Beyond Naive Replay: SS-RL for Robust and Effective Continual Offline Reinforcement Learning

1st Yang Yu
School of Artificial Intelligence
Jilin University
Changchun, China
yyu23@mails.jlu.edu.cn

2nd Jifeng Hu*
School of Artificial Intelligence
Jilin University
Changchun, China
hujf21@mails.jlu.edu.cn

3rd Sinuo Zhang
School of Artificial Intelligence
Jilin University
Changchun, China
zsn24@mails.jlu.edu.cn

4th Zhejian Yang
School of Artificial Intelligence
Jilin University
Changchun, China
zjyang22@mails.jlu.edu.cn

5th Shengjie Wang
School of Artificial Intelligence
Jilin University
Changchun, China
sjwang24@mails.jlu.edu.cn

6th Hechang Chen*
School of Artificial Intelligence
Jilin University
Changchun, China
chenhc@jlu.edu.cn

***Abstract*—To address the increasing demands of mastering potential decision-making tasks based on existing well-trained models, continual offline reinforcement learning (CORL) is proposed to effectively utilize offline data for long-term learning and decision-making. Facing the challenges of mitigating catastrophic forgetting when learning sequentially from fixed offline datasets and overcoming performance limitations imposed by sub-optimal data quality within these datasets, we propose an innovative method, Trajectory Stitching and PCA-Selective Replay for RL (SS-RL), to tackle the above challenges and increase the generalization ability. Specifically, a multi-head neural network structure is adopted to promote agents' knowledge sharing among multiple subtasks or goals, thereby improving the strategy's stability and expressiveness. Secondly, an experience replay mechanism is introduced to enhance the long-term stability of the strategy by reusing previous tasks' offline data. Finally, increasing the dataset's quality by stitching sub-optimal trajectories, then holding the performance or avoiding rehearsal experience overfitting with diverse trajectories selected by PCA. Experimental results show that the proposed method significantly improves learning efficiency and strategy performance in standard CORL benchmark tasks, and has strong generalization ability.**

***Index Terms*—Continual Offline Reinforcement Learning, Experience Replay, Data Augmentation.**

I. INTRODUCTION

Artificial intelligence has made remarkable progress over the past few decades. However, current AI systems often excel at specific tasks but struggle to adapt to new tasks and continually learn in different contexts. In contrast, humans possess a unique ability to continually learn and adapt to new tasks in diverse situations, even generalizing quickly to unseen scenarios with limited experience. This capacity is one of core objectives of Artificial General Intelligence (AGI) [1] [2] . As AGI evolves, a critical challenge lies in enabling AI systems to learn new tasks continually, much like humans, without forgetting previously acquired knowledge or being constrained by it. Although current reinforcement learning (RL) methods can learn complex strategies through interaction with the environment, they often suffer from "catastrophic forgetting" [3] [4] when confronted with task changes, meaning that the knowledge gained from previous tasks can be erased or severely disrupted by new learning experiences.

Traditional continual RL algorithms often rely on extensive online interaction, which incurs prohibitive computational costs and significant safety risks in large-scale applications. In safety-critical domains such as autonomous driving [5], [6] and medical systems [7], [8], exploratory online actions can lead to catastrophic consequences. To mitigate these issues, Continual Offline RL (CORL) [9] has emerged. By leveraging pre-collected static datasets, CORL synergizes the safety and efficiency of offline learning with the knowledge retention goals of continual learning, providing a robust framework for sequential learning in complex, real-world environments.

Various methods have been proposed to mitigate catastrophic forgetting in continual learning (CL), typically categorized into regularization, structural, and rehearsal-based approaches. Among these, rehearsal-based methods are particularly effective due to their flexibility and ability to review past experiences. However, in Continual Offline RL (CORL), the performance of these methods is highly sensitive to the quality and distribution of the replay buffer. Simply selecting trajectories with the highest rewards often leads to distribution shift and overfitting due to a lack of diversity. To address these issues, we propose a rehearsal-based strategy that balances trajectory quality, diversity, and representativeness. We first employ model-based trajectory stitching (MBTS) to enhance the offline dataset by synthesizing high-quality transitions. Then, we utilize Principal Component Analysis (PCA) to evaluate trajectory similarity within the augmented dataset. By selecting n trajectories that maximize both cumulative rewards and structural diversity, our approach ensures a high-performing and comprehensive replay buffer, facilitating robust knowledge retention across sequential tasks. The main contributions of this paper can be summarized as follows:

- We introduce SS-RL, a CORL framework that improves knowledge retention by elevating the quality of replayed experiences, especially in low-quality offline settings.
- We present a hybrid selection strategy combining Trajectory Stitching and PCA analysis to maximize buffer diversity while maintaining high data utility, thus boosting model generalization.
- We empirically validate SS-RL on various benchmarks, demonstrating its superior performance and adaptability in complex continual reinforcement learning scenarios.

II. Related Work

a) Offline Reinforcement Learning: Offline reinforcement [10] [11] [12] learning is becoming an important branch of RL. Traditional reinforcement learning algorithms provide us with a basic online learning template, which is not applicable in all scenarios. In quite a few cases, we cannot perform online interactions because data collection is expensive (such as in robots, educational agents, or healthcare) or dangerous (for example, in autonomous driving or healthcare). On the other hand, even in scenarios where online interactions are feasible, we may still use previously collected data because pre-collected data can greatly improve training efficiency and save training time.

Previous works proposed to constrain the learning strategy to move closer to the behavior strategy by adding KL-divergence [13], MSE, Decision Transformer [14], or regularization of action selection. These methods exhibit degraded performance when trained on suboptimal data, illustrating the challenges of learning from sub-optimal trajectories. In CORL, this performance degradation problem is even more serious. Here we use a model-based data enhancement strategy, trajectory stitch (MBTS) [15], to improve the quality of suboptimal historical trajectories, and use the PCA correlation algorithm [16] to select the *n* trajectories with the lowest similarity and store them.

b) Continual Reinforcement Learning: Continual reinforcement learning (CRL) [17] is a key research direction in the field of RL. In traditional reinforcement learning, the agent gradually learns new strategies by continually interacting with the environment. It is usually assumed that the environment is static or single, and the transfer or forgetting of knowledge is not considered. However, in the continual learning scenario, the agent not only must effectively learn the current new task, but also must maintain the performance of the old task in the process of learning the new task (i.e., avoid catastrophic forgetting).

Continual learning methods are generally categorized into three main approaches. Regularization-based methods incorporate a regularization term to constrain model parameters, ensuring that they do not deviate significantly from the values optimized for previous tasks. Modular approaches allocate specific subsets of parameters exclusively to individual tasks, maintaining task-specific performance without interference. Rehearsal-based methods, on the other hand, train the agent by combining data from previously learned tasks with data from the current task. While these strategies have shown effectiveness, they primarily focus on online reinforcement learning settings. Previous studies have proven that the rehearsal-based method is the most effective method for continual learning: storing partial samples from previous tasks and using interleaved updates between new task samples and previous task samples.

III. Preliminary

a) Continual Offline Reinforcement Learning: In this paper, we investigate CORL [18] [19], which learns a sequence of RL tasks $\mathcal{T} = (T_1, \cdots, T_N)$. More specifically, our focus is on task-aware continual learning, where distinct task boundaries are clearly defined. Each task T_n is described as a Markov Decision Process (MDP) represented by a tuple of $\mathcal{M}_n = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}_n, \rho_{0,n}, r_n, \gamma \rangle$, where we use subscript n to differentiate different tasks, N is the number of total tasks, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}_n : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the transition function, $\rho_{0,n}$ is the distribution of the initial state, $r_n : \mathcal{S} \times \mathcal{A} \to [R_{\min}, R_{\max}]$ is the reward function, and $\gamma \in (0, 1]$ is the discount factor. In this work, we only consider sequential tasks with the same task domain, that is, they have different $\mathcal{P}_n, \rho_{0,n}$ and r_n, but share the same $\mathcal{S}, \mathcal{A}$, and γ.

In the offline RL setting, we can only train the policy through the existing static dataset $\mathcal{D}_n = \{e_n\}, e_n = (s_n, a_n, r_n)$, which are offline datasets of different tasks obtained by unknown policies, denoted as $\pi_n^\beta(a \mid s)$. We aim to train sequential tasks all over $[T_1, \cdots, T_N]$ sequentially to get a high mean performance $\sum_i^N \mathbb{E}_\pi[\sum_{t=0}^H \gamma^t r(s_t^i, a_t^i)]$, where H is the horizon, and minimize forgetting across all learned tasks while relying solely on a small buffer instead of access to data from previous tasks.

IV. Method

To address data quality challenges in continual offline reinforcement learning, we introduce SS-RL (Trajectory Stitching and PCA-Selective Replay). By integrating TS-based augmentation with PCA-based similarity metrics, SS-RL selects diverse, high-quality trajectories to ensure robust knowledge retention. We describe the DBC multi-head framework and our selective replay strategy, followed by the formal algorithm and its implementation details.

A. Trajectory Stitching

To build a high-quality and diverse trajectory experience replay buffer, we combined the model-based trajectory stitching (MBTS) data augmentation method proposed by [15] and principal component analysis (PCA) as the basis for similarity calculation, and modified it appropriately to adapt to our specific method. First, we will introduce the core idea of MBTS.

Trajectory stitching is to stitch together high-value areas of different trajectories to enhance the static dataset $\mathcal{D}$. Here we first define *stitching event*. The *stitching event* is achieved by searching the static $\mathcal{D}$ for a candidate next state that can

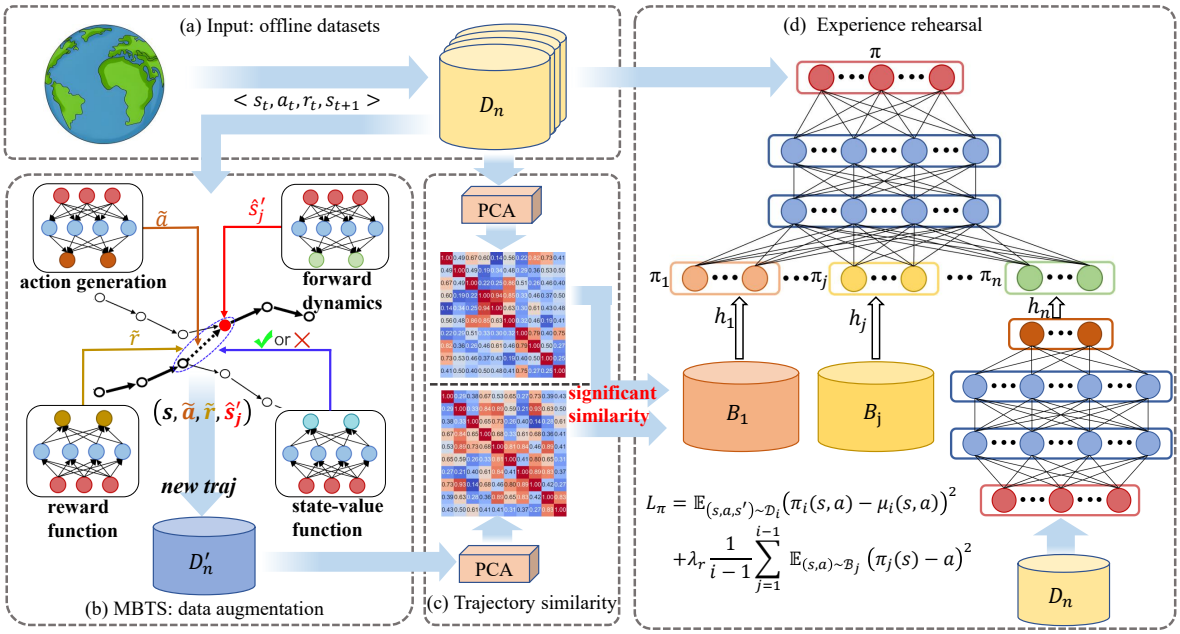


Fig. 1: SS-RL architecture.

produce a credible and higher report, i.e., a reward value. These high-quality states are determined by the state value function $V(s)$, which is trained using static data $\mathcal{D}$. This function is unaffected by distribution shift because it only evaluates states within the distribution. Assuming that in a certain transition (s, a, s') of a trajectory τ in $\mathcal{D}$, for which the joint density function is $p(s, a, s') \propto p(s' \mid s)\, p(a \mid s, s')$; here, $p(s' \mid s)$ represents the forward dynamics of the environments and $p(a \mid s, s')$ is its inverse dynamics.Based on the aforementioned concept, we want to replace the original s and a with the new candidate next state $\hat{s}'$ and the new action $\hat{a}$ for connection. Here are two principles for executing events.

Principle 1, Credibility: The next candidate state must be at least as observable as the original $\hat{s}'$, or more credible, that is, $p(\hat{s}' \mid s) \geq p(s' \mid s)$.

Principle 2: High quality: To ensure that the stitching event is effective, in addition to ensuring that the new candidate state $\hat{s}'$ is under our distribution, we must also ensure that it can produce a better reward report than s'. We give its formal definition based on the state value function $V(\hat{s}') > V(s')$.

When the above two principles are satisfied, we need to find a new $\hat{a}$ to connect s and $\hat{s}'$ by finding an action that maximizes the inverse dynamics, i.e. $\arg\max_{\hat{a}} p(\hat{a} \mid s, \hat{s}')$. Let's take an example. Now there are two trajectories τ_1 and τ_2. At a certain transition (s, a, s') in τ_1, it is found that there is a better point in τ_2 that meets the credible conditions. Therefore, τ_1 and τ_2 are stitched together at this point.

$$\left(s_1^{(1)}, a_1^{(1)}, s_2^{(1)}, \ldots, s_{t-1}^{(1)}, a_{t-1}^{(1)}, s_t^{(1)}, \hat{a}, s_m^{(2)}, a_m^{(2)}, s_{m+1}^{(2)}, \ldots, a_{M-1}^{(2)}, s_M^{(2)}\right)$$

Here, we use neural networks to estimate the forward dynamics, inverse dynamics, reward function, and state-value function as in the original paper.

For the forward dynamics model, using an ensemble strategy according to [20] [21] can be shown to take into account epistemic uncertainty, so N models are trained and maximum likelihood estimation is used

$$\mathcal{L}_{\hat{p}}(\xi) = \mathbb{E}_{(s,s')\sim\mathcal{D}}\big[(\mu_\xi(s) - s')^T \Sigma_\xi^{-1}(s)(\mu_\xi(s) - s') + \log|\Sigma_\xi(s)|\big] \tag{1}$$

to ensure

$$\min_{i\in\mathcal{E}} \hat{p}_\xi^i(\hat{s}' \mid s) > \text{mean}_{i\in\mathcal{E}}\, \hat{p}_\xi^i(s' \mid s) \tag{2}$$

Here, $\mathcal{E}$ indicates the parameters of the neural network.

For the state-value function, training by minimizing the Bellman error [22]:

$$\mathcal{L}_V(\theta) = \mathbb{E}_{(s,r,s')\sim\mathcal{D}}\left[\left(r + \gamma V_\theta(s') - V_\theta(s)\right)^2\right] \tag{3}$$

For the reward function, use the Wasserstein GAN [23] [24]to fit according to the original method. The discriminator takes in the state, action, reward, next state, and determines whether this transition is from the dataset. The generator loss function is

$$\mathcal{L}_G(\phi) = \mathbb{E}_{z\sim p(z),(s,a,s')\sim\mathcal{D},\tilde{r}\sim G_\phi(z,s,a,s')}\left[D_\psi(s, a, s', \tilde{r})\right] \tag{4}$$

The discriminator loss function is:

$$\mathcal{L}_D(\psi) = \mathbb{E}_{s,a,r,s'\sim\mathcal{D}}[D_\psi(s, a, s', r)] - \mathbb{E}_{\substack{z\sim p(z)\\(s,a,s')\sim\mathcal{D}\\\tilde{r}_t\sim G_\phi(z,s,a,s')}}[D_\psi(s, a, s', \tilde{r})]. \tag{5}$$

After training, we only keep the generator, which predicts rewards when *stitching event* occurs.

For the inverse dynamics of generating new actions for connection, we train a conditional variational autoencoder (CVAE), consisting of an encoder q_{ω_1} and a decoder p_{ω_2} where ω_1 and ω_2 are the respective parameters of the neural networks. The training goal is

$$\begin{aligned}&\max_{\omega_1,\omega_2} \log p(a \mid s, \hat{s}', z) \\ &\quad \geq \max_{\omega_1,\omega_2} \mathbb{E}_{z\sim q_{\omega_1}}[\log p_{\omega_2}(a \mid s, \hat{s}', z)] \\ &\qquad - D_{\mathrm{KL}}[q_{\omega_1}(z \mid a, s, \hat{s}') \parallel P(z \mid s, \hat{s}')]\end{aligned} \tag{6}$$

This process ensures that the most plausible action is generated conditional on s and $\hat{s}'$.

B. Similarity Calculation

To quantify trajectory similarity, we first represent each trajectory by concatenating its constituent state and action sequences into a high-dimensional feature vector. We then employ Principal Component Analysis (PCA) to project these vectors into a lower-dimensional latent space, capturing the essential structural characteristics of the trajectories. Finally, a symmetric similarity matrix is constructed based on the projections in this reduced subspace, providing a comprehensive measure of the inter-trajectory relationships within the offline dataset.

a) Trajectory Representation: For each trajectory τ_i in the dataset, let s_i represent the state at the i-th time step, and a_i denote the corresponding action taken at that moment. At each time step, we concatenate the state-action pair (s_i, a_i) into a high-dimensional vector, embedding both the state and the action into a single representation. This process ensures that each time step is captured in a unified vector space, enabling the preservation of both the temporal dynamics of the state sequence and the corresponding control decisions made throughout the trajectory.

$$x_j = \mathrm{concat}\,(s_j, a_j) \tag{7}$$

Next, the entire trajectory τ_i is represented as a sequence of state-action vectors, where each vector combines the state s_i and the corresponding action a_i at the i-th time step.

$$X_i = [x_1, x_2, \ldots, x_n] \tag{8}$$

This step maps each trajectory into a high-dimensional space, providing a foundation for the subsequent calculation of trajectory similarities.

b) PCA Dimensionality Reduction: Given that the state and action vectors within a trajectory may have high dimensionality, directly computing the similarity between trajectories could lead to sparse high-dimensional data and substantial computational overhead. To mitigate these challenges, we employ Principal Component Analysis (PCA) [25] to reduce the dimensionality of the trajectory representations. This dimensionality reduction technique preserves the most significant variance in the data, effectively retaining the key structural information while eliminating less informative components. By projecting the high-dimensional trajectory vectors onto a lower-dimensional subspace, we not only enhance computational efficiency but also improve the robustness of similarity calculations.Through PCA, we map the representation X_i of each trajectory to a low-dimensional space X_i^{PCA}:

$$X_i^{\mathrm{PCA}} = X_i W_k \tag{9}$$

Where W_k is the eigenvector matrix of the first k principal components, and X_i^{PCA} is the trajectory representation after dimensionality reduction.

c) Trajectory Similarity Calculation: We evaluate the similarity between two trajectories, τ_i and τ_j, by computing the Euclidean distance between their respective PCA-reduced representations, X_i^{PCA} and X_j^{PCA}. Operating in the PCA subspace allows the metric to focus on the most significant structural features, where smaller distances signify higher trajectory similarity and shared underlying patterns:

$$\mathrm{S}_{ij} = EuclideanDistance\,(T_i, T_j) = \|X_i^{\mathrm{PCA}} - X_j^{\mathrm{PCA}}\| \tag{10}$$

d) Similarity Matrix: By calculating the similarity between all pairs of trajectories, we obtain a symmetric trajectory similarity matrix S, where each element S_{ij} represents the similarity between trajectory τ_i and trajectory τ_j. This matrix comprehensively quantifies the similarity relationship between trajectories in the dataset. Using this similarity matrix, we can selectively select the n trajectories with the lowest similarity to each other, thereby maximizing the diversity of the replay buffer while maintaining data quality, to promote effective learning in continual tasks.

$$S = \begin{bmatrix} S_{11} & S_{12} & \ldots & S_{1N} \\ S_{21} & S_{22} & \ldots & S_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ S_{N1} & S_{N2} & \ldots & S_{NN} \end{bmatrix} \tag{11}$$

C. Multi-head Policy Network

We utilize a multi-head architecture for the policy network π, comprising a shared backbone π_s and task-specific heads $\{h_i\}_{i=1}^n$. The policy for the i-th task is defined by the pair $[\pi_s, h_i]$, where π_s extracts common features across tasks and h_i specializes in task-specific requirements. This design facilitates efficient knowledge transfer while maintaining individual task flexibility.

Conventionally, rehearsal-based CORL balances stability and plasticity by training the current task i via offline RL (e.g., CQL) on $[\pi_s, h_i]$ while replaying experiences from previous buffers $B_{1:i-1}$. However, both literature and our empirical findings indicate significant performance decay as task number and complexity increase. We attribute this degradation to the naive application of offline RL objectives directly to the multi-head structure, which fails to handle the intricate dynamics of sequential task learning.

As is well-known, reinforcement learning algorithms based on the Bellman equation iteratively approximate the optimal policy π^* through a trial-and-error process [26]:

$$Q(s,a) = r(s,a) + \gamma * \max_{a'} Q\left(s', a'\right) \tag{12}$$

However, when applying this approach in the context of continual learning with a multi-head network, this iterative training process can overwrite the network's shared component π_s. The original intent behind using π_s is to capture the shared knowledge across the N tasks. Yet, when employing the Bellman equation to learn the i-th task, this training methodology can inadvertently disrupt or degrade the previously learned shared knowledge, causing significant performance degradation on earlier tasks. This issue arises because the reinforcement learning process updates the shared component π_s to undermine its generalization capability across multiple tasks, leading to a catastrophic forgetting effect.

Therefore, we adopt the Dual Behavior Cloning (DBC) architecture in [27]: we will not directly train the RL algorithm on $[\pi_s, h_i]$, but first use an additional MLP network to learn the current task, obtain the policy μ_i, and then learn the policy to our multi-head neural network through supervised learning, giving a formal loss function:

$$\begin{aligned} L_\pi = \mathbb{E}_{(s,a,s')\sim\mathcal{D}_i}\left(\pi_i(s,a) - \mu_i(s,a)\right)^2 \\ + \lambda_r \frac{1}{i-1}\sum_{j=1}^{i-1} \mathbb{E}_{(s,a)\sim\mathcal{B}_j}\left(\pi_j(s) - a\right)^2 \end{aligned} \tag{13}$$

where λ_r is a coefficient for the behavioral cloning term. It is important to note that the continual learning policy π aims to replicate the behavior of both μ_i simultaneously, the current task's policy, and the previous experiences stored in the buffer B_1 to B_{i-1}.

D. Algorithm Summary

When considering a new task T_n, we first utilize the DBC framework to learn the two policy networks μ_n and π. Then, we will train the MBTS components until convergence and use them to perform data augmentation operations to obtain $\hat{D}$. Finally, we select the top $2m$ trajectories with the highest cumulative rewards from the original datasets D and $\hat{D}$, and then use PCA to select the n most dissimilar trajectories, adding them to the replay buffer. The overall process is given in Algorithm 1.

Algorithm 1: SS-RL Algorithm

Input: Number of tasks N; Tasks $\mathcal{T} = \{T_1, \ldots, T_N\}$; Dataset D
Output: Multi-head policy network π with heads $\{h_1, \ldots, h_N\}$

for $i = 1$ **to** N **do**
  Initialize offline dataset D_i, replay buffer $B_i = \emptyset$, and new head h_i;
  Initialize MBTS components: $\hat{p}^n_\xi$, V_θ, $G(\phi)$, $D(\psi)$, and $(q_{\omega_1}, p_{\omega_2})$;
  `// Phase 1: Policy Training`
  while *not converged* **do**
    Update μ_i and Q_i via TD3+BC using D_i;
    Update π via Eq. (13) by replaying $B_0, \ldots, B_{i-1}$;
  end
  `// Phase 2: MBTS Components Update`
  while *not converged* **do**
    Update $\hat{p}^n_\xi, V_\theta$ via Eq. (1), (3);
    Update $G(\phi), D(\psi)$ via Eq. (4)-(5);
    Update $(q_{\omega_1}, p_{\omega_2})$ via Eq. (6);
  end
  `// Phase 3: Data Enhancement`
  Enhance D_i to $\hat{D}_i$ using MBTS component;
  Select top m trajectories $d_i, \hat{d}_i$ from $D_i, \hat{D}_i$;
  Calculate similarity matrices $\mathcal{S}$ via PCA:; $\mathcal{S}_{ij} = \text{Similarity}_{\text{PCA}}(\tau_i, \tau_j)$;
  Select $n/2$ dissimilar trajectories and add to B_i;
end

V. Experiments

In this section, we evaluate the performance of our proposed method within the Continual Offline Reinforcement Learning (CORL) setting. Extensive experiments are conducted to demonstrate its effectiveness and assess its capacity to maintain stability and plasticity simultaneously.

A. Setup

Baselines We evaluate the proposed method against representative continual learning (CL) baselines, categorized into: rehearsal-based (CRIL, DGR, t-DGR), regularization-based (EWC, Finetune), and architecture-based (PackNet, Multi-task) methods:

- **CRIL** [28]: A rehearsal-based method that aligns feature representations via knowledge distillation to enhance knowledge transfer.
- **DGR** [29]: A generative replay approach that uses a generator (GAN/VAE) to synthesize pseudo-samples of past tasks, avoiding raw data storage.
- **t-DGR** [30]: A variant of DGR that focuses on improving the efficiency and scheduling of generative replay.
- **EWC** [31]: A regularization method that penalizes changes to weights important for previous tasks, estimated by the Fisher Information Matrix.
- **PackNet** [32]: An architecture-based method that isolates task-specific parameters through iterative pruning and weight masking.
- **Finetune**: A naive baseline that sequentially trains tasks without any forgetting mitigation, serving as a performance lower bound.
- **Multi-task**: This approach assumes that data from all tasks can be acquired simultaneously, and a globally optimal model can be trained on the joint dataset. This is

an ideal scenario without sequential learning constraints and forgetting problems.

Offline Sequential Datasets We evaluate our method on three distinct task sets derived from widely-used continuous control environments, following the setup presented in [33]:

- **Ant-2D Direction**: Train a simulated ant, typically with 8 articulated joints, to run towards specific target directions in the 2D plane.
- **Walker-2D Direction**: Train a simulated bipedal walker, usually with 6 articulated joints, to maintain balance while walking towards designated 2D directions.
- **Half-Cheetah Velocity**: Train a simulated planar cheetah-like robot, typically with 6 articulated joints, to run forward along the x-axis at specific target velocities.

For each environment, we created task sequences ($T_1 - T_5$) by randomly sampling five distinct tasks. To explore the impact of data quality, we collected datasets corresponding to Medium (M) and Medium-Random (M-R) qualities by selecting data from different time periods within an online training buffer.

Metrics Following [34], we adopt the average performance (PER) and the backward transfer (BWT) as evaluation metrics:

$$\begin{aligned} \mathrm{PER} &= \frac{1}{N}\sum_{n=1}^{N} a_{N,n}, \\ \mathrm{BWT} &= \frac{1}{N-1}\sum_{n=1}^{N-1}(a_{n,n} - a_{N,n}) \end{aligned} \tag{14}$$

where $a_{i,j}$ represents the final cumulative reward obtained on task j after the agent has completed training sequentially up to task i. This value is used to PER, for which higher values indicate a stronger ability to learn new tasks, and BWT, for which lower absolute values signify less forgetting of previous tasks. Together, these metrics evaluate the capacity for learning new skills while maintaining performance on acquired ones.

B. Overall Results

This section presents a comprehensive analysis of the performance metrics for all evaluated methods across three distinct environments and under two offline data quality conditions (M and M-R Quality), with detailed results tabulated in Tables I and II.

TABLE I: Comparison Experimental Results (M-R Quality)

Model	**Ant_dir**		**Walker_dir**		**Cheetah_vel**	
	PER ↑	BWT ↓	PER ↑	BWT ↓	PER ↑	BWT ↓
PackNet	1020.2	500.3	1031.5	314.5	-404.8	34.4
EWC	1016.1	450.2	953.6	342.1	-451.5	48.3
CRIL	1340.4	350.1	1449.4	474.3	-334.2	94.5
Finetune	840.3	602.7	851.5	722.4	-569.4	100.7
DGR	1390.5	337.5	1352.4	481.1	-400.5	69.1
t-DGR	1408.3	302.5	1507.6	296.4	-372.6	57.4
Multitask	**1593.4**	–	1583.2	–	**-234.5**	–
SS-RL	1540.3	**239.4**	**1601.4**	**268.5**	-257.7	**32.3**

TABLE II: Comparison Experimental Results (M Quality)

Model	**Ant_dir**		**Walker_dir**		**Cheetah_vel**	
	PER ↑	BWT ↓	PER ↑	BWT ↓	PER ↑	BWT ↓
PackNet	1149.5	534.7	1204.1	345.7	-384.4	39.4
EWC	1143.1	493.4	1149.0	492.3	-361.1	68.4
CRIL	1474.2	371.3	1584.7	510.4	-336.8	92.1
Finetune	1004.5	565.4	1042.3	756.3	-446.1	94.5
DGR	1499.4	391.6	1512.7	550.2	-374.4	71.3
t-DGR	1549.5	337.5	1659.1	361.4	-300.4	61.7
Multitask	**1737.4**	–	1693.7	–	**-204.2**	–
SS-RL	1647.8	**307.5**	**1756.1**	**317.4**	-221.5	**41.2**

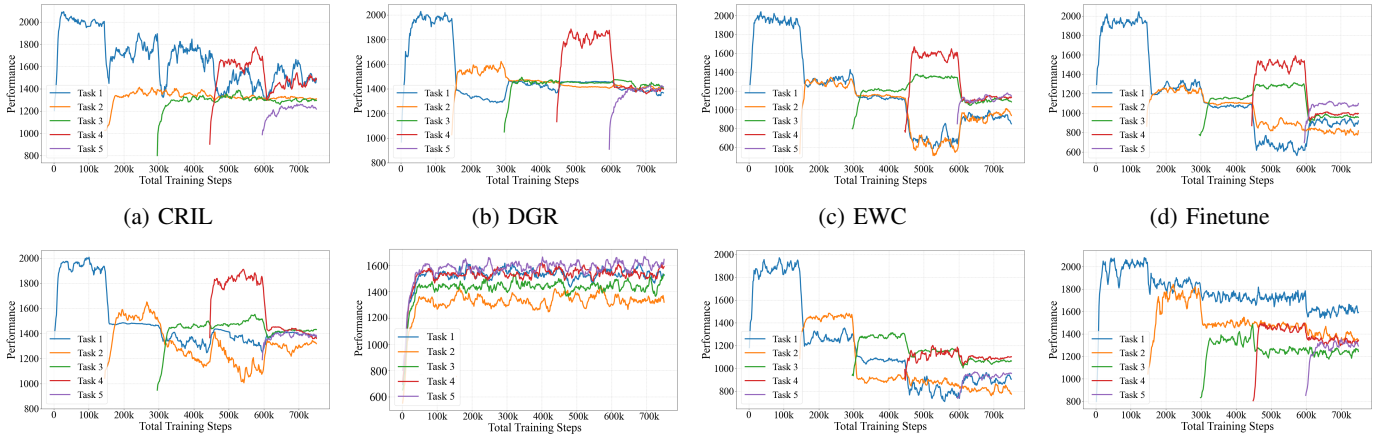
A careful examination of these results leads to several key observations. First, our proposed method consistently outperforms the baseline approaches in most experimental settings. This robust performance advantage underscores our proposed framework's general effectiveness and superiority. Secondly, we observe that the performance degradation of our method when transitioning from the higher-quality 'M' dataset to the lower-quality 'M-R' dataset is minimal; its performance remains remarkably comparable across both data regimes. This finding strongly suggests that the integrated trajectory stitching mechanism effectively enhances the quality of the initial offline dataset and the subsequently utilized replay buffer, thereby mitigating the negative impact of suboptimal data and consequently boosting overall model performance.

Furthermore, to more intuitively assess learning efficiency and forgetting patterns, the learning curves for all methods are presented in Fig. ref curve. Analyzing these trajectories helps differentiate how effectively each approach learns new tasks while preserving prior knowledge over time. For brevity and due to page constraints, we only include the curves for the Ant-Dir task within the main text as a representative case.

C. Ablation Study Analysis

To evaluate the specific contributions of the Model-Based Trajectory Stitching (MBTS) and PCA-based diversity selection modules to mitigating catastrophic forgetting and enhancing model performance, we conduct an ablation study by comparing three variants of the SS-RL framework. The configurations are defined as follows:

- **Vanilla ER**: A baseline experience replay method that randomly samples trajectories from the original offline dataset into the replay buffer without any enhancement or diversity filtering.
- **SS-RL w/o PCA**: This variant utilizes MBTS for data augmentation but selects trajectories based solely on cumulative rewards, without considering trajectory similarity.
- **SS-RL w/o MBTS**: This variant omits trajectory stitching and applies PCA-based similarity metrics directly to the original dataset to select "high-reward and low-similarity" trajectories.
- **SS-RL**: The complete framework integrating both MBTS-based augmentation and PCA diversity selection.



(a) CRIL (b) DGR (c) EWC (d) Finetune

(e) t-DGR (f) Multitask (g) PackNet (h) SS-RL

Fig. 2: Process of learning five sequential tasks in the Ant-Dir environment, where our method is compared against seven baselines on the medium-random dataset. Each task was trained for 150K steps under a continual learning protocol.

TABLE III: Ablation Results of Core Modules (M-R Quality)

Model	**Ant_dir**		**Walker_dir**		**Cheetah_vel**	
	PER ↑	BWT ↓	PER ↑	BWT ↓	PER ↑	BWT ↓
Vanilla ER	1202.4	417.9	1273.8	524.7	-431.2	93.2
SS-RL w/o PCA	1442.1	266.6	1477.5	314.2	-318.9	41.7
SS-RL w/o MBTS	1273.5	310.4	1338.7	396.0	-374.2	73.9
SS-RL	**1540.3**	**239.4**	**1601.4**	**268.5**	**-257.7**	**32.3**

The experimental results in Table III demonstrate that MBTS is the primary driver for elevating the performance upper bound. The fact that *SS-RL w/o PCA* outperforms *SS-RL w/o MBTS* suggests that, under the constraints of sub-optimal offline data, enhancing data quality via MBTS contributes more significantly to overall performance gains.

Furthermore, the PCA-based selection mechanism serves as a critical regulator for model stability. Although *SS-RL w/o PCA* achieves high PER scores, its Backward Transfer (BWT) metrics remain inferior to the full SS-RL framework, indicating more pronounced forgetting. In summary, the synergy between MBTS and PCA selection allows the SS-RL framework to significantly enhance model plasticity while maintaining robust system stability during multi-task continual learning.

D. Parameter Analysis

1) Size of Buffer B_n: We first analyze the sensitivity of our model to the replay buffer capacity B_n, a critical hyperparameter in CORL that governs the stability-plasticity trade-off. B_n dictates the volume and diversity of past experiences available for rehearsal, fundamentally influencing the agent's ability to mitigate catastrophic forgetting while learning from sequential offline datasets. To determine the optimal balance between memory efficiency and performance, we evaluate B_n across capacities of 50, 100, 150, 200, and 300 trajectories. The empirical results, summarized in Table IV, quantify the impact of buffer size on the final model performance.

TABLE IV: Performance Comparison with Different Buffer Sizes (M-R Quality)

Model	**Buffer**	**Ant_dir**		**Walker_dir**		**Cheetah_vel**	
		PER ↑	BWT ↓	PER ↑	BWT ↓	PER ↑	BWT ↓
SS-RL	50	1442.6	327.9	1400.8	318.7	-383.9	71.6
	100	1521.4	295.7	1585.1	328.5	-324.1	37.3
	150	1540.3	239.4	1601.4	268.5	-257.7	32.3
	200	1563.2	204.5	1647.2	240.1	-246.1	33.4
	300	**1615.7**	**199.4**	**1692.1**	**220.7**	**-243.5**	**30.4**

Our findings suggest that a buffer size equivalent to 150 trajectories strikes an effective balance, delivering robust performance without significant gains observed from further capacity increases.

2) Replay Coefficient λ_r : The coefficient λ_r in Eq.13 is a critical hyperparameter, balancing the preservation of knowledge from previous tasks against learning the policy π_n for the current task n. We set $\lambda_r = 1.0$ in our primary CORL experiments, aligning with common replay-based approaches and achieving the strong performance reported earlier. To assess sensitivity to this parameter, we evaluated performance with λ_r set to 0.3, 0.7, 1.0, 3.0, and 5.0; as detailed in Table V. The empirical results confirm that $\lambda_r = 1.0$ provides the most balanced performance within the tested range, validating its consistent application in this paper.

VI. Conclusion

In this paper, we introduce a robust framework named SS-RL to address catastrophic forgetting and data quality challenges in Continual Offline Reinforcement Learning. Initially, the method employs Model-Based Trajectory Stitching (MBTS) to augment static datasets by synthesizing high-quality transitions from sub-optimal data. Subsequently, Principal Component Analysis (PCA) is utilized to evaluate trajectory similarity, selecting diverse and representative samples for

TABLE V: Performance Comparison with Different λ_r Values (M-R Quality)

Model	λ_r	Ant_dir		Walker_dir		Cheetah_vel	
		PER ↑	BWT ↓	PER ↑	BWT ↓	PER ↑	BWT ↓
SS-RL	0.3	1437.3	**209.6**	1425.5	295.6	-342.4	79.1
	0.7	**1598.8**	274.6	**1645.3**	270.5	-273.8	**31.3**
	1.0	1540.3	239.4	1601.4	**268.5**	**-257.7**	32.3
	3.0	1449.1	307.1	1401.7	289.7	-308.3	58.2
	5.0	1319.3	234.2	1219.2	257.6	-328.9	77.4

the replay buffer. Furthermore, a multi-head policy network is incorporated to facilitate efficient knowledge transfer across sequential tasks. Lastly, an experience replay mechanism is applied to enable stable long-term learning by reusing the optimized historical data. Experimental results demonstrate that SS-RL significantly improves learning efficiency and generalization ability compared to state-of-the-art baselines.

ACKNOWLEDGMENT

This work is supported in part by the International Cooperation Project of Jilin Province (No. 20250205015GH).

REFERENCES

[1] B. Goertzel, "Artificial general intelligence: concept, state of the art, and future prospects," *Journal of Artificial General Intelligence*, vol. 5, no. 1, p. 1, 2014.

[2] S. Sonko, A. O. Adewusi, O. C. Obi, S. Onwusinkwue, and A. Atadoga, "A critical review towards artificial general intelligence: Challenges, ethical considerations, and the path forward," *World Journal of Advanced Research and Reviews*, vol. 21, no. 3, pp. 1262–1268, 2024.

[3] A. Cahill, "Catastrophic forgetting in reinforcement-learning environments," Ph.D. dissertation, University of Otago, 2011.

[4] T. Korbak, H. Elsahar, G. Kruszewski, and M. Dymetman, "On reinforcement learning and distribution matching for fine-tuning language models with no catastrophic forgetting," *Advances in Neural Information Processing Systems*, vol. 35, pp. 16 203–16 220, 2022.

[5] J. Zhao, W. Zhao, B. Deng, Z. Wang, F. Zhang, W. Zheng, W. Cao, J. Nan, Y. Lian, and A. F. Burke, "Autonomous driving system: A comprehensive survey," *Expert Systems with Applications*, vol. 242, p. 122836, 2024.

[6] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[7] M. Hu, J. Zhang, L. Matkovic, T. Liu, and X. Yang, "Reinforcement learning in medical image analysis: Concepts, applications, challenges, and future directions," *Journal of Applied Clinical Medical Physics*, vol. 24, no. 2, p. e13898, 2023.

[8] C. Yu, J. Liu, S. Nemati, and G. Yin, "Reinforcement learning in healthcare: A survey," *ACM Computing Surveys (CSUR)*, vol. 55, no. 1, pp. 1–36, 2021.

[9] T. Zhang, K. Z. Shen, Z. Lin, B. Yuan, X. Wang, X. Li, and D. Ye, "Replay-enhanced continual reinforcement learning," *arXiv preprint arXiv:2311.11557*, 2023.

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

[11] J. Hu, Y. Sun, S. Huang, S. Guo, H. Chen, L. Shen, L. Sun, Y. Chang, and D. Tao, "Instructed diffuser with temporal condition guidance for offline reinforcement learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–11, 2026.

[12] J. Hu, S. Huang, Z. Yang, S. Hu, L. Shen, H. Chen, L. Sun, Y. Chang, and D. Tao, "Analytic energy-guided policy optimization for offline reinforcement learning," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. [Online]. Available: https://openreview.net/forum?id=lcUpF96w7Z

[13] Y. Bu, S. Zou, Y. Liang, and V. V. Veeravalli, "Estimation of kl divergence: Optimal minimax rate," *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2648–2674, 2018.

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

[15] C. A. Hepburn and G. Montana, "Model-based trajectory stitching for improved offline reinforcement learning," *arXiv preprint arXiv:2211.11603*, 2022.

[16] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (pca)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[17] K. Khetarpal, M. Riemer, I. Rish, and D. Precup, "Towards continual reinforcement learning: A review and perspectives," *Journal of Artificial Intelligence Research*, vol. 75, pp. 1401–1476, 2022.

[18] J. Hu, S. Huang, L. Shen, Z. Yang, S. Hu, S. Tang, H. Chen, L. Sun, Y. Chang, and D. Tao, "Tackling continual offline rl through selective weights activation on aligned spaces," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[19] Y. Wang, C. Yang, Y. Wen, Y. Liu, and Y. Qiao, "Critic-guided decision transformer for offline reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 14, 2024, pp. 15 706–15 714.

[20] J. Buckman, D. Hafner, G. Tucker, E. Brevdo, and H. Lee, "Sample-efficient reinforcement learning with stochastic ensemble value expansion," *Advances in neural information processing systems*, vol. 31, 2018.

[21] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," *Advances in neural information processing systems*, vol. 31, 2018.

[22] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[23] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.

[24] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[25] K. L. Elmore and M. B. Richman, "Euclidean distance as a similarity metric for principal component analysis," *Monthly weather review*, vol. 129, no. 3, pp. 540–549, 2001.

[26] J. Clifton and E. Laber, "Q-learning: Theory and applications," *Annual Review of Statistics and Its Application*, vol. 7, no. 1, pp. 279–301, 2020.

[27] S. Gai, D. Wang, and L. He, "Oer: offline experience replay for continual offline reinforcement learning," in *ECAI 2023*. IOS Press, 2023, pp. 772–779.

[28] C. Gao, H. Gao, S. Guo, T. Zhang, and F. Chen, "Cril: Continual robot imitation learning via generative and prediction model. in 2021 ieee," in *RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 6747–5754.

[29] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Advances in neural information processing systems*, vol. 30, 2017.

[30] W. Yue, B. Liu, and P. Stone, "t-dgr: A trajectory-based deep generative replay method for continual learning in decision making," *arXiv preprint arXiv:2401.02576*, 2024.

[31] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[32] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 7765–7773.

[33] E. Mitchell, R. Rafailov, X. B. Peng, S. Levine, and C. Finn, "Offline meta-reinforcement learning with advantage weighting," in *International Conference on Machine Learning*. PMLR, 2021, pp. 7780–7791.

[34] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," *Advances in neural information processing systems*, vol. 30, 2017.