Decision Mixer: Integrating Long-term and Local Dependencies via Dynamic Token Selection for Decision-Making

Hongling Zheng [1] **Li Shen** [2] **Yong Luo** [1] **Deheng Ye** [3] **Bo Du** [1] **Jialie Shen** [4] **Dacheng Tao** [5]

Abstract

The Conditional Sequence Modeling (CSM) paradigm, benefiting from the transformer's powerful distribution modeling capabilities, has demonstrated considerable promise in offline Reinforcement Learning (RL) tasks. Depending on the task's nature, it is crucial to carefully balance the interplay between inherent local features and long-term dependencies in Markov decision trajectories to mitigate potential performance degradation and unnecessary computational overhead. In this paper, we propose Decision Mixer (DM), which addresses the conflict between features of different scales in the modeling process from the perspective of dynamic integration. Drawing inspiration from conditional computation, we design a plug-and-play dynamic token selection mechanism to ensure the model can effectively allocate attention to different features based on task characteristics. Additionally, we employ an auxiliary predictor to alleviate the short-sightedness issue in the autoregressive sampling process. DM achieves state-of-the-art performance on various standard RL benchmarks while requiring significantly fewer computational resources, offering a viable solution for building efficient and scalable RL foundation models. Code is available at here.

1. Introduction

Transformer (Vaswani, 2017) is widely regarded for its capacity to capture complex data distributions and long-term temporal dependencies, becoming a foundational architecture in fields such as Natural Language Processing (NLP) (Brown et al., 2020; Achiam et al., 2023; Xu et al., 2021) and Computer Vision (CV) (Liu et al., 2021; Si et al., 2022). Inspired by this success, Decision Transformer (DT) (Chen et al., 2021; Li et al., 2023) and its variants (Lee et al., 2022; Xie et al., 2023) introduce the transformer to the field of offline Reinforcement Learning (RL), demonstrating its powerful capabilities in Conditional Sequence Modeling (CSM). Specifically, DT integrates cumulative rewards, states, and actions into a tuple and trains on offline datasets autoregressively to output appropriate actions. DT's limitation lies in overlooking the inherent local associations between adjacent timestep tokens in offline RL data, which are crucial for the model's learning of transition and reward functions. Decision Convformer (DC) (Kim et al., 2024) employs causal convolution filters instead of attention modules for data modelling to address this limitation and outperforms DT on specific tasks. However, its overemphasis on capturing local features leads to suboptimal performance on long-term sequence modeling tasks. Such tasks typically do not adhere to the standard Markov property indicated by DC but rather follow a non-standard Markov process (Ching et al., 2013), where the current state may depend on multiple previous time steps. Consequently, a fixed convolution range may fail to adapt to the specific characteristics of different tasks (Figure 1). Furthermore, existing works rarely consider the presence of suboptimal segments within trajectories, and trajectory selection and concatenation would substantially improve the efficiency of

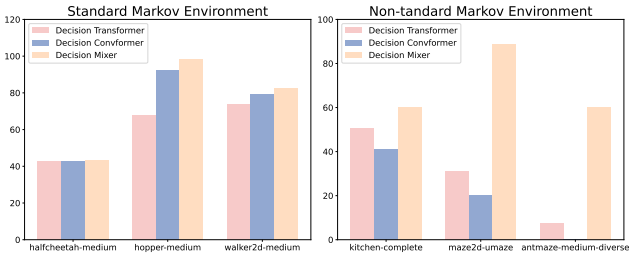

Figure 1. The evaluation results for DT, DC, and DM in tasks with standard Markov properties and non-standard Markov properties. DM consistently secures the top performance across all tested environments, showcasing its superiority.

[1]School of Computer Science, National Engineering Research Center for Multimedia Software and Hubei Key Laboratory of Multimedia and Network Communication Engineering, Wuhan University, Wuhan, China [2]School of Cyber Science and Technology, Shenzhen Campus of Sun Yat-sen University, Shenzhen, China [3]Tencent Inc., Shenzhen, China [4]Department of Computer Science, City, University of London, London, United Kingdom [5]Nanyang Technological University, Singapore, Singapore. Correspondence to: Li Shen <shenli6@mail.sysu.edu.cn>, Yong Luo <luoyong@whu.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

both attention and convolution operations.

Although increasing the number of parameters in foundation models has demonstrated success across a broad range of tasks (Bommasani et al., 2021; Muennighoff et al., 2023), this scaling law (Kaplan et al., 2020; Ye et al., 2020a;b) remains insufficiently explored in the context of deep RL. Blindly expanding the architecture of DT not only escalates computational demands but may also lead to a decline in performance. Conditional computation algorithms (Ainslie et al., 2023; Lei et al., 2023), which statically or dynamically allocate the number of parameters used during model training, appear promising for achieving computational efficiency while enhancing model performance. However, whether such architectures can better capture the inherent Markov patterns in offline RL datasets while preserving the ability to learn long-term temporal dependencies remains unclear.

In this paper, we propose Decision Mixer (DM), a low-complexity architecture that dynamically balances long-term dependency features and local Markov features through a token selection mechanism. For tokens in a given sequence, DM dynamically selects whether a token should participate in the current layer's attention calculation or be passed directly to the next layer. Specifically, drawing inspiration from the Mixture of Experts (MoE) (Zhou et al., 2022), DM utilizes a router to assign weights to each token in the sequence. At the same time, a hypernetwork determines the number of tokens k to be selected based on data features. The final selection is achieved by combining these two components with a weight-based top-k mechanism. This dynamic token selection mechanism ensures that the model can allocate attention to long-term dependencies and local features based on the characteristics of the task. After being selected, tokens are fed into the attention layer in their original order, which can be viewed as a concatenation mechanism and allows the model to learn features of optimal trajectories from suboptimal ones. The unselected tokens are passed through the residual link and then combined with the tokens processed through attention. During testing, since the entire sequence cannot be obtained at once in autoregressive sampling, we have designed an auxiliary network that directly predicts at the token level whether the token should participate in the attention computation.

The main contributions of this work are as follows:

- We reconsidered the trade-off between capturing long-term dependencies and extracting local Markov features in CSM methods from an experimental perspective, proposing a selection, stitching, and computation mechanism.
- Drawing inspiration from MOE architecture, we significantly reduced the computational burden by dynamically selecting important tokens at each layer of the transformer, providing a direction for exploring the scaling law in offline RL.
- We demonstrate the effectiveness of DM through intensive experiments on a broad spectrum of benchmarks, highlighting its competitive performance in Offline RL scenarios.

2. Preliminary

2.1. Conditional Computation

The transformer architecture has become a cornerstone in driving the artificial intelligence revolution, but its high computational cost has sparked significant interest in improving efficiency. The concept of conditional computation (Bengio et al., 2015) has been proposed as a promising approach to address the abovementioned issues, with learned mechanisms determining when and how computation is expended. Mixture of Experts (MOE) (Zhou et al., 2022) is a representative method of conditional computation, where tokens are routed to one of several experts. This sparse activation architecture maintains a constant total computation cost while expanding the number of parameters (Dai et al., 2024; Raposo et al., 2024; Zheng et al., 2025). Some other work (Schuster et al., 2022; Ainslie et al., 2023) focuses on the design of an early exit mechanism within the transformer, where the model learns to decide when to terminate the computation for a given token, allowing the token to skip any remaining transformer layers once the exit decision is made. Our approach can be viewed as a token-level sparse activation strategy. In this strategy, a routing mechanism similar to that in MoE, along with an hypernetwork, determines whether a specific token should pass through the transformer layers or skip them.

Our approach is also related to token dropping, which was initially proposed to reduce BERT inference costs (Press et al., 2022; Wang et al., 2021) and later adapted to improve training efficiency (Hou et al., 2022). Random-LTD (Yao et al., 2022) further advanced this idea by introducing random-layer token dropping combined with learning rate scheduling. While prior work focuses on static efficiency strategies for vision and language tasks (Zhong et al., 2023; Liu et al., 2024), they often lack dynamic adaptation, which can lead to semantic disruption. In contrast, DM dynamically selects tokens using a router and hypernetwork, aligning with the Markovian nature of offline RL for improved performance. Its plug-and-play design and synchronized training offer greater flexibility than existing methods.

2.2. Conditional Sequence Modeling for Offline RL

In contrast to online RL, offline RL (Agarwal et al., 2020; Wei et al., 2021; Qu et al., 2023) focuses on training models and performing trial-and-error using offline data without

environmental interaction to arrive at appropriate strategies. Recently, conditional sequence modeling for RL (Hu et al., 2024b; Brandfonbrener et al., 2022), represented by the transformer architecture, has further demonstrated the advantages of data-driven policy learning. DT (Chen et al., 2021) is trained on an offline dataset of triplets encapsulating return-to-go $\hat{r}_t$, state s_t, and action a_t, and outputs the optimal action. The $\hat{r}_t$ token quantifies the cumulative reward from the current time step to the end of the episode. During training, DT processes a trajectory sequence τ_t in an auto-regressive manner, which encompasses the most recent K-step historical context:

$$\tau_t = (\hat{r}_{t-K+1}, s_{t-K+1}, a_{t-K+1}, \ldots, \hat{r}_t, s_t, a_t) \quad (1)$$

The prediction head associated with a state token s_t is trained to predict the corresponding action a_t. Subsequent work has made various improvements to DT, including prompt tuning (Xu et al., 2022; Zheng et al., 2024), trajectory concatenation (Wu et al., 2024), and value regularization (Chebotar et al., 2023; Hu et al., 2024a). These approaches often involve more complex modifications of DT to adapt it to specific tasks.

The most relevant work to ours is Decision Convformer (Kim et al., 2024), which solely utilizes convolutional filters to capture local patterns but shows limitations in tasks requiring long-term dependencies. Additionally, the fixed and finite convolution range limits its performance on tasks with non-standard Markov features. Our approach's distinction lies in designing an innovative token selection mechanism, which dynamically balances the trade-off between long-term and local features by retaining the attention mechanism. This innovation addresses the inherent limitations of DT and DC, is orthogonal to previous methods, and can be seamlessly integrated into existing architectures.

2.3. Rethinking the Trade-offs of Features in CSM

RL tasks with the standard Markov property imply that the current state contains all the information needed to predict future states. Therefore, using fixed convolutional kernels to capture features across adjacent time steps has performed well in such tasks. However, in scenarios where the Markov property is non-standard, the current state depends not only on the previous state but also on multiple past states or actions. In such cases, focusing on the long-term dependencies in the data is crucial for ensuring the model's performance. LSDT (Wang et al., 2025) experimentally integrates self-attention and dynamic convolution via a branch design for decision-making, further demonstrating the effectiveness of combining features at different scales. We visualized the attention scores of the DT's first layer on two tasks: Gym HalfCheetah (with standard Markov properties) and Maze2D (with non-standard Markov properties) to show the degree of token associations across sequences.

As shown in Figure 2, compared to the halfcheetah-medium, the attention matrices obtained from the maze2d-umaze dataset exhibited strong correlations between tokens that are far apart. This observation aligns with our intuition that, in non-standard Markov tasks, the relationships between tokens are not strictly local. Additionally, we perform a visualization analysis on datasets of varying quality from the same task and discover that the relationships between tokens in lower-quality data sequences are more chaotic and unevenly distributed. Therefore, a feasible concatenation mechanism could help the model capture compelling local or long-term features from suboptimal data.

Our proposed Decision Mixer aims to use a dynamic token selection mechanism to concatenate high-quality sequences from suboptimal trajectory sequences. The sequence's quality and length that enter the attention layer reflect a balance between local and long-term features, and the model can decide how to select and concatenate tokens based on the task's specific characteristics. In cases where the sequence length is extremely short (only two adjacent time-step tokens are selected and concatenated), DM degenerates into a DC-like architecture that utilizes the attention mechanism to capture more extreme local features than DC. When the sequence length matches the original context length, DM degenerates into DT.

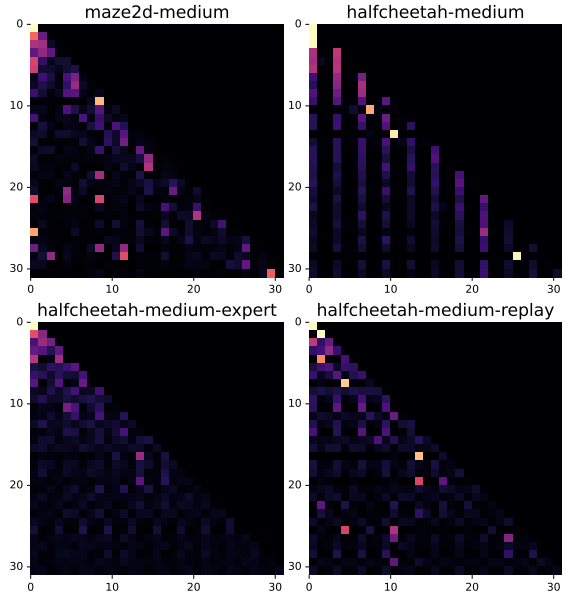

Figure 2. Visualization of attention scores in the first layer of the model for Maze2D and HalfCheetah tasks. We visualize the attention scores for the top 30 tokens, with color depth representing the correlation between tokens (return-to-go, state, and action). These scores serve as alignment measures, indicating the strength of association between each target token and source token. The Query index i (or Key index j) is ordered such that $i = 1$ corresponds to $\hat{r}_{t-K+1}$, $i = 2$ corresponds to state s_{t-K+1}, $i = 3$ corresponds to action a_{t-K+1}, and so on, continuing until the end. Given the application of causality, the attention matrix is lower-triangular.

3. Methodology

This section provides a comprehensive description of the proposed Decision Mixer. We analyze how CSM methods learn from historical trajectories from the perspective of reweighting. We integrate our dynamic token selection mechanism into the mixer layer to reduce computational burden while ensuring a reasonable trade-off between long-term dependencies and local features. Considering the short-sightedness during sampling, we also design an auxiliary predictor trained in parallel to compensate for this limitation.

3.1. A Reweighting Perspective

We let the e_t be the set of the history of states, actions and rewards up to step t along with s_t. When the trajectory is evident from the context, we use e to refer to a generic set, with a as a generic action. The essence of CSM is to learn a distribution $P_\beta(a \mid e)$, where β is the behavior policy that generated the data, and P_β refers to the joint distribution over states, actions, and rewards induced by β. Here, $f(e)$ is the conditioning function used during sampling to adjust the policy. It is often chosen to be constant at the initial state and decrease with the observed reward along the trajectory. By factoring this distribution, we can express the optimal policy π_f^{CSM} for a specific conditioning function $f(e)$ as:

$$\begin{aligned}\pi_f^{\mathrm{CSM}}(a \mid e) = P_\beta(a \mid e, f(e)) &= \frac{P_\beta(a \mid e) P_\beta(f(e) \mid e, a)}{P_\beta(f(e) \mid e)} \\ &= P_\beta(a \mid e) \frac{P_\beta(f(e) \mid e, a)}{P_\beta(f(e) \mid e)}.\end{aligned} \tag{2}$$

CSM thus can be viewed as reweighting the behavior based on the distribution of future returns. DT directly uses the attention mechanism to learn the internal weighting relationships in the data. However, the absence of explicit constraints means that suboptimal data can still disrupt the model's final decision. Filtering the data before the attention mechanism could better guide the model's learning, which motivates the design of the token selection mechanism.

3.2. Mixer Layer

We draw inspiration from the expert routing mechanism in the MOE architecture. Given the input X^l of length $S = 3K$ at the current mixer layer l, the router R_l assigns a weight $w_i^l = R_l(x_i^l)$ to each token x_i^l in X^l. We also design a hypernetwork H_l, which takes X^l as input and generates a threshold $k = H_l(X^l)$. We use simple MLPs to construct R_l and H_l, with the detailed architecture shown in Table B. R_l and H_l enable the model to adaptively design the threshold based on the data characteristics. This allows the model to select the top-k tokens for the attention mechanism based on the weights output by the router while skipping the unselected tokens in the current layer. The selected tokens are arranged in the original order to form $\hat{X}^l$, which is then passed through the first subblock, consisting of layer normalization LN_1^l and an attention layer Att^l, yielding Z_1^l. Z_1^l is subsequently passed through the second subblock, comprising layer normalization LN_2^l and a feed-forward network FFN^l, to produce Z_2^l in Equation 3.

$$Z_1^l = \mathrm{Att}^l(\mathrm{LN}_1^l(\hat{X}^l)) + \hat{X}^l, \quad Z_2^l = \mathrm{FFN}^l(\mathrm{LN}_2^l(Z_1^l)). \tag{3}$$

In the specific implementation, we multiply Z_2^l by the router weights and perform a generalized residual connection with the unselected tokens from X^l, preserving the original order. By placing the router weights along the gradient path, we enable the router to receive feedback from gradient descent during training, which allows for updates. The token selection mechanism allows the DM to use the subsequent attention module to further mine the relationships between relatively important tokens, avoiding interference from irrelevant or suboptimal tokens. Meanwhile, the re-connection of Z_2^l with the unselected tokens in X^l ensures that usable information continues to be passed to the next layer, further guaranteeing the retention of important context. We integrate the dynamic selection mechanism, attention mechanism, and forward network into a single layer, referred to as the mixer layer, and show the data flow through it in Equation 4.

$$X_i^{l+1} = (w_i^l Z_{2,i}^l) \cdot \mathbb{I}(w_i^l \in \text{top-}k) + X_i^l \cdot \mathbb{I}(w_i^l \notin \text{top-}k), \tag{4}$$

where $Z_{2,i}^l$ represents the token at the i-th position in Z_2^l.

From a computational burden perspective, if we reduce the length of the token sequence entering the attention layer to half of the original, the FLOP consumption during the key and value matrix multiplication process is reduced to 25% of the original. Similar calculations can determine the FLOP savings for the MLP.

3.3. Sampling Process

The token selection mechanism significantly reduces computational costs while enabling dynamic feature balancing. However, the generation of threshold k depends on the information from the entire sequence, meaning that both past and future tokens influence the selection of a specific token. While this is straightforward during training, the autoregressive sampling, which generates tokens step by step, obstructs the implementation of the token selection mechanism. To overcome this issue, we introduce an auxiliary predictor θ_{aux}^l that receives a single token as input and directly predicts whether the current token will be selected for the attention mechanism layer at the token level. Specifically, the auxiliary predictor outputs logits $\hat{y}_i \in \mathbb{R}^2$, representing the probabilities for token selection and non-selection, respectively. The auxiliary predictor is trained separately and in parallel with the primary model training, using the token selection results from each round in real time. The

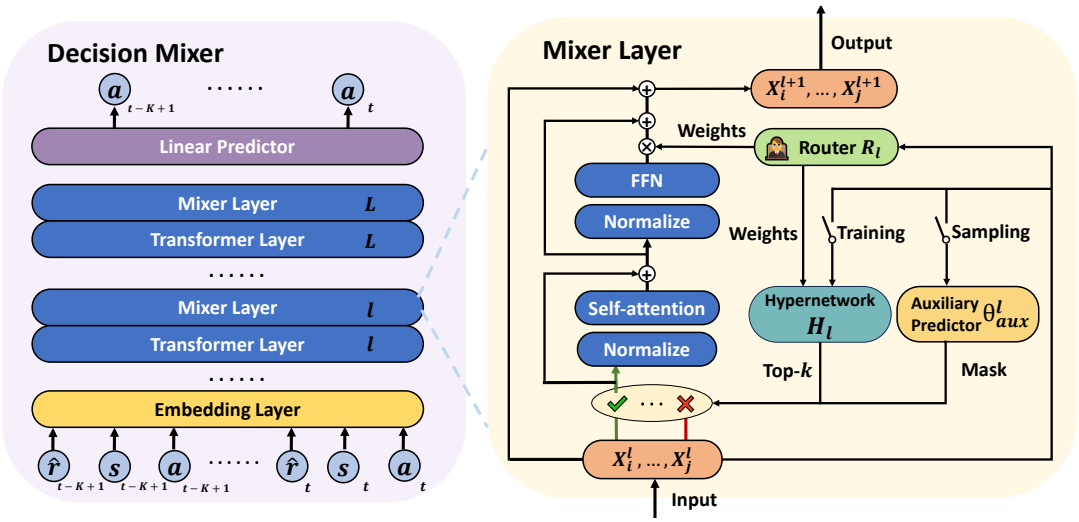

Figure 3. The architecture of Decision Mixer. During the training phase, the input sequence $X^l = (X_i^l, \ldots, X_j^l)$ undergoes token selection based on the router R_l and the hypernetwork H_l for a specific mixer layer l. During sampling, DM selects tokens from the X^l based on the auxiliary predictor θ^l_{aux}. Each mixer layer independently trains its own R_l, H_l, and θ^l_{aux}.

loss function for training is cross-entropy loss, applied to the predicted logits and the corresponding binary labels:

$$\mathcal{L}_{\text{aux}} = -\frac{1}{S}\sum_{i=1}^{S}\left[z_i \log\left(\sigma\left(\hat{y}_i\right)\right) + \left(1 - z_i\right)\log\left(1 - \sigma\left(\hat{y}_i\right)\right)\right], \tag{5}$$

where $\sigma(\hat{y}_i)$ is the sigmoid function that converts the logits $\hat{y}_i$ into probabilities. z_i is the binary label for each token in the mask, where $z_i = 1$ indicates the selected token and $z_i = 0$ indicates the unselected token. We concisely outline the pipeline of DM in Alg 1, 2.

3.4. Model Architecture

We adopt the same data format as DT in Equation 1 to maximize the algorithm's feasibility and highlight the advantages of our designed components. The input data τ_t is processed by a structure formed by alternating mixer layers and standard transformer layers, with the final layer incorporating a state-based prediction head to generate feasible actions. The purpose of alternating layers is to ensure model stability. Our ablation experiments observe that using only mixer layers as the main structure led to significant fluctuations during training despite achieving similar overall performance. A potential explanation is that the dynamic token selection mechanism prunes the entire sequence, which may cause parameter distribution to fluctuate. Intermittently deploying attention mechanisms over the entire sequence helps the model retain comprehensive input information and accelerates convergence. The loss function L_{DM} is shown in Equation 6.

$$\mathcal{L}_{\text{DM}} = \mathbb{E}_{\tau\sim D}[\frac{1}{K}\sum_{i=t-K+1}^{t}(a_t - (\pi_\theta(\tau_t))_i)^2]. \tag{6}$$

4. Experiment

In this section, we extensively evaluate our proposed Decision Mixer using the widely recognized D4RL benchmark (Fu et al., 2020). Our main objective is to assess the effectiveness of DM across various domains. Additionally, we execute empirical ablation studies to dissect and understand the individual contributions of the core components of our method.

Datasets. We consider five different domains of tasks in the widely used D4RL benchmark: Gym, Adroit, Kitchen, AntMaze and Maze2D. A detailed introduction to these five environments is presented in the appendix D.

Baselines. We compare our approach with representative offline RL algorithms from value-based and CSM methods. Each algorithm excels in specific domains but performs sub-optimal in others. For value-based methods, including BEAR (Kumar et al., 2019), BCQ (Fujimoto et al., 2019), CQL (Kumar et al., 2020), IQL (Kostrikov et al., 2022), TD3+BC (Fujimoto & Gu, 2021), MoRel (Kidambi et al., 2020), O-RL (Brandfonbrener et al., 2021) and COMBO (Yu et al., 2021). For CSM methods, including DT, DC, DD (Ajay et al., 2023), EDAC (An et al., 2021), D-QL (Wang et al., 2023), MPPI (Pravitra et al., 2020), StAR (Shang et al., 2022), GDT (Hu et al., 2023), LSDT (Wang et al., 2025) and CGDT (Wang et al., 2024). The performance scores for these baseline methods are sourced from the best results published in their respective papers or from our runs, ensuring a fair comparison.

Implementation details. All experiments are carried out on a server with 8 NVIDIA 3090 GPUs, each with 24GB of

Table 1. The performance of DM and SOTA baselines on D4RL Gym, Adroit, Kitchen, Maze2D, and AntMaze tasks. Results for DM correspond to the mean and standard errors of normalized scores over 30 random rollouts (3 independently trained models and 10 trajectories per model) for all tasks, which generally exhibit low variance in performance. Our method outperforms all prior methods by a clear margin in almost all domains, including the conventional value-based and CSM methods.

Dataset	**Value-Based Methods**						**Conditional Sequence Modeling Methods**						
Gym Tasks	**BEAR**	**BCQ**	**CQL**	**IQL**	**TD3+BC**	**MoRel**	**DT**	**StAR**	**GDT**	**CGDT**	**LSDT**	**DC**	**DM**
halfcheetah-medium-replay-v2	38.6	34.8	37.5	44.1	**44.6**	40.2	36.6	36.8	40.5	40.4	42.9	41.3	39.6$_{\pm 0.2}$
hopper-medium-replay-v2	33.7	31.1	95.0	92.1	60.9	93.6	82.7	29.2	85.3	93.4	93.9	94.2	**95.4**$_{\pm 0.4}$
walker2d-medium-replay-v2	19.2	13.7	77.2	73.7	81.8	49.8	79.4	39.8	77.5	78.1	74.7	76.6	**85.5**$_{\pm 2.1}$
halfcheetah-medium-v2	41.7	41.5	44.0	47.4	**48.3**	42.1	42.6	42.9	42.9	43.0	43.6	43.0	43.5$_{\pm 0.7}$
hopper-medium-v2	52.1	65.1	58.5	63.8	59.3	95.4	67.6	59.5	77.1	96.9	87.2	92.5	**98.1**$_{\pm 3.6}$
walker2d-medium-v2	59.1	52.0	72.5	79.9	83.7	77.8	74.0	73.8	76.5	79.1	81.0	79.2	**83.8**$_{\pm 0.8}$
halfcheetah-medium-expert-v2	53.4	69.6	91.6	86.7	90.7	53.3	86.8	93.7	93.2	93.6	93.2	93.0	**93.9**$_{\pm 0.1}$
hopper-medium-expert-v2	96.3	109.1	105.4	91.5	98.0	108.7	107.6	111.1	111.1	107.6	111.7	110.4	**111.8**$_{\pm 0.5}$
walker2d-medium-expert-v2	40.1	67.3	108.8	109.6	110.1	95.6	108.1	109.0	107.7	109.3	109.8	109.6	**112.7**$_{\pm 1.3}$
Average	48.2	53.8	77.6	76.5	75.3	72.9	76.2	66.2	79.1	82.4	82.0	82.2	**84.7**
Adroit Tasks	**BEAR**	**BCQ**	**CQL**	**IQL**	**O-RL**	**MoRel**	**EDAC**	**BC**	**DT**	**D-QL**	**StAR**	**GDT**	**DM**
pen-human-v1	-1.0	66.9	37.5	71.5	90.7	-3.2	52.1	63.9	79.5	72.8	77.9	92.5	**125.4**$_{\pm 5.1}$
hammer-human-v1	2.7	0.9	4.4	1.4	0.2	2.3	0.8	1.2	3.7	0.2	3.7	5.5	**6.1**$_{\pm 0.2}$
door-human-v1	2.2	-0.05	9.9	4.3	-0.3	2.3	10.7	2.0	14.8	0.0	1.5	20.6	**24.0**$_{\pm 1.8}$
pen-cloned-v1	-0.2	50.9	39.2	37.3	60.0	-0.2	68.2	37.0	75.8	57.3	33.1	86.2	**117.0**$_{\pm 3.1}$
hammer-cloned-v1	2.3	0.4	2.1	2.1	2.0	2.3	0.3	0.6	3.0	3.1	0.3	8.9	**9.5**$_{\pm 2.8}$
door-cloned-v1	2.3	0.01	0.4	1.6	-0.1	2.3	9.6	0.0	16.3	0.0	0.0	19.8	**23.7**$_{\pm 2.7}$
Average	1.0	19.8	15.6	19.7	25.5	1.0	23.6	17.5	32.2	22.2	19.4	38.9	**51.0**
Kitchen Tasks	**BEAR**	**BCQ**	**CQL**	**IQL**	**O-RL**	**TD3+BC**	**BC**	**DT**	**DD**	**StAR**	**GDT**	**DC**	**DM**
kitchen-complete-v0	0.0	8.1	43.8	62.5	2.0	0.0	65.0	50.8	65.0	40.8	43.8	40.9	**65.3**$_{\pm 0.3}$
kitchen-partial-v0	13.1	18.9	49.8	46.3	35.5	0.0	33.8	57.9	57.0	12.3	73.3	66.8	**75.0**$_{\pm 0.2}$
Average	6.6	13.5	46.8	54.4	18.8	0.0	51.5	54.4	61.0	26.6	58.6	58.7	**70.2**
Maze2D Tasks	**BEAR**	**BCQ**	**CQL**	**IQL**	**TD3+BC**	**COMBO**	**BC**	**MPPI**	**DT**	**QDT**	**GDT**	**DC**	**DM**
maze2d-umaze-v1	65.7	49.1	86.7	42.1	14.8	76.4	85.7	33.2	31.0	57.3	50.4	20.1	**86.9**$_{\pm 1.9}$
maze2d-medium-v1	25.0	17.1	41.8	34.9	62.1	68.5	38.3	10.2	8.2	13.3	7.8	38.2	**95.2**$_{\pm 7.7}$
Average	45.35	33.1	64.3	38.5	38.5	72.5	63.6	21.7	19.6	35.3	29.1	57.6	**91.1**
AntMaze Tasks	**BEAR**	**BCQ**	**CQL**	**IQL**	**TD3+BC**	**O-RL**	**BC**	**DT**	**RvS**	**StAR**	**GDT**	**DC**	**DM**
antmaze-umaze-v0	73.0	78.9	74.0	87.1	78.6	64.3	54.6	59.2	65.4	51.3	76.0	85.0	**100.0**$_{\pm 0.5}$
antmaze-umaze-diverse-v0	61.0	55.0	84.0	64.4	71.4	60.7	45.6	66.2	60.9	45.6	69.0	78.5	**100.0**$_{\pm 0.5}$
antmaze-medium-diverse-v0	8.0	0.0	53.7	**70.0**	0.0	0.0	0.0	7.5	67.3	0.0	0.0	0.0	60.0$_{\pm 1.3}$
Average	47.3	44.6	70.6	73.8	50.0	41.7	33.4	44.3	75.0	32.3	48.3	54.5	**86.7**

memory. The experimental hyperparameter configurations of DM are shown in Appendix A.

4.1. Main Results

We compare our DM with the baselines on five domains of tasks and report the results in Table 1. To ensure fair comparisons, we normalize the scores according to the protocol established in D4RL (Fu et al., 2020), where a score of 100 corresponds to an expert policy. We analyze the performance of DM on standard Markov tasks and non-standard Markov tasks separately.

Standard Markov Environment. We evaluate DM in Gym and Adroit environments with dense rewards. Our DM consistently achieves or approaches sota performance in all datasets, demonstrating the effectiveness of our architecture. Although most baseline models demonstrate proficiency in the Gym environment, DM still exhibits outstanding performance on almost all tasks. DM's slightly lower performance compared to TD3+BC on certain datasets is primarily due to CSM's inability to adapt well to datasets with low quality or insufficient coverage of the state space in scenarios where the value function is completely absent. On the other hand, the Adroit environment is characterized by a limited scope of human demonstrations, which leads to extrapolation errors that particularly challenge offline RL. It is precisely for this reason that DM's excellent performance across all Adroit tasks can be attributed to its high expressiveness and more effective token selection mechanism. Both DC and DM demonstrate that capturing local information can significantly enhance DT's performance in standard Markov

Table 2. Ablation study on model components. For simplicity, we have removed the version numbers after each task, which does not affect understanding. All experiments are repeated three times, and the average value is taken.

Dynamic selection	Auxiliary predictor	hopper-medium	walker2d-medium-expert	pen-cloned	maze2d-medium	antmaze-umaze
		67.6	108.1	75.8	8.2	32.1
	✓	33.6	70.9	75.3	29.9	59.6
✓		93.7	109.5	105.4	85.0	88.9
✓	✓	**98.1**	**112.7**	**117.0**	**95.2**	**100.0**

environments.

Non-standard Markov Environment. We use the Kitchen, Maze2d, and more complex AntMaze environments to evaluate DM's stitching and long-term credit assignment capabilities on non-standard Markov datasets. For Kitchen tasks requiring generalization to unseen states and long-term value optimization, the DM outperforms the CSM and Value-Based Methods. These results demonstrate that DM can learn useful data features from offline trajectories, enhancing generalization and stability. For the Maze2d environment, which serves as a benchmark to evaluate the capacity of offline RL algorithms to stitch segments of disparate trajectories effectively, the performance of DM significantly outperforms other methods, demonstrating the advantage of the token selection mechanism in stitching high-quality trajectories. The AntMaze environment is characterized by sparse rewards and many suboptimal trajectories, which presents an even more significant challenge. In this context, DC performs poorly due to its neglect of long-term information and an overreliance on data quality based on prior assumptions. The performance results of DM demonstrate the effectiveness and generalizability of the architecture we designed, particularly in antmaze-umaze-diverse tasks.

4.2. Ablation Study

Role of Different Components. As shown in Table 2, we observe a significant decline in the model's performance when we do not use the auxiliary predictor. The main reason is that the autoregressive testing process leads to insufficient predictive scope for the router, causing biases that are particularly noticeable in the early stages of testing when the number of visible tokens is limited. Additionally, increasing the sequence length of the training data is essential for enabling the CSM method to exhibit performance comparable to that of language or vision foundation models in RL tasks. The auxiliary predictor compensates for the performance degradation caused by the shorter data length available during sampling compared to the training data length. In the second row, we remove the dynamic token selection mechanism and fix the number of tokens model can select, setting it to 50% of the training trajectory length. This change allows the auxiliary predictor to function normally. However, we observe a significant performance drop, which aligns with our intuition: the fixed number of token selections prevents the model from adapting well to the varying nature of tasks and the complex distribution of sample quality.

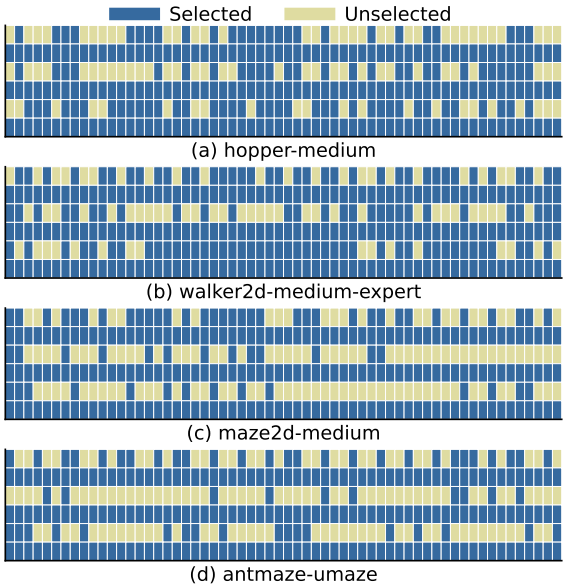

Figure 4. The visualization results of the token selection mechanism. The mixer and standard transformer layers are arranged alternately, from bottom to top, corresponding to the first to the sixth layer. The input sequence length is S=60, with tokens marked in blue representing the selected tokens and those marked in beige representing the unselected tokens.

Visualization of the token selection mechanism. To evaluate the performance of the token selection mechanism across tasks with different properties, we visualized the token selection at each layer for four tasks: hopper-medium and walker-medium-expert (with standard Markov properties), and maze2d-medium and antmaze-umaze (with non-standard Markov properties). The visualization results in Figure 4 reveal that the selected tokens exhibit a certain degree of positional proximity in standard Markov tasks. In contrast, the positional proximity of selected tokens is lower in non-standard Markov tasks, which aligns with our motivation, as the spatial relationships between relevant tokens follow a discrete distribution in these tasks. Interestingly, selected tokens tend to be picked in multiple mixer layers

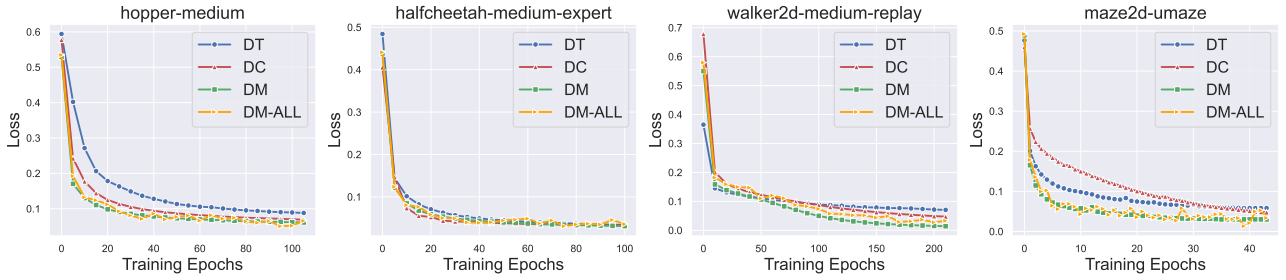

Figure 5. Visualization of loss curves for DT, DC, DM, and DM with all mixer layers (DM-ALL) across four tasks.

across all tasks, while some tokens tend to bypass all layers. We hypothesize that this phenomenon is closely related to the varying information quality within the sequence. We present the average selected token numbers for each mixer layer across all tasks in Table 10 and Figure 6. The mixer layer exhibits a disparity in the number of tokens selected for data of different qualities within the same task, which validates the potential of our method for high-quality trajectory selection.

Computational complexity. Table 3 shows the memory usage, parameter size, and FLOPs for several methods in the hopper-medium task. The FLOPs of DM are nearly halved compared to the original DT. Although DM has more parameters than DC, its unique token selection mechanism results in fewer FLOPs during training. Furthermore, due to the better alignment of transformers with the scaling law, DM outperforms the convolution-based DC regarding scalability and generalization. When we replace all transformer layers with mixer layers (DM-ALL), memory usage and FLOPs are further reduced. DM adopts a more suitable approach for offline RL data characteristics, achieving improved performance while reducing computational resource requirements. Figure 5 presents the training loss curves for several methods. We observe that DM-ALL exhibits significant fluctuations during training, which inspires us to choose alternating layers as the core architecture for DM to stabilize the training process.

Table 3. Ablation on the computational complexity.

Complexity	DT	DC	DM-ALL	DM	Δ%
Memory ↓	7128M	4830 M	3004M	4474M	↓ 37.2%
Params ↑	43.3M	28.5M	43.6M	43.5M	↑ 0.5%
Flops ↓	752.5 G	436.9G	330.1G	398.5 G	↓ 47.0%

Ability to unseen tasks. The model's performance on unseen tasks is a key criterion for evaluating its generalization ability and potential scalability. Inspired by Prompt-DT (Xu et al., 2022), we conduct experiments in three meta-learning environments: Cheetah-vel, Antdir, and MetaWorld. After training on the training tasks, we test the model on tasks that do not overlap with the training tasks. A description of the environments and the training/testing task division is provided in Appendix D. The results in Table 4 affirm DM's comprehensive enhancements across test tasks. Furthermore, the transformer architecture ensures that DM exhibits more substantial generalization capabilities as its parameter scale increases. We also conducted an ablation study on the context length K used in DM (Table 11), which shows performance improvements with an appropriate increase in context length. However, increasing K beyond a certain threshold, such as $K = 120$, may lead to performance degradation due to the prevalence of suboptimal trajectories.

Table 4. Ablation on the zero-shot generalization ability. We denote the model size as having x layers, y attention heads, and an embedding dimension of z, represented as (x, y, z) in the table.

Model Size	Game	BC	DT	DC	DM
(3,1,256)	Cheetah-vel	-147.41	-138.05	-154.20	**-135.62**
	Ant-dir	171.91	169.64	**173.32**	166.91
	MetaWorld 10	350.36	352.67	339.09	**373.53**
	MetaWorld 50	275.93	280.57	257.91	**305.28**
(6,4,256)	Cheetah-vel	-161.84	-148.24	-146.67	**-139.18**
	Ant-dir	168.33	**170.66**	142.09	167.73
	MetaWorld 10	337.40	343.16	255.21	**345.32**
	MetaWorld 50	297.14	279.70	290.73	**319.36**
(12,12,768)	Cheetah-vel	-140.59	-137.54	-142.30	**-134.67**
	Ant-dir	165.92	169.50	169.73	**177.24**
	MetaWorld 10	327.48	330.25	257.86	**365.04**
	MetaWorld 50	291.07	313.82	308.80	**347.53**

Combination with Other Methods. DM addresses the inherent trade-off problem in CSM methods, enhancing DT's performance from a foundational perspective. It is worth noting that DM is orthogonal to most previous works rather than conflicting with them and can improve the performance of past methods in a plug-and-play manner. We select two methods to demonstrate this. QT (Hu et al., 2024a) introduces Q-value regularization to optimize action selection on top of DT and excels in handling long time horizons and sparse reward tasks. We refer to the QT enhanced with

DM as QDM. ODT (Zheng et al., 2022) combines offline pretraining with online fine-tuning by introducing entropy regularization and hindsight experience replay mechanisms, optimizing the policy to adapt to tasks. We refer to the ODT enhanced with DM as ODM. As shown in Table 5, the methods enhanced with DM outperform the original methods on nearly all tasks, indicating that the DM architecture can serve as an efficient and feasible auxiliary mechanism to further improve model performance without compromising the advantages of the original model.

Table 5. Ablation on the model portability. DT architecture in QT and ODT is replaced with DM architecture to examine whether the DM method can enhance other methods in a non-conflicting manner. For simplicity, we have removed the version numbers after each task, which does not affect understanding.

Datasets	DT	DM	QT	QDM	ODT	ODM
halfcheetah-medium-replay	36.6	39.1	48.9	49.3	40.4	41.1
hopper-medium-replay	82.7	95.4	102.0	104.5	88.9	95.9
walker2d-medium-replay	79.4	85.5	98.5	90.6	76.9	80.3
halfcheetah-medium	42.6	43.5	51.4	52.9	42.7	43.8
hopper-medium	67.6	98.1	96.9	99.4	97.5	98.3
walker2d-medium	74.0	83.8	88.8	90.8	76.8	81.0
halfcheetah-medium-expert	86.8	93.9	96.1	97.2	87.1	90.7
hopper-medium-expert	107.6	111.8	113.4	113.5	111.0	112.3
walker2d-medium-expert	108.1	112.7	112.6	114.2	109.6	111.0
Average	74.6	84.7	91.4	94.4	82.7	84.7

Training–inference strategy comparison. We designed three schemes: (1) The hypernetwork uses a single token as input for training and inference without needing an auxiliary predictor. (2) The hypernetwork uses a causal sequence as input for training and inference without needing an auxiliary predictor. (3) The hypernetwork uses the entire sequence as input for training, providing data for the binary classification training of the auxiliary predictor, which is then used for inference (adopted by DM). As shown in Table 6, (3) demonstrated the best performance and stability across all tasks. Although (1) and (2) maintained consistency between training and inference, they struggled with convergence due to insufficient utilization of global information from the training data. In contrast, (3) adopts a hierarchical task decomposition strategy for training and inference. The auxiliary predictor is updated iteratively using predictions from the hypernetwork and router, effectively mitigating potential distribution shifts and ensuring stable performance.

Table 6. Ablation on training–inference strategies. We compare three schemes to investigate the impact of consistent or hierarchical training–inference designs on task performance.

Datasets	(1)	(2)	(3)
halfcheetah-medium	27.5	40.9	43.5
hopper-medium	82.7	94.7	98.1
walker2d-medium	55.2	83.4	83.8
maze2d-umaze	59.1	83.2	86.9
antmaze-umaze	80.3	75.0	100.0
Average Score	61.0	75.4	82.5

5. Conclusion

In this study, we propose Decision Mixer (DM), a novel approach that addresses the challenge of modelling both long-term dependencies and local Markov properties in offline RL tasks. DM optimizes attention allocation and significantly reduces computational complexity by dynamically selecting and concatenating tokens at each layer of the transformer architecture. Extensive experiments on standard RL benchmarks demonstrate that DM outperforms existing methods and highlights its ability to effectively balance feature conflicts, providing a viable path forward for scaling RL models with reduced computational overhead.

Limitation. Further enhancing DM's robustness, particularly in incomplete or noisy data scenarios, could improve its adaptability to diverse situations.

Acknowledgements

This project is supported by the STI 2030—Major Projects (No. 2021ZD0201405), the National Natural Science Foundation of China (Grant No. U23A20318, 62276195 and 62225113), the Shenzhen Basic Research Project (Natural Science Foundation) Basic Research Key Project (NO. JCYJ20241202124430041), the Science and Technology Major Project of Hubei Province (Grant No. 2024BAB046), the Tencent JR2025TEG002,the Foundation for Innovative Research Groups of Hubei Province (Grant No. 2024AFA017) and the National Research Foundation, Singapore, under its NRF Professorship Award No. NRF-P2024-001. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

Impact Statement

Decision Mixer contributes to developing more efficient and scalable offline RL foundation models, enabling broader applications in autonomous systems, robotics, and decision-making processes. While the immediate societal implications are primarily related to improving computational efficiency in RL tasks, it is crucial to remain mindful of the ethical considerations surrounding deploying RL models in real-world scenarios. These include the potential risks of unintended biases in decision-making, the impact of automated systems on jobs and industries, and the need for responsible AI practices to ensure fairness, transparency, and accountability in autonomous systems.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Agarwal, R., Schuurmans, D., and Norouzi, M. An optimistic perspective on offline reinforcement learning. In *International conference on machine learning*, pp. 104–114. PMLR, 2020.

Ainslie, J., Lei, T., de Jong, M., Ontanon, S., Brahma, S., Zemlyanskiy, Y., Uthus, D., Guo, M., Lee-Thorp, J., Tay, Y., Sung, Y.-H., and Sanghai, S. CoLT5: Faster long-range transformers with conditional computation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5085–5100, Singapore, 2023. Association for Computational Linguistics.

Ajay, A., Du, Y., Gupta, A., Tenenbaum, J. B., Jaakkola, T. S., and Agrawal, P. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023.

An, G., Moon, S., Kim, J.-H., and Song, H. O. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

Bengio, E., Bacon, P.-L., Pineau, J., and Precup, D. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Brandfonbrener, D., Whitney, W., Ranganath, R., and Bruna, J. Offline rl without off-policy evaluation. *Advances in neural information processing systems*, 34:4933–4946, 2021.

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

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Chebotar, Y., Vuong, Q., Hausman, K., Xia, F., Lu, Y., Irpan, A., Kumar, A., Yu, T., Herzog, A., Pertsch, K., et al. Q-transformer: Scalable offline reinforcement learning via autoregressive q-functions. In *Conference on Robot Learning*, pp. 3909–3928. PMLR, 2023.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Ching, W.-K., Huang, X., Ng, M. K., Siu, T.-K., Ching, W.-K., Huang, X., Ng, M. K., and Siu, T.-K. Higher-order markov chains. *Markov Chains: Models, Algorithms and Applications*, pp. 141–176, 2013.

Dai, D., Deng, C., Zhao, C., Xu, R., Gao, H., Chen, D., Li, J., Zeng, W., Yu, X., Wu, Y., Xie, Z., Li, Y., Huang, P., Luo, F., Ruan, C., Sui, Z., and Liang, W. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1280–1297, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

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

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Hou, L., Pang, R. Y., Zhou, T., Wu, Y., Song, X., Song, X., and Zhou, D. Token dropping for efficient BERT pretraining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3774–3784, Dublin, Ireland, 2022. Association for Computational Linguistics.

Hu, S., Shen, L., Zhang, Y., and Tao, D. Graph decision transformer. *arXiv preprint arXiv:2303.03747*, 2023.

Hu, S., Fan, Z., Huang, C., Shen, L., Zhang, Y., Wang, Y., and Tao, D. Q-value regularized transformer for offline reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 19165–19181. PMLR, 21–27 Jul 2024a.

Hu, S., Shen, L., Zhang, Y., Chen, Y., and Tao, D. On transforming reinforcement learning with transformers: The development trajectory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024b.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Kidambi, R., Rajeswaran, A., Netrapalli, P., and Joachims, T. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Kim, J., Lee, S., Kim, W., and Sung, Y. Decision convformer: Local filtering in metaformer is sufficient for decision making. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=af2c8EaKl8.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

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

Lee, K.-H., Nachum, O., Yang, M. S., Lee, L., Freeman, D., Guadarrama, S., Fischer, I., Xu, W., Jang, E., Michalewski, H., et al. Multi-game decision transformers. *Advances in Neural Information Processing Systems*, 35: 27921–27936, 2022.

Lei, T., Bai, J., Brahma, S., Ainslie, J., Lee, K., Zhou, Y., Du, N., Zhao, V., Wu, Y., Li, B., et al. Conditional adapters: Parameter-efficient transfer learning with fast inference. *Advances in Neural Information Processing Systems*, 36: 8152–8172, 2023.

Li, W., Luo, H., Lin, Z., Zhang, C., Lu, Z., and Ye, D. A survey on transformers in reinforcement learning. *Transactions on Machine Learning Research*, 2023.

Liu, T., Shi, L., Hong, R., Hu, Y., Yin, Q., and Zhang, L. Multi-stage vision token dropping: Towards efficient multimodal large language model. *arXiv preprint arXiv:2411.10803*, 2024.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Muennighoff, N., Rush, A., Barak, B., Le Scao, T., Tazi, N., Piktus, A., Pyysalo, S., Wolf, T., and Raffel, C. A. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36:50358–50376, 2023.

Pravitra, J., Ackerman, K. A., Cao, C., Hovakimyan, N., and Theodorou, E. A. L1-adaptive mppi architecture for robust and agile control of multirotors. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7661–7666. IEEE, 2020.

Press, O., Smith, N., and Lewis, M. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=R8sQPpGCv0.

Qu, Y., Wang, B., Shao, J., Jiang, Y., Chen, C., Ye, Z., Linc, L., Feng, Y., Lai, L., Qin, H., et al. Hokoff: Real game dataset from honor of kings and its offline reinforcement learning benchmarks. *Advances in Neural Information Processing Systems*, 36:22166–22190, 2023.

Raposo, D., Ritter, S., Richards, B., Lillicrap, T., Humphreys, P. C., and Santoro, A. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024.

Schuster, T., Fisch, A., Gupta, J., Dehghani, M., Bahri, D., Tran, V., Tay, Y., and Metzler, D. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.

Shang, J., Kahatapitiya, K., Li, X., and Ryoo, M. S. Starformer: Transformer with state-action-reward representations for visual reinforcement learning. In *European conference on computer vision*, pp. 462–479. Springer, 2022.

Si, C., Yu, W., Zhou, P., Zhou, Y., Wang, X., and Yan, S. Inception transformer. *Advances in Neural Information Processing Systems*, 35:23495–23509, 2022.

Vaswani, A. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Wang, H., Zhang, Z., and Han, S. Spatten: Efficient sparse attention architecture with cascade token and head pruning. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 97–110. IEEE, 2021.

Wang, J., Karanasou, P., Wei, P., Gatti, E., Plasencia, D. M., and Kanoulas, D. Long-short decision transformer: Bridging global and local dependencies for generalized decision-making. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wang, Y., Yang, C., Wen, Y., Liu, Y., and Qiao, Y. Critic-guided decision transformer for offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 15706–15714, 2024.

Wang, Z., Hunt, J. J., and Zhou, M. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

Wei, H., Ye, D., Liu, Z., Wu, H., Yuan, B., Fu, Q., Yang, W., and Li, Z. Boosting offline reinforcement learning with residual generative modeling. In *30th International Joint Conference on Artificial Intelligence, IJCAI 2021*, pp. 3574–3580. International Joint Conferences on Artificial Intelligence, 2021.

Wu, Y.-H., Wang, X., and Hamaya, M. Elastic decision transformer. *Advances in Neural Information Processing Systems*, 36, 2024.

Xie, Z., Lin, Z., Ye, D., Fu, Q., Wei, Y., and Li, S. Future-conditioned unsupervised pretraining for decision transformer. In *International Conference on Machine Learning*, pp. 38187–38203. PMLR, 2023.

Xu, K., Zhang, Y., Ye, D., Zhao, P., and Tan, M. Relation-aware transformer for portfolio policy learning. In *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, pp. 4647–4653, 2021.

Xu, M., Shen, Y., Zhang, S., Lu, Y., Zhao, D., Tenenbaum, J., and Gan, C. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pp. 24631–24645. PMLR, 2022.

Yao, Z., Wu, X., Li, C., Holmes, C., Zhang, M., Li, C., and He, Y. Random-ltd: Random and layerwise token dropping brings efficient training for large-scale transformers. *arXiv preprint arXiv:2211.11586*, 2022.

Ye, D., Chen, G., Zhang, W., Chen, S., Yuan, B., Liu, B., Chen, J., Liu, Z., Qiu, F., Yu, H., et al. Towards playing full moba games with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 621–632, 2020a.

Ye, D., Liu, Z., Sun, M., Shi, B., Zhao, P., Wu, H., Yu, H., Yang, S., Wu, X., Guo, Q., et al. Mastering complex control in moba games with deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 6672–6679, 2020b.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

Zheng, H., Shen, L., Luo, Y., Liu, T., Shen, J., and Tao, D. Decomposed prompt decision transformer for efficient unseen task generalization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Zheng, H., Shen, L., Tang, A., Luo, Y., Hu, H., Du, B., Wen, Y., and Tao, D. Learning from models beyond fine-tuning. *Nature Machine Intelligence*, pp. 1–12, 2025.

Zheng, Q., Zhang, A., and Grover, A. Online decision transformer. In *international conference on machine learning*, pp. 27042–27059. PMLR, 2022.

Zhong, Q., Ding, L., Liu, J., Liu, X., Zhang, M., Du, B., and Tao, D. Revisiting token dropping strategy in efficient BERT pretraining. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10391–10405, Toronto, Canada, July 2023. Association for Computational Linguistics.

Zhou, Y., Lei, T., Liu, H., Du, N., Huang, Y., Zhao, V., Dai, A. M., Le, Q. V., Laudon, J., et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

Appendix

The appendix is organized into several sections, each providing additional insights and details related to different aspects of the main work.

A. Hyperparameters Configuration

Table 7. Common hyperparameters configuration of Decision Mixer.

Hyperparameters	Value
K (length of context)	20
training batch size	512
learning rate	1e-3
weight dacay	1e-4
pct_traj	1
number of layers	6
number of mixer layers	3
number of transformer layers	3
number of attention heads	4
embedding dimension	256
activation	GeLU
dropout	0.1
num_workers	64

B. Network Architecture Details

Table 8. Network architecture details.

Network	**Layer**	**Input**	**Output**
R	Linear	embed_dim	1
H	Linear	context_length $\times$ embed_dim	512
	LeakyReLU	–	–
	Linear	512	1
θ_{aux}	Linear	embed_dim	embed_dim//2
	SiLU	–	–
	Linear	embed_dim//2	2

C. Algorithm Pseudocode

Algorithm 1 Training Process of Decision Mixer

Input: Training data $\mathcal{D}$
Initialize: Model π_θ, Router R_l, Hypernetwork H_l, Auxiliary predictor θ^l_{aux}
for each epoch **do**
  for each batch (X_b, y_b) in training data **do**
    Initialize $X^l = X_b$ (input sequence)
    for each layer $l = 1$ to L **do**
      Obtain the router weights: $w_i^l = R_l(x_i^l)$
      Compute the dynamic threshold:$k = H_l(X^l)$
      Perform token selection:
        $\hat{X}^l \leftarrow$ Select tokens based on w_i^l and k
      Apply attention mechanism and Feed-Forward Network by Equation 3
      Update token sequence by Equation 4
    end for
    Train the auxiliary predictor independently:
      Compute the $\mathcal{L}_{\text{aux}}$ by Equation 5
      Update θ^l_{aux} using $\mathcal{L}_{\text{aux}}$ (independent training, no gradient propagation to π_θ)
    Compute the $\mathcal{L}_{\text{DM}}$ by Equation 6
    Backpropagation to update π_θ
  end for
end for

Algorithm 2 Sampling Process of Decision Mixer

Initialize: Model π_θ, Router R_l, Auxiliary predictor θ^l_{aux}
for each test sample X **do**
  Initialize $X^l = X$ (input sequence)
  for each layer $l = 1$ to L **do**
    Obtain the router weights: $w_i^l = R_l(x_i^l)$
    Use the auxiliary predictor θ^l_{aux} to decide token selection:
      $\hat{\mathbb{I}}_i \leftarrow$ Auxiliary predictor output (whether token x_i^l is selected)
    Apply token selection based on the predicted $\hat{\mathbb{I}}_i$
    Apply attention mechanism and Feed-Forward Network by Equation 3
    Update token sequence by Equation 4
  end for
  Compute predicted action $\hat{a}$ using X^L
end for

D. Environment Details

D.1. Main Environment

- **Gym tasks**: The Gym-MuJoCo tasks (Hopper, HalfCheetah, Walker2d) are popular benchmarks used in offline deep RL. They are relatively straightforward and characterized by datasets with a significant proportion of near-optimal trajectories and smooth reward functions.

- **Adroit tasks**: The Adroit domain involves controlling a 24-DoF simulated Shadow Hand robot to perform tasks such as hammering a nail, opening a door, twirling a pen, or picking up and moving a ball. This domain is chosen to study the impact of narrow expert data distributions and human demonstrations on sparse-reward, high-dimensional robotic manipulation tasks. Since these tasks are primarily derived from human behavior, they exhibit a limited state-action space, requiring robust policy regularization to ensure consistent agent performance.

- **Kitchen tasks**: The Kitchen domain involves controlling a 9-DoF Franka robot in a kitchen environment with everyday household items such as a microwave, kettle, overhead light, cabinets, and an oven. The goal is to interact with these items to achieve a desired state configuration. This domain benchmarks the impact of multitasking behaviour in a realistic, non-navigation environment, where the "stitching" challenge arises from complex paths through the state space. Consequently, algorithms must generalize to unseen states rather than rely solely on training trajectories. The environment requires the agent to complete multiple sequential sub-tasks, further emphasizing the need for robust generalization.

- **Maze2D tasks**: The Maze2D domain is a navigation task in which a 2D agent must reach a fixed goal location. It tests offline RL algorithms' ability to stitch together previously collected sub-trajectories to find the shortest path to the goal. Three maze layouts are provided: the "maze," "medium," and "large " mazes. These tasks evaluate the algorithm's capability to effectively combine sub-trajectories and identify the shortest path to the set goal.

- **AntMaze tasks**: The AntMaze domain extends the Maze2D task by replacing the 2D ball with a more complex 8-DoF "Ant" quadruped robot, presenting a more demanding navigation challenge. This domain is introduced to test the stitching challenge with a morphologically complex robot, better representing real-world robotic navigation tasks. The task uses a sparse 0-1 reward, activated upon reaching the goal.

D.2. Meta-RL Environment

- **Cheetah-vel**: There are 40 tasks in Cheetah-vel with different goal velocities. The target velocities are uniformly sampled from the interval [0,3]. The agent is penalized with l2 errors to the target velocity. We hold out 5 tasks to construct the testing set and train with the remaining 35 tasks.

- **Ant-dir**: There are 50 tasks in Ant-dir with different goal directions uniformly sampled in 2D space. The 8-joints ant is rewarded with high velocity along the goal direction. We sample 5 tasks for testing and leave the rest for training.

- **Meta-World ML10**: In Meta-World ML10, the task is to control a Sawyer robot's end-effector to reach a target position in 3D space. The agent directly controls the XYZ location of the end-effector. Each task has a different goal position. We train in 10 tasks and test in unseen 3 tasks.

- **Meta-World ML45**: In Meta-World ML45, each task has a different goal position. We train in 45 tasks and test in unseen 5 tasks.

Table 9. Training and testing task indexes when testing the generalization ability in unseen tasks.

Cheetah-vel	
Training set of size 35	$[0-1, 3-6, 8-14, 16-22, 24-25, 27-39]$
Testing set of size 5	$[2, 7, 15, 23, 26]$
Ant-dir	
Training set of size 45	$[0-5, 7-16, 18-22, 24-29, 31-40, 42-49]$
Testing set of size 5	$[6, 17, 23, 30, 41]$
Meta-World ML10	
Training set of size 10	$[0, 9, 19, 29, 33, 36, 39, 40, 48, 49]$
Testing set of size 3	$[11, 24, 41]$
Meta-World ML45	
Training set of size 45	$[0-10, 12-16, 18-24, 26-35, 37-40, 42-49]$
Testing set of size 5	$[11, 17, 25, 36, 41]$

E. Supplementary Experiment

Another perspective on the token selection mechanism. It is worth noting that token selection can also be viewed from the stitching perspective. Existing methods (Wu et al., 2024; Hu et al., 2024a) demonstrate effectiveness in stitching capabilities but often require additional optimization, such as complex representation learning objectives and statistical measures, which complicate the training process and increase the computational burden. In contrast, DM employs an intuitive and straightforward token selection mechanism to reorganize the trajectories, capturing specific features that lead to performance improvements. Significant inconsistencies in sequence lengths are observed in standard and non-standard Markov task datasets of the same quality collected under the same policy. This further demonstrates that DM focuses on the internal features of the data rather than merely the quality of the trajectories themselves.

Table 10. The average number of tokens selected on the three mixer layers by the token selection mechanism when the model size is (6, 4, 256). For simplicity, we have removed the version numbers after each task, which does not affect understanding.

Dataset		1th Mixer Layer	2nd Mixer Layer	3rd Mixer Layer	Average
Gym Tasks	halfcheetah-medium-replay	15.07	31.99	37.29	28.12
	hopper-medium-replay	40.59	33.60	15.76	29.98
	walker2d-medium-replay	37.26	16.10	28.25	27.20
	halfcheetah-medium	25.15	34.65	3.47	21.09
	hopper-medium	36.54	38.62	6.29	27.15
	walker2d-medium	30.48	44.51	5.04	26.68
	halfcheetah-medium-expert	37.95	35.36	31.70	34.67
	hopper-medium-expert	18.41	10.60	41.66	23.56
	walker2d-medium-expert	33.81	23.64	45.49	34.31
Adroit Tasks	pen-human	17.17	34.42	2.83	18.14
	hammer-human	40.97	27.43	10.93	26.44
	door-human	13.32	44.12	45.03	34.16
	pen-cloned	22.04	10.69	9.07	13.93
	hammer-cloned	21.18	13.27	26.81	20.42
	door-cloned	9.38	53.63	3.88	22.30
Kitchen Tasks	kitchen-complete	33.89	42.88	7.15	27.97
	kitchen-partial	29.03	28.55	32.34	29.97
Maze2D Tasks	maze2d-umaze	19.74	19.98	22.67	20.80
	maze2d-medium	21.08	32.77	41.23	31.69
AntMaze Tasks	antmaze-umaze	21.36	12.87	30.37	21.53
	antmaze-umaze-diverse	21.70	20.27	32.00	24.66
	antmaze-medium-diverse	10.56	40.59	57.40	36.18

Impact of context length K As shown in Table 11, we conducted experiments in the Gym environment with DM context lengths of 8, 20, 60, and 120, keeping the other parameters consistent with those in Table A. We observed that DM performance improved to varying degrees with increasing context length, indicating that DM, like DT, exhibits excellent extendability. However, when K was set to 120, the model's performance decreased compared to K=60. This highlights that an indiscriminate increase in context length can negatively impact model performance. Therefore, it is important to carefully analyze the relationship between data scale and data quality and design appropriate hyperparameters and architecture sizes

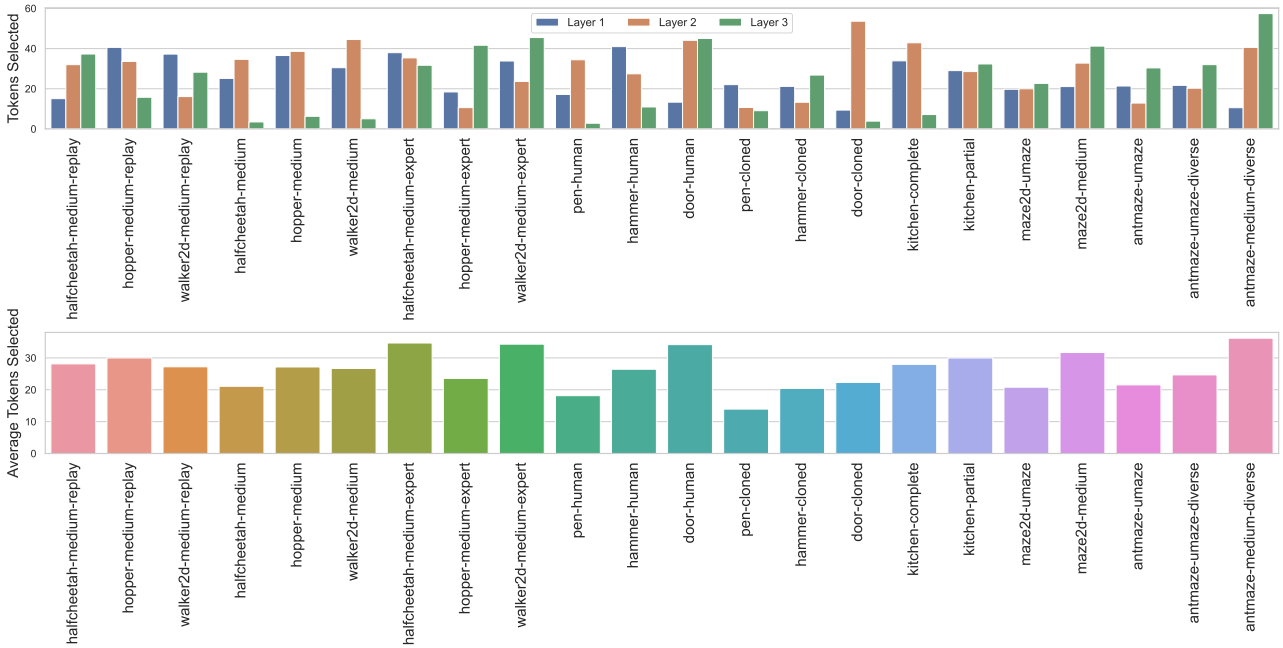

Figure 6. Average tokens selected across three mixer layers for token selection mechanism.

Table 11. Experimental results of DM with different context lengths (K) in Gym tasks.

Datasets	**DT(20)**	**DM (8)**	**DM (20)**	**DM (60)**	**DM(120)**
halfcheetah-medium-replay	36.6	36.9	39.6	39.2	39.5
hopper-medium-replay	82.7	95.6	95.4	94.5	96.6
walker2d-medium-replay	79.4	82.4	85.5	87.4	82.9
halfcheetah-medium	42.6	44.2	43.5	43.8	44.0
hopper-medium	67.6	96.0	98.1	99.1	99.7
walker2d-medium	74.0	85.1	83.8	84.1	85.5
halfcheetah-medium-expert	86.8	91.1	93.9	92.4	93.6
hopper-medium-expert	107.6	112.2	111.8	112.4	113.2
walker2d-medium-expert	108.1	114.0	112.7	114.6	110.6
Average	76.2	84.2	84.7	85.3	85.1

to ensure the token selection mechanism can fully function.

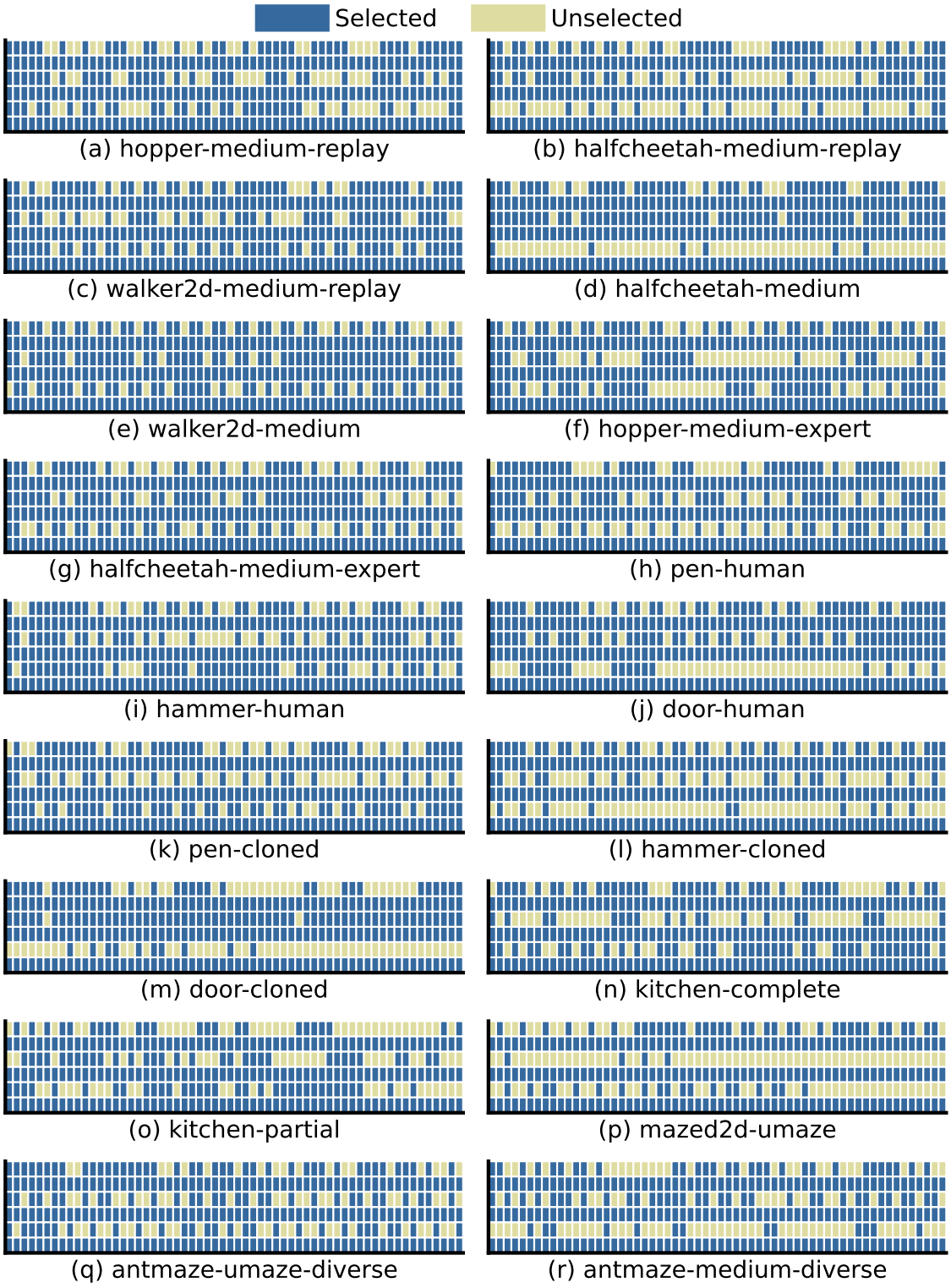

Figure 7. The visualization results of the token selection mechanism. The mixer and standard transformer layers are arranged alternately, from bottom to top, corresponding to the first to the sixth layer. The input sequence length is 60, with tokens marked in blue indicating those selected and tokens marked in beige indicating those not selected.