# Unraveling Max-Return Sequence Modeling via Return Consistency


Zifeng Zhuang
*Westlake University, Zhejiang University*
Hangzhou, China
zhuangzifeng@westlake.edu.cn

Dengyun Peng
*Harbin Institute of Technology*
Harbin, China
dypeng@ir.hit.edu.cn

Jiacheng Liu
*Zhejiang University*
Hangzhou, China
liujiacheng@westlake.edu.cn

Xiao He
*Westlake Robotics*
Hangzhou, China
hexiao18@mails.ucas.ac.cn

Ting Wang†
*Westlake University*
Hangzhou, China
wangting@westlake.edu.cn

Donglin Wang†
*Westlake University*
Hangzhou, China
wangdonglin@westlake.edu.cn



***Abstract*—Offline reinforcement learning (RL) learns from fixed datasets without interaction with online environment, enabling supervised solutions for offline RL. Decision Transformer (DT) casts offline RL as return-conditioned supervised sequence modeling, thereby sidestepping optimal value fitting and policy gradients. This paradigm overlooks RL's core objective of return maximization, which yields brittle behavior on suboptimal trajectories and limited stitching ability. Rein*former* reorients this objective through max-return sequence modeling: during inference, the model conditions on the predicted maximum achievable returns to generate the optimal actions. To better understand both the SOTA performance of this paradigm and its occasional dramatic failures, we adopt a supervised perspective and introduce the return consistency to assess whether similar state-action pairs have similar returns. Indeed, high return consistency guarantees the maximized return reliably cues the optimal action, while low consistency may lead to suboptimal action selection. Through visualizations, two different consistency modes are exposed and we quantify this via the return standard deviation of the data cluster with highest return mean. Furthermore, we reveal the relationship between this metric and 1) final performance, 2) context lengths, 3) model architectures through a systematic study. Finally, we improve return consistency by explicitly decreasing the return standard deviation, thereby further increasing the performance.**

***Index Terms*—Offline RL, Max-return Sequence Modeling**


## I. Introduction

Classical online reinforcement learning (RL) methods such as Q-learning [1] or policy gradient [2] are built on the Markov Decision Process (MDP) [3] formulation, which differs fundamentally from supervised learning. Offline RL [4, 5], learning from static datasets, moves closer to this paradigm. On the basis, sequence modeling [6], also Decision transformer (DT) [6], formulates policy learning as likelihood maximization over trajectories. However, it suffers from manual target return specification [7] and weak return maximization, leading to poor trajectory stitching [8]. Reinforced Transformer (Rein*former*) [9] addresses these issues by introducing max-return sequence modeling, directly predicting optimal returns to guide action generation, thus removing the need for preset targets and improving performance.

Moreover, when and why max-return sequence modeling outperforms or underperforms conventional offline RL remains an open question, stalling further improvement. To address this, we conduct an in-depth investigation into the inference stage from the data perspective. Intuitively, similar state-action pairs should carry similar returns; only under this condition does maximizing predicted return guarantee selecting the truly optimal action. If the assumption is violated, the model may blindly pick actions whose nearest neighbors map to low returns, triggering an abrupt performance collapse. We refer to the scenario where similar state-action pairs yield similar returns as **high return consistency**. Indeed, visualizations across different datasets expose two distinct return-consistency regimes and we quantify this via *the return std of the cluster with highest return mean*.

Surrounding the return consistency, we explore the relationships between this concept and 1) the performance, 2) context length, as well as 3) model architecture. In summarize, datasets with higher consistency tend to exhibit superior performance for max-return sequence modeling, which also favors longer context length and benefit more from architectures that focus on global information. Furthermore, the performance of max-return sequence modeling is clear to improve. Enhancing return consistency, also reducing the standard deviation of returns, exactly suggests that classical variance-reduction technique from traditional RL is helpful. Specifically, we substitute raw returns with advantages, the expectation of return minus the value baseline. This replacement substantially improves return consistency and leads to significant gains in final performance. This phenomenon, achieved with only return replacement, constitutes indirect validation of the return consistency itself.

## II. Preliminary

### A. Offline Reinforcement Learning

Offline RL [4] forbids the interaction with the environment and only a fixed offline dataset full of trajectories $\mathcal{D} = \{(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_t, a_t, r_t \cdots)\}$ is provided . Here $s_t$ is the current state at timestep $t$, $a_t$ is the action

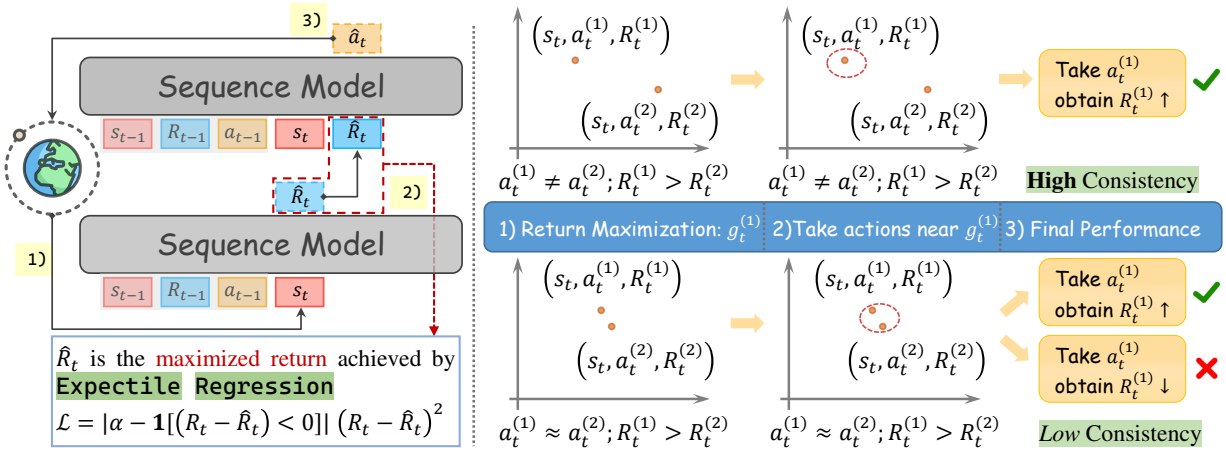



Fig. 1: **Left:** Max-return sequence modeling, during inference, first predicts the maximized return using expectile regression. And then this predicted maximized return is reintroduced back into the same transformer to guide the optimal action generation. **Right:** The visualization of two possible action selection process under **high** or *Low* return consistency scenarios.

and $r_t \doteq r(s_t, a_t)$ is the reward of current state and action. The objective of offline RL is to learn a policy $\pi(a_t|s_t)$ that maximizes the expected returns $\mathbb{E}_\pi\left[\sum_{t=0}^{T} r(s_t, a_t)\right]$. Compared to the traditional online RL [3], this setting is more challenging since the agent is unable to explore the environment and collect extra feedback.

### B. Sequence Modeling

Sequence modeling [6] breaks the traditional Markov property, where the prediction of the current action $a_t$ is based on the previous $K$ (also called context length) timesteps trajectories $\tau_{t-K} = (R_{t-K+1}, s_{t-K+1}, a_{t-K+1}, \cdots, R_{t+1}, s_{t+1}, a_{t+1})$ where $R_t \doteq \sum_{t'=t}^{T} r_{t'}$ represents the sum of future rewards from the current timestep $t$, known as returns-to-go (or simply returns). Sequence modeling directly maximizes the likelihood of actions conditioned on not only the current state $s_t$ and returns-to-go $R_t$, but also the historical trajectories $\tau_{t-K}$:

$$\mathcal{L}_{\mathrm{DT}} = -\mathbb{E}_t\Big[\log \pi(a_t|\tau_{t-K}, s_t, R_t)\Big]. \tag{1}$$

This loss indicates offline RL is solved from the supervised perspective, rather than traditional RL paradigm. Besides, $\pi$ is implemented based on sequence model transformer [10].

For the **Inference**, the initial target returns $\hat{R}_0$ must be determined first. Given $\hat{R}_0$ and the initial environment state $s_0$, the next action is generated by the model $\pi\left(a_1|\hat{R}_0, s_0\right)$. Once the action $a_1$ is executed, the environment returns the next state $s_1$ and reward $r_1$ are returned. The next return-to-go is updated as $\hat{R}_1 = \hat{R}_0 - r_1$. This process is repeated until the episode terminates.

### C. Max-return Sequence Modeling

Since supervised sequence modeling does not explicitly consider return maximization, the core objective of RL, the concept of max-return sequence modeling is introduced. The key lies in utilizing the maximized return to guide the generation of next actions during inference. Concretely, Reinforced Transformer (Rein*former*) [9] adopt the following historical trajectories $\tau_{t-K} = (s_{t-K+1}, R_{t-K+1}, a_{t-K+1}, \cdots, s_{t+1}, R_{t+1}, a_{t+1})$ where the state $s_t$ is placed before the returns-to-go $\hat{R}_t$, different from the original DT. The main advantage is that the return can be predicted through the state without the need for prior specification. During training, Rein*former* introduces a return loss based on expectile regression in addition to the action probability maximization loss:

$$\begin{aligned}\mathcal{L}_{\text{Rein}former} = &-\mathbb{E}_t \log \pi(a_t|\tau_{t-K}, s_t, R_t) \\ &+\mathbb{E}_t\left[|\alpha - \mathbb{1}(\Delta R_t < 0)| \Delta R_t^2\right],\end{aligned} \tag{2}$$

where $\Delta R_t = R_t - \pi\left(\hat{R}_t|\tau_{t-K}, s_t\right)$ is the difference between the oracle return $R_t$ and its prediction $\hat{R}_t$. Here $\alpha \in (0, 1)$ is the hyperparameter of expectile regression. When $\alpha = 0.5$, expectile regression degenerates into standard MSE loss. But when $\alpha > 0.5$, this asymmetric loss gives more weight to $R_t$ larger than $\hat{R}_t$. It can be proved that this additional return loss can make the model predict the maximum returns-to-go when $\alpha \to 1$, similar to the return maximization objective in RL.

When **inference**, given the initial state $s_0$, the maximized initial target return is predicted $\pi\left(\hat{R}_0|s_0\right)$ rather than manually specified. Since $\hat{R}_0$ is maximized, the next action $\pi\left(a_1|\hat{R}_0, s_0\right)$ will approach to the optimal one. The next state $s_1$ is returned and this process is repeated until termination.

## III. Return Consistency

We first conduct a detailed analysis of the potential issues during the inference phase of max-return sequence modeling. Based on this, the concept of high return consistency is introduced. We also develop a concretely metric to measure return consistency with the help of clustering. In addition, we explore the relationships between return consistency and the performance, context length, model architecture, drawing several conjectures. Finally, we enhance return consistency by deploying the classical RL variance-reduction trick, which is strikingly simple yet measurably strong.

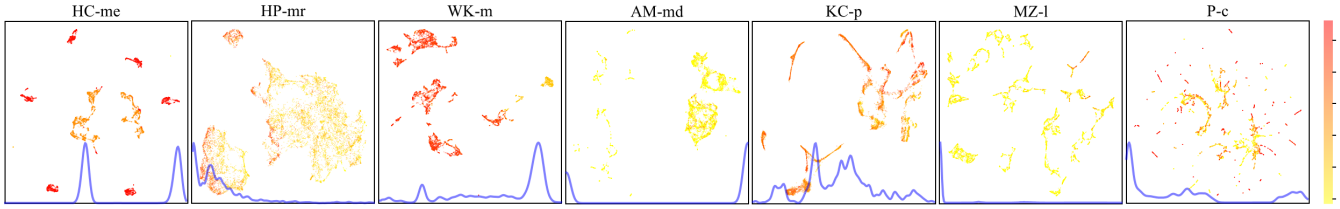



Fig. 2: The state-action distribution, obtained through dimensionality reduction using Umap [11], is visualized alongside the KDE curves (blue) of the returns. `AM-mp` and `P-h` are omitted here due to the similarity with `AM-md` and `P-c`.

*a) Analysis of Inference Stage:* At timestep $t$, the max-return sequence modeling employs the following inputs and outputs during training:

$$\textbf{Input:} \ \langle \tau_{t-K}, s_t, R_t \rangle \xrightarrow{\pi} \textbf{Output:} \ \left\langle \hat{R}_t, \hat{a}_t \right\rangle. \quad (3)$$

The model attempts to establish a connection between return $R_t$ and action $a_t$. During the inference phase, max-return sequence modeling first predicts a maximized return $\hat{R}_t^{\max}$ and then attempts to forecast the optimal action $\hat{a}_t^*$ under the guidance of this maximized return:

$$\textbf{Input:} \ \left\langle \tau_{t-K}, s_t, \hat{R}_t^{\max} \right\rangle \xrightarrow{\pi} \textbf{Output:} \ \langle \hat{a}_t^* \rangle. \quad (4)$$

We illustrate the principle and potential issues of this inference with the example in right part of Figure 1. Assume we have two datapoints $\left(\tau_{t-K}, s_t, a_t^{(1)}, R_t^{(1)}\right)$ and $\left(\tau_{t-K}, s_t, a_t^{(2)}, R_t^{(2)}\right)$ where $R_t^{(1)} \geq R_t^{(2)}$. At inference, two scenarios may occur:

- **Problematic Case** $a_t^{(1)} \approx a_t^{(2)}$: But when $a_t^{(1)}, a_t^{(2)}$ are similar, the process of predicting $a_t^{(1)}$ may erroneously yield $a_t^{(2)}$, leading to a decline in performance.
- **Ideal Case** $a_t^{(1)} \neq a_t^{(2)}$: Max-return sequence modeling first predicts a value close to $R_t^{(1)}$ as the maximized return. Since $a_t^{(1)}, a_t^{(2)}$ are not similar, the model $\pi$ will accurately predict $a_t^{(1)}$ under the guidance of $R_t^{(1)}$, thereby maximizing the return.

The above **Ideal Case** can be rigorously formulated using the following definition:

**Definition III.1.** *Given a dataset $D = \{(s_t, a_t, R_t)\}_{t=1}^N$ consisting of state-action-return triplets, where $(s_t, a_t)$ is a state-action pair, $R_t$ is the corresponding return, $d_{sa} : (\mathcal{S} \times \mathcal{A}) \times (\mathcal{S} \times \mathcal{A}) \to \mathbb{R}^+$ is a distance metric in the state-action space, $d_g : \mathcal{G} \times \mathcal{G} \to \mathbb{R}^+$ is the Euclidean distance. The dataset belongs to **Ideal Case** if the following holds:*

$$\forall \varepsilon > 0, \ \exists \delta > 0, \ \textit{s.t.} \sup_{t \in [1,N]} \max_{t' \in B_\delta(t)} d_g(R_t, R_{t'}) < \varepsilon, \quad (5)$$

*where the neighborhood $B_\delta(t) = \{t' \in [1, N] \mid d_{sa}[(s_t, a_t), (s_{t'}, a_{t'})] < \delta\}$.*

Simply put, this definition implies that when state-action pairs are sufficiently close to each other, their corresponding returns are also close. Under such scenarios, the max-return sequence modeling can distinguish the quality of different actions. Therefore, we refer to datasets that satisfy this definition as having **high return consistency**, and those that do not as having *low return consistency*.

*b) Visualization and Metric:* In this paper, we consider 9 representative datasets including 1) common `Halfcheetah-medium-expert`, `Hopper-medium-replay`, `Walker2d-medium`, 2) trajectory stitching domain `Antmaze-medium-play`, `Antmaze-medium-diverse` and 3) more diverse `Kitchen-partial`, `maze2d-large`, `Pen-human` and `Pen-cloned` with various characteristic[1]. To verify whether these datasets meet the definition of **high** return consistency, we visualize the state-action distributions in Figure 2, with colors indicating the magnitude of the returns.

In Figure 2, we can observe that for `HC-me` (the first one), the outer ring consists entirely of high returns, while the inner ring comprises medium returns, with a very clear boundary. This aligns with the definition of **high** return consistency. In contrast, for `AT-md` (the fourth one), sporadic red dots are interspersed among the yellow dots, indicating a clear presence of state-action pairs that are similar yet have significantly different returns. So `AT-md` is *low* return consistency.

Inspired by the above visualization, we further develop a metric for return consistency across different datasets. DBSCAN [12] is first applied to cluster state-action pairs. Then, for each cluster, we compute the mean return (to identify high-return regions) and standard deviation of returns (to measure variability). We use the std of the highest-mean-return cluster as the metric for return consistency, since 1) RL focus on high-return regions and 2) std quantifies the spread of returns—smaller values indicate better consistency. Since the returns are normalized, allowing us to define return consistency as follows:

$$\textbf{High return consistency}: \ \text{std} \approx 0; \quad (6)$$

$$\textit{Low return consistency}: \ \text{std} \approx 1. \quad (7)$$

*c) The relationship between return consistency and performance, context length, model architecture.:* In datapoint $\left(\tau_{t-K}, s_t, a_t^{(1)}, R_t^{(1)}\right)$, the context length affects $\tau_{t-K}$. The longer the context length $K$, the more informative $\tau_{t-K}$ is, thereby making the following mapping more robust:

$$\left\langle \tau_{t-K}, s_t, \hat{R}_t^{(1)} \right\rangle \xrightarrow{\pi} \left\langle \hat{a}_t^{(1)} \right\rangle. \quad (8)$$

Additionally, when the data exhibit high consistency, the validity of this mapping positively impacts performance. Therefore, the higher return consistency and the longer context

[1]These datasets are abbreviated as `HC-me`, `HP-mr`, `WK-m`, `AM-mp`, `AM-md`, `K-p`, `MZ-l`, `P-h`, `P-c`.

TABLE I: The return consistency defined by the return std of the cluster with highest return mean.

| Datasets | HC-me | HP-mr | WK-m | AT-md | KC-p | MZ-l | P-c |
|---|---|---|---|---|---|---|---|
| Return Range | $[-2.67, 1.22]$ | $[-1.25, 3.00]$ | $[-3.29, 1.20]$ | $[-0.21, 4.73]$ | $[-2.45, 2.46]$ | $[-0.32, 5.67]$ | $[-0.96, 1.73]$ |
| Return Mean | 1.07 | 2.05 | 0.58 | 0.06 | 1.88 | 3.16 | 1.73 |
| Return Std | 0.03 | 0.94 | 0.05 | 1.12 | 0.07 | 1.00 | 0.00 |
| ***consistency*** | **High** | *Low* | **High** | *Low* | **High** | *Low* | **High** |

length, the better performance. We consider context length $K = 2, 5, 10, 20$, with the maximum value of 20 being the default of DT [6], and the minimum of 2 corresponding to the shortest length.

Moreover, when the return consistency is high and the context length $K$ is long, model architecture that can capture global information gain a significant advantage. In our experiments, we investigated the impact of three model architectures on the final performance, including the Transformer [10], One-dimensional convolutional layers (Conv) [13, 14], and the linear Recurrent Neural Network Mamba [15–19]. Their corresponding sequence models are called Rein*former*, Reinconver and Reimba. Since the Transformer itself lacks the ability to understand relative positional relationships, we augmented it with positional encoding, while the other two models not.

**Conjectures**

1) The performance of max-return sequence modeling is positively correlated with **return consistency**. 2) **High** consistency datasets prefer **longer** context length $K$. 3) Architecture that excels at understanding **global** information are more compatible with **high** consistency.

*d) Return consistency with advantage:* Previous first conjecture about performance is verified in section V-A, motivating a simple yet effective upgrade to max-return sequence modeling: increasing return consistency by decreasing the standard deviation of returns inside the highest-mean-return cluster. More simply, we can reduce the global std of the return by subtracting a baseline from its expectation, a common technique in traditional RL. Concretely, we replace the raw return with the advantage estimated by IQL [20] and the updated consistency is presented in Table II. The corresponding performance improvement is exhibited in Table IV.

## IV. Related Work

Offline Reinforcement Learning [4] breaks free from the traditional paradigm of online interaction [3] and learns policy from fixed offline dataset collected by arbitrary or even unknown process [5]. Most offline RL algorithms are developed based on classical online algorithms, such as CQL [21] based on SAC [22], TD3+BC [23] based on TD3 [24] and BPPO [25] based on PPO [26]. In contrast, Decision Transformer (DT) [6] directly maximizes the action likelihood, solving offline RL from supervised sequence modeling paradigm. Following upside-down RL [27, 28], DT considers returns when predicting the action. Some works equip DT with classical RL components including dynamics programming [29], critic guidance [30, 31], return maximization [9], online finetuning [7] and trajectory stitching [32]. On the other hand, DT is investigated from supervised learning perspective such as unsupervised pretraining [33] and scaling ability [34, 35]. As for model architecture, LSTM [36], one-dimension convolution network [14, 37] and linear RNN [16–19] are adopted to replace the transformer [10] in DT. Some work also bring the critic or advantage back to sequence modeling, such as Q-learning DT [29], CGDT [30], ACT [38]. Besides, QT [31] using the Q-function as the loss of sequence modeling, which strictly belongs to traditional RL with transformer policy.

## V. Results and Discussion

In this section, we present the performance of max-return sequence modeling across 9 datasets, 3 architectures, and 4 different context lengths. Based on this, we conduct an in-depth analysis to uncover the underlying principles of performance, gradually validating the above Conjectures III-0c. This leads us to draw conclusions regarding the relationship between performance, context length, model architecture, and return consistency. Finally, we demonstrate the significant improvement in trajectory stitching performance of Rein*former* after the return consistency is improved.

### A. Main Results

Table III presents the performance of max-return sequence modeling with different parameters on diverse data distribution. We also compare the performance of sequence modeling algorithms with the most classic offline RL algorithm IQL [20], highlighting scores below IQL in gray. IQL significantly outperforms sequence modeling on datasets with *Low* return consistency including HP-mr, AT-mp, AT-md and MZ-l. Moreover, these datasets are heavily emphasize trajectory stitching ability, which corresponds to the definition of *Low* consistency. In contrast, datasets with **high** consistency either

TABLE II: The return consistency with advantage from IQL rather then original return.

| *Low* Datasets | HP-mr | AT-md | MZ-l |
|---|---|---|---|
| Return Range → Advantage Range | $[-1.25, 3.00] \rightarrow [-2.86, 1.25]$ | $[-0.21, 4.73] \rightarrow [-1.69, 1.96]$ | $[-0.32, 5.67] \rightarrow [-2.23, 6.05]$ |
| Return Mean → Advantage Mean | $2.05 \rightarrow 0.87$ | $0.06 \rightarrow 1.70$ | $3.16 \rightarrow 4.26$ |
| Return Std → Advantage Std | $0.94 \rightarrow \mathbf{0.11}$ | $1.12 \rightarrow \mathbf{0.10}$ | $1.00 \rightarrow \mathbf{0.01}$ |
| ***consistency*** | *Low* → **High** | *Low* → **High** | *Low* → **High** |

TABLE III: The normalized score of max-return sequence modeling on 9 datasets (HC-me, HP-mr, WK-m, AT-mp, AT-md, KC-p, MZ-l, P-h, P-c) with 3 different architectures and 4 context-lengths ($K$). Datasets with **high** return consistency are red while *Low* is green. We report the mean of normalized score for five seeds. For each seed, the normalized score is calculated by the mean of 10 evaluation trajectories for Gym and Adroit while 100 for Antmaze, Maze2d and Kitchen. The scores below IQL are in gray. The best result is blue and the **bold** result means the best result among one sequence model with different $K$. The last row represents how many results outperforms IQL.

| model | $K$ | HC-me | HP-mr | WK-m | AT-mp | AT-md | KC-p | MZ-l | P-h | P-c |
|---|---|---|---|---|---|---|---|---|---|---|
| Reinformer | 2 | 91.23 | **70.92** | 79.84 | **5.80** | 2.00 | 68.05 | 61.80 | 62.77 | 64.49 |
| Reinformer | 5 | 90.99 | 68.80 | **79.91** | 4.20 | 3.40 | 73.00 | 64.95 | **75.15** | **86.55** |
| Reinformer | 10 | 91.87 | 53.02 | 79.82 | 3.80 | **5.60** | **74.05** | 62.00 | 68.25 | 75.17 |
| Reinformer | 20 | **92.81** | 40.84 | 72.25 | 1.60 | 4.20 | 66.20 | **64.99** | 71.94 | 74.79 |
| Reinconver | 2 | 91.83 | 84.24 | 72.28 | 6.20 | **5.40** | 65.20 | 32.69 | 73.64 | 68.52 |
| Reinconver | 5 | 92.26 | **84.44** | 74.09 | **7.80** | 4.20 | 34.55 | 22.45 | **82.23** | 71.68 |
| Reinconver | 10 | **92.90** | 54.02 | **75.88** | 4.40 | 5.20 | **65.85** | **34.74** | 76.27 | 62.58 |
| Reinconver | 20 | 92.78 | 49.22 | 75.38 | 2.00 | 2.60 | 65.25 | 32.39 | 75.29 | **83.38** |
| Reimba | 2 | 91.79 | **81.95** | **77.81** | 5.20 | 2.60 | 40.75 | **59.00** | 84.89 | 59.60 |
| Reimba | 5 | 92.91 | 74.24 | **80.03** | 12.40 | 5.00 | **45.10** | 41.04 | 82.91 | **71.28** |
| Reimba | 10 | **93.05** | 55.99 | 75.59 | 13.80 | 5.00 | 29.70 | 43.59 | **97.31** | 71.02 |
| Reimba | 20 | 92.42 | 49.47 | 73.35 | **15.60** | **9.00** | 29.05 | 43.14 | 91.61 | 70.57 |
| IQL | 1 | 86.70 | **94.70** | 78.30 | **65.80** | **73.80** | 46.30 | 61.70 | 71.50 | 37.30 |
| | | (12/12) | (0/12) | (4/12) | (0/12) | (0/12) | (6/12) | (4/12) | (10/12) | (12/12) |

matches or even surpasses the performance of IQL. This experiment results confirm the validity of our first conjecture.

**Conclusion I**

Return consistency significantly impacts the performance of max-return sequence modeling. *Low* return consistency datasets often exhibit notably inferior performance while **high** consistency datasets achieve comparable performance and even surpass RL.

### B. Context Length

In this section, we investigate the impact of the historical sequence length, also known as context length $K$, on performance. Sequence modeling and MDPs have distinct perspectives on the historical trajectory when predicting actions, making context length a crucial factor in sequence modeling.

*1) Context Length Comparison:* We plot the performance curves and the return distribution on HP-mr in Figure 3. The quality of HP-mr is widely distributed, ranging from random to expert, with a peak less than 20 normalized score. The distribution of HP-mr is akin to an online replay buffer, which places higher demands on learning from suboptimal trajectories. Correspondingly, a smaller context length aligns more closely with the Markov Decision Process (MDP) framework, and thus performs better.

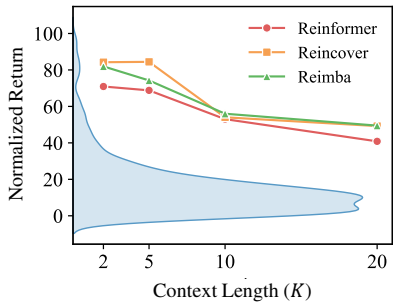



Fig. 3: The data distribution (blue shade) and normalized evaluation score on HP-mr.

Upon considering all datasets, it becomes evident that no context length is universally applicable across 9 datasets (Figure 4a). For high-quality datasets, such as HC-me, a longer context length facilitates better training convergence and leads to improved score in Figure 4c. For trajectory stitching, shorter trajectories are preferred 4b. Taking into account longer historical trajectories increases the influence of past actions on subsequent behaviors, which may hinder the adoption of trajectories that deviate from historical ones. This is detrimental to the stitching process. In other words, longer historical trajectories can be seen as conservatism.

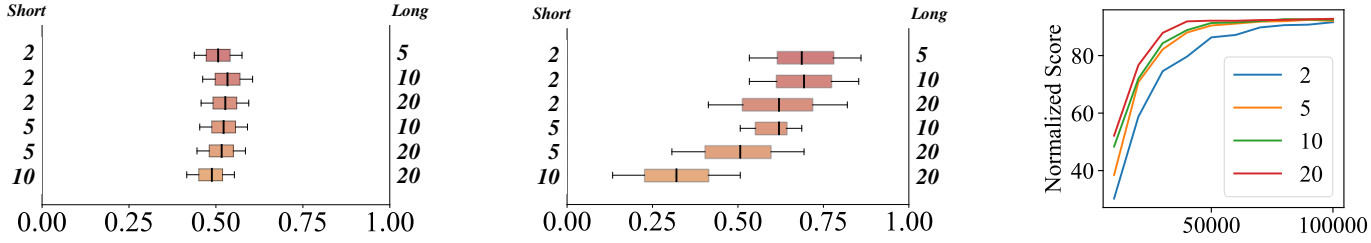
Based on the experiments outlined above, we can derive the second following conclusion:

**Conclusion II**

For datasets with **high** consistency, longer context length $K$ generally lead to better performance. Conversely, shorter $K$ tend to be more effective for *Low* consistency.

*2) Long Training Context Length while Short Inference Context Length:* All previous models have considered the historical trajectory length to be the same during inference as in the training phase. ODT [7] discovers that, in some cases, a shorter sequence length during inference can improve performance. EDT [32] also proposes the concept of dynamically adjusting the sequence length during inference. Inspired by these, we explore the performance of masking some historical tokens with the model trained by context length $K = 20$.

During the inference phase, we use historical trajectories




(a) 9 datasets (b) AT-mp (c) HC-me

Fig. 4: (a) represents the probability of the left $K$ superior to the right one across all datasets. (b) represents the probability on AT-mp (c) represents evaluation scores with different $K$ on HC-me.

of length $K_1 < 20$, padding the empty tokens with zeros to meet the model's input length requirement. This can be viewed as masking a segment of length $20 - K_1$. In Figure 5, as the horizontal axis increases, the input trajectory length $K_1$ decreases, while the zero-padded portion increases accordingly. Notably, the model's performance improves significantly under these conditions, surpassing all the different $K$.

### C. Architecture

In this section, we explore the impact of model architecture on the performance of sequence modeling. Prior research has indicated that the sequence modeling with Convformer [14] and Mamba architecture [16–19] outperform transformer. However, these conclusions were drawn without considering the diverse data distribution. Therefore, we will re-examine these findings across 9 datasets.

*1) Attention on Historical Tokens:* We analyze which part of the historical trajectory different model constructions specifically focus on. We selected the model trained with $K = 10$ on the Antmaze environment. Let $t$ represent the time step of the current token, and $t - 9$ represents the token furthest from the current time step. By masking a token at a certain position with 0, we calculate the difference bet This difference can, to some extent, reflect the importance of the masked token to the current output value. Then, based on this difference, we can determine whether the model pays more attention to global or local information. We have plotted the heatmaps of the differences in state, return, and action for the three models in the right figure.

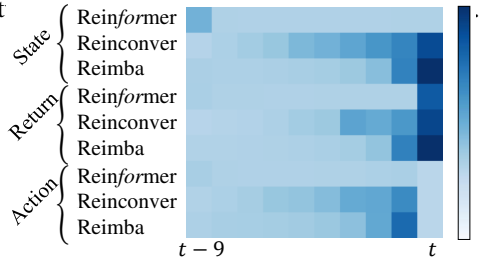
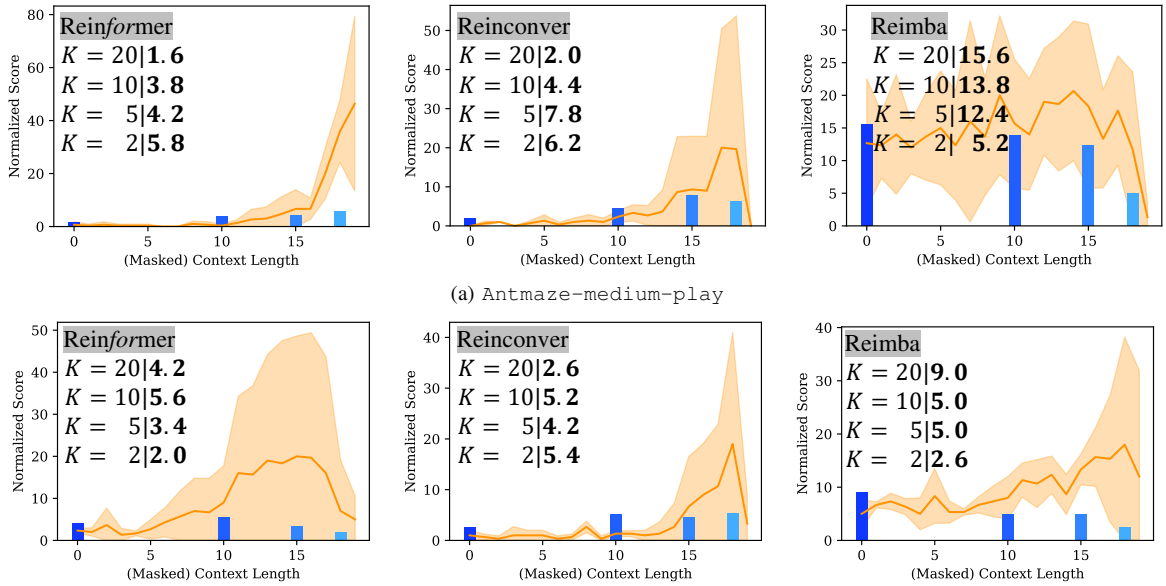



Fig. 6: This heatmap illustrates the impact of token zero masking on the final output.

The heatmap reveals that Reinconver and Reimba exhibit a significant increase in values at the current timestep, indicating




(a) Antmaze-medium-play

(b) Antmaze-medium-diverse

Fig. 5: This figure displays the performance of masking the first $(20 - K_1)$ tokens in a sequence model with $K = 20$. The averages and corresponding standard deviations of three seeds are represented by the solid yellow line and its shaded area. Additionally, we compare this with models trained and evaluated normally with $K = 20, 10, 5, 2$ (blue bar values). The horizontal axis increases from left to right as the masked tokens increases and the remaining context length $K_1$ decreases.

TABLE IV: Performance on *low* return consistency datasets with other baselines. The best results among sequence models are **bold**. Some results are reproduced using public code.

| Datasets | IQL | Rein*former*-best | Rein*former* + $A$ | DT | ODT | EDT | ACT |
|---|---|---|---|---|---|---|---|
| HP-mr | 94.7 | 84.44 | 88.24 ± 5.21(+ 3.80) | 82.7 | 86.6 | 91.2 | **98.4** |
| AT-mp | 65.8 | 15.60 | **43.60** ± **13.89**(+28.00) | 0.8 | 0.0 | 0.0 | 1.8 |
| AT-md | 73.8 | 9.00 | **56.50** ± **6.36**(+47.50) | 0.5 | 0.0 | 0.0 | 2.4 |
| MZ-l | 61.7 | 64.99 | **116.92** ± **48.30**(+51.93) | 35.7 | 39.4 | 26.8 | 58.3 |

the focus on local information. In contrast, the Rein*former* does not show a marked rise in differences, suggesting that the impact of masking any token is relatively uniform. Thus, Rein*former* focuses on global information.

*2) Architecture Comparison:* In Figure 7, we illustrate the probability that the architecture on the left outperforms the right one. The closer the box is to the right side, the better the performance of the model on the left, and vice versa. A central position indicates that the two architectures have comparable performance. Considering the nine datasets collectively, no single model demonstrates an absolute advantage. In other words, the superiority of a model cannot be discussed independently of the characteristics of the dataset.

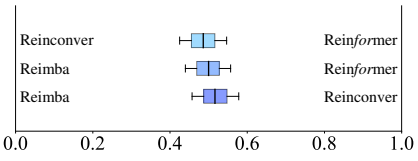
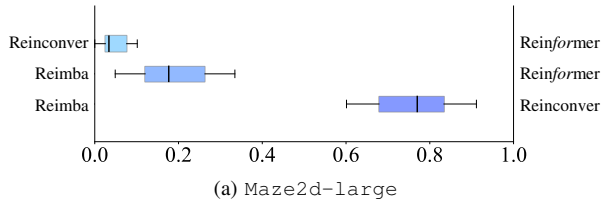
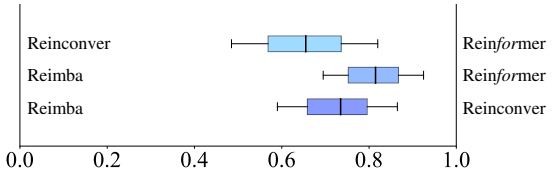



Fig. 7: The improvement probability of architectures across all the 9 datasets.

On the MZ-l dataset, the Rein*former* demonstrates a significant advantage. This is because the maze-large dataset inherently exhibits non-Markovian properties, where decisions based on the current state are correlated with historical waypoints. The Reinformer's focus on global historical trajectory information is particularly adept at considering and utilizing waypoint-related information effectively.




(a) Maze2d-large




(b) Antmaze-medium-play

Fig. 8: The probability of the model on the left superior to the model on the right across (a) Maze2d-large and (b) Antmaze-medium-play.

In contrast, on the Antmaze-medium-play dataset, which emphasizes trajectory stitching, models like Reincover and Reimba that focus on local information perform better. This is attributed to the fact that extensive historical sequence information leads to more conservative model outputs, reducing the likelihood of generating new decisions that deviate from historical trajectories.

**Conclusion III**

For **high** consistency datasets, architectures that focus on global information tend to better. Conversely, it is preferable to focus on local for *low* consistency datasets.

### D. Results with Advantage

As indicated in Table II, replacing the return with the advantage can significantly reduce the std of the cluster with the highest mean, thereby transforming the scenario from *low* return consistency to **high**. In Section V-A, we have already revealed the positive correlation between the performance of max-return sequence modeling and return consistency. Therefore, we investigate here whether the performance of max-return sequence modeling, with the aid of the advantage, far surpasses that based on the return. This can also serve as a validation of Conclusion I.

*a) Baselines:* Rein*former*-best denotes the best performance achieved for each dataset in Table I, regardless of the context length and model architecture. Rein*former* + $A$ indicates that the returns-to-go are replaced with the advantage during training. Elastic Decision Transformer (EDT) [32] explicitly consider trajectory stitching through dynamically adjusts historical trajectories during inference. ACT [38] also adopts the advantage of IQL.

As anticipated before, replacing the return with the advantage significantly enhances return consistency, which in turn leads to a substantial improvement in trajectory stitching capabilities. Thus, we have not only elucidated the reasons behind the suboptimal performance of max-return sequence models but also proposed a simple yet highly effective solution.

## VI. Conclusion

In summary, our investigation into max-return sequence modeling reveals that return consistency is a crucial factor influencing performance. Datasets with **high** return consistency tend to achieve better performance in max-return sequence modeling. These datasets also benefit more from longer context lengths and architectures that emphasize global information. Additionally, we find that replacing the return with the advantage function can enhance return consistency, leading to significant performance improvements and stronger stitching ability. These findings provide valuable insights for advancing

sequence modeling and bridging the gap with classical offline RL algorithms.

## VII. Acknowledgements

This work was supported by the Brain Science and Brain-like Intelligence Technology — National Science and Technology Major Project (Grant No. 2022ZD0208800)

## References

[1] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992.

[2] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in neural information processing systems*, vol. 12, 1999.

[3] R. S. Sutton, A. G. Barto *et al.*, "Introduction to reinforcement learning," 1998.

[4] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[5] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

[6] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," *Advances in neural information processing systems*, vol. 34, pp. 15 084–15 097, 2021.

[7] Q. Zheng, A. Zhang, and A. Grover, "Online decision transformer," in *international conference on machine learning*. PMLR, 2022, pp. 27 042–27 059.

[8] D. Brandfonbrener, A. Bietti, J. Buckman, R. Laroche, and J. Bruna, "When does return-conditioned supervised learning work for offline reinforcement learning?" *Advances in Neural Information Processing Systems*, vol. 35, pp. 1542–1553, 2022.

[9] Z. Zhuang, D. Peng, Z. Zhang, D. Wang *et al.*, "Reinformer: Max-return sequence modeling for offline rl," *arXiv preprint arXiv:2405.08740*, 2024.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30.

[11] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[13] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, and S. Yan, "Metaformer is actually what you need for vision," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 819–10 829.

[14] J. Kim, S. Lee, W. Kim, and Y. Sung, "Decision convformer: Local filtering in metaformer is sufficient for decision making," *arXiv preprint arXiv:2310.03022*, 2023.

[15] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*.

[16] J. Cao, Q. Zhang, Z. Wang, J. Wang, H. Cheng, Y. Shao, W. Zhao, G. Han, Y. Guo, and R. Xu, "Mamba as decision maker: Exploring multi-scale sequence modeling in offline reinforcement learning," *arXiv preprint arXiv:2406.02013*, 2024.

[17] S. Huang, J. Hu, Z. Yang, L. Yang, T. Luo, H. Chen, L. Sun, and B. Yang, "Decision mamba: Reinforcement learning via hybrid selective sequence modeling," *arXiv preprint arXiv:2406.00079*.

[18] T. Ota, "Decision mamba: Reinforcement learning via sequence modeling with selective state spaces," *arXiv preprint arXiv:2403.19925*.

[19] Q. Lv, X. Deng, G. Chen, M. Y. Wang, and L. Nie, "Decision mamba: A multi-grained state space model with self-evolution regularization for offline rl," *arXiv preprint arXiv:2406.05427*, 2024.

[20] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*.

[21] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[22] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[23] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 20 132–20 145, 2021.

[24] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[25] Z. Zhuang, K. Lei, J. Liu, D. Wang, and Y. Guo, "Behavior proximal policy optimization," *arXiv preprint arXiv:2302.11312*.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[27] R. K. Srivastava, P. Shyam, F. Mutz, W. Jaśkowski, and J. Schmidhuber, "Training agents using upside-down reinforcement learning," *arXiv preprint arXiv:1912.02877*, 2019.

[28] J. Schmidhuber, "Reinforcement learning upside down: Don't predict rewards–just map them to actions," *arXiv preprint arXiv:1912.02875*, 2019.

[29] T. Yamagata, A. Khalil, and R. Santos-Rodriguez, "Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl," in *International Conference on Machine Learning*. PMLR, 2023, pp. 38 989–39 007.

[30] Y. Wang, C. Yang, Y. Wen, Y. Liu, and Y. Qiao, "Critic-guided decision transformer for offline reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, no. 14, 2024, pp. 15 706–15 714.

[31] S. Hu, Z. Fan, C. Huang, L. Shen, Y. Zhang, Y. Wang, and D. Tao, "Q-value regularized transformer for offline reinforcement learning," *arXiv preprint arXiv:2405.17098*, 2024.

[32] Y.-H. Wu, X. Wang, and M. Hamaya, "Elastic decision transformer," *arXiv preprint arXiv:2307.02484*, 2023.

[33] Z. Xie, Z. Lin, D. Ye, Q. Fu, Y. Wei, and S. Li, "Future-conditioned unsupervised pretraining for decision transformer," in *International Conference on Machine Learning*. PMLR, 2023, pp. 38 187–38 203.

[34] K.-H. Lee, O. Nachum, M. S. Yang, L. Lee, D. Freeman, S. Guadarrama, I. Fischer, W. Xu, E. Jang, H. Michalewski *et al.*, "Multi-game decision transformers," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 921–27 936.

[35] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 785–799.

[36] M. Siebenborn, B. Belousov, J. Huang, and J. Peters, "How crucial is transformer in decision transformer?" *arXiv preprint arXiv:2211.14655*, 2022.

[37] T. Yan, Z. Ruan, Y. Cai, Y. Han, W. Li, and Y. Zhang, "Q-value regularized decision convformer for offline reinforcement learning," *arXiv preprint arXiv:2409.08062*, 2024.

[38] C.-X. Gao, C. Wu, M. Cao, R. Kong, Z. Zhang, and Y. Yu, "Act: empowering decision transformer with dynamic programming via advantage conditioning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 11, 2024, pp. 12 127–12 135.