Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn engineering.cae.cn www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Significance extraction based on data augmentation for reinforcement learning*

Yuxi HAN†, Dequan LI†‡, Yang YANG

Faculty of Artificial Intelligence, Anhui University of Science and Technology, Huainan 232000, China

†E-mail: hanyuxi0712@163.com; leedqcpp@126.com

Received May 17, 2024; Revision accepted Sept. 18, 2024; Crosschecked Feb. 13, 2025; Published online Mar. 6, 2025

Abstract: Deep reinforcement learning has shown remarkable capabilities in visual tasks, but it does not have a good generalization ability in the context of interference signals in the input images; this approach is therefore hard to be applied to trained agents in a new environment. To enable agents to distinguish between noise signals and important pixels in images, data augmentation techniques and the establishment of auxiliary networks are proven effective solutions. We introduce a novel algorithm, namely, saliency-extracted Q-value by augmentation (SEQA), which encourages the agent to explore unknown states more comprehensively and focus its attention on important information. Specifically, SEQA masks out interfering features and extracts salient features and then updates the mask decoder network with critic losses to encourage the agent to focus on important features and make correct decisions. We evaluate our algorithm on the DeepMind Control generalization benchmark (DMControl-GB), and the experimental results show that our algorithm greatly improves training efficiency and stability. Meanwhile, our algorithm is superior to state-of-the-art reinforcement learning methods in terms of sample efficiency and generalization in most DMControl-GB tasks.

Key words: Deep reinforcement learning; Visual tasks; Generalization; Data augmentation; Significance; DeepMind Control generalization benchmark

https://doi.org/10.1631/FITEE.2400406 **CLC number:** TP391.4

1 Introduction

Visual observation based reinforcement learning (RL) (Arulkumaran et al., 2017) has achieved tremendous success in various fields such as gaming (Mnih et al., 2013), robotic manipulation (Levine et al., 2016; Kalashnikov et al., 2018; Nair et al., 2018), autonomous navigation (Zhu et al., 2016; Yang W et al., 2019), and natural language processing (NLP) (Luketina et al., 2019). This learning approach continually optimizes decision-making processes by enabling the agent to interact with the environment to maximize long-term objectives. Despite excelling in specific virtual scenarios, RL algorithms face numerous challenges when applied in the real world. The complexity and variability of real-world environments mean that even a well-trained agent in one environment may struggle to adapt to new environments (Zhao et al., 2022). Research works indicate severe deficiencies in the generalization capability of agents (Farebrother et al., 2018; Cobbe et al., 2019; Gamrian and Goldberg, 2019; Song et al., 2020), as agents may fail to ignore noise factors and to focus attention on critical state features. Consequently, agents are unable to explore effectively in unknown environments (Henderson et al., 2017).

Recent research advancements have demonstrated that the generalization capability of RL models can be effectively enhanced through domain randomization (Tobin et al., 2017; Pinto et al., 2018) and data augmentation techniques (Yarats et al.,

‡ Corresponding author

* Project supported by the Academic and Technical Leaders and Backup Candidates Program of Anhui Province, China (No. 2019h211) and the Natural Science Foundation of Anhui Province, China (No. 2208085ME128)

ORCID: Yuxi HAN, https://orcid.org/0009-0008-5319-006X; Dequan LI, https://orcid.org/0000-0002-4329-864X

2021; Zhou, 2024). By increasing the randomness and diversity of the training data, these methods effectively help the agents better adapt to the new environment and thus enable the agents to try more states. Despite continuous efforts to improve the generalization performance of RL models (Hansen et al., 2021b; Yarats et al., 2021; Bertoin et al., 2022), existing algorithms still have shortcomings in focusing on important pixels and exploring the state space. These issues limit their effectiveness in real-world applications. Therefore, data augmentation techniques play a crucial role in driving the development of large-scale RL. Future research needs to further explore how to leverage these techniques more effectively such that agents can better understand and adapt to complex and dynamic real-world environments.

In this study, we propose a novel algorithm named saliency-extracted Q-value by augmentation (SEQA). Its core lies in guiding the agent to focus on important pixels and use noise augmentation in state space exploration, with the aim to enhance the learning and generalization capabilities of the agents. By mixing attention mechanisms, our algorithm enables the agents to identify and extract key features more quickly and accurately, greatly enhances the estimation of Q-value by convolutional neural networks, and improves the training efficiency of agents under the condition of extremely low overhead without interference.

SEQA is applied to soft actor–critic (SAC) agents and extensively experimented on the DeepMind Control Suite (Tassa et al., 2018), including testing on the DMControl generalization benchmark (DMControl-GB) (Hansen and Wang, 2021). In most DMControl tasks, our algorithm achieves superior sample efficiency, performance, and generalization capability compared to state-of-the-art methods.

The results indicate that the SEQA algorithm holds significant practical value in the field of RL, particularly in enhancing the learning efficiency and generalization capability of agents in complex tasks. By precisely guiding the agent to focus on important state features and effectively exploring the state space, the SEQA algorithm provides an effective strategy for addressing RL problems in the real world. It serves as an efficient nonpolicy RL data augmentation framework. SEQA consists of three essential components, as follows:

1. By combining attention mechanism and a mask decoder, key pixel information is extracted and encoded into features, making the agents aware of which pixels are important, thereby enabling more precise decision-making.

2. Noise augmentation is applied to observation states and state rewards to encourage exploration and increase the diversity of training data.

3. SEQA performs consistency regularization on the Q-values of the augmented states and the Q-values of unaugmented subsequent states. For SEQA using the actor–critic algorithm, the critic optimizes the actor network through shared parameters.

2 Related works

2.1 Representation learning

Chen et al. (2020) introduced nonlinear transformation between representation and contrast loss and conducted a comprehensive study of data enhancement (such as random cropping and image distortion). Fu et al. (2021) leveraged cooperative reconstruction to learn visual features and separate reward features from the background. Wang et al. (2021) focused on extracting visual foreground to provide clear and unchanging visual representation for strategy learning, making it easier for agents to learn task-related stable features. Gelada et al. (2019) trained their model by minimizing two manageable losses, the reward prediction loss and the distributed prediction loss of the next potential state, enabling the model to better understand the dynamic changes in the environment and predict future rewards to make more informed decisions.

2.2 Visual learning in RL

Learning task-relevant feature representations from raw visual inputs is often more challenging than learning directly from vectorized features (Ze et al., 2023). Recent research suggests that the success of self-supervised learning methods in visual representation learning can be leveraged in RL to improve algorithm generalization ability and sample efficiency (Yarats et al., 2019; Laskin et al., 2020a; Hansen and Wang, 2021; Hansen et al., 2021a; Zhang A et al., 2021). Yarats et al. (2019) proposed that joint training of autoencoders with RL can

significantly improve the sample efficiency of model-free RL. Laskin et al. (2020a) achieved nearly equivalent sample efficiency between model-free RL and model-based RL by combining autoencoders with RL to assist contrastive learning. Zhang A et al. (2021) proposed to learn behavioral similarity embedding through dual simulation metrics and contrast learning, which helps agents identify and imitate effective strategies. Zhou (2024) modified the underlying deep deterministic policy gradient (DDPG) algorithm to distributed distributional DDPG (D4PG) and integrated double Q-learning to reduce the overestimation bias of the critic value function, which improves the robustness of the DDPG algorithm to hyperparameters. Hansen and Wang (2021) trained agents to learn visual invariance through assisted prediction tasks. While auxiliary tasks have been shown to enhance RL performance, their effectiveness often relies on specific RL tasks (Lin et al., 2019; Hansen et al., 2021a). Our algorithm achieves satisfactory results on multiple tasks without the need to select auxiliary tasks according to specific RL tasks, demonstrating the potential of adaptive auxiliary learning strategies.

2.3 Generalization in RL

Although RL algorithms have achieved significant success in solving complex tasks, they tend to overfit the training environment (Farebrother et al., 2018; Cobbe et al., 2019), leading to performance deterioration in new environments and presenting significant deficiencies in generalization (Kirk et al., 2023). Xing et al. (2021) improved generalization capability by adjusting agents' learning strategy through domain adaptation. OpenAI et al. (2019) enhanced generalization capability by training control policies through automatic domain randomization. Yang SZ et al. (2023) successfully adapted a visualization model based strategy during the test and experimented in four scenarios. Zhang A et al. (2018) measured generalization errors in continuous control environments by injecting training diversity. These studies suggest that agents may fail to achieve satisfactory rewards when facing environmental changes or untrained environments. We propose a novel algorithm that focuses attention on important pixels in policy learning and enhances generalization capability through loss function constraints. This not only enables agents to identify critical state features more effectively but also enhances their adaptability and robustness in diverse environments.

2.4 Data augmentation and randomization

Laskin et al. (2020b) showed that training RL agents with augmented data can bring significant benefits. Yarats et al. (2021) further improved learning performance by averaging over randomly cropped Q-functions and their targets. Hansen and Wang (2021) added self-supervised learning to the SAC algorithm and applied data enhancement only to the Q-value estimation; further, they proposed a method that strictly uses the enhanced data to calculate the Q-target. Hansen et al. (2021b) proposed a deep Q-learning data augmentation framework aimed at enhancing learning efficiency and generalization performance through augmentation techniques. Almuzairee et al. (2024) increased the input of both actor and critic, reducing the instability of training. Lee et al. (2020) enhanced data diversity by applying random convolutional layers during training to modify the texture of observation results. Recent data augmentation studies (Yarats et al., 2021; Hansen et al., 2023) have shown that some weak augmentation techniques, such as random translation, improve sample efficiency but sacrifice generalization capability. On the other hand, some strong augmentation techniques, such as random convolution, enhance generalization ability but at the cost of reduced sample efficiency. In this paper, we propose a novel algorithm that successfully improves generalization ability while maintaining sample efficiency. Our algorithm demonstrates how to balance the intensity of data augmentation to enhance the generalization performance of agents across various tasks without sacrificing sample efficiency. This algorithm provides a new perspective on the trade-off between sample efficiency and generalization ability in RL, laying the foundation for future research and applications.

2.5 Masking in visual RL

In visual RL, masking techniques are used to enhance agents' generalization capability. By masking certain parts of the input, agents are compelled to focus on more important features in the environment, reducing the interference of irrelevant information and enhancing agents' ability to recognize

critical states. Yu et al. (2022) adopted the method of randomly masking part of the input and then reconstructed these masked pixels through the auxiliary loss function. This strategy not only improves the generalization ability of the agent but also increases the robustness of the model to the input data. Bertoin et al. (2022) established attribution maps to attribute decisions to inputs, using binarized attribute mappings to predict saliency maps of Q-values. Grooten et al. (2024) designed a lightweight network generation mask to mask interference, enabling the agents to explore in unknown environments.

2.6 Noise augmentation

Recent research has shown that introducing noise during the training process allows algorithms to make better decisions under uncertainty and variability. Noise augmentation helps agents adapt better to new environments, enhancing their robustness and generalization ability when facing unknown situations (Zhang H et al., 2020; Antotsiou et al., 2021). Sinha et al. (2022) explored the application of various data augmentation methods to seven nonvisual RL problems, with particular focus on three noise augmentation techniques. Khraishi and Okhrati (2023) proposed a generic wrapper for noise augmentation, which increases the diversity of training samples, facilitating broader exploration by the agents and thus improving the generalization ability of learned policies.

3 Preliminaries

3.1 Problem formulation

We formulate the image-based control problem as an infinite horizontal partially observable Markov decision process (POMDP) (Kaelbling et al., 1998; Kurniawati, 2022). A Markov decision process (MDP) is represented by a tuple $M=\langle O,A,P,r,\gamma\rangle$, which is used to describe the decision-making process of an agent in an uncertain environment, where O represents the high-dimensional state observation space (image pixels), A represents the action space, P represents the state transition function from state s_t to state s_{t+1}, r represents the reward function $O\times A\to\mathbb{R}$, and $\gamma\in[0,1)$ represents the discount factor, which is used to regulate future rewards. The entire MDP can be described as the agent interacting with the environment at each time step, obtaining a state $s_t=\{o_t,o_{t-1},o_{t-2},...\}$, taking an action $a_t\in A$ in the current state, then making a decision with probability $P(s_{t+1}|s_t,a_t)$ to transition to the next state s_{t+1}, and receiving a reward $r_t=r(s_t,a_t)$. The objective of the agent is to maximize the discounted expected return $E[\sum_{t=0}^{\infty}\gamma^t r_t|a_t\sim\pi(\cdot|s_t),s_{t+1}\sim P(\cdot|s_t,a_t)]$, in which π is parameterized by a set of learnable parameters, termed θ (Sutton and Barto, 1998).

Our further research goal is to explore how to learn the structural parameters π_θ of the MDP so that the agent can achieve good generalization ability even when the observation space is disturbed. Within the MDP framework, we assume that the agent can adapt to perturbations by learning the structure and dynamic properties of the environment. We focus on a generalization $\bar{M}=\langle\bar{S},A,P,r,\gamma\rangle$, wherein the space $\bar{M}\sim M$ and states $\bar{s_t}\in\bar{S}$ of the MDP are constructed from observations $\bar{o_t}\in\bar{O}$, with O being a subset of the perturbed observation space $\bar{O}$. In many environments wherein $o_t=s_t$, agents can have full observability, allowing them to capture key features of the environment while maintaining robustness to perturbations in the observation space.

3.2 SAC algorithm

SAC is a model-free, off-policy actor–critic algorithm that aims to learn the state–action value function Q_θ and estimate the optimal state–action value function Q^*. SAC optimizes the policy to obtain higher cumulative returns, while maximizing the entropy target. The critic and shared encoder have the same parameters $\phi_{\text{tgt}}\leftarrow(1-\tau)\phi_{\text{tgt}}+\tau\theta$ in the target network. In this work, we describe our method based on the SAC.

4 Method

We propose the algorithm SEQA, a general deep RL framework focused on enhancing visual learning performance through data and noise augmentation. The core of SEQA lies in leveraging attention mechanisms to identify and extract key features, which further map these features as masks to the input states using a mask decoder network, followed by consistent regularization of the Hadamard product

of the masked image with the original image and the original state image. Additionally, we introduce noise perturbations to observation states and state rewards, aiming to increase the diversity of training data and encourage exploration. The design of SEQA enables seamless integration with any standard off-policy RL algorithm without the need for additional parameters or complex forward pass processes.

4.1 Overview of the framework

Fig. 1 illustrates an overview of the SEQA framework. We first apply random shifting and overlay augmentation to the images and then input the augmented images into an encoder network equipped with a mixed attention mechanism. The resulting feature vector is fed into a mask decoder, which generates a new state observation $\tilde{s}$ by performing Hadamard multiplication between the obtained masked image M and the original image s. Subsequently, the state observation $\tilde{s}$ is inputted into a state encoder f_θ for computation of the Q-function Q_θ. For state–action pairs $(\tilde{s}, a_t)$, we can compute the corresponding Q-values by predicting q_t, while we define a target state-value function q_t^{tgt} as the target Q-value, wherein the parameter φ is an exponential moving average of θ. To ensure consistency between the predicted Q-values and the target Q-values, we regularize q_t and q_t^{tgt} with consistency regularization and update the parameters of the critic network using the consistency regularization objective, ensuring that the critic can more accurately assess the value of state–action pairs.

$$q_t \triangleq Q_\theta(f_\theta(\tilde{s}), a_t), \tag{1}$$

$$q_t^{\mathrm{tgt}} \triangleq r(s_t, a_t) + \gamma \max_{a'_t} Q_\varphi^{\mathrm{tgt}}(f_\varphi(s_{t+1}), a'). \tag{2}$$

4.2 Mixed attention mechanism

Hansen et al. (2021b) integrated neural networks with a visual Transformer (ViT) to enhance sample efficiency. While ViT performs well in handling large-scale image data, it requires substantial computational resources and training data for model training. Additionally, when dealing with small-sample datasets, ViT may encounter overfitting issues, necessitating the introduction of auxiliary tasks for regularization. However, this greatly increases the computational overhead.

In this study, we use distinct methodologies to enhance the attentional mechanisms of the agents and optimize algorithmic efficiency. Our algorithm introduces a mixed attention mechanism that combines channel attention and spatial attention into the encoder f, which can effectively extract important spatial and semantic features from images, enabling convolutional neural networks to learn more efficiently from key pixels while ignoring disturbing information.

4.2.1 Channel attention

As shown in Fig. 2, global maximum pooling and average pooling are performed on the spatial dimension of $H \times W \times C$ feature map $\boldsymbol{X}$ to extract global information in the height H and width W dimensions, while information in dimension C is retained. Maximum pooling highlights the most significant features in each channel, while average pooling captures the overall spatial distribution. Subsequently, the resulting two feature maps are inputted into a shared multilayer perceptron (MLP), resulting

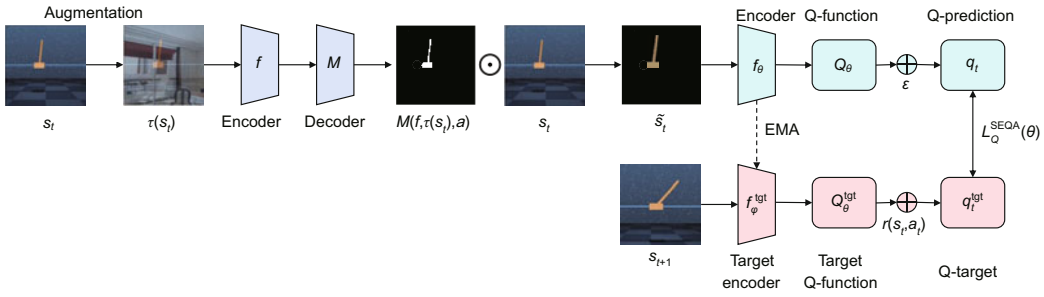


Fig. 1 An overview of the saliency-extracted Q-value by augmentation (SEQA) framework. The augmented data $\boldsymbol{\tau(s_t)}$ are encoded and decoded to generate $\boldsymbol{M(f,\tau(s_t),a)}$, and the Q-functions of the augmented state $\boldsymbol{\tilde{s}_t}$ and the unaugmented state $\boldsymbol{s_{t+1}}$ are regularized consistently to obtain $\boldsymbol{L_Q^{\mathrm{SEQA}}(\theta)}$. Here, $\boldsymbol{\odot}$ represents the Hadamard product and $\boldsymbol{\oplus}$ represents addition to $\boldsymbol{Q}$. EMA: exponential moving average

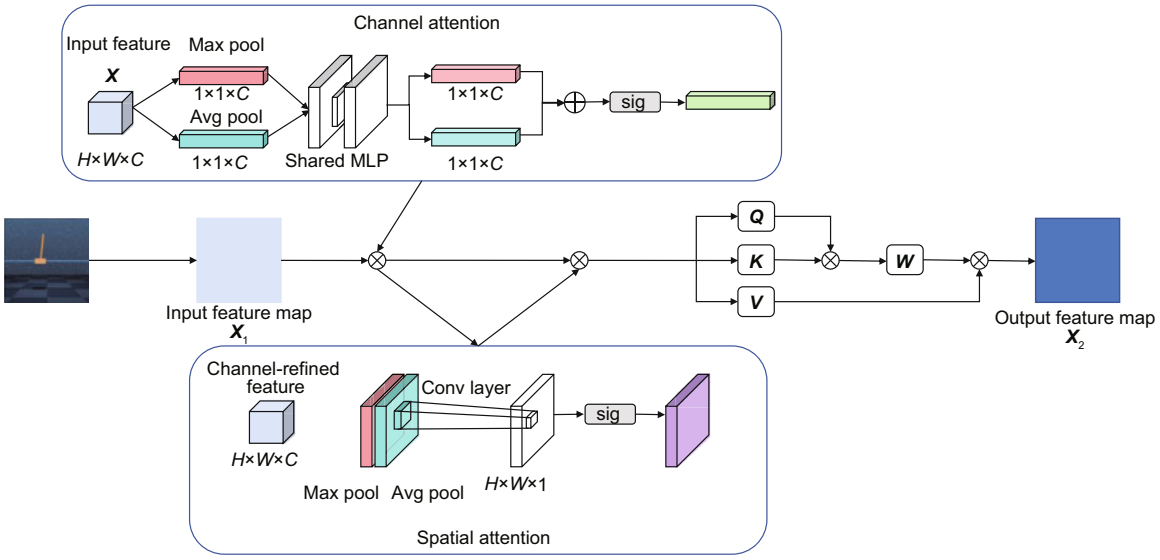


Fig. 2 Attention model. The detailed design of the attention mechanism is given, including how channel attention, spatial attention, and mixed attention are implemented

in two $1\times1\times C$ separate feature maps corresponding to the outputs of maximum pooling and average pooling separately. These MLP-generated feature maps are concatenated and pass through a sigmoid activation function to obtain the channel attention weight matrix $\boldsymbol{M}_{\mathrm{c}}$:

$$\boldsymbol{M}_{\mathrm{c}}=\mathrm{sig}(\mathrm{MLP}(\mathrm{MaxPool}(\boldsymbol{X}))+\mathrm{MLP}(\mathrm{AvgPool}(\boldsymbol{X}))). \tag{3}$$

4.2.2 Spatial attention

The channel-refined features obtained from the channel attention module undergo global maximum pooling and average pooling across the channel dimension. The resulting two feature maps are concatenated along the channel dimension to obtain a fused feature map that incorporates spatial information extracted from both pooling operations. The two obtained $H\times W\times1$ feature graphs are joined on channel dimensions to obtain a feature graph that integrates the spatial information extracted by the two pooling operations. The fused feature map is subject to a 7×7 convolution operation, and the output of this convolution passes through a sigmoid activation function to obtain the spatial attention weight matrix $\boldsymbol{M}_{\mathrm{s}}$:

$$\boldsymbol{M}_{\mathrm{s}}=\mathrm{sig}(f^{7\times7}([\mathrm{MaxPool}(\boldsymbol{X});\mathrm{AvgPool}(\boldsymbol{X})])). \tag{4}$$

4.2.3 Mixed attention

As shown in Fig. 2, integrating the image information, the feature maps processed through channel and spatial attention are linearly transformed to obtain three different representations: query ($\boldsymbol{Q}$), key ($\boldsymbol{K}$), and value ($\boldsymbol{V}$). The attention score is calculated using queries and keys, and the resulting attention score is weighted and summed with the values, weighting different regions of the original feature map to obtain a matrix $\boldsymbol{M}_{\mathrm{f}}$. By computing the correlations between different regions, the model can capture global dependencies within the image. After fusing the obtained $\boldsymbol{M}_{\mathrm{c}}$, $\boldsymbol{M}_{\mathrm{s}}$, and $\boldsymbol{M}_{\mathrm{f}}$, a complete feature map $\boldsymbol{X}_2$ is obtained:

$$\boldsymbol{M}_{\mathrm{f}}=\mathrm{softmax}\Big(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_{\mathrm{k}}}}\Big)\boldsymbol{V}, \tag{5}$$

$$\boldsymbol{X}_2=\boldsymbol{M}_{\mathrm{c}}\otimes\boldsymbol{M}_{\mathrm{s}}\otimes\boldsymbol{M}_{\mathrm{f}}, \tag{6}$$

where $\sqrt{d_{\mathrm{k}}}$ refers to the dimensionality of the key.

4.3 Mask decoder

We design a mask decoder that assigns an importance weight to each pixel in the image to distinguish and emphasize the importance of the pixel. As shown in Fig. 3, the mask decoder consists of six main layers, the specific structure of which is shown in this subsection.

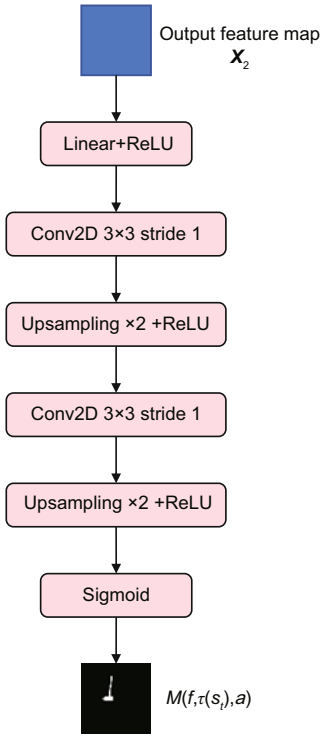


Fig. 3 Mask decoder network

4.3.1 Linear layer

The initial layer is a linear layer that has 32 output channels and is responsible for mapping the features of the encoder to a $32 \times 21 \times 21$ feature map.

4.3.2 Convolutional layers and upsampling blocks

Starting from the second layer, there are two convolutional layers and two upsampling blocks. Each convolutional layer contains 64 3×3 kernels with padding of 1. Each upsampling block has an upsampling factor of 2. These convolutional and upsampling blocks progressively increase the spatial resolution of the feature maps while learning higher-level feature information.

4.3.3 Output layer

The final layer uses the sigmoid activation function and outputs a single-channel feature map, scaling each pixel value to be within the range of [0, 1]. The purpose of this layer is to transform the generated feature map into a mask image of the same size as the original image. Each pixel value in the mask represents the importance level of the corresponding pixel in the original image.

Overall, the mask decoder gradually decodes and upsamples the encoded feature maps to generate a refined mask image, which helps the agent focus its attention on key information in the image while ignoring unimportant or distracting information.

During the training, there is no need to define a separate loss function to train the mask. The parameters of the mask decoder are updated by changes in the critic loss function. When the mask network masks important pixels, the critic loss increases, reminding the network that these pixels are significant pixels. When the mask network masks distracting pixels, the Q-values obtained by the agent either increase or remain unchanged, encouraging the mask network to continue masking these pixels. With each decision made by the agent, the mask network updates its parameters, thereby more effectively masking irrelevant information while preserving important information. This algorithm significantly enhances the performance and generalization capabilities of the agent, enabling it to exhibit greater robustness when dealing with visual information.

4.4 Noise augmentation

To encourage the agents to explore more possibilities and increase the diversity of training data, in this paper we inject random uniform observation noise into the state space and random normal reward noise into the reward space. The noise level is controlled by the noise rate parameter $p \in [0, 1]$, which determines the probability of injecting noise.

4.4.1 Random uniform observation noise

With a probability p, uniform noise ϵ is added to each state of the agent, resulting in $\tilde{o_t} \leftarrow o_t + \epsilon$, where $\epsilon \in U(\alpha, \beta)$, and α and β are important hyperparameters. When $p = 1$ and $\alpha = -\beta$, random uniform observation noise corresponds to the uniform noise augmentation proposed by Sinha et al. (2022). By adding uniform noise in the state space, the agent can explore more uncertainty during training, thereby increasing the diversity of training data and improving its generalization ability. The values of the hyperparameters in this paper are provided in Section 5.

4.4.2 Random normal reward noise

Adding zero-mean Gaussian noise to the reward with the probability p changes the reward signal to

obtain $\tilde{r}_t \leftarrow r_t + \varepsilon$, where $\varepsilon \in \mathcal{N}(0, \sigma^2)$ and σ is an important hyperparameter. Adding random normal distribution noise to the reward space can make the agent not only rely on the fixed reward signal to learn but also adapt to the strategy under different reward signals more flexibly, preventing overfitting and effectively improving the generalization ability.

By controlling the probability of noise injection, the relationship between exploration and utilization can be effectively balanced. When the agent overfits or falls into the local optimal solution, increasing the noise rate can promote the agent to explore more possibilities. When the noise rate is low, the intelligence can make better use of the information it already knows. By adjusting the noise rate, the exploration and utilization strategies of the intelligent agent can be balanced according to the needs of the task, which helps optimize its learning process.

4.5 Consistency regularization objective

As shown in Fig. 1, our algorithm is built upon the Q-network architecture, which divides the neural network and value function into two parts: the encoder network $f_\theta : S \to Z$ and the Q-function $Q_\theta : Z \times A \to \mathbb{R}$. The encoder f_θ maps the original observation space S to a low-dimensional latent representation space Z, effectively extracting crucial information from the states. The Q-function built on the encoder takes actions on the potential representation to predict the corresponding Q-value.

The original paper on deep Q-network (DQN) (Mnih et al., 2015) pointed out that high-variance target values can adversely affect Q-learning algorithms. To reduce the variance of Q-function predictions, we replace a single function with multiple states having the same Q-value. For any state distribution $\mu(\cdot)$ and policy π_θ, instead of relying solely on the estimates of expectation from individual samples (s, a) drawn from $\mu(\cdot)$ and $\pi_\theta(\cdot|s)$, we use multiple enhanced states generated through transformation to obtain an estimate with a lower variance and a greater stability than the original expectation:

$$E_{\substack{s\sim\mu(\cdot)\\ a\sim\pi_\theta(\cdot|s)}}[Q(s,a)] \approx \frac{1}{K}\sum_{k=1}^{K} Q(f_\theta(\tilde{s}), a_k), \quad (7)$$

where $a_k \sim \pi_\theta(\cdot|f_\theta(\tilde{s}))$ and K is the number of Q augmentations. In the process of learning objectives, we use two sets of independent data: the unaugmented data s and the augmented data $\tilde{s}$; here, the latter strictly uses the augmented data for representation learning, while the unaugmented data are used to generate the target function q_t^{tgt} to decouple the flow of training data:

$$q_t^{\text{tgt}} = r(s_t, a_t) + \gamma \max_{a'} Q_\varphi^{\text{tgt}}(f_\varphi^{\text{tgt}}(s_{t+1}), a'). \quad (8)$$

The consistency regularization objective $L_Q^{\text{SEQA}}(\theta)$ is obtained by calculating the Q-estimates obtained from representation learning with the target function q_t^{tgt} derived from the unaugmented data:

$$\begin{aligned} &L_Q^{\text{SEQA}}(\theta)\\ &= \frac{1}{KN}\sum_{k=1}^{K}\sum_{t=1}^{N}\Big[||Q_\theta(f_\theta(s \odot M(f, \tau(s_t), a_t)), a_k)\\ &\quad + \varepsilon - q_t^{\text{tgt}}||^2\Big], \end{aligned} \quad (9)$$

where $\tau(s_t)$ represents the data augmentation function, N represents mini-batch size, and $\odot$ represents the Hadamard product of the mask and the original state operation.

When the strategy π_θ is learned in the benchmark algorithm, $L_\pi(\theta)$ is optimized solely on s_t and does not alter the objective. To prevent the nonstationary gradients of $L_\pi(\theta)$ from interfering with the Q-estimates, we update f_θ using gradient descent. The consistency regularization loss complements the previous data augmentation, encouraging the network to focus its decisions on important pixels.

Algorithm 1 gives the pseudocode of SEQA, where the innovative adjustment is in lines 10–19.

5 Experiments

In this section, we provide a detailed explanation of the experimental setup and the implementation process. To comprehensively evaluate the training efficiency, performance, and generalization capability of the SEQA algorithm, we conduct a series of experiments on DMControl-GB (Hansen and Wang, 2021). SEQA is compared with state-of-the-art methods for continuous action generalization. We aim to demonstrate the effectiveness of SEQA in extracting important features to mask interference and improve the generalization ability of RL algorithms.

Algorithm 1 SEQA based on SAC

Input: total number of environment steps T, mini-batch size N, learning rate η, target network update rate ζ, number of Q augmentations K, and data augmentation function τ
Randomly initialize all the network parameters, $\theta, \theta_\pi, \varphi, M$ // $\varphi \leftarrow \theta$
for each timestep $t = 1$ to T **do**
  // act
  $a_t \sim \pi_\theta(\cdot|f_\theta(s_t))$ // Sample the action from the // policy
  $s'_t \sim p(\cdot|s_t, a_t)$ // Sample the transition from the // environment
  $B \leftarrow B \cup \{(s_t, a_t, r(s_t, a_t), s'_t)\}$ // Add transition // to the replay buffer
  // update
  $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^{N} \sim B$ // Sample a mini-batch
  for transition $i = 1$ to N **do**
    $\theta_\pi \leftarrow \theta_\pi - \eta\nabla_{\theta_\pi} L_\pi(s_i)$ // Optimize π_θ with // stochastic gradient descent (SGD)
    $q_i^{\text{tgt}} = r(s_i, a_i) + \gamma \max_{a'_i} Q_\varphi^{\text{tgt}}(f_\varphi^{\text{tgt}}(s'_i), a'_i)$ // Compute Q-target
    $s_i = s_i + \epsilon$ // $\epsilon \in U(\alpha, \beta)$
    $s_i^{\text{aug}} = \tau(s_i)$ // Apply augmentation
    $\tilde{s}_i = M(f, s_i^{\text{aug}}, a) \odot s_i$
    $L_Q^{\text{SEQA}}(\theta) = \frac{1}{NK} \sum_k \sum_i \left[\| Q_\theta(f_\theta(\tilde{s}_i), a_k) + \varepsilon - q_i^{\text{tgt}} \|^2 \right]$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$
    $\theta \leftarrow \theta - \eta\nabla_\theta L_Q^{\text{SEQA}}(\theta)$ // Optimize f_θ, Q_θ, and // M with SGD
    $\varphi \leftarrow (1-\zeta)\varphi + \zeta\theta$ // Update φ using exponential // moving average (EMA) of θ
  end for
end for

5.1 Setup

The methods compared in this study all use SAC as the base algorithm, and all agents are configured with the same architecture and hyperparameters (if applicable). We apply random overlay augmentation (Bertoin et al., 2022) to all methods except for data-regularized Q (DrQ), which uses random shifting, as DrQ shows better performance using random shifting (Yarats et al., 2021).

5.2 Network architecture

We adopt our network architecture from Grooten et al. (2024). The shared encoder f_θ is an 11-layer convolutional neural network (ConvNet) that processes a stack of three RGB frames rendered at $9 \times 84 \times 84$ and outputs spatial features of size $32 \times 21 \times 21$. Each convolutional layer consists of 32 channels, using 3×3 kernels, with a stride of 1 and no padding. The stride of the first layer is set to 2 to expedite the reduction in channel size. Rectified linear unit (ReLU) activation functions are applied between all convolutional layers.

5.3 Hyperparameters

Four additional hyperparameters are introduced into SAC by SEQA: mask decoder network learning rate η, mask decoder update frequency, random uniform observation noise ϵ, and random normal reward noise ε. In Table 1, we provide a detailed introduction to the hyperparameters related to the experiments.

5.4 Environment

DMControl-GB is based on and extends the DeepMind Control Suite, offering various vision-based continuous control tasks. It is widely used for studying the performance of image-based RL algorithms in continuous action spaces.

5.5 Training and evaluation

In DMControl-GB, we train all agents for 5×10^5 steps in fixed background training environments and evaluate their performance on 12 environments. Additionally, we assess algorithm performance and generalization ability in six environments by replacing the training background with other backgrounds. Specifically, we set up five random seeds to train the model and evaluate our algorithm on video_easy and video_hard benchmarks within DMControl-GB.

6 Results

6.1 Stability on DMControl

We compare the effects of SAC, DrQ, stabilized Q-value estimation under augmentation (SVEA), soft data augmentation (SODA), saliency-guided Q-networks (SGQN), and SEQA algorithms in 12 environments of DMControl-GB without visual interference in Fig. 4. The results show that the SEQA agent achieves the best asymptotic performance and sample efficiency in all the 10 environments. The SEQA algorithm achieves not only better results, but also the lowest variance of the trained agent, which indicates that the improved strategy of the SEQA algorithm is beneficial to the training efficiency and stability.

6.2 Generalization on DMControl

We evaluate the generalization capability of SEQA on DMControl-GB for video_easy and video_hard benchmarks. As depicted in Fig. 5, the video_easy benchmark replaces only the fixed background with a natural image, while the video_hard benchmark replaces the picture as a whole with an everyday picture (including the background and the ground).

At the top of Table 2, we report the total average rewards after 5×10^5 steps of training

Table 1 Hyperparameters used in the experiments on DMControl-GB

Hyperparameter	Value
Frame rendering resolution	84×84×3
Number of tacked frames	3
Random shift	Up to ±4 pixels
Optimizer (θ of SAC)	Adam (β_1=0.9; β_2=0.999)
Optimizer (α of SAC)	Adam (β_1=0.5; β_2=0.999)
Learning rate η (actor, critic, and mask decoder)	10^{-3}
Learning rate (for α)	10^{-4}
Number of action repetitions	4 (walker_walk, walker_stand, ball_in_cup), 8 (cartpole_swingup, cartpole_balance), 2 (finger_spin)
Discount factor γ	0.99
Number of initial collection steps	1000
Number of environment steps	5×10^5
Replay buffer size	5×10^5
Batch size	128
Random uniform observation noise ϵ	α=–0.001; β=0.001; p=1.00
Random normal reward noise ε	σ=0.001; p=0.05
Actor update frequency	2
Critic update frequency	1
Mask decoder update frequency	1
Target φ update frequency	2
Target φ momentum coefficient	0.05 (encoder), 0.01 (critic)

Table 2 Comparison with state-of-the-art methods

Benchmark	Environment	Episode return						p-value
		SAC	DrQ	SODA	SVEA	SGQN	SEQA	
video_easy	walker_walk	276 ± 135	799 ± 23	771 ± 66	865 ± 43	**894 ± 26**	874 ± 58	–20 (2%)
	walker_stand	462 ± 126	936 ± 21	965 ± 7	969 ± 5	945 ± 17	**970 ± 3**	+1 (0%)
	ball_in_cup	563 ± 91	694 ± 141	750 ± 98	857 ± 62	861 ± 55	**919 ± 26**	+58 (7%)
	cartpole_balance	916 ± 25	932 ± 33	961 ± 10	963 ± 2	965 ± 4	**971 ± 4**	+6 (1%)
	cartpole_swingup	398 ± 24	459 ± 81	742 ± 73	753 ± 45	761 ± 28	**780 ± 23**	+19 (2%)
	finger_spin	206 ± 20	479 ± 67	494 ± 100	609 ± 27	**646 ± 26**	629 ± 59	–17 (3%)
	Average	470	717	781	836	845	**857**	+12 (1%)
Benchmark	Environment	Episode return						p-value
		SAC	DrQ	SODA	SVEA	SGQN	SEQA	
video_hard	walker_walk	144 ± 37	117 ± 31	307 ± 47	491 ± 61	**731 ± 39**	691 ± 44	–40 (6%)
	walker_stand	231 ± 36	331 ± 52	706 ± 83	797 ± 35	849 ± 20	**874 ± 39**	+25 (3%)
	ball_in_cup	156 ± 37	144 ± 57	361 ± 143	468 ± 74	790 ± 44	**863 ± 53**	+73 (9%)
	cartpole_balance	302 ± 24	350 ± 28	456 ± 46	599 ± 34	703 ± 11	**728 ± 21**	+25 (4%)
	cartpole_swingup	158 ± 17	145 ± 26	403 ± 61	453 ± 38	538 ± 43	**566 ± 35**	+28 (5%)
	finger_spin	23 ± 10	38 ± 13	309 ± 49	307 ± 24	**549 ± 34**	522 ± 12	–27 (5%)
	Average	169	188	424	519	693	**707**	+14 (2%)

These algorithms were trained on fixed backgrounds and evaluated on video_easy and video_hard benchmarks from DMControl-GB. We took average values and standard errors in the experiments in five random seeds. The optimal results are in bold. Here, p-value=optimal result−suboptimal result, and the content in parentheses=(optimal result−suboptimal result)/(suboptimal result)×100%

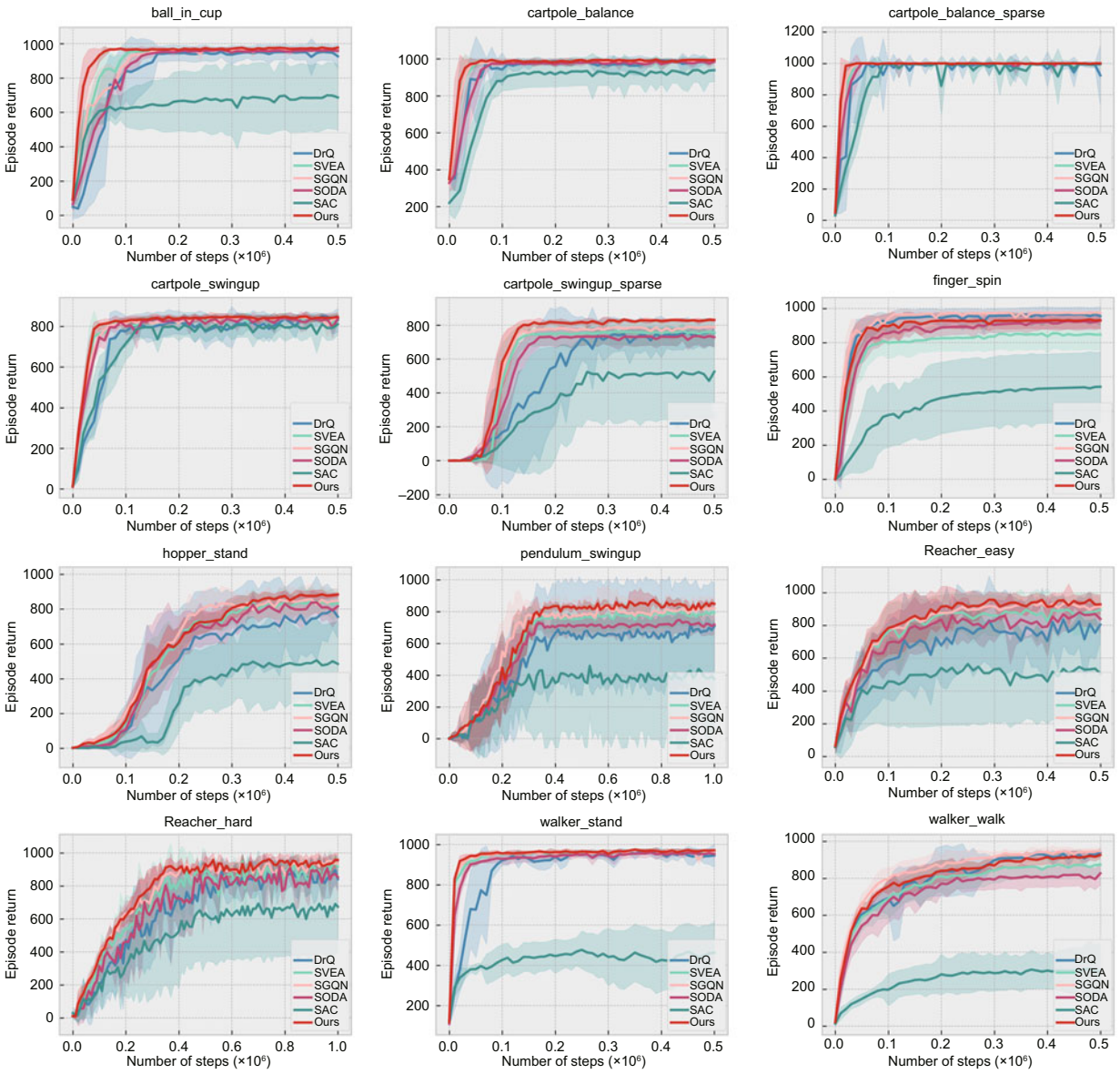


Fig. 4 Performance on training. SEQA achieves the best performance in almost all environments (References to color refer to the online version of this figure)

on the video_easy benchmark. In terms of performance, SEQA outperforms state-of-the-art algorithms in four environments and slightly falls behind the SGQN algorithm in the walker_walk and finger_spin environments. SGQN converts saliency maps into a self-supervised learning objective, deeply coupling them with Q-network's decision-making process. Although this approach consumes more computational resources and is less efficient, it excels in environments such as finger_spin and walker_walk wherein highly precise control strategies are required.

At the bottom of Table 2, we report the total average rewards after 5×10^5 steps of training on the video_hard benchmark. Fig. 6 visually illustrates the reward results for each agent on the video_hard benchmark. Due to the removal of ground and shadows, the video_hard benchmark results in a messier and more challenging distribution shift, and the rewards for SODA and SVEA show a sharp downward trend. SEQA is less sensitive to changes in the environment because the mask alerts the agent about which pixel is more important, encouraging the agent to make the right decision. In summary,

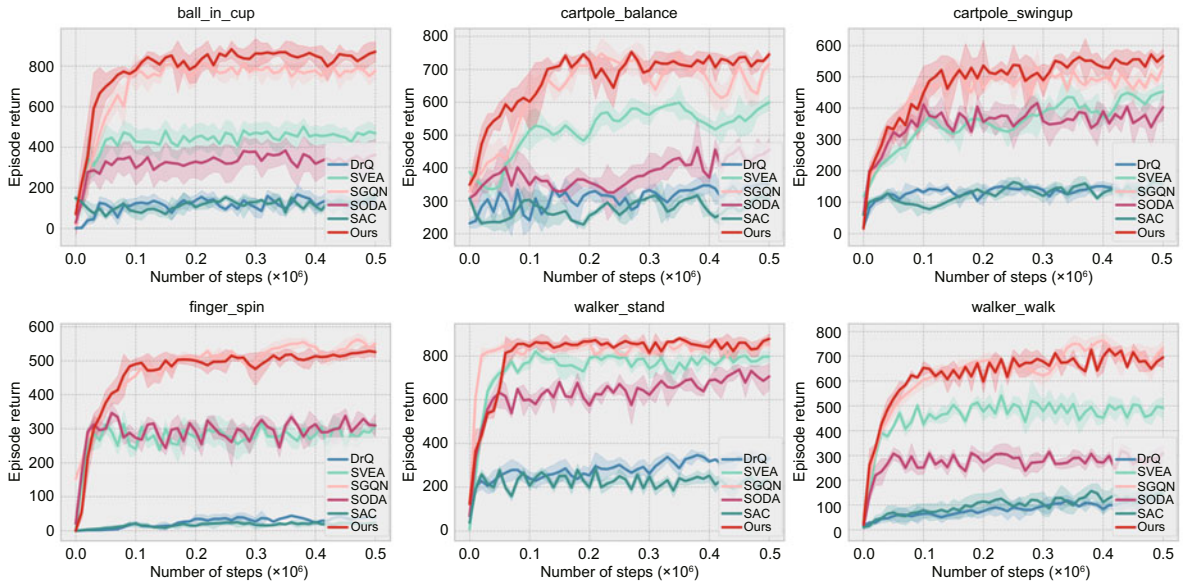
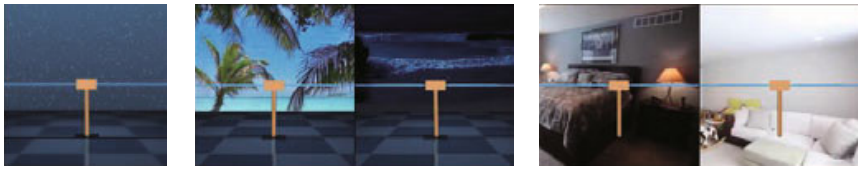


Fig. 5 Performance on the video_hard benchmark. SEQA performs significantly better than the other algorithms in four environments and slightly worse than the optimal algorithm SGQN in the other two environments. References to color refer to the online version of this figure



(a) (b) (c)

Fig. 6 Examples of training and testing environments: (a) training; (b) video_easy; (c) video_hard

SEQA significantly improves the generalization capability of the algorithm and lays foundation for the practical application.

In RL, agent training is fundamentally an exploration–exploitation process, wherein both the environment and the interactions between the agents and the environment are inherently stochastic and uncertain. As shown in Figs. 4 and 5, as a baseline algorithm, SAC typically requires extended training to explore the environment and occasionally discovers high-value actions that lead to significant rewards. However, it may also become trapped in local optima by repeatedly exploiting what it perceives as high-reward actions without sufficient exploration, resulting in higher variance during training. On the video_hard benchmark, SAC's generalization capability is compromised due to the complex interference factors, making it challenging to overcome these disturbances. This results in the agents consistently receiving fewer rewards in this environment and thus exhibiting lower variance compared to the training phase. By introducing noise into the state and reward, our algorithm enhances the exploration–exploitation mechanism. This improvement helps the agents avoid local optima more effectively and demonstrates stronger generalization capabilities. Consequently, our algorithm performs better in the training phase and on the video_easy and video_hard benchmarks.

6.3 Ablation study

SEQA is improved in three main aspects: noise augmentation, mask coverage, and consistency regularization in critic loss. To evaluate the effect of each improvement, we test the performance and generalization ability of each improvement by running five random seeds in six environments, with the results shown in Table 3. The experimental results demonstrate that the proposed improvements effectively

Table 3 Ablation experiment results

Environment	Benchmark	Episode return				
		SAC	SAC+noise	SAC+mask	SAC+consistency	SEQA
walker_walk	training	302 ± 97	562 ± 62	802 ± 54	631 ± 77	910 ± 26
	video_easy	276 ± 135	465 ± 101	726 ± 96	563 ± 132	874 ± 58
	video_hard	144 ± 37	406 ± 73	548 ± 61	462 ± 57	691 ± 44
walker_stand	training	447 ± 162	771 ± 63	914 ± 33	857 ± 110	972 ± 7
	video_easy	462 ± 126	683 ± 55	897 ± 42	847 ± 94	970 ± 3
	video_hard	231 ± 36	544 ± 83	738 ± 48	685 ± 72	874 ± 39
ball_in_cup	training	691 ± 198	745 ± 137	953 ± 21	865 ± 74	978 ± 4
	video_easy	563 ± 91	677 ± 89	793 ± 55	741 ± 76	919 ± 26
	video_hard	156 ± 37	505 ± 122	632 ± 101	563 ± 83	863 ± 53
cartpole_swingup	training	821 ± 58	827 ± 43	833 ± 47	836 ± 34	851 ± 24
	video_easy	398 ± 24	597 ± 73	661 ± 58	703 ± 47	780 ± 23
	video_hard	158 ± 17	238 ± 51	301 ± 61	361 ± 44	566 ± 35
cartpole_balance	training	940 ± 51	953 ± 36	971 ± 17	977 ± 13	996 ± 3
	video_easy	916 ± 25	901 ± 21	943 ± 11	956 ± 7	971 ± 4
	video_hard	302 ± 24	517 ± 37	574 ± 33	621 ± 46	728 ± 21
finger_spin	training	530 ± 201	580 ± 124	811 ± 88	784 ± 73	917 ± 38
	video_easy	206 ± 20	316 ± 112	467 ± 103	531 ± 62	629 ± 59
	video_hard	23 ± 10	163 ± 41	385 ± 56	402 ± 33	522 ± 12
Average	training	622	740	881	825	937
	video_easy	470	607	748	724	857
	video_hard	169	396	530	543	707

Each algorithm was trained, and we evaluated the average undiscounted reward and standard error for each algorithm across six environments on training, video_easy, and video_hard

enhance the stability and generalization capability of the SAC algorithm, with consistency regularization showing the most significant improvement.

7 Conclusions

In vision-based RL, ignoring distracting features poses a significant challenge. We aim to enable agents to avoid erroneous decisions by focusing attention on important pixels, thus improving efficiency and reducing computational overhead. We propose SEQA, an algorithm that relies primarily on three key components: noise augmentation, mask coverage, and consistency regularization. These components collectively encourage that the agent concentrates on important features during its decision-making process while disregarding distract factors. We experimentally evaluate SEQA and state-of-the-art algorithms on DMControl-GB. Across various environments, the SEQA agent consistently demonstrates superior training efficiency and generalization capabilities compared to other algorithms. SEQA is compatible with any standard off-policy RL algorithm and does not require additional parameters or forward passes, making it a promising candidate for future development in RL.

Our algorithm is not as effective as SGQN in walker_walk and finger_spin environments in DMControl-GB, which may be related to self-supervised learning and needs to be optimized in the future. Future research will be focused on the following aspects:

1. We will further explore how to optimize the selection of auxiliary tasks and how to combine these strategies with other RL methods to achieve wider applications.

2. Our algorithm is based on the SAC algorithm, and in future work, we may base the SEQA agent on the policy gradient method proximal policy optimization (PPO) and add a better self-supervised learning module.

Contributors

Yuxi HAN designed the research and drafted the paper. Yang YANG and Dequan LI helped organize the paper. Yuxi HAN and Dequan LI revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Almuzairee A, Hansen N, Christensen HI, 2024. A recipe for unbounded data augmentation in visual reinforcement learning. https://arxiv.org/abs/2405.17416

Antotsiou D, Ciliberto C, Kim TK, 2021. Adversarial imitation learning with trajectorial augmentation and correction. IEEE Int Conf on Robotics and Automation, p.4724-4730. https://doi.org/10.1109/ICRA48506.2021.9561915

Arulkumaran K, Deisenroth MP, Brundage M, et al., 2017. Deep reinforcement learning: a brief survey. *IEEE Signal Process Mag*, 34(6):26-38. https://doi.org/10.1109/MSP.2017.2743240

Bertoin D, Zouitine A, Zouitine M, et al., 2022. Look where you look! Saliency-guided Q-networks for generalization in visual reinforcement learning. Proc 36th Int Conf on Neural Information Processing Systems, Article 2225.

Chen T, Kornblith S, Norouzi M, et al., 2020. A simple framework for contrastive learning of visual representations. Proc 37th Int Conf on Machine Learning, p.1597-1607.

Cobbe K, Klimov O, Hesse C, et al., 2019. Quantifying generalization in reinforcement learning. Proc 36th Int Conf on Machine Learning, p.1282-1289.

Farebrother J, Machado MC, Bowling M, 2018. Generalization and regularization in DQN. https://arxiv.org/abs/1810.00123

Fu X, Yang G, Agrawal P, et al., 2021. Learning task informed abstractions. Proc 38th Int Conf on Machine Learning, p.3480-3491.

Gamrian S, Goldberg Y, 2019. Transfer learning for related reinforcement learning tasks via image-to-image translation. Proc 36th Int Conf on Machine Learning, p.2063-2072.

Gelada C, Kumar S, Buckman J, et al., 2019. DeepMDP: learning continuous latent space models for representation learning. Proc 36th Int Conf on Machine Learning, p.2170-2179.

Grooten B, Tomilin T, Vasan G, et al., 2024. MaDi: learning to mask distractions for generalization in visual deep reinforcement learning. Proc 23rd Int Conf on Autonomous Agents and Multiagent Systems, p.733-742.

Hansen N, Wang XL, 2021. Generalization in reinforcement learning by soft data augmentation. IEEE Int Conf on Robotics and Automation, p.13611-13617. https://doi.org/10.1109/ICRA48506.2021.9561103

Hansen N, Jangir R, Sun Y, et al., 2021a. Self-supervised policy adaptation during deployment. Proc 9th Int Conf on Learning Representations.

Hansen N, Su H, Wang XL, 2021b. Stabilizing deep Q-learning with ConvNets and vision Transformers under data augmentation. Proc 35th Int Conf on Neural Information Processing Systems, Article 281.

Hansen N, Yuan ZC, Ze YJ, et al., 2023. On pre-training for visuo-motor control: revisiting a learning-from-scratch baseline. Proc 40th Int Conf on Machine Learning, Article 506.

Henderson P, Islam R, Bachman P, et al., 2017. Deep reinforcement learning that matters. Proc 32nd AAAI Conf on Artificial Intelligence, Article 392. https://doi.org/10.1609/aaai.v32i1.11694

Kaelbling LP, Littman ML, Cassandra AR, 1998. Planning and acting in partially observable stochastic domains. *Artif Intell*, 101(1-2):99-134. https://doi.org/10.1016/S0004-3702(98)00023-X

Kalashnikov D, Irpan A, Pastor P, et al., 2018. Scalable deep reinforcement learning for vision-based robotic manipulation. Proc 2nd Conf on Robot Learning, p.651-673.

Khraishi R, Okhrati R, 2023. Simple noisy environment augmentation for reinforcement learning. https://arxiv.org/abs/2305.02882

Kirk R, Zhang A, Grefenstette E, et al., 2023. A survey of zero-shot generalisation in deep reinforcement learning. *J Artif Intell Res*, 76:201-264. https://doi.org/10.1613/jair.1.14174

Kurniawati H, 2022. Partially observable Markov decision processes and robotics. *Ann Rev Contr Rob Auton Syst*, 5:253-277. https://doi.org/10.1146/annurev-control-042920-092451

Laskin M, Srinivas A, Abbeel P, 2020a. CURL: contrastive unsupervised representations for reinforcement learning. Proc 37th Int Conf on Machine Learning, Article 523.

Laskin M, Lee K, Stooke A, et al., 2020b. Reinforcement learning with augmented data. Proc 34th Int Conf on Neural Information Processing Systems, Article 1669.

Lee K, Lee K, Shin J, et al., 2020. Network randomization: a simple technique for generalization in deep reinforcement learning. Proc 8th Int Conf on Learning Representations.

Levine S, Finn C, Darrell T, et al., 2016. End-to-end training of deep visuomotor policies. *J Mach Learn Res*, 17(1):1334-1373.

Lin X, Baweja HS, Kantor GA, et al., 2019. Adaptive auxiliary task weighting for reinforcement learning. Proc 33rd Conf on Neural Information Processing Systems, p.4772-4783.

Luketina J, Nardelli N, Farquhar G, et al., 2019. A survey of reinforcement learning informed by natural language. Proc 28th Int Joint Conf on Artificial Intelligence, p.6309-6317.

Mnih V, Kavukcuoglu K, Silver D, et al., 2013. Playing Atari with deep reinforcement learning. https://arxiv.org/abs/1312.5602

Mnih V, Kavukcuoglu K, Silver D, et al., 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529-533. https://doi.org/10.1038/nature14236

Nair A, Pong VH, Dalal M, et al., 2018. Visual reinforcement learning with imagined goals. Proc 32nd Int Conf on Neural Information Processing Systems, p.9209-9220.

OpenAI, Akkaya I, Andrychowicz M, et al., 2019. Solving Rubik's cube with a robot hand. https://arxiv.org/abs/1910.07113

Pinto L, Andrychowicz M, Welinder P, et al., 2018. Asymmetric actor critic for image-based robot learning. https://arxiv.org/abs/1710.06542

Sinha S, Mandlekar A, Garg A, 2022. S4RL: surprisingly simple self-supervision for offline reinforcement learning in robotics. Proc 5th Conf on Robot Learning, p.907-917.

Song XY, Jiang YD, Tu S, et al., 2020. Observational overfitting in reinforcement learning. Proc 8th Int Conf on Learning Representations.

Sutton RS, Barto AG, 2018. Reinforcement learning: an introduction. *IEEE Trans Neur Netw*, 9:1054.

Tassa Y, Doron Y, Muldal A, et al., 2018. DeepMind Control Suite. https://arxiv.org/abs/1801.00690

Tobin J, Fong R, Ray A, et al., 2017. Domain randomization for transferring deep neural networks from simulation to the real world. IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.23-30. https://doi.org/10.1109/IROS.2017.8202133

Wang XD, Lian L, Yu SX, 2021. Unsupervised visual attention and invariance for reinforcement learning. IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.6673-6683. https://doi.org/10.1109/CVPR46437.2021.00661

Xing JW, Nagata T, Chen KX, et al., 2021. Domain adaptation in reinforcement learning via latent unified state representation. Proc 35th AAAI Conf on Artificial Intelligence, p.10452-10459. https://doi.org/10.1609/aaai.v35i12.17251

Yang SZ, Ze YJ, Xu HZ, 2023. MoVie: visual model-based policy adaptation for view generalization. Proc 37th Int Conf on Neural Information Processing Systems, Article 940.

Yang W, Wang XL, Farhadi A, et al., 2019. Visual semantic navigation using scene priors. Proc 7th Int Conf on Learning Representations.

Yarats D, Zhang A, Kostrikov I, et al., 2019. Improving sample efficiency in model-free reinforcement learning from images. Proc 35th AAAI Conf on Artificial Intelligence, p.10674-10681. https://doi.org/10.1609/aaai.v35i12.17276

Yarats D, Kostrikov I, Fergus R, 2021. Image augmentation is all you need: regularizing deep reinforcement learning from pixels. Proc 9th Int Conf on Learning Representations.

Yu T, Zhang ZZ, Lan CL, et al., 2022. Mask-based latent reconstruction for reinforcement learning. Proc 36th Conf on Neural Information Processing Systems, p.25117-25131.

Ze YJ, Hansen N, Chen YB, et al., 2023. Visual reinforcement learning with self-supervised 3D representations. *IEEE Rob Autom Lett*, 8(5):2890-2897. https://doi.org/10.1109/LRA.2023.3259681

Zhang A, Ballas N, Pineau J, 2018. A dissection of overfitting and generalization in continuous reinforcement learning. https://arxiv.org/abs/1806.07937

Zhang A, McAllister RT, Calandra R, et al., 2021. Learning invariant representations for reinforcement learning without reconstruction. Proc 9th Int Conf on Learning Representations.

Zhang H, Chen HG, Xiao CW, et al., 2020. Robust deep reinforcement learning against adversarial perturbations on state observations. Proc 34th Int Conf on Neural Information Processing Systems, Article 1765.

Zhao J, Zhao YP, Wang WX, et al., 2022. Coach-assisted multi-agent reinforcement learning framework for unexpected crashed agents. *Front Inform Technol Electron Eng*, 23(7):1032-1042. https://doi.org/10.1631/FITEE.2100594

Zhou ZH, 2024. Continuous control reinforcement learning: distributed distributional DrQ algorithms. https://arxiv.org/abs/2404.10645

Zhu YK, Mottaghi R, Kolve E, et al., 2016. Target-driven visual navigation in indoor scenes using deep reinforcement learning. IEEE Int Conf on Robotics and Automation, p.3357-3364. https://doi.org/10.1109/ICRA.2017.7989381