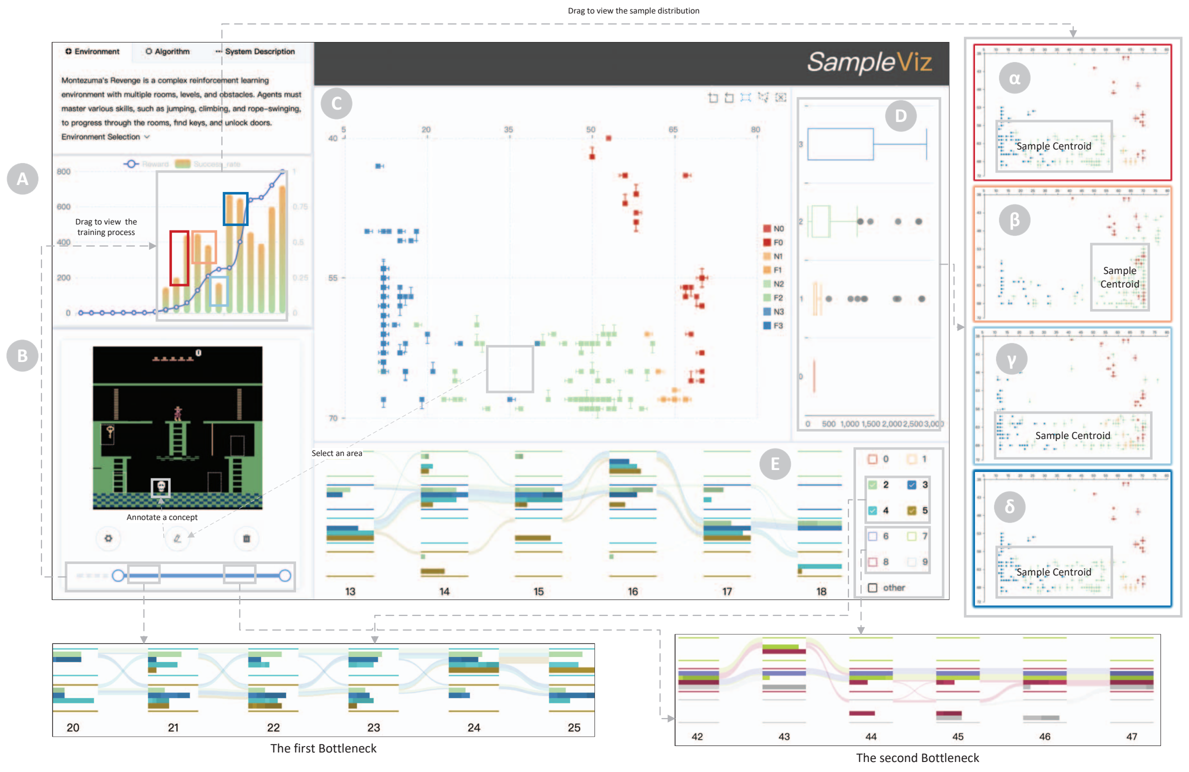
SampleViz: Concept based Sampling for Policy Refinement in Deep Reinforcement Learning

Zhaohui Liang[1,2] * Guan Li[1,2] † Ruiqi Gu[1] ‡ Yang Wang[1,2] § Guihua Shan[1,2] ¶

1) Computer Network Information Center, Chinese Academy of Sciences
2) University of Chinese Academy of Sciences



Montezuma's Revenge is a complex reinforcement learning environment with multiple rooms, levels, and obstacles. Agents must master various skills, such as jumping, climbing, and rope-swinging, to progress through the rooms, find keys, and unlock doors.

Figure 1: *SampleViz*: (A, D) the Statistics View displays the local result changes of the model throughout the selected training process; (B) the Grid View superimposes a grid onto the reinforcement learning environment above the designated area, streamlining the analysis of particular challenges faced by the agent; (C) the Distribution View investigates the distribution data of the trajectory; (E) the Flow View assesses the progression and disparities among sequences.

Abstract

Deep reinforcement learning (DRL) aims to train software agents that can understand environments and learn effective strategies, and has achieved significant breakthroughs in performance and capabilities, particularly in areas such as Go, Atari games, and autonomous vehicles. Unlike traditional deep learning, the goals of reinforcement learning can be more abstract and require careful modification of reward functions. The training process involves unstructured sequential data, which can be difficult for human experts to analyze and gain insights from. To address this challenge, we propose *SampleViz*, a visual analytics system that enables flexible interaction between human experts and DRL sequence data, allowing for the extraction of crucial concepts from massive amounts of data and their provision to the agent. *SampleViz* transforms the tedious task of modifying reward functions and policy debugging into an engaging concept exploration process, allowing for the efficient integration of human expertise with automatic sampling algorithms for effective model improvement. Through case studies and expert feedback, we demonstrate that *SampleViz* can effectively assist experts in concept extraction and model improvement, and enables the incorporation of interpretability and human-in-the-loop concepts into DRL policy settings.

*e-mail: zhliang@cnic.cn
†e-mail: liguan@sccas.cn
‡e-mail: rqgu@cnic.cn
§e-mail: wangyang@sccas.cn (corresponding author)
¶e-mail: sgh@sccas.cn

Index Terms: Human-centered computing—Visualization—Visual Analytics;

1 Introduction

Reinforcement learning (RL) is an area of machine learning that seeks to train a software agent capable of interacting with an environment, learning from these interactions, and accomplishing user-defined goals. Recent advances in deep neural networks (DNNs) have significantly improved the capabilities of deep reinforcement learning (DRL) agents to understand complex environments and develop effective strategies [31]. DRL has achieved multiple breakthroughs, particularly in playing Go [27] and Atari games [18].

In contrast to deep learning, RL involves more abstract and goal-oriented tasks that often require complex reward engineering. The key to training an RL agent is providing relevant samples so that the agent can learn from them and improve its performance. However, existing DRL approaches may face a major challenge: the problem of sparse rewards [8]. In some environments, the desired goal or optimal behavior is difficult to obtain by random exploration, which leads to the agent receiving few positive reward samples, making it challenging to learn the value functions and policies. While human experts may be able to identify these crucial samples, they may not have the tools to directly interact [19] with DRL agents. Furthermore, training a DRL agent for long-horizon problems may require a substantial amount of demonstration data, which can be mentally and physically taxing for human experts. One of the emerging areas of DRL is how experts can interact more effectively with DRL agents and refine their policies with greater ease. One of the primary methods used to analyze the training process of deep reinforcement learning involves analyzing the vast sequence data that the agent learns from [16]. However, this data is frequently rife with failed decisions and meaningless repetitions, while genuinely valuable subsequences are rare and scarce. Thus, enhancing the agent's learning efficiency is synonymous with improving the efficiency of discovering these valuable sequences, which is referred to as improving sample efficiency [26].

There are three major challenges in improving the training efficiency of DRL models. **Firstly**, one strategy for improving sample efficiency in RL over long time horizons is to exploit the hierarchical structure [22] of the problem. Different environments and neural network architectures may correspond to different optimal hierarchies. However, in the past, experts had to rely on intuition and iterative experiments to determine the optimal hierarchies, and few tools were available to DRL researchers for efficiently exploring long game episodes. Utilizing visualization methods to generate summaries of the training process will effectively assist in abstracting specific difficulties and bottlenecks, thereby providing a basis for experts to improve the model. **Secondly**, to avoid overfitting, DRL agents cannot train on the same sample repeatedly during training [7, 14], even if the sample is valuable. This means that experts need to provide a large number of similar but not identical samples, which can be a tedious and inefficient process. Sample points in trajectory data contain multi-dimensional information. Presenting both the trajectory and the information of the trajectory points simultaneously poses a challenge for experts analyzing training data. The lack of effective visualization methods hinders experts' understanding of potential imbalances in samples and model overfitting. **Thirdly**, DRL researchers need to examine the sampling efficiency [28] and training results of the DRL agents to gain a better understanding of the current situation and potential solutions. Improving model interpretability can also facilitate model refinement. It is a challenge to use visualization methods to help experts understand data at both the macro level of the agent's overall learning patterns and the micro level of specific difficulties.

In order to address these difficulties and challenges, we propose *SampleViz* designed to assist experts in their interaction with DRL agents. We introduce human-defined concepts into our system and set them as sub-goals for agents to learn. Different agents learn different concepts, ultimately combining to complete the entire task. This approach helps agents improve sample efficiency. Our system enables a novel visualization method for the analysis and comparison of the trajectory data, facilitating the effective combination of expert knowledge and automated search algorithms in order to improve sampling efficiency. **In summary, our contributions include solutions to three problems in interactive deep reinforcement learning**.

- Incorporating human concepts as training targets for agents: Our approach enhances training efficiency, model interpretability, and enables the model to train in directions preferred by humans.
- Designing a category-based vector sample point trend visualization method: Our visualization method can concurrently present the trajectory distribution and the features of trajectory points, and it enables the visualization of the agent's learning state through the patterns of sample point changes.
- Implementing a visualization analysis system that integrates overall and overview perspectives: The use of a visual analytics (VA) system further enhances this control by enabling users to filter and blend samples, thereby assisting them in identifying optimal subgoals. This leads to improved sampling efficiency and faster convergence.

Our approach was validated through quantitative comparison experiments and expert interviews, which demonstrated that the use of our visual analysis system resulted in significant improvements in training efficiency and model robustness. The experts found that employing concepts as a means to improve the model aligns well with their intuition after using our system. Utilizing our system to refine the reward function not only enhanced training efficiency but also led to some creative behaviors by the agents, which greatly surprised them.

2 Related Work

In this section, we describe works related to our approach, focusing on sample analysis in machine learning and sampling policies in DRL.

2.1 Sample Analysis for Machine Learning

We summarize several sample analysis works for machine learning by comparing them based on four factors as shown in Tab. 1.

Concept Annotation. Concept annotation is commonly used in the context of explaining how deep learning models function, serving to label objects or semantic information in images, text, or videos [9]. For instance, some works in RL, including DQNViz [31], have utilized concept annotation to analyze environments like Breakout [20] in reinforcement learning. Nevertheless, such environments are relatively simplistic, and only a few studies have delved deeply into the analysis of concepts, such as obstacles and reward items, in reinforcement learning environments.

Samples Comparison. Almost all works that explain machine learning samples support comparison, however, the range and granularity of these sample comparisons vary greatly. Works based on deep learning mainly compare the processing results of samples [34], such as comparing the recognition results of traffic light images to improve model performance [6]. However, works based on RL are quite different, mainly comparing sequences [5], such as comparing the trajectory sequences of left and right movements in the ping-pong environment [32]. Such differences are caused by the different tasks of the two field models, and few works can compare the results of each step in the reinforcement learning sequence, making it difficult to explore and analyze the learning progress of the model in detail.

Sequence and Trajectory. The analysis of sequences and trajectories has been widely utilized in interpreting the workings of

Table 1: Sample visualization in machine learning

Area	Method	Concept Annotation	Comparison	Sequence	High Dimensional Data
DL	RNNVis [17]	✗	✓	✓	✗
	Bluff [4]	✗	✓	✗	✗
	InstanceFLow [24]	✓	✓	✓	✗
	ConceptExtract [35]	✓	✓	✗	✗
	VATLD [6]	✓	✓	✓	✗
	VASS [9]	✓	✓	✗	✗
	ConceptExplainer [10]	✗	✓	✗	✗
	OoDAnalyzer [3]	✓	✓	✗	✗
RL	DQNViz [31]	✗	✓	✓	✗
	DRLIVE [32]	✗	✓	✓	✗
	SampleViz	✔	✔	✔	✔

reinforcement learning models. While previous studies have employed videos and t-SNE to compare sequences, they often lack detailed analysis of variations across sequences and explanations for underlying behavioral causes [29]. In InstanceFLow [24], an improved Sankey diagram was utilized to analyze changes in class labels of the same sample during the training process. This analysis provides valuable insights into the learning process of both the model and the sample sequences. To gain insights into an agent's perception and decision-making processes, Pardo *et al.* [21] employed a gridded heatmap visualization to illustrate the Q-values of each frame. This approach allowed for a comprehensive visualization of the agent's Q-values and enabled researchers to identify the underlying reasons for the agent's actions within the environment. Few studies are capable of providing explanations and comparisons for trajectory data across a wide range of categories over a long period.

High Dimensional Data. Reinforcement learning involves high-dimensional information such as position, reward, state, and action probability. Warnell *et al.* propose an approach that combines interactive learning from demonstrations and reinforcement learning, enabling human experts to guide and shape an autonomous agent's behavior [33]. This approach is useful in scenarios where defining an objective function is challenging, and the state space is too large for exhaustive search methods. Previous visualization studies have addressed this issue through multiple views analysis, resulting in incomplete information and difficulties in providing comprehensive comparisons for users.

2.2 Sampling Policy in DRL

Sampling is a crucial component in the RL process. It allows agents to explore their environment, estimate the value of different actions, and learn an optimal policy that maximizes their cumulative reward.

Exploration-exploitation Dilemma. In RL, the exploration-exploitation dilemma refers to the trade-off between taking actions that are known to lead to high rewards (exploitation) and taking actions that are uncertain but have the potential to lead to even higher rewards (exploration) [2]. The dilemma arises because, in order to maximize the long-term reward, the agent must select actions that it believes will lead to the highest expected reward, but it must also explore new actions to learn more about the environment and improve its estimates of the expected reward. If the agent always chooses the action with the highest expected reward, it may miss out on better actions that are currently uncertain [12]. On the other hand, if the agent always explores, it may waste time and resources trying out actions that are known to be suboptimal.

Human-in-the-loop Sampling. Human-in-the-loop sampling is a critical technique in RL, which aims to train intelligent agents to learn optimal decision-making policies through interaction with their environments. In this approach, the agent is initially given a set of pre-defined rules to follow, and as it interacts with the environment, it receives feedback from humans who adjust the rules based on their observations. This feedback can be in the form of reward signals, corrective actions, or feedback on the agent's performance. Human-in-the-loop sampling is particularly useful in scenarios where the agent operates in complex or unfamiliar environments, where pre-defined rules may not be sufficient to capture the full range of possible actions and outcomes [11]. Aytar *et al.* explores the use of human demonstrations in training reinforcement learning agents to play challenging video games and proposes a novel approach that leverages human gameplay footage from YouTube as a source of demonstrations for training agents to explore and solve complex game environments [1]. Human-in-the-loop sampling represents a promising approach for developing intelligent agents that can adapt to changing environments and make effective decisions in complex situations.

3 Background

This section provides background information on our proposed method, outlines the algorithms employed, and describes the interaction mechanisms between users and the model.

3.1 Hierarchical Deep Q-network (H-DQN)

Hierarchical Reinforcement Learning (HRL) is a popular approach in which the agent learns to perform a task by breaking it down into smaller subtasks [23] or goals. This hierarchical decomposition allows for more efficient learning by reducing the complexity of the overall task and enabling the agent to learn at multiple levels of abstraction. Deep Q-Network (DQN) is a deep learning algorithm that combines Q-learning with DNNs to learn optimal policies in complex and high-dimensional environments.

Learning goal-directed behavior in sparse feedback environments is the primary challenge faced by reinforcement learning algorithms. The main difficulty arises from insufficient exploration, which hinders the agent's ability to learn robust value functions. Agents with intrinsic motivation can explore new behaviors for their own benefit rather than directly solving the problem. Such intrinsic behavior can ultimately help the agent solve tasks in the environment. A hierarchical deep Q-network (H-DQN) framework [13], as shown in Fig. 2, is proposed to address this challenge. This framework integrates hierarchical value functions at different time scales, with intrinsic deep reinforcement learning motivation. The high-level value functions learn policies rather than intrinsic goals [30], while the low-level functions learn atomic actions that satisfy the given goals. The relationship between hierarchical model structures and the agent's environment, as well as the delineation of these hierarchies, presents a challenge for expert understanding, exacerbated by the absence of specialized visualization methods.

3.2 Concept Annotation

In deep learning, concept annotation is usually achieved through the use of neural networks, which can learn to automatically associate input data with corresponding concepts. This process often involves training the network on a large annotated dataset, and then using the trained network to annotate new data. Concept annotation is widely used in various fields, including information retrieval, text classification, image recognition, and speech recognition. In the context of DRL, concept annotation refers to the process of annotating human domain knowledge or common-sense knowledge into textual or data inputs to enable machine comprehension and processing. Concept annotation can significantly enhance the performance of DRL algorithms by providing prior knowledge, reducing the sample complexity, and improving the interpretability of the learned policies.

Specifically, concept annotation can be used to improve the exploration-exploitation trade-off in DRL, by providing prior knowledge about the state space and action space and helping the agent to identify the most promising actions to take. Concept annotation can also be used to regularize the learning process, by incorporating domain-specific constraints or priors into the objective function of

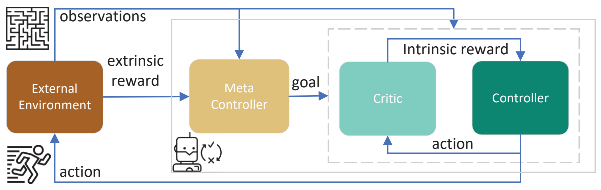


Figure 2: H-DQN framework [13]. The external environment is the agent's operating environment, which provides the agent with observations, and receives actions from the agent. The meta controller is responsible for guiding the learning process of the controllers, and selecting the most appropriate controller to handle the current task. The controllers are responsible for selecting actions based on the current observation and the learned Q-values, which are estimated by the critic network. The critic network is responsible for estimating the Q-values of the current state-action pairs, and providing feedback to the controllers to improve their action selection.

the DRL algorithm. Moreover, concept annotation can enable the learning of more interpretable policies, by providing explicit rules or guidelines for decision-making, which can be useful in safety-critical applications or when explaining the agent's behavior to human users. In our approach, we represent concepts using collections of sample points. Visualization methods aid experts in abstracting common features and patterns from a large number of sample points, thereby establishing a correspondence between the sample collections and user-defined concepts.

4 Requirements

We conducted two meetings with three domain experts (E1-E3) in RL to distill their requirements of a desired visual analytics system for hierarchical reinforcement learning (HRL). All the experts have 3+ years of experience in RL. These experts typically analyze certain trends in the training process, such as the commonly used reward function, to assess the model's training status. However, gaining intuitive insights into the details of the training process remains a challenge, which often results in difficulties in identifying the turning points and anomalies. Even for tasks that are simple for humans, reinforcement learning agents may require millions of training iterations, and the entire training process can be filled with repetitive and erroneous operations and judgments. Despite containing a wealth of valuable information, the vast number of training samples is rarely efficiently utilized. Identifying high-value samples and their inherent patterns can significantly assist reinforcement learning experts in optimizing the model. When training a DRL agent, the experts can understand and analyze the agent's learning patterns and progress through the event sequences of the training process. The learning of the agent occurs through essentially random attempts, while the experts understand how to perform the correct actions and can guide the agent to attempt these high-value actions faster by defining concepts. However, in complex reinforcement learning environments, agents often require millions of training episodes, and sequences of tens of millions of steps can leave experts at a loss for where to start. These long sequences are filled with failed attempts and meaningless repetitions, making it crucial to identify valuable samples for the agent from the sequences.

Over these discussions, we elicited and iteratively refined three themes of design requirements on (R1) experiment overview and samples exploration, (R2) concepts investigation based on subgoals, bottleneck, and specifications, and (R3) interactive policy refinement.

R1: Providing an overview of agent's evolvement. Obtaining a comprehensive overview of the samples throughout the entire training process is an essential prerequisite for experts. By monitoring and analyzing the distribution and characteristics of the training data, experts can gain valuable insights into the behavior of the learning algorithms and identify potential issues or biases.

- R1.1: How does the training progress evolve and is there a clear hierarchy of training phases.
- R1.2: Why is the agent stuck and how does the agent conquer the obstacles.
- R1.3: How do samples play a role in the training process.

R2: Empowering analysis and comparisons of massive trajectories. This requirement primarily aims to support users in gaining an intuitive understanding of long sequences and massive samples. Once users have a deep understanding of the composition and structure of the trajectories, they can then focus on analyzing the differences between the sequences and samples to gain insight into the agent's learning progress.

- R2.1: Revealing the main components of a trajectory and the distribution of each component.
- R2.2: Being able to present the hierarchical structure of a trajectory.
- R2.3: Supporting users to compare and analyze the expanded and missing parts among trajectories.

R3: Revealing the concepts in the training process and parameterizing them to subgoals, bottlenecks, and specifications. We introduce the concept to define a set of samples and stages of training processes, experts can extract or define these concepts. The concept can be split into subgoals, bottlenecks, and specifications. Users can convert subgoals and bottlenecks to the hierarchical structure of the RL task. For the agent, the hierarchical structure shows the different levels of difficulty in different areas of the environment, and the experts analyze the agent's learning patterns and difficulties by viewing and understanding this hierarchical structure. Specifications are restrictions on Agent behavior and can define successful actions and failures.

- R3.1: Revealing the hierarchical architecture by agent's subgoals and bottleneck.
- R3.2: Being able to show the trajectories and present the distance between trajectories and expected path.

5 Approach Overview

In this section, we present the main process and experimental method of our approach. Informed by the experts' requirements and detailed discussions, we formulated a three-stage approach, utilizing the challenging 'Montezuma's Revenge' game as our experimental environment.

5.1 System Pipeline

Fig. 3 depicts the main functionalities of our system and how experts use the *SampleViz* to understand and improve the model. Initially, the model is subjected to testing and training, and the data collected during the training process is provided to the visual analytics system. Then the process of utilizing *SampleViz* by experts can be divided into three main stages, which are explained as follows:

S1: Overview the training process and samples (R1,R2). We initially designed an overview view to demonstrate whether the experiment achieved its expected objectives and to identify potential significant areas for improvement. This was primarily achieved by showcasing the overall training process through reward and success rate curves, highlighting specific aspects such as the intelligent agent's learning rate, training time, training process fluctuations, and model robustness. We believe that a comprehensive understanding

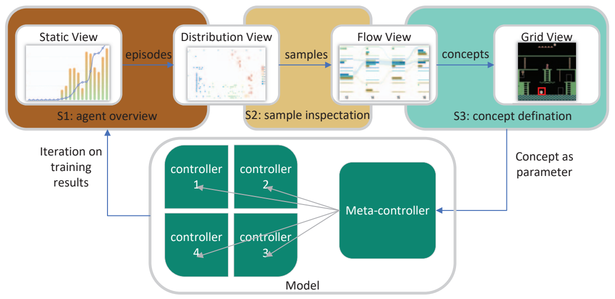


Figure 3: System Pipeline. The three boxes below correspond to S1, S2, and S3, respectively, and ultimately transmit the hierarchical structure to the meta controller.

of the intelligent agent's trajectory can be obtained by examining the overall distribution of samples. It is possible that certain regions may be excessively sparse or dense, or that a significant number of samples may deviate from the expected trajectory.

S2: Investigate the evolution of samples' distribution and categories (R1, R2). To analyze the complex distribution, we designed flexible zooming and selection methods to assess the density of the distribution, with denser areas typically corresponding to bottlenecks and obstacles encountered by the agent during the exploration process, while sparser areas usually represent regions with which the agent is already familiar. The investigation of intersection involves the overlap and misalignment of samples among sub-goals, which may be the result of the agent choosing the wrong direction to explore, leading to movements in the opposite direction. In addition, we implemented a grid-based method for map partitioning to analyze the changes in samples corresponding to each partition, where more significant changes indicate higher exploration intensity of the agent.

S3: Modify the hierarchical structure and optimize the model (R3). We adjusted the model's network structure and training objectives to enable the agent to reach each sub-goal sequentially. Moreover, we introduced Grid View to interactively define concepts, where a concept is designated by selecting a region through drawing a bounding box, which subsequently becomes a sub-goal on the agent's path.

5.2 Training Process and Data Collection

Montezuma's Revenge. Agent's goal is to acquire Montezuma's treasure by making your way through a maze of chambers. The environment is filled with a variety of obstacles and the path from the start to the finish is winding.

The Montezuma's Revenge environment simulates the real physical world and is a typical environment that is easy for humans to understand and operate, but difficult for machines to implement. The agent needs to explore extensively in this environment to understand the specific conditions of the entire environment, as well as learn and understand different physical objects such as ropes and ladders. This environment has long been considered a challenging problem to overcome and requires extensive training time [25]. In addition, due to the complexity of the agent's action space, there have been few breakthroughs in analyzing the agent's learning process in this environment. The main reason we selected this environment is to test the possibility of using human expert knowledge to assist in agent training.

Experiment Settings and Data Collection. We trained a model from [15] to enable the agent to obtain the key on the left side of the map. We first conducted training tests to collect data and improve the model. In each test step, we collected six types of data as follows

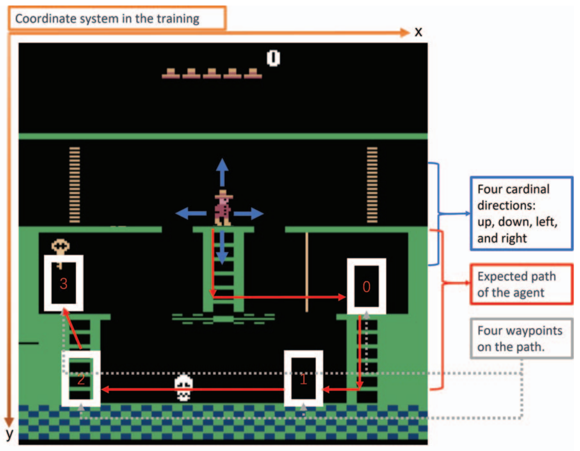

Coordinate system in the training
x
y
Four cardinal directions: up, down, left, and right
Expected path of the agent
Four waypoints on the path.
3
0
2
1

Figure 4: Experiment settings. The agent commences its journey from a specific location on the map, successfully completing a sequence of four sub-goals (0, 1, 2, and 3) before finally acquiring the key located in the middle of the left side of the map. The agent must also learn to utilize various tools, such as ladders and ropes, while avoiding obstacles such as moving skulls. Action selection for the agent is based on the model, and the projected actions are constrained to the four cardinal directions: up, down, left, and right. The coordinate axes for the location information are denoted by the arrows shown in the figure.

(as shown in Fig. 4).

- **action:** a value of 0, 1, 2, 3, 4, 5, 11, 12 representing *no action*, *jump*, *up*, *right*, *left*, *down*, *jump right* and *jump left*.
- **reward:** a value for the reward from an action.
- **terminal:** a boolean value indicating if an episode ends or not (avoid falling from high places or touching hazardous objects).
- **probability:** the predicted probability (a floating-point value) for each action in the current step.
- **location:** two integers, denoted as x and y respectively, where x ranges from 5 to 80 and y ranges from 38 to 72.
- **sub-goal:** a value of 0, 1, 2, 3 representing four different sub-goals.

We set four sub-goals for the agent in the model, distributed along the trajectory from the starting point to the endpoint. The agent needs to learn how to sequentially reach each sub-goal and eventually reach the endpoint. In the experiment, we introduced penalties to prevent the agent from encountering obstacles or remaining stationary. In this environment, contact with skulls or falling results in the termination of the episode.

6 Visual Analytics System

Following the requirements (Sec. 4), we design and develop a visual analytics system called *SampleViz*. The entire system supports multi-granularity analysis of the sequence from a global perspective to local details. It is noteworthy that, unlike previous studies, we have incorporated a variety of methods in the design of our visualization system to facilitate the understanding of the decision-making logic of agents by experts.

6.1 Statistics View

To analyze a DRL model, it is necessary to first examine its learning process (S1), which primarily involves analyzing changes in reward, success rate, and data distribution during training.

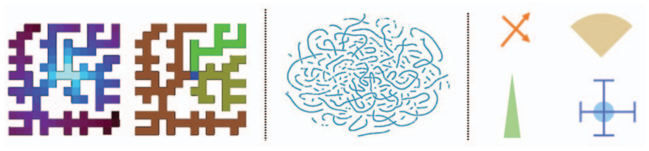


Figure 5: Different designs for the samples in trajectory. The figure demonstrates the effects of using heatmap, dimensionality reduction methods, and directed point visualization of trajectories, respectively.

Line Chart: Training Process Overview. The experts tend to monitor the progress of the training process by using reward curves and success rate curves. In fact, the trends of these two curves often show a strong correlation. To facilitate experts' observation of this highly synchronized change, we present both curves in the same view.

Box Chart: Sub-goal Distribution Overview. Boxplots provide a clear and concise summary of the data distribution, including information about the median, quartiles, and potential outliers. The primary function of the box chart is to illustrate the distribution and variation of samples throughout the training process, as well as at specific stages and grids. Utilizing a box chart can provide valuable information, such as model robustness and learning strategies. The clear and intuitive outlier can effectively assist experts in inspecting the anomalies in data.

6.2 Distribution View: Trajectory Distribution Overview

Analyzing the agent's evolution requires visualizing not only its current location but also examining its decision-making process and the subtle differences between different decisions at the same location for each sample (S2). The agent executes the decision with the highest Q-value to obtain the maximum reward. By determining the agent's level of approval for all decisions based on the Q-value, we can visualize the differences in approval for different decisions and analyze the discrepancies between decisions at the same time and the variations in the same decision at different time points.

In this study, we explored different visualization designs for analyzing high-dimensional trajectories, as illustrated in Fig. 5. While the qmap [21] method utilizes a heatmap to visualize agent decision-making based on q-values, such heat maps are challenging for comparison and analysis of overall and detailed trajectory differences due to low spatial utilization. We also attempted to use dimensionality reduction techniques to reduce the complexity of the trajectories to 2D; however, the resulting visualizations were disorganized and challenging to interpret, hindering expert analysis and interaction. To address these issues, we improved the scatterplot by gridding the region according to the distribution of the scatter points and adding visual mappings, such as arrows, to enhance interpretability. We explored various visual mappings, including unidirectional arrows, sectors, and triangles, and ultimately selected the metaphor of neurons, to enable experts to compare the probabilities of the four directions and analyze the model's decision certainty and robustness from multiple perspectives.

The distribution view displays the complete information of trajectories, and mainly supports detail searching among massive samples. The view supports zooming and selection functions to facilitate experts' inspection and analysis of local details. The visualization mapping we adopted helps users intuitively understand the decision probabilities of agents in different directions, but we found in our experiments that there might be cases where the lengths are almost equal, causing confusion for users. We applied a binarization method to the maximum probability direction and other directions to make the final decision more evident. Experts uniformly sampled the trajectories of 20 rounds during the training process as visualization samples.

We introduce a grid-based design in the distribution to enable users to align the distribution with the actual environment map and conduct a comprehensive analysis by integrating and combining it with the flow view. For example, users can choose to view the data distribution in a certain grid to learn about the agent's learning process in that grid. The left section displays the specific information of each sample, with color encoding as right. The legend (color encoding) on the right side allows users to control the display of subgoals (1, 2, 3, 4) and whether to visualize the probability of moving in each direction at the next step. Different color codes can correspond to different sub-goals or different grids. The color codes can also help users align and integrate the distribution with the flow view for a comprehensive analysis. For example, users can choose to view the data distribution in a certain grid to learn about the agent's learning process in that grid. Users have the option to zoom in on specific areas or perform selection to analyze a particular group of samples and extract pattern information.

6.3 Flow View: Sub-goal and Bottleneck Inspection

The sequential information within trajectory data is crucial, as experts require detailed analysis of sequence evolution. In addition, the experts hope to find subtle differences between the expected sequence and the current sequence, in order to analyze the validity of the experimental setup (S3). To analyze such differences, we have made modifications to the Sankey diagram [24]. A Sankey diagram of the agent trajectory changes during the selected training period is a helpful tool for experts to analyze the evolution of the agent, which illustrates the model's performance. Initially, the experts compare distribution changes across different sample grids within the same sequence; subsequently, they analyze temporal variations of the same category across different sequences; finally, they examine the high analytical value of changes within the same category across various episodes.

In the Flow View, the experts initially need to design an anticipated ideal sequence reflecting their expected experimental outcomes, based on model settings. For instance, they might envisage a final sequence of 100 samples uniformly distributed across 10 grids. The system then compares this ideal sequence with the actual experimental sequence data, delineating deviations via flow dynamics. An illustrative case would be if the 9th sample, expected in the first grid, ends up in a different grid, with the Flow View using varying color codes to highlight such discrepancies. The Flow View offers varying granularity levels. At its basic level, it employs a Sankey diagram to visualize category shifts and depicts the projected distribution for each category. The 20 trajectory data each with 50 steps serve as the visualization data for the Flow View. The bar chart displays the distribution of grids between the current step and the grids from the previous step within the sequence, aiding users in understanding the evolutionary changes within the same trajectory.

Flow view patterns, summarized from the experts' feedback and depicted in Fig. 6, reveal diverse agent behaviors. Pattern A signifies stability or a bottleneck situation where the agent is impeded, necessitating situation-specific analysis. Pattern B, characterized by repetition and cycling, suggests the agent is continually seeking the optimal decision. Pattern C represents an ideal evolutionary approach, with the agent progressively exploring and refining its model. In contrast, Pattern D exhibits a more rapid evolutionary pace but is prone to greater risks of abrupt learning disruptions.

The "Class to Display" option allows users to select the category to be analyzed and displayed, where categories 0-9 correspond to different grids, -1 corresponds to termination, and other corresponds to the grids where the agent should not appear. Users can interact by selecting the category of interest. The "Epoch to Display" option in the grid view selects the epoch interval to be analyzed, with a maximum of 50 epochs displayed based on performance limitations.

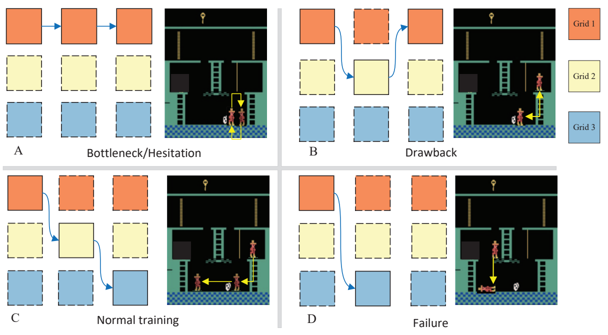


Figure 6: **Concepts** in Flow View. A, B, C, and D correspond to four different iteration modes and learning rates in the Flow View. **A. Bottleneck or Hesitation** indicates that the agent has encountered an insurmountable barrier, resulting in its stagnation. **B. Drawback** implies that the agent is not progressing as planned but instead moving closer to the starting point, indicating a misdirection in exploration. **C. Normal training** signifies that the agent is gradually learning in the anticipated direction. **D. Failure** indicates that the agent has chosen an unexpected route, which typically signifies a failure, although in rare cases, it may lead to unexpected success.

6.4 Grid View: Grid and Sub-goal settings

Grid-based analysis and sub-goal setting are important components of our method. In practical operations, it is necessary to adjust these settings in a timely manner in order to complete the entire analysis process. In the Grid View, experts can grid the map of the environment to define subgoals by these grids.

The primary function is to discretize the reinforcement learning environment into a grid and number the grid based on the trajectory of the agent. The setting button is for setting the size and color of grids and sub-goals. Adjusting each grid's size appropriately is crucial for optimizing the usability of experimental data. The edit button supports adding new sub-goals by clicking the button and drawing a box. The delete button clears the sub-goal setting before. Upon partitioning the training environment, users can utilize the statistics view to conduct an initial analysis on the resulting partitions. These charts allow for a comprehensive analysis of the current grid partitioning, especially in cases of highly imbalanced category distributions or abnormal changes in category distribution during the training process, which may necessitate a reevaluation of the grid partitioning strategy.

We aim to facilitate experts in discovering better grid-based analysis and sub-goal settings through the aforementioned functions. The objective of these settings is to create distinct exploration areas for different controllers, aiming to reduce overlap and minimize the agent's backtracking process, thereby enhancing exploration efficiency in the correct direction.

7 Case Study

We focus on a game called Montezuma's Revenge in this work. Owing to the complexity of the action and state spaces in these environments, as well as the protracted training process, previous works have achieved only limited success in improving the analysis effect in such settings.

During the design phase of our system, we collaborated with three domain experts (E1, E2, and E3) on two case studies. The two case studies were designed to enhance the understanding and refinement of the DRL process, aiming to deepen insights into the evolution of reinforcement learning models through visual interactive methods. Furthermore, our approach enhances model performance by effectively providing expert-extracted concepts to the model using appropriate algorithms. We employed qualitative and quantitative experiments to demonstrate the effectiveness of our system. In addition, our method makes the process from understanding the model to improving the model transparent and intuitive, achieving interpretability throughout the entire process. These two cases demonstrate that our system is a valuable tool for experts studying the primary tasks of deep reinforcement learning.

7.1 Analyzing the fluctuations during the training process

The first case study focuses on the balance between exploration and exploitation in the agent's behavior. We aim to leverage human experts' knowledge to assist the agent in exploring more efficiently and achieving better training performance.

The experts' feedback indicates that they often encounter fluctuations in rewards and success rates during the experiment. However, they cannot identify the specific situation that caused the turning point, making it difficult to analyze the reasons for the turning point. The experts are very concerned about whether there are methods to analyze the trajectory of the agent during these time periods to understand the reasons for the sudden drop in success rate and thus help improve the model.

During the experiment, the experts noticed plateaus in the learning process, characterized by a lack of improvement in the reward or penalty signal over time (as illustrated in Fig. 1 A). Additionally, the success rate of the agent reaching the key exhibited two sharp drops (as illustrated in Fig. 1 A), which resulted in a substantial increase in the model's training time. The experts were confounded by the root cause of the agent's recurrent failures, despite having previously demonstrated proficiency in the task. However, the experts are confronted with the challenge of handling massive amounts of training data, and reviewing every recording of the training process appears to be almost impossible. To reduce irrelevant information, the experts used the Statistics View to select intervals where fluctuations increased during the training process.

The experts conducted an in-depth analysis of the first wave, examining the data corresponding to the preceding peaks (Fig. 1 α, β), the trough (Fig. 1 γ), and the subsequent peak (Fig. 1 δ). Based on visual analysis, the experts have obtained the following insights:

Expanding or limiting search space: The experts observed fluctuations attributed to the agent's extensive search range expansion when navigating the skull obstacle. Initially, this expansion led to counterproductive exploration, with the agent deviating significantly from its path, often returning to the start.

In hierarchical reinforcement learning, intrinsic rewards guide specific actions, while extrinsic rewards direct the meta-controller towards overarching goals. Despite correct goal-setting by the meta-controller, early-stage explorations often veer in reverse, influenced by the stochastic nature of the agent's decisions and formidable obstacles near sub-goals. This behavior indicates a limited agent understanding of sub-goals akin to human interpretation. However, such reverse explorations are beneficial for mastering complex environments. Theoretically, without predefined sub-goals, a single controller could suffice for the agent to reach its final destination. Yet, in hierarchical contexts, sub-goals effectively channel exploration in a focused direction, albeit underexplored in current research. The case study supports the viability of this strategy.

Exploration and Exploitation Dilemma: The experts found that the success rate of agents reaching the endpoint had reached 60 percent at the previous peak, but it appears that the agent was not satisfied with such a strategy. At this stage, the agent exhibited obvious hesitation, repeatedly attempting to move back and forth between grids 4 and 5, which delayed its progress and ultimately led to failure when it encountered an obstacle in grid 6 (as shown in Flow View Fig. 1 *The First Bottleneck*). The distribution view also reveals

Table 2: Case2 result. This table documents the sub-goals chosen by experts during the experiment and the corresponding results. Blank cells indicate a failure of the experiment with no data recorded.

Experiment ID	Sub-goal Concept	Success Rate	Training Steps	Time Consuming	Fluctuations	Episode Steps
0	Upon the right ladder Skull Lower left ladder Key	1.0	211080	4.74	2.5	134
1	Born Point Lower Right Ladder Lower left ladder Key	0.6	227802	3.23	1.5	193
2	Rope Lower Right Ladder Lower left ladder Key	0	-	-	-	-
3	Lower Right Ladder Skull Lower left ladder Key	0.3	225313	4.28	2	186

the same trend, as shown in Fig. 1 α, where the agent's decision-making exhibits significant disorder and randomness near obstacles. From one peak to the next, the agent altered its strategy for skull avoidance. Initially, it employed a fixed jumping strategy (dense green dots), but adaptability was required due to the skulls' dynamic nature. Over time, the agent learned to navigate this obstacle more flexibly (sparse green dots). We mandated a 90 percent success rate for task completion, prompting strategic optimization by the agent. This shift illustrates the agent's learning pattern transitioning from fixed, to random, and back to fixed, indicative of the model's robustness and learning capacity. As training progressed, the agent's approach to the obstacle stabilized, reflecting in a consistent jumping action."

Setting Sub-goals: The visual analytics results demonstrate that the use of sub-goals in agent-based reinforcement learning allows for the exploration of diverse regions of the environment, employing different models for each sub-goal. This method proves to be more efficient in sampling when there are fewer intersections between trajectories of different sub-goals. Furthermore, the use of sub-goals can effectively reduce the agent's repetitive operations. Upon reaching the end point of a sub-goal, the agent switches to the next model for control, which further reduces redundant repetition. Experimental results reveal that this setting effectively reduces redundant repetition in comparison to traditional reinforcement learning methods that lack the use of sub-goals. It should be noted, however, that different sub-goal settings may result in highly variable experimental outcomes. In light of this, the experts have conducted Case 2 to further examine the effects of sub-goal settings on experimental results.

7.2 Improving agent's performance by changing sub-goals

Traditionally, model interpretability has centered on elucidating model functionality, with experts enhancing models based on these understandings. However, this transition from insight to improvement, primarily occurring in the experts' minds, remains opaque and its effectiveness in enhancing models is challenging to assess. The experts wonder whether it is possible to establish a channel so that insights obtained in a visual analysis system can be passed to the model in the form of parameters, enabling direct model optimization. In hierarchical reinforcement learning, setting diverse sub-goals, traditionally based on experts' intuition, is crucial for improving sampling efficiency and speeding up training. However, research on optimal sub-goal formulation is limited. The experts are concerned about whether there is a way to guide them in improving the setting of sub-goals, and whether such improvements can enhance model performance.

Traditional reinforcement learning methods optimize a single objective function to enable an agent to obtain high-quality control policies for a specific task. However, in practical tasks, there may be multi-level control problems. These problems require the task to be decomposed into smaller sub-tasks, so that the agent can effectively learn control strategies. However, due to the challenges of complexity and hierarchy, achieving multi-level control is still a challenging problem. Multi-level control methods decompose the task into multiple sub-goals, each of which is controlled by a controller. To avoid interference between sub-goals, each sub-goal is trained and adjusted separately. The experts interactively explore the training process based on the visualization system, finding anomalies or patterns of interest. The training settings of the agents are also adjusted in combination with the experts' experience, setting different subgoals to accelerate the agent's exploration process and make the converged model more stable.

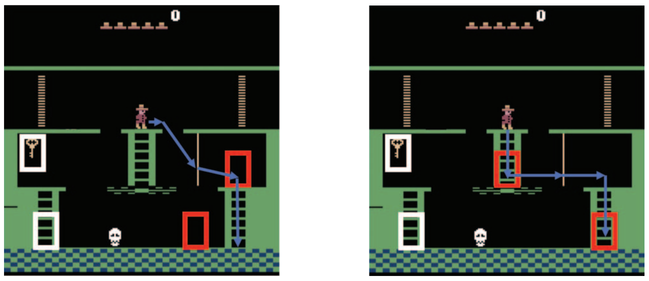

(a) Sub-goal settings 0 (b) Sub-goal settings 1

Figure 7: Different trajectories in Case2. (a)the agent jumps down to the right from the birthplace and uses a rope to move to the right; (b)the agent uses a ladder to move downwards from the birthplace, and then uses a rope to move to the right.

In the first case study, the experts noticed two different paths taken by the agent during the early stages of training, indicating that different sub-goals could lead to varied outcomes and affect the model's performance. They chose regions on the map that correspond to common concepts such as ropes, barriers, ladders, platforms, and keys, as well as typical areas along the path, identifying these as potential sub-goals. During the experiments, selecting sub-goals at random from these options generally resulted in similar paths, but occasionally produced unique trajectories. The experts conducted a detailed analysis of these unique cases, and the findings are presented in Fig. 7. These results were unexpected and provided several key insights, which are detailed below.

1. The sub-goals used in the experiment have a significant impact on the trajectory of the agent.

2. The selection of sub-goals should be carefully considered to ensure that they reflect the overall objectives of the task.

3. The introduction of diverse sub-goals can increase the robustness of the model and improve its performance.

4. The identification and analysis of anomalous experimental settings can provide valuable insights into the behavior and performance of the agent.

We summarized the sub-goal settings obtained by the experts' exploration, and obtained four different settings, each of which was tested 10 times. The success rate represents the probability of achieving a success rate of more than 90 percent for training the agent to obtain the key in 10 trials. Training steps and Time consuming represent the average training steps and time, respectively. Fluctuations represent the number of times the success rate decreases by more than 20 percent during the experiment. Fewer fluctuations indicate a more stable training process. Episode steps represent the steps of the agent's trajectory in the final result of the training. Fewer steps indicate that the agent has learned a better route and action. Training results can be found in Tab. 2. The experts analyzed the experimental data and found that the overall performance of sub-goal setting 0 was optimal, achieving a 100 percent success rate in the experiments. Although there was no advantage in terms of training steps and time compared to other sub-goal settings, the final policy corresponding to this setting had far fewer steps than other settings. The experts analyzed the results and found that the reason for this outcome was that in sub-goal setting 0, the agent chose a unique path, as shown in the left image compared to the paths in other sub-goal settings shown in the right image. Upon consulting with relevant experts, they confirmed that the agent had discovered a highly creative shortcut, which was unimaginable for most human players. In fact, learning this shortcut also costs the agent more training time, but the experts generally believe that the model has paid very little time cost for a significant improvement. The results obtained from the left-side configuration exhibited a noteworthy improvement in both training time and stability when compared to those from the right-side configuration.

Traditionally, model training direction has been guided by the reward function. However, this experiment demonstrates that adjusting sub-goals can effectively steer training, offering a more efficient approach to model optimization. Nonetheless, in this experiment, the experts could guide the agent to learn in the direction they desired by pre-setting subgoals, which considerably reduces the time required for parameter adjustment.

8 Feedback and Discussion

This section discusses the experts' feedback after using our system as well as limitations and future work.

8.1 Expert Review

Following the case study, in-depth interviews were conducted with the experts to gather their insightful feedback. Overall, all experts believed that our tool helped them understand the model and gave them insights that they had not previously discovered. In addition, the experts gave very high ratings for the concise operation and intuitive effect of using our tool to optimize the model.

The experts unanimously agree that the distribution view is the most helpful module in our system for them to understand the model. *"The distribution view has freed me from the tedious process of observing experiment videos, and I will use this tool in my daily model building work."*, commented by E2. Regarding the use of the metaphorical visualization of neurons in our distribution view, E1 commented, *"Not only can I see the agent's position and actions, but I can even know how it's thinking."* Experts generally agree that this view provides them with a new understanding of the learning process of the agent. For example, after completing case study 1, E3 was surprised to discover *"the complex reasons behind the commonly observed fluctuations in rewards"*.

The experts found the grid view to be both intriguing and practical for policy enhancement. *"The process of using Grid View is as easy and intuitive as playing an electronic game or assembling a Lego set"*, commented by E3. Experts also found that the integrated interaction among grid view, distribution view, and flow view was very user-friendly, to the extent that *"it seemed unnecessary to provide any training or instruction for them to adjust the sub-goals using this tool"*. Experts pointed out through discussions that the sub-goal setting in the experiment should be uniformly distributed to avoid interference between exploration in different areas. E1 suggested that *"sub-goals should be set in front of difficult obstacles or bottlenecks"* to allow the controller to focus on overcoming these obstacles.

In addition, the experts also pointed out the additional features they desired and provided some suggestions for improvement. E2 suggested that adding filters to the distribution view would be a useful feature, as in practical use, it may not only be necessary to select based on time and region, but also to filter specific action types or sequences of actions. Both E1 and E3 believe that if our tool can support more complex environments or reinforcement learning models in the real world, it will be a huge breakthrough. E3 added, *"This tool may bring significant changes to route optimization and human-machine interaction for drones and autonomous driving tools."*

8.2 Discussion, Limitations and Future Work

Generalization. While *SampleViz* has demonstrated proficiency in environments that necessitate intricate mapping and prolonged exploration strategies, its application to tasks requiring precise manipulation and control in three-dimensional spaces has been limited. A significant direction for strategic enhancements aimed at broadening the system's utility and effectiveness involves the integration of immersive interaction techniques. Such advancements are not solely about extending support to three-dimensional analyses but also about enriching the system's capacity to interpret and visualize the multifaceted dynamics of samples within these environments. Through immersive interaction, *SampleViz* is poised to offer users a more intuitive and engaging platform for dissecting and understanding complex reinforcement learning phenomena, thereby improving the interpretability of neural representations and enhancing the overall analytical depth. Additionally, incorporating human expert demonstrations into *SampleViz* bridges the gap between human expertise and algorithmic learning, allowing for a detailed comparison of decision-making strategies between humans and AI agents. Such comparative analysis is expected to yield insights that guide the development of more refined and human-centered H-DQN strategies, offering users valuable perspectives for optimizing RL processes.

The case of Montezuma's Revenge illustrates *SampleViz*'s capacity to manage complex learning tasks, showcasing its wide-ranging potential. Our approach, centered around grid-based trajectory analysis, offers broad applicability for understanding agent behavior in diverse settings. The distribution view, in particular, stands out for its ability to encode complex decisions and actions, making it versatile for various decision-making scenarios. This flexibility, combined with sophisticated decision dynamics analysis, establishes *SampleViz* as a comprehensive tool for delving into the intricacies of advanced H-DQN environments, heralding significant advancements in the domain of HRL.

Concept abstraction and labeling. Our significant contribution is the introduction of concept abstraction and annotation into DRL, which bridges the communication gap between human experts and models through concepts. However, the definitions of concepts in our system are currently limited to a certain temporal and spatial range, and there is not enough exploration of the actions and physical properties of the samples themselves. Additionally, for concept abstraction, we plan to incorporate regular expressions in the future to enhance the usability and practical value of user-defined abstract concepts.

More future works. We are concerned with the ethical and moral implications of artificial intelligence, and our system aims to enhance the transparency and controllability of AI models. We are also interested in Reinforcement Learning with Hindsight Feedback (RLHF). With the recent application of RLHF in the field of large language models, concerns regarding ethical risks and controllability have been raised among researchers. We aim to incorporate concept extraction and labeling into RLHF to make the model fairer, unbiased, and transparent, while taking human feedback as an information source to clarify the position of human control and enhance functional outcomes.

9 Conclusion

In this work, we propose *SampleViz*, a visual analytics system that helps experts understand, abstract, and label concepts in deep reinforcement learning models. Based on hierarchical reinforcement learning theory, we set conceptual sub-goals for the model and integrate a complete pipeline so that concepts labeled by the user can be directly fed back to the model. Through case studies and experimental verification, we find that users can control the development and evolution of the model's training to match their personal preferences by manipulating concepts. Additionally, the model can effectively improve performance by understanding and learning these concepts.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62202446).

References

[1] Y. Aytar, T. Pfaff, D. Budden, T. Paine, Z. Wang, and N. De Freitas. Playing hard exploration games by watching youtube. *Advances in neural information processing systems*, 31, 2018. 3

[2] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016. 3

[3] C. Chen, J. Yuan, Y. Lu, Y. Liu, H. Su, S. Yuan, and S. Liu. Oodanalyzer: Interactive analysis of out-of-distribution samples. *IEEE transactions on visualization and computer graphics*, 27(7):3335–3349, 2020. 3

[4] N. Das, H. Park, Z. J. Wang, F. Hohman, R. Firstman, E. Rogers, and D. H. P. Chau. Bluff: Interactively deciphering adversarial attacks on deep neural networks. In *2020 IEEE Visualization Conference (VIS)*, pp. 271–275. IEEE, 2020. 3

[5] H. Fujiyoshi, T. Hirakawa, and T. Yamashita. Deep learning-based image recognition for autonomous driving. *IATSS research*, 43(4):244–252, 2019. 2

[6] L. Gou, L. Zou, N. Li, M. Hofmann, A. K. Shekar, A. Wendt, and L. Ren. Vatld: A visual analytics system to assess, understand and improve traffic light detection. *IEEE transactions on visualization and computer graphics*, 27(2):261–271, 2020. 2, 3

[7] S. Gupta, G. Singal, and D. Garg. Deep reinforcement learning techniques in diversified domains: a survey. *Archives of Computational Methods in Engineering*, 28(7):4715–4754, 2021. 2

[8] J. Hare. Dealing with sparse rewards in reinforcement learning. *arXiv preprint arXiv:1910.09281*, 2019. 2

[9] W. He, L. Zou, A. K. Shekar, L. Gou, and L. Ren. Where can we help? a visual analytics approach to diagnosing and improving semantic segmentation of movable objects. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1040–1050, 2021. 2, 3

[10] J. Huang, A. Mishra, B. C. Kwon, and C. Bryan. Conceptexplainer: Interactive explanation for deep neural networks from a concept perspective. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):831–841, 2022. 3

[11] J. Koch, A. Lucero, L. Hegemann, and A. Oulasvirta. May ai? design ideation with cooperative contextual bandits. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2019. 3

[12] V. Krakovna, L. Orseau, R. Ngo, M. Martic, and S. Legg. Avoiding side effects by considering future tasks. *Advances in Neural Information Processing Systems*, 33:19064–19074, 2020. 3

[13] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016. 3, 4

[14] Y.-H. Lai, T.-C. Wu, C.-F. Lai, L. T. Yang, and X. Zhou. Cognitive optimal-setting control of aiot industrial applications with deep reinforcement learning. *IEEE Transactions on Industrial Informatics*, 17(3):2116–2123, 2020. 2

[15] H. Le, N. Jiang, A. Agarwal, M. Dudík, Y. Yue, and H. Daumé III. Hierarchical imitation and reinforcement learning. In *International conference on machine learning*, pp. 2917–2926. PMLR, 2018. 5

[16] M. Mahmud, M. S. Kaiser, A. Hussain, and S. Vassanelli. Applications of deep learning and reinforcement learning to biological data. *IEEE transactions on neural networks and learning systems*, 29(6):2063–2079, 2018. 2

[17] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu. Understanding hidden memories of recurrent neural networks. In *2017 IEEE conference on visual analytics science and technology (VAST)*, pp. 13–24. IEEE, 2017. 3

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 2

[19] E. Mosqueira-Rey, E. Hernández-Pereira, D. Alonso-Ríos, J. Bobes-Bascarán, and Á. Fernández-Leal. Human-in-the-loop machine learning: a state of the art. *Artificial Intelligence Review*, pp. 1–50, 2022. 2

[20] N. D. Nguyen, S. Nahavandi, and T. Nguyen. A human mixed strategy approach to deep reinforcement learning. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 4023–4028. IEEE, 2018. 2

[21] F. Pardo, V. Levdik, and P. Kormushev. Scaling all-goals updates in reinforcement learning using convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 5355–5362, 2020. 3, 6

[22] S. Pateria, B. Subagdja, A.-h. Tan, and C. Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys (CSUR)*, 54(5):1–35, 2021. 2

[23] A. P. Pope, J. S. Ide, D. Mićović, H. Diaz, D. Rosenbluth, L. Ritholtz, J. C. Twedt, T. T. Walker, K. Alcedo, and D. Javorsek. Hierarchical reinforcement learning for air-to-air combat. In *2021 international conference on unmanned aircraft systems (ICUAS)*, pp. 275–284. IEEE, 2021. 3

[24] M. Pühringer, A. Hinterreiter, and M. Streit. Instanceflow: Visualizing the evolution of classifier confusion at the instance level. In *2020 IEEE visualization conference (VIS)*, pp. 291–295. IEEE, 2020. 3, 6

[25] T. Salimans and R. Chen. Learning montezuma's revenge from a single demonstration. *arXiv preprint arXiv:1812.03381*, 2018. 5

[26] M. Schwarzer, A. Anand, R. Goel, R. D. Hjelm, A. Courville, and P. Bachman. Data-efficient reinforcement learning with momentum predictive representations. *arXiv preprint arXiv:2007.05929*, 2, 2020. 2

[27] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016. 2

[28] Y. Tsurumine, Y. Cui, E. Uchibe, and T. Matsubara. Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation. *Robotics and Autonomous Systems*, 112:72–83, 2019. 2

[29] G. Vecchio, S. Palazzo, D. Giordano, F. Rundo, and C. Spampinato. Mask-rl: Multiagent video object segmentation framework through reinforcement learning. *IEEE transactions on neural networks and learning systems*, 31(12):5103–5115, 2020. 3

[30] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pp. 3540–3549. PMLR, 2017. 3

[31] J. Wang, L. Gou, H.-W. Shen, and H. Yang. Dqnviz: A visual analytics approach to understand deep q-networks. *IEEE transactions on visualization and computer graphics*, 25(1):288–298, 2018. 2, 3

[32] J. Wang, W. Zhang, H. Yang, C.-C. M. Yeh, and L. Wang. Visual analytics for rnn-based deep reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics*, 28(12):4141–4155, 2021. 2, 3

[33] G. Warnell, N. Waytowich, V. Lawhern, and P. Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018. 3

[34] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10):1084–1102, 2018. 2

[35] Z. Zhao, P. Xu, C. Scheidegger, and L. Ren. Human-in-the-loop extraction of interpretable concepts in deep learning models. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):780–790, 2021. 3