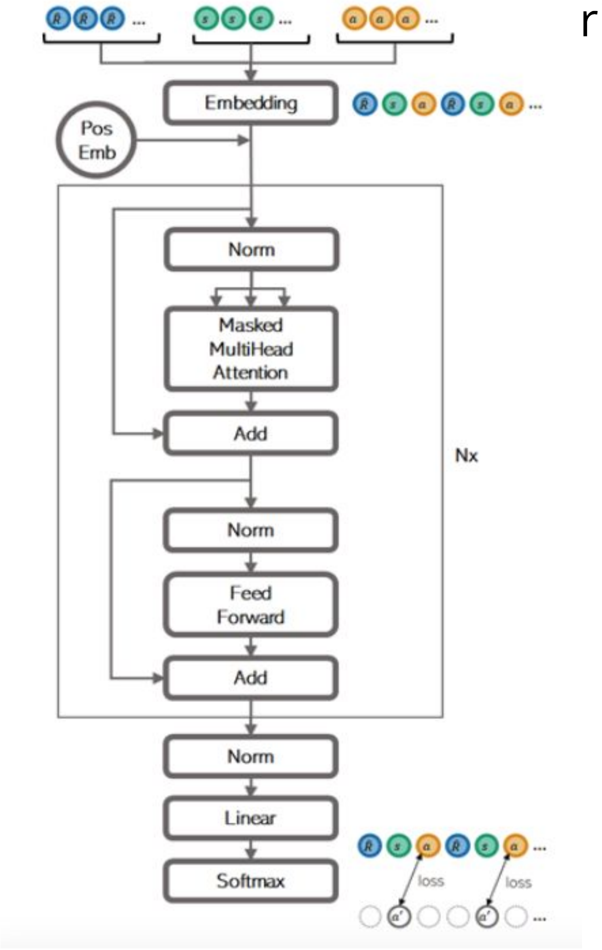
- **Rewards**
  +1 when get the ball across the opponent
  -1 when the player misses the ball

Decision Transformer

ring

- Based on minGPT
- Model Input
  - State
  - Action
  - Reward to go(rtg)
  - Timesteps





Decision Transformer

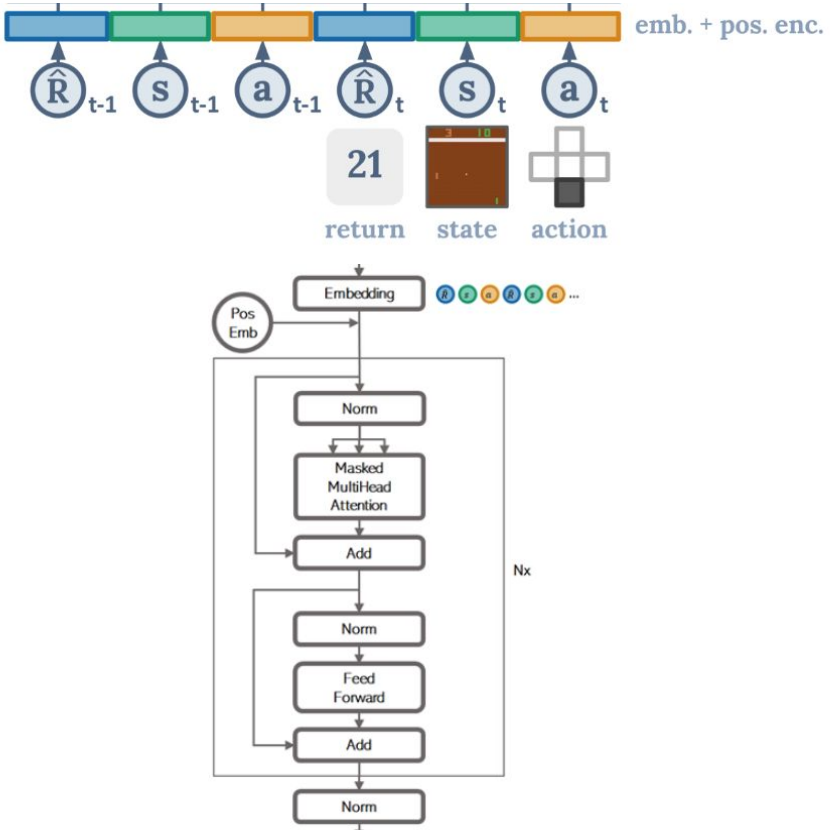
TEXAS A&M UNIVERSITY Engineering

- Embedding
  - States: Using CNN
  - Action: Embedding Matrix
  - Reward to go: Single-layer linear network

- Generate a sequence of tokens
  - [s], [a], [R] → [R, s, a, R, s, a, ...]

- Position Embedding

- MultiHead Masked Self-Attention
- MLP
- Add & Norm



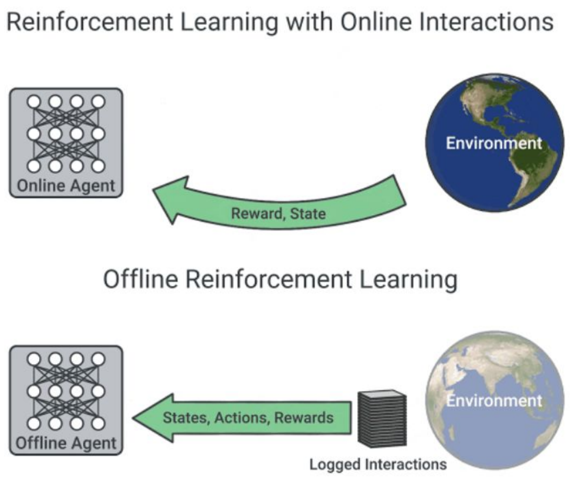
Training

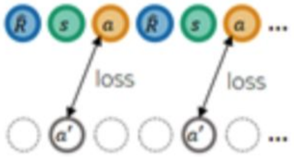
- [DQN Replay Dataset]
  - Agarwal, R., Schuurmans, D. & Norouzi, M.. (2020). An Optimistic Perspective on Offline Reinforcement Learning *International Conference on Machine Learning (ICML)*.

- Data Process
  - Game frames Stacking

- Loss Function
  - Cross-entropy loss function

- Arcade-Learning-Environment(ALE)





Results

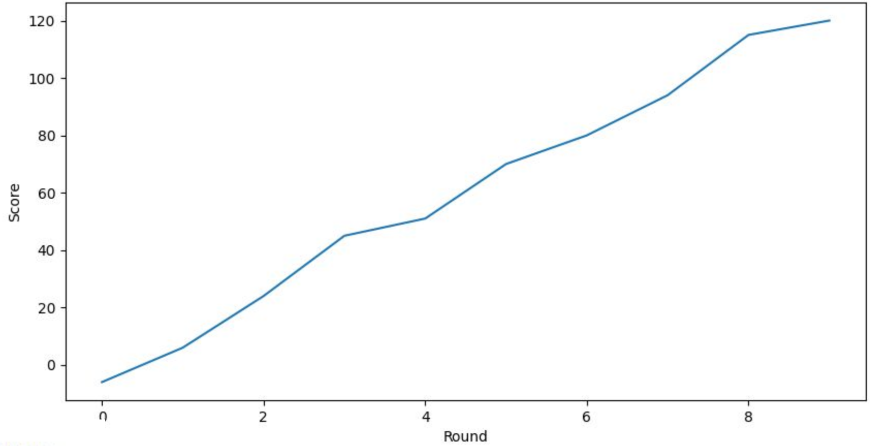
Evaluation

- Reward

```
Total return of 10 round: 120, Average return: 12.0
Total return of 10 round: 62, Average return: 6.2
```

Game	DT (Ours)	CQL	QR-DQN	REM
Pong	120	111.9	18.0	0.5

DT compared to other algorithms[1]



- Pseudocode[1]

```
target_return = 1  # for instance, expert-level return
R, s, a, t, done = [target_return], [env.reset()], [], [1], False
while not done:  # autoregressive generation/sampling
    # sample next action
    action = DecisionTransformer(R, s, a, t)[-1]  # for cts actions
    new_s, r, done, _ = env.step(action)

    # append new tokens to sequence
    R = R + [R[-1] - r]  # decrement returns-to-go with reward
    s, a, t = s + [new_s], a + [action], t + [len(R)]
    R, s, a, t = R[-K:], ...  # only keep context length of K
```

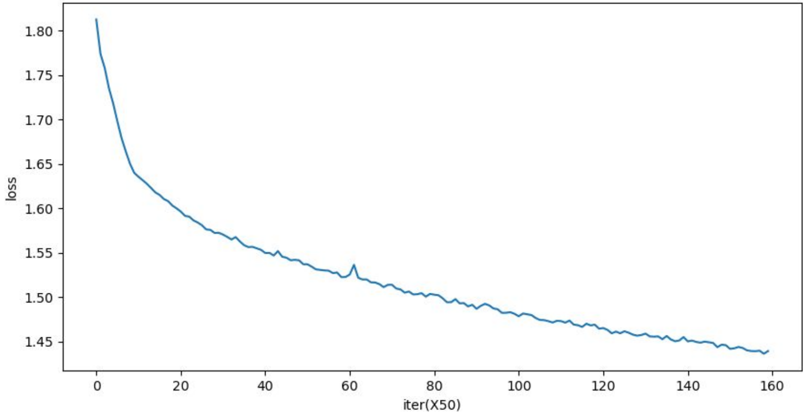
Context length K = 50

[1] Chen, Lili, et al. "Decision transformer: Reinforcement learning via sequence modeling." *Advances in neural information processing systems* 34 (2021): 15084-15097.

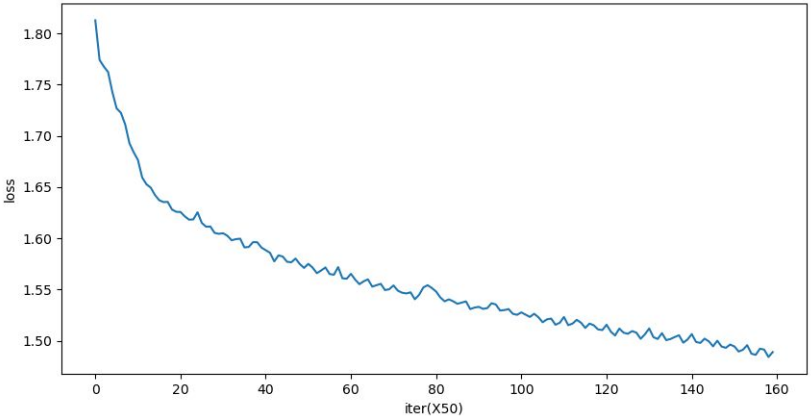
Results

Training

- Loss



lr = 1e-4



lr-decay

- Pseudocode[1]

```
# training loop
for (R, s, a, t) in dataloader:  # dims: (batch_size, K, dim)
    a_preds = DecisionTransformer(R, s, a, t)
    loss = F.cross_entropy(logits.reshape(-1, logits.size(-1)), labels.reshape(-1))
    optimizer.zero_grad(); loss.backward(); optimizer.step()
```

Results

TEXAS A&M UNIVERSITY
Engineering

Ablation Study - Context Length

Context Length	Total Return	Average Return
1	-129	-12.9
10	-15	-1.5
30	58	5.8
40	115	11.5
50	**120**	**12.0**

- RL and MDP traditionally need context length = 1
- Decision transformer performs *significantly* better with *longer context length*
- Hypothesis:
  - Helps model identify which policy generated previous context
  - Better estimate policy distribution
  - Longer context length allows model to capture temporal dependencies

Model Pruning

Apply L1 unstructured pruning to all convolutional layers and linear layers.

Trade-off between efficiency and accuracy.

Pruning Rate	No pruning	0.01	0.1	0.2
Average Return	12.7	12.1	7.1	-18.9
Inference Time per Step(ms)	6.79	6.74	6.12	5.97

Conclusion

TEXAS A&M UNIVERSITY
Engineering

- We reimplemented and proposed improvements on Decision Transformer, which outperforms strong offline RL algorithms
- Scope change
- Our improvements on Decision Transformer
  - Learning rate decay, optimizing for Atari Pong, memory efficiency
- Future Work:
  - Test and compare DT in online RL settings
  - Sophisticated embeddings - return distribution

Thank you