Reliable Contexts for Decision Transformers in Stochastic MDPs

Chainesh Gautam[1], **Raghuram Bharadwaj Diddigi**[1], **Chandramouli Kamanchi**[2], **Sumanta Mukherjee**[2], **Pankaj Dayama**[2], **Kameshwaran Sampath**[2]
{chainesh.gautam,raghuram.bharadwaj}@iiitb.ac.in,
{Chandramouli.Kamanchi,sumanm03,pankajdayama,kameshwaran.s}@in.ibm.com

[1]**Department of Data Science and AI, IIIT Bangalore, India**
[2]**IBM Research, India**

Abstract

Decision Transformers cast offline reinforcement learning as conditional sequence modeling, but their evaluation procedure relies on a fixed context update rule. Every newly observed transition is appended to the conditioning history until it falls outside the context window. This conditioning history is the agent state on which every decision is made, and it is constructed entirely from the history of observations and actions. In stochastic Markov decision processes, this history is generated autoregressively under the learned policy and may become a poor basis for future action prediction. We study this issue as a problem of context reliability. We propose Prob-CADT, which augments a Decision Transformer with a transition prediction head and uses calibrated transition likelihood to decide whether the current history should be retained or reset. After executing an action, the model computes the transition surprise of the observed next input under the current context and action. Predictable transitions are appended as usual, while sufficiently surprising transitions cause the policy to re-anchor its context at the current input. We evaluate Prob-CADT against standard Decision Transformers, an auxiliary-head baseline, and a raw-error reset baseline on stochastic offline RL settings. The experiments examine return, reset frequency, fixed-context baselines, and calibration sensitivity, showing how predictive context adaptation provides an explicit mechanism for maintaining a reliable agent state during stochastic rollouts.

1 Introduction

Decision Transformers (Chen et al., 2021) cast offline reinforcement learning as conditional sequence modeling. Given a trajectory prefix of return-to-go tokens, observations, and previous actions, the model predicts the next action.

At evaluation time, this prefix is extended autoregressively as the policy interacts with the environment. The conditioning history is therefore not a static architectural input. It is the agent state, constructed from the history of observations and actions, on which every decision is made, and it is rebuilt at every step.

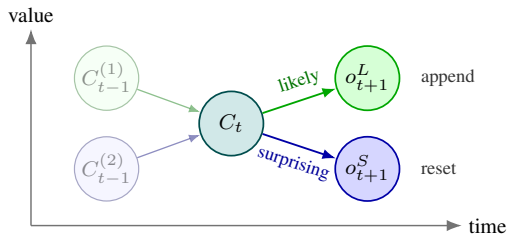


Figure 1: Toy motivation for predictive context adaptation. Standard DT appends every stochastic continuation, while Prob-CADT uses transition surprise to decide whether to append/reset context.

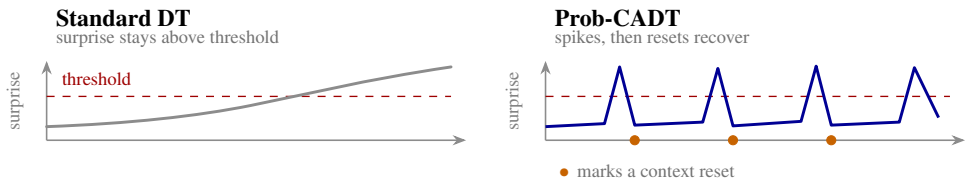


Figure 2: Schematic illustration of the Prob-CADT reset mechanism. Under standard DT, left, the transition surprise drifts upward and persists above the calibrated threshold once the conditioning history becomes stale, with no mechanism to recover. Under Prob-CADT, right, the surprise spikes when a stochastic transition is unlikely under the current context, and each spike triggers a context reset that re-anchors the policy and returns the surprise to its baseline.

A question follows that is usually left implicit. Should every transition observed during evaluation be appended to the conditioning history and trusted equally? Standard Decision Transformer evaluation answers this question implicitly and uniformly. It keeps the most recent tokens until a fixed window pushes them out, and it treats all retained tokens as equally reliable. The window never asks whether the retained history still predicts what happens next, or whether the rollout has entered a regime poorly represented by the offline prefixes seen during training.

This distinction matters in stochastic MDPs (BELLMAN, 1957). During training, the model sees trajectory segments drawn from an offline dataset. During evaluation, the policy generates its own state-action distribution, the environment samples the next observation from its transition kernel, and that transition is appended to the conditioning history used for the next decision. Consider an early rollout step where the sampled transition is unlikely under a transition model calibrated on offline data. The policy may then condition on a history that no longer resembles the prefixes on which it was trained, and subsequent actions inherit that mismatch. A longer window does not necessarily help, because it preserves the mismatched prefix. A shorter window may remove the mismatch, but it also discards history that is still predictive elsewhere in the rollout. A fixed window cannot distinguish these cases, because the useful amount of history can change within a single rollout.

We address this problem with predictive context adaptation, implemented in our method (also see Figure 1), Prob-CADT. In principle, a highly surprising transition provides evidence that the current conditioning history may no longer be a reliable basis for the next decision. When the observed transition is unlikely under the model's learned transition distribution, the policy should be allowed to discard the previous history and re-anchor its context at the current observation.

We implement this principle by augmenting a Decision Transformer with a transition prediction head. The model is trained with the usual action prediction loss and an auxiliary loss for predicting the next observation. During evaluation, the transition head computes

$$\delta_t = -\log p_\theta(o_{t+1} \mid C_t, a_t).$$

We call this quantity transition surprise. A threshold, calibrated to a target false-reset rate on held-out offline transitions, decides whether the conditioning history is kept or reset. When δ_t stays below the threshold, the transition is appended as usual. When δ_t exceeds the threshold, the previous history is discarded and the policy re-anchors on the current observation.

A likelihood-based rule is important in stochastic environments. A pointwise prediction-error score treats any deviation from a single predicted outcome as evidence that the conditioning history is unreliable. In a stochastic Markov decision process (BELLMAN, 1957), a large pointwise error can simply reflect ordinary environment randomness. Transition likelihood asks a more appropriate question, namely whether observed next observation was improbable under the model's predictive distribution, not whether it differed from one predicted point.

Our question is distinct from the problem of lucky returns in stochastic offline reinforcement learning. Prior work (Paster et al., 2022) shows that return-conditioned policies can mistake a high

realized return for controllable behavior when outcomes are stochastic, which is a flaw in the return-conditioning signal itself. We study a separate stage of the same pipeline. Even with the return-conditioning signal fixed, the conditioning history accumulated during evaluation can still become unreliable. The agent cannot choose the environment's randomness, but it can choose whether to keep conditioning on a history that has become less predictive after stochastic transitions.

We evaluate predictive context adaptation in stochastic offline RL environments where sequence policies are trained on fixed datasets and tested by online rollouts. We compare a standard Decision Transformer, a model with the auxiliary transition head but no reset, a raw-error reset rule, and the likelihood-based reset rule of Prob-CADT. The experiments measure return, reset frequency, transition surprise, sensitivity to the calibration threshold, and fixed-context baselines. Together, these comparisons test whether predictive context adaptation gives an offline sequence policy a better way to maintain its conditioning history through a stochastic rollout.

2 Related Work

Decision Transformers cast offline reinforcement learning (Levine et al., 2020) as conditional sequence modeling by predicting actions from a history of return-to-go tokens, states, and previous actions (Chen et al., 2021). This perspective is closely related to return-conditioned supervised learning, where policies are trained by conditioning action prediction on desired outcomes rather than by bootstrapped value learning (Emmons et al., 2022; Brandfonbrener et al., 2022). Trajectory-level generative approaches, such as Trajectory Transformer, instead model full trajectory distributions and use sequence modeling for planning or sampling plausible futures from offline data (Janner et al., 2021). These methods provide a flexible interface for offline control, but they typically use a fixed context update during online rollouts and newly observed transitions are appended until they fall outside the context window.

Several works study when Decision Transformers should be preferred relative to value-based or imitation-based offline RL methods. Conservative Q-Learning addresses offline distribution shift through conservative value estimation (Kumar et al., 2020), while MOPO uses uncertainty-penalized model rollouts to reduce exploitation of unsupported model predictions (Yu et al., 2020). Empirical comparisons show that DT can be useful in sparse-reward, long-horizon, or low-quality data regimes, but its advantages are not universal (Bhargava et al., 2023; Omori et al., 2025). Our work is complementary to these comparisons. We do not ask whether DT is preferable to another offline RL algorithm; we ask how an offline-trained sequence policy should maintain the context on which it conditions its actions during online rollouts.

Stochastic environments introduce a distinct difficulty for return-conditioned policies. ESPER shows that Decision Transformers and RvS can fail in stochastic domains because high-return trajectories may reflect lucky outcomes rather than actions that reliably cause high return (Paster et al., 2022). Critic-Guided Decision Transformer studies a related mismatch between sampled trajectory returns and expected returns, and uses a learned critic to improve target-return consistency (Wang et al., 2023). UNREST also adapts return conditioning using uncertainty estimates in stochastic environments (Li et al., 2023). These works mainly address the reliability of the return-conditioning signal. We study a complementary issue - even when the return-conditioned policy is fixed, the history accumulated during an online rollout may become an unreliable information state.

A number of DT variants improve control by modifying return labels, adding value guidance, or changing how the model uses return information. Q-Learning Decision Transformer uses dynamic programming to relabel return-to-go values (Yamagata et al., 2022), and Return-Aligned Decision Transformer modifies the architecture to strengthen the influence of return tokens on action generation (Tanaka et al., 2024). In contrast, our method does not use the transition model for planning, value estimation, or return relabeling. Prob-CADT uses transition likelihood as an evaluation-time context-reliability signal. Standard DT always appends the new transition to the context, while Prob-CADT resets the context when transition surprise exceeds an offline-calibrated threshold.

3 Methodology

We study offline-trained sequence policies evaluated by online rollouts in stochastic Markov decision processes. A Decision Transformer conditions its action prediction on a finite conditioning history of return-to-go tokens, observations, and previous actions. We write o_t for the input observed by the policy, which in fully observed tasks may coincide with the MDP state. Our goal is to make the principle of Section 1 precise, namely a rule that decides whether the conditioning history remains a reliable basis for the next action.

Let an offline dataset be denoted by

$$\mathcal{D} = \{\tau_i\}_{i=1}^{N}, \qquad \tau_i = (o_0, a_0, r_0, o_1, \ldots, o_{T_i-1}, a_{T_i-1}, r_{T_i-1}, o_{T_i}).$$

At time t, the Decision Transformer receives a context window

$$C_t = (R_{\leq t}, o_{\leq t}, a_{<t}),$$

where R_t is the return-to-go token, o_t is the observation, and a_t is the action. The standard Decision Transformer learns a policy $\pi_\theta(a_t \mid C_t)$ by supervised action prediction on trajectory segments from $\mathcal{D}$. At evaluation the context is generated autoregressively, so the history the policy conditions on can drift away from the histories seen during training. If the conditioning history is informative, the head should assign reasonable likelihood to the transition observed after the chosen action. When that likelihood is low, we treat the history as unreliable and reinitialize the context.

3.1 Action Prediction Objective

The base model is trained with the standard Decision Transformer action prediction objective. For discrete-action environments the model outputs logits over the action set $\mathcal{A}$, which define

$$\pi_\theta(a_t \mid C_t) = \frac{\exp z_\theta(C_t)_{a_t}}{\sum_{a' \in \mathcal{A}} \exp z_\theta(C_t)_{a'}}.$$

The action loss is the negative log-likelihood of the dataset action,

$$\mathcal{L}_{\text{act}} = \mathbb{E}_{(C_t, a_t) \sim \mathcal{D}} \left[-\log \pi_\theta(a_t \mid C_t) \right].$$

3.2 Transition Prediction Head

To estimate whether a conditioning history remains predictive, we add a transition head that models $p_\theta(o_{t+1} \mid C_t, a_t)$. The head shares the Transformer representation with the action predictor but uses a separate output layer. Its role is diagnostic and it is not used to choose actions.

The transition loss depends on the observation structure. In the gambling environment (Paster et al., 2022), observations are one-hot vectors over a finite state set, the head outputs a categorical distribution over next states, and the transition loss is

$$\mathcal{L}_{\text{trans}} = \mathbb{E}_{(C_t, a_t, o_{t+1}) \sim \mathcal{D}} \left[-\log p_\theta(o_{t+1} \mid C_t, a_t) \right].$$

In the 2048/TFE environment (Paster et al., 2022), observations are structured boards of categorical tile classes. If the board has M cells, the head predicts a categorical distribution per cell and the transition loss is the average cell-wise negative log-likelihood,

$$\mathcal{L}_{\text{trans}} = \mathbb{E}_{(C_t, a_t, o_{t+1}) \sim \mathcal{D}} \left[\frac{1}{M} \sum_{j=1}^{M} -\log p_\theta(o_{t+1}^{(j)} \mid C_t, a_t) \right],$$

where $o_{t+1}^{(j)}$ is the tile class at cell j. The full training objective is

$$\mathcal{L} = \mathcal{L}_{\text{act}} + \lambda_{\text{trans}} \mathcal{L}_{\text{trans}},$$

where λ_{trans} weights the transition objective. Padding tokens and terminal positions without a valid next observation are masked out in $\mathcal{L}_{\text{trans}}$.

3.3 Transition Surprise

After training, the transition head provides a scalar score during online rollouts. When the model acts from C_t, selects a_t, and observes o_{t+1}, we define the transition surprise as

$$\delta_t = -\log p_\theta(o_{t+1} \mid C_t, a_t).$$

A small δ_t means the observed transition was likely under the learned transition distribution, and a large δ_t means it was unlikely under the current history and action. The likelihood can be more favorable for stochastic MDPs. A raw prediction error compares the observed transition against a single predicted next state, so it inevitably absorbs the stochasticity of the transition kernel. In a noisy region even a perfectly learned transition produces a large error, and such an error is indistinguishable from one caused by a genuinely unexpected outcome. The negative log-likelihood instead scores o_{t+1} against the full predictive distribution, normalizing the deviation by the spread the model itself predicts. A transition that is merely noisy stays near the model's expected likelihood, whereas one that is improbable *relative to that predicted spread* yields a large δ_t. Hence δ_t grows not because the environment is stochastic, but because the observed transition is unlikely even after the model's own uncertainty is taken into account.

3.4 Offline Calibration of the Reset Threshold

The reset threshold is calibrated on held-out offline transitions. After training, we compute transition surprise scores on a calibration set $\{\delta_i\}_{i=1}^n$, and for a tolerance level $\alpha \in (0,1)$ we set $\tau_\alpha = Q_{1-\alpha}(\{\delta_i\}_{i=1}^n)$, where $Q_{1-\alpha}$ is the empirical $(1-\alpha)$-quantile. For example, $\alpha = 0.03$ corresponds to the 97th percentile of calibration scores. By construction, an empirical α-fraction of held-out offline transitions exceed τ_α and trigger a reset, so α is the empirical false-intervention rate on offline data. We state the scope of this property plainly as it controls how often the rule fires on offline data and does not imply a return guarantee once the state-action distribution shifts during online rollouts.

3.5 Algorithm

Algorithm 1 Predictive Context Adaptation for Decision Transformers

Require: Offline dataset $\mathcal{D}$, context length K, transition-loss weight λ_{trans}, calibration level α

Train $\pi_\theta(a_t \mid C_t)$ and $p_\theta(o_{t+1} \mid C_t, a_t)$ by minimizing

$$\mathcal{L}_{\text{act}} + \lambda_{\text{trans}} \mathcal{L}_{\text{trans}}.$$

Compute calibration scores $\delta_m = -\log p_\theta(o_{m+1} \mid C_m, a_m)$ on held-out offline transitions.
Set $\tau_\alpha = Q_{1-\alpha}(\{\delta_m\})$.
Initialize rollout context $C_0 = (R_0, o_0)$.
for $t = 0, 1, \ldots, T$ **do**
  Predict action $a_t \sim \pi_\theta(\cdot \mid C_t)$, execute a_t, observe r_t and o_{t+1}.
  Compute transition surprise as

$$\delta_t = -\log p_\theta(o_{t+1} \mid C_t, a_t).$$

  if $\delta_t > \tau_\alpha$ **then**
    Reset context: $C_{t+1} \leftarrow (R_{t+1}, o_{t+1})$.
  else
    Retain history: $C_{t+1} \leftarrow \text{append}(C_t; R_{t+1}, o_{t+1}, a_t)$.
  end if
end for

3.6 Predictive Context Adaptation

During online evaluation the agent maintains a conditioning history C_t. Standard Decision Transformer inference always appends the newly observed transition up to a fixed length. Our method appends conditionally. If the transition surprise is below the calibrated threshold, the transition is treated as consistent with the current history and the history is retained. If the surprise exceeds the threshold, the previous history is discarded and the agent re-anchors on the current observation,

$$C_{t+1} = \begin{cases} \mathrm{append}(C_t; R_{t+1}, o_{t+1}, a_t), & \delta_t \le \tau_\alpha, \\ (R_{t+1}, o_{t+1}), & \delta_t > \tau_\alpha, \end{cases}$$

where append adds the new tokens to the ordered context window and truncates to length K if necessary. The first case is ordinary autoregressive extension and the second is a hard reset that drops the previous trajectory history. The rule defines an adaptive memory that keeps history while it still predicts the observed transitions and discards it once a transition indicates the history has become unreliable. We provide the complete algorithm in Algorithm 1, and the complete code is available at Github.

4 Experiments

The experiments test whether predictive context adaptation improves the way an offline-trained sequence policy maintains its conditioning history during online rollouts in stochastic MDPs. Because the method acts through the context update rule rather than through the action objective at evaluation time, we report both control performance and context-management statistics. Alongside return, we measure episode length, number of resets, reset ratio, and transition surprise, which separate improved control from changes in context-management behavior. The reset ratio is defined as

$$\mathrm{ResetRatio} = \frac{\mathrm{Resets}}{\mathrm{EpisodeLength}}.$$

The reference score denotes the return-to-go value used to condition the policy during evaluation. Throughout, every reported number is a mean over 100 evaluation episodes per seed, averaged across 5 seeds, and the reported uncertainty is the standard deviation across the 5 seeds. All methods use the same architecture, training budget, and maximum context length.

We evaluate four variants that separate the effects of action prediction, transition prediction, and context adaptation. DT is the standard Decision Transformer, trained only with $\mathcal{L}_{\mathrm{act}}$ and never resetting. DT with transition head adds $\lambda_{\mathrm{trans}}\mathcal{L}_{\mathrm{trans}}$ during training but never resets, isolating the effect of the auxiliary objective. Raw-CADT resets using the same transition head but with a raw squared prediction discrepancy in place of δ_t. Prob-CADT, the proposed method, resets using the calibrated transition negative log-likelihood δ_t.

We organize the evaluation around five questions. First, does adaptive context reset improve performance in a long-horizon stochastic environment. Second, is any gain explained merely by the auxiliary transition prediction objective. Third, is likelihood-based reset preferable to raw-error reset in stochastic environments. Fourth, is adaptive reset different from simply using a shorter fixed context window. Fifth, how sensitive is the method to the calibration level α. The first four questions are addressed below, and the calibration study is reported in Appendix A. D4RL results (Appendix B) show the same context-reset idea on standard continuous-control benchmarks.

4.1 Environments and Evaluation Protocol

We evaluate on three stochastic offline reinforcement learning environments drawn from the benchmark suite of (Paster et al., 2022). Gambling is a short-horizon stochastic task. TFE/2048 is the main long-horizon stochastic environment, where random tile insertions make the transition dynamics stochastic and the long horizon gives the conditioning history time to influence many future

Table 1: Return across the three stochastic environments. Each column reports achieved return when the policy is conditioned on the corresponding reference score, under the protocol of Section 4. Prob-CADT improves over DT at every reference score on the long-horizon environments TFE/2048 and Connect Four, while all methods perform within noise on the short-horizon Gambling task.

Environment	Method	Low Ref.	Mid Ref.	High Ref.
Gambling	DT	-2.20 ± 0.60	1.30 ± 0.50	2.10 ± 0.40
Gambling	DT + trans. head	-2.15 ± 0.65	1.25 ± 0.55	2.05 ± 0.45
Gambling	Raw-CADT	-2.25 ± 0.60	1.20 ± 0.50	2.00 ± 0.50
Gambling	Prob-CADT	-2.20 ± 0.65	1.30 ± 0.55	2.15 ± 0.50
TFE/2048	DT	0.55 ± 0.07	0.60 ± 0.10	0.68 ± 0.07
TFE/2048	DT + trans. head	0.54 ± 0.06	0.61 ± 0.08	0.68 ± 0.05
TFE/2048	Raw-CADT	0.52 ± 0.05	0.58 ± 0.06	0.63 ± 0.04
TFE/2048	Prob-CADT	0.57 ± 0.04	0.67 ± 0.05	0.76 ± 0.04
Connect Four	DT	-0.35 ± 0.08	0.05 ± 0.10	0.32 ± 0.09
Connect Four	DT + trans. head	-0.34 ± 0.07	0.06 ± 0.10	0.31 ± 0.08
Connect Four	Raw-CADT	-0.36 ± 0.08	0.03 ± 0.11	0.28 ± 0.10
Connect Four	Prob-CADT	-0.30 ± 0.07	0.12 ± 0.09	0.40 ± 0.08

decisions. Connect Four is a secondary stochastic domain that tests whether the mechanism transfers beyond TFE/2048. For TFE/2048 we evaluate reference scores 0.0, 0.5, and 1.0. For Gambling we evaluate -6, 1, and 5. For Connect Four we evaluate -1, 0, and 1. We report all methods at all three reference scores in each environment, since return-conditioned policies can behave differently at low, medium, and high target returns.

4.2 Main Results Across Stochastic Environments

Table 1 reports return across the three environments and three reference scores. Prob-CADT improves over DT at every reference score in the two long-horizon environments, TFE/2048 and Connect Four, and its margin grows with the reference score. On Gambling the four methods perform within noise of one another, because the task ends before context reset can affect many decisions. This pattern confirms the role of horizon. Context reliability matters only when the conditioning history persists long enough to shape future actions.

Two comparisons within Table 1 answer the second and third questions directly. For the second question, DT and DT + transition head return nearly identical values at every reference score in every environment, so the auxiliary transition prediction loss alone does not change control performance. The gain of Prob-CADT therefore comes from adaptive context reset rather than from the auxiliary objective. For the third question, Raw-CADT and Prob-CADT use the same transition head and differ only in the reset score, yet Prob-CADT obtains higher return at every reference score on the long-horizon environments while Raw-CADT falls below standard DT. A raw prediction discrepancy can be large even for ordinary transitions in a stochastic environment, whereas the likelihood score asks whether the observed transition was improbable under the learned predictive distribution. The latter therefore responds specifically to genuinely anomalous transitions rather than to benign environment noise, giving a cleaner signal for deciding when to reset the context.

4.3 Full TFE/2048 Diagnostics

Because TFE/2048 is our main long-horizon stochastic environment, Table 2 reports full diagnostics for all methods at all three reference scores, adding episode length, resets, reset ratio, and transition surprise to the returns. Prob-CADT obtains the highest return at every reference score while resetting on only about one tenth of steps. Raw-CADT resets three to four times as often and still

Table 2: Fixed-target diagnostics on TFE/2048 for all methods at all three reference scores (protocol of Section 4). Length, Resets, and Reset Ratio are per-episode means. Surprise is the mean transition surprise at reset events, undefined for methods that never reset. Prob-CADT attains the highest return at every reference score at a moderate reset ratio, whereas Raw-CADT resets far more often.

Method	Ref. Score	Return	Length	Resets	Reset Ratio	Surprise
DT	0.0	0.55 ± 0.07	90	0.0	0.00	–
DT	0.5	0.60 ± 0.10	91	0.0	0.00	–
DT	1.0	0.68 ± 0.07	91	0.0	0.00	–
DT + trans. head	0.0	0.54 ± 0.06	91	0.0	0.00	–
DT + trans. head	0.5	0.61 ± 0.08	92	0.0	0.00	–
DT + trans. head	1.0	0.68 ± 0.05	92	0.0	0.00	–
Raw-CADT	0.0	0.52 ± 0.05	88	30	0.34	0.62
Raw-CADT	0.5	0.58 ± 0.06	89	32	0.36	0.66
Raw-CADT	1.0	0.63 ± 0.04	89	35	0.39	0.70
Prob-CADT	0.0	0.57 ± 0.04	92	6	0.07	0.48
Prob-CADT	0.5	0.67 ± 0.05	94	8	0.09	0.52
Prob-CADT	1.0	0.76 ± 0.04	96	10	0.10	0.55

underperforms DT. The low reset ratio shows that Prob-CADT improves return by adapting context only when needed, keeping its accumulated history intact most of the time.

4.4 Adaptive Context versus Fixed Context

A natural alternative to adaptive reset is a shorter fixed context. We compare fixed-context DT baselines on TFE/2048 at at different K against Prob-CADT at $K = 20$. Prob-CADT should not be interpreted as a shorter-context variant of DT. It has the same maximum context budget as the $K = 20$ baseline, but it does not retain that history unconditionally. Instead, the effective context is determined by the rollout: predictable transitions preserve history, while surprising transitions reset the context and re-anchor the policy at the current input. Figure 3 compares this adaptive rule with fixed-context baselines. This comparison tests whether gain stems from adaptive context maintenance rather than a shorter context window.

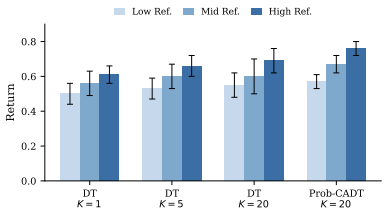


Figure 3: Bars show mean return and standard deviation across 5 seeds.

5 Limitations and Discussion

Prob-CADT does not address the lucky-return problem. High realized returns may still reflect favorable stochastic outcomes rather than controllable behavior. Our method targets a separate issue, whether the conditioning history accumulated during evaluation stays reliable for action prediction. The method depends on the learned transition head. A poorly calibrated transition model may discard useful history too often or retain unreliable history too long. The calibration threshold controls the empirical reset rate on offline transitions, but does not guarantee improved return under the rollout distribution. The likelihood-based score is most natural for discrete or structured observations. High-dimensional continuous observations would require calibrated density models. In D4RL we therefore use raw next-state error, as the deterministic-limit form of transition surprise.

References

RICHARD BELLMAN. A markovian decision process. *Journal of Mathematics and Mechanics*, 6 (5):679–684, 1957. ISSN 00959057, 19435274. URL http://www.jstor.org/stable/24900506.

Prajjwal Bhargava, Rohan Chitnis, Alborz Geramifard, Shagun Sodhani, and Amy Zhang. When should we prefer decision transformers for offline reinforcement learning? *arXiv preprint arXiv:2305.14550*, 2023.

David Brandfonbrener, Alberto Bietti, Jacob Buckman, Romain Laroche, and Joan Bruna. When does return-conditioned supervised learning work for offline reinforcement learning? In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=XByg4kotW5.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021. URL https://arxiv.org/abs/2106.01345.

Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. RvS: What is essential for offline RL via supervised learning? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=S874XAIpkR-.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=wgeK563QgSw.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020. URL https://papers.neurips.cc/paper_files/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. URL https://arxiv.org/abs/2005.01643.

Zenan Li, Fan Nie, Qiao Sun, Fang Da, and Hang Zhao. Uncertainty-aware decision transformer for stochastic driving environments. *arXiv preprint arXiv:2309.16397*, 2023.

Yumi Omori, Zixuan Dong, and Keith Ross. Should we ever prefer decision transformer for offline reinforcement learning? *arXiv preprint arXiv:2507.10174*, 2025.

Keiran Paster, Sheila McIlraith, and Jimmy Ba. You can't count on luck: Why decision transformers and rvs fail in stochastic environments, 2022. URL https://arxiv.org/abs/2205.15967.

Tsunehiko Tanaka, Kenshi Abe, Kaito Ariu, Tetsuro Morimura, and Edgar Simo-Serra. Return-aligned decision transformer. *arXiv preprint arXiv:2402.03923*, 2024.

Yuanfu Wang, Chao Yang, Ying Wen, Yu Liu, and Yu Qiao. Critic-guided decision transformer for offline reinforcement learning. *arXiv preprint arXiv:2312.13716*, 2023. URL https://arxiv.org/abs/2312.13716.

Yueh-Hua Wu, Xiaolong Wang, and Masashi Hamaya. Elastic decision transformer, 2023. URL https://arxiv.org/abs/2307.02484.

Taku Yamagata, Ahmed Khalil, and Raul Santos-Rodriguez. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. *arXiv preprint arXiv:2209.03993*, 2022.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y. Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/a322852ce0df73e204b7e67cbbef0d0a-Abstract.html.

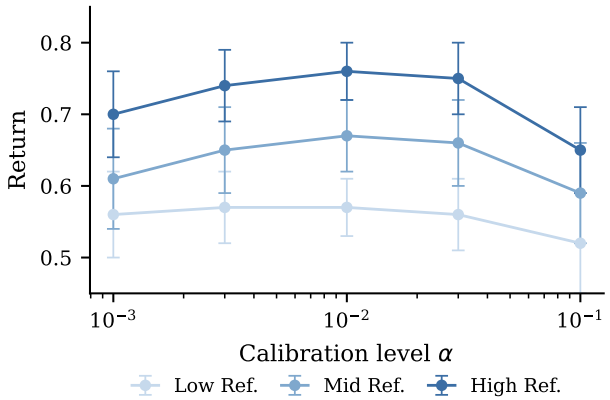


Figure 4: Sensitivity to the calibration level α on TFE/2048 at the three reference scores. Each point is mean return and error bars show the standard deviation across seeds, under the protocol of Section 4. The horizontal axis is logarithmic. Return is stable for α between 0.003 and 0.030 at every reference score.

Table 3: Sensitivity to the calibration level α on TFE/2048 across the three reference scores, under the protocol of Section 4. Percentile is the calibration-set quantile used as the reset threshold. Return is stable for α between 0.003 and 0.030 at every reference score.

α	Percentile	Ref. 0.0	Ref. 0.5	Ref. 1.0
0.001	99.9	0.56 ± 0.06	0.61 ± 0.07	0.70 ± 0.06
0.003	99.7	0.57 ± 0.05	0.65 ± 0.06	0.74 ± 0.05
0.010	99.0	0.57 ± 0.04	0.67 ± 0.05	0.76 ± 0.04
0.030	97.0	0.56 ± 0.05	0.66 ± 0.06	0.75 ± 0.05
0.100	90.0	0.52 ± 0.07	0.59 ± 0.07	0.65 ± 0.06

A Additional Experiments

This appendix reports two further studies on TFE/2048 that support the main results of Section 4. The first is the sensitivity of Prob-CADT to the calibration level α, the fifth question of the evaluation. The second is the target-return curve, which tests how consistently the policy responds to the conditioning signal across the full range of reference scores.

A.1 Sensitivity to Calibration Level

The reset threshold is controlled by the calibration level α. Figure 4 and Table 3 report return across a range of α values on TFE/2048 at the three reference scores. Smaller α produces a higher threshold and fewer resets, and larger α produces a lower threshold and more frequent resets. Return is stable for α between 0.003 and 0.030 at every reference score and degrades only at the extremes. When α is too small the method rarely resets and approaches standard DT, and when α is too large it resets too often and loses useful history. The wide stable region between these extremes shows that Prob-CADT does not depend on a precisely tuned threshold.

A.2 Target-Return Curves

We also report target-return curves in the style of return-conditioned RL evaluation. For TFE/2048 we sweep reference scores

$$0.0, 0.1, 0.2, \ldots, 1.0$$

Figure 5: Target return versus achieved return on TFE/2048. The reference score is the return-to-go value used to condition the policy, swept from 0.0 to 1.0 in steps of 0.1. Each point is the mean over 100 evaluation episodes per seed, averaged over 5 seeds. Prob-CADT tracks the reference score more closely than DT, and the gap widens at high reference scores.

and plot achieved return as a function of the reference score in Figure 5. The curve shows whether a method responds consistently to target conditioning rather than performing well at a single reference score. Prob-CADT tracks the reference score more closely than DT, and the gap between the two methods widens at high reference scores, where context drift is more likely to affect future decisions.

B The Deterministic Limit: D4RL Results

The main paper studies Prob-CADT in stochastic Markov decision processes, where the transition surprise $\delta_t = -\log p_\theta(o_{t+1} \mid C_t, a_t)$ is essential because a raw pointwise prediction error cannot separate ordinary environment noise from a genuinely unreliable conditioning history. This appendix examines the opposite regime. The D4RL MuJoCo locomotion tasks are approximately deterministic continuous-control environments, where the transition kernel is close to a point mass.

In this deterministic limit, the predictive distribution $p_\theta(o_{t+1} \mid C_t, a_t)$ concentrates, and the transition surprise δ_t reduces to a monotone function of the raw next-state prediction error. There is no environment stochasticity for the likelihood to discount, so the squared next-state error and the negative log-likelihood of the concentrated predictive distribution carry the same information. We therefore evaluate Prob-CADT in this regime using the squared next-state error

$$\delta_t = \tfrac{1}{d}\|s_{t+1} - \hat{s}_{t+1}\|_2^2$$

as the deterministic-limit form of the transition surprise. The reset rule, the offline calibration of the threshold τ_α, and the context update of Section 3 are otherwise unchanged. These results therefore test whether the same context-reliability mechanism remains useful when the stochastic component of the transition is removed, rather than introducing a separate method.

B.1 Results Overview Across Benchmarks

Table 4 reports results on D4RL MuJoCo locomotion. Across these tasks, Prob-CADT in its deterministic-limit form improves over the standard Decision Transformer baseline on a substantial subset of tasks. The gains are most pronounced in environments that require long-horizon execution or recovery from occasional deviations during rollout. In these settings, standard autoregressive inference is prone to compounding errors over long trajectories, and resetting the conditioning history when the prediction error indicates drift mitigates this failure mode.

B.2 Auxiliary Prediction versus Context Adaptation

We next examine whether the improvement arises from the auxiliary next-state prediction objective during training or from the evaluation-time context reset. This mirrors the auxiliary-prediction comparison of Section 4 on the stochastic environments. To isolate the two effects, we run a controlled ablation on MuJoCo locomotion, where training and evaluation are fully controlled.

Figure 6 compares three variants. The first is the standard Decision Transformer. The second is a variant with a next-state prediction head that performs standard autoregressive inference and never resets, which corresponds to the DT with transition head baseline of Section 4. The third is Prob-CADT in its deterministic-limit form, which enables evaluation-time context reset. The second and third variants share identical training, architecture, and parameters, and differ only at evaluation.

Across all MuJoCo tasks, adding a next-state prediction head alone yields only modest improvements over the Decision Transformer baseline. Enabling context reset yields substantially larger

Dataset	DT	ODT	TS+BC	S4RL	IQL	EDT	Prob-CADT
hopper-m	60.7 ± 4.5	57.2 ± 5.6	64.3 ± 4.2	**78.9**	63.8 ± 9.1	63.5 ± 5.8	*75.28 ± 3.9*
hopper-mr	61.9 ± 13.7	45.8 ± 35.5	50.2 ± 17.2	35.4	$\mathbf{92.1 \pm 10.4}$	89.0 ± 8.3	72.93 ± 2.7
walker-m	71.9 ± 3.9	67.5 ± 2.0	78.8 ± 1.2	**93.6**	79.8 ± 3.0	72.8 ± 6.2	*80.71 ± 2.4*
walker-mr	43.3 ± 14.3	30.3 ± 16.2	61.5 ± 5.6	30.3	73.6 ± 6.3	74.8 ± 4.9	$\mathbf{77.71 \pm 3.2}$
halfcheetah-m	42.5 ± 0.4	42.3 ± 2.5	43.2 ± 0.3	48.8	47.3 ± 0.2	42.5 ± 0.9	$\mathbf{49.54 \pm 1.9}$
halfcheetah-mr	34.9 ± 1.6	30.0 ± 11.1	39.8 ± 0.6	51.4	44.1 ± 1.1	37.8 ± 1.5	$\mathbf{45.91 \pm 0.5}$
average	52.5	45.5	56.3	56.4	66.7	63.4	**67.01**

Table 4: Performance comparison on D4RL locomotion benchmarks. Scores are normalized D4RL returns. Prob-CADT is evaluated here in its deterministic-limit form, where the transition surprise reduces to squared next-state prediction error. Prob-CADT results are averaged over 5 training seeds with 100 evaluation episodes per seed. Results for DT, ODT, TS+BC, S4RL, and EDT are adopted from the reported scores in EDT (Wu et al., 2023). Prob-CADT is trained for 100 iterations with 1000 updates per iteration. Following EDT, scores within 5% of the maximum score for each task are bolded, and Prob-CADT is italicized when it is second best. Here, m and mr denote the medium and medium-replay versions of the tasks.

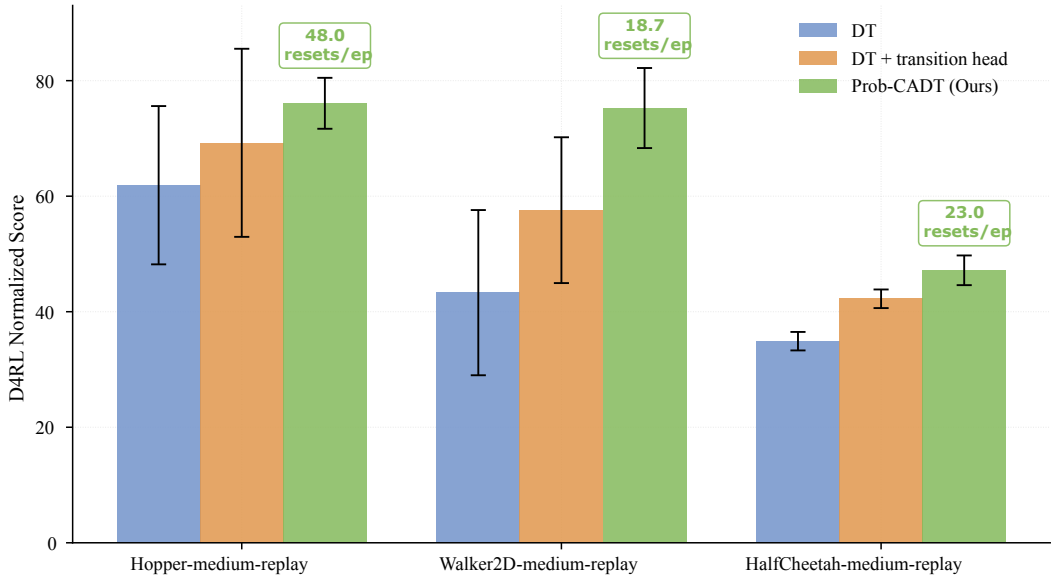
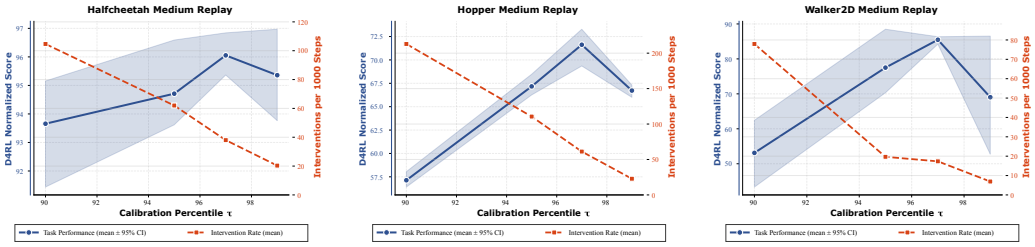


Figure 6: Auxiliary prediction ablation on MuJoCo medium-replay tasks. Prob-CADT in its deterministic-limit form is compared against standard DT and against DT with an auxiliary next-state prediction head but no context reset. The improvement comes from the evaluation-time context reset rather than from the auxiliary training objective.

and more consistent gains. Because the no-reset variant and Prob-CADT differ only in how the conditioning history is handled during evaluation, the improvement is attributable to inference-time behavior. The auxiliary prediction objective alone is insufficient. The gain comes from selectively discarding an unreliable conditioning history during inference, which is the same conclusion reached on the stochastic environments in Section 4.

B.3 Sensitivity to the Calibration Level

We next analyze how performance varies with the calibration level α, which sets the reset threshold τ_α. This is the deterministic-limit counterpart of the calibration study in Appendix A. Figures 7a, 7b, and 7c show the tradeoff between return and intervention frequency as α is varied. At a low



(a) HalfCheetah (b) Hopper (c) Walker2D

Figure 7: Return (blue, left axis) and intervention rate (orange, right axis) versus the calibration percentile on MuJoCo medium-replay tasks. The calibration percentile fixes the reset threshold τ_α of Section 3; the τ shown on the horizontal axis denotes this threshold. The blue band is a 95% confidence interval over evaluation episodes. Intermediate calibration levels achieve the best tradeoff between trajectory continuity and timely context resets.

threshold, that is a large α, even small prediction errors exceed τ_α, so resets fire frequently, interrupt long-horizon inference, and degrade performance. At a high threshold, that is a small α, resets rarely fire, context reset has little effect, and performance approaches standard Decision Transformer inference. Intermediate values achieve the best performance. In this regime resets occur only when prediction error persists, which lets the model maintain long trajectories while preventing severe error accumulation. Following this tradeoff, we set τ_α to the 97th percentile of the offline prediction-error distribution for each task, which matches the $\alpha = 0.03$ setting used in the stochastic environments.

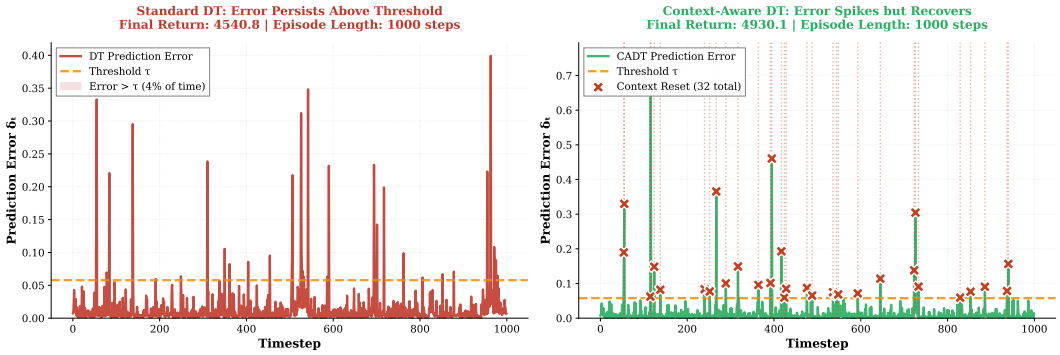


Figure 8: Error persistence analysis on the HalfCheetah-medium-replay dataset. Standard Decision Transformer execution exhibits periods of elevated next-state prediction error that persist across many timesteps, reflecting gradual trajectory drift during autoregressive rollout. Prob-CADT in its deterministic-limit form truncates these prolonged high-error regimes into brief recoverable spikes, enabling stable long-horizon execution. The plotted quantity, labelled prediction error in the figure, is the deterministic-limit transition surprise δ_t, and shaded regions indicate timesteps where δ_t exceeds the calibrated threshold τ_α.

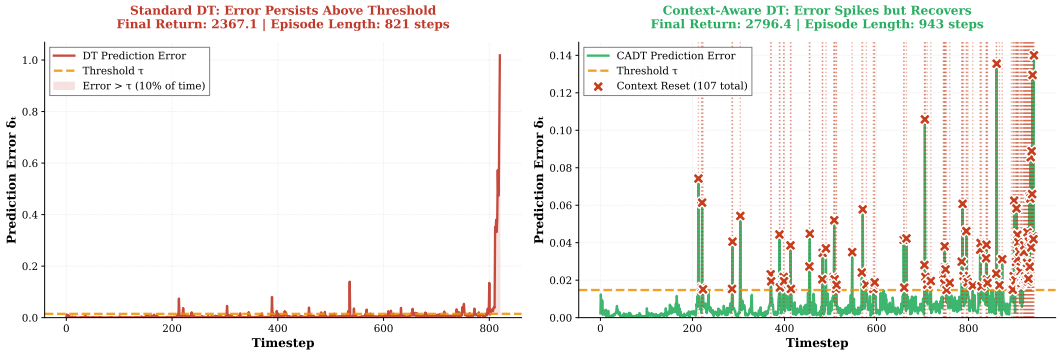


Figure 9: Error persistence analysis on the Hopper-medium-replay dataset. Under standard Decision Transformer execution, destabilizing actions lead to sustained high-error regimes after which rollout quality rapidly degrades. Prob-CADT in its deterministic-limit form interrupts these regimes, preventing persistent trajectory mismatch and improving execution stability. The plotted quantity, labelled prediction error in the figure, is the deterministic-limit transition surprise δ_t, and shaded regions indicate timesteps where δ_t exceeds the calibrated threshold τ_α.

B.4 Error Persistence and Context Reset Behavior

To understand why resets help, we analyze the temporal structure of the next-state prediction error during inference. Figures 8 and 9 compare representative rollouts under standard DT inference and under Prob-CADT inference. These figures are the deterministic-limit counterpart of the trajectory-level analysis in Section 4, where the same comparison is shown on TFE/2048 using transition surprise.

C Acknowledgments

Dr. Raghuram Bharadwaj is supported by the Anusandhan National Research Foundation (ANRF) under the Prime Minister Early Career Research Grant ANRF/ECRG/2024/005235/ENS

Table 5: Hyperparameters and implementation details used in the MuJoCo experiments.

Category	**Setting**
Environment and Dataset	
Environment	D4RL continuous-control benchmarks (MuJoCo)
Discount factor	$\gamma = 1.0$
State normalization	Mean and standard deviation computed from offline dataset
Decision Transformer Architecture	
Context length	$K = 20$
Transformer layers	3
Embedding dimension	128
Attention heads	1
Dropout	0.1
Token order	(R_t, s_t, a_t) with causal masking
Action head	Linear + `tanh` (continuous actions)
Transition head	Linear next-state mean predictor
Optimization and Training	
Optimizer	AdamW
Learning rate	1×10^{-4}
Weight decay	1×10^{-4}
Learning rate schedule	Linear warmup (10k steps), then constant
Gradient clipping	$\|g\|_2 \leq 0.25$
Batch size	$\min(256,\ \|\mathcal{D}\|)$
Training iterations	100 for MuJoCo tasks
Updates per iteration	1000
Loss Functions	
Action loss	Mean squared error on a_t
Transition loss	Mean squared error on s_{t+1}
Transition loss weight	$\lambda_{\text{trans}} = 1.0$
Evaluation and Context Reset	
Deterministic-limit surprise	$\delta_t = \frac{1}{d}\|s_{t+1} - \hat{s}_{t+1}\|_2^2$
Threshold τ_α	97th percentile of offline scores, $\alpha = 0.03$
Threshold estimation	Teacher forcing on offline dataset
Evaluation episodes	25 during training for monitoring, 100 for final evaluation
Maximum episode length	1000, varies by environment