2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

X-neuron: Interpreting, Locating and Editing of Neurons in Reinforcement Learning Policy

Yuhong Ge[1], Xun Zhao[2], Jiangmiao Pang[2], Mingguo Zhao[3] and Dahua Lin[4]

Abstract— **Despite the impressive performance of Reinforcement Learning (RL), the black-box neural network backbone hinders users from trusting and deploying trained agents in real-world applications where safety is crucial. In order to make agents more trustworthy and controllable, for a given RL-trained policy, we propose to enhance its interpretability and make it human-controllable without retraining. We accomplish this goal by following a 3-step pipeline: 1) We *interpret* neurons by analyzing the causal effect of neurons on the kinematic attributes; To help agents unlock novel skills and enable human to assist agents in accomplishing tasks, 2) we *locate* the *X-neuron*, the optimal neuron that is capable of evoking the desired behavior; 3) and *edit* its activation values to achieve the precise control. We evaluate our method on various RL tasks ranging from autonomous driving to robot locomotion, and the results display that our approach outperforms previous work regarding almost all evaluation metrics. Through enhancing interpretability and introducing human control, the agents can improve safety and performance, even in unseen environments and novel tasks. For locomotion robots simply trained to walk forward, our method unlocks diverse controllable behaviors ranging from jump to backflip.**

I. INTRODUCTION

Reinforcement Learning (RL), a machine learning paradigm where an agent learns an optimal policy to maximize the reward via interacting with the environment, has been applied to various critical fields ranging from autonomous driving [1], [2], [3] to robotics [4], [5], [6], [7], [8]. Although RL algorithms have achieved impressive performance, the opaque nature of neural networks impedes understanding of their inner working mechanisms and decision-making processes [9], [10]. The lack of transparency undermines users' trust in the trained agents, which hinders successful deployment in real-world applications. Moreover, when interacting with unseen environments, the agent lacks sufficient transferability and generalizability [11], which may lead to catastrophic consequences in safety-critical applications.

To address these issues, existing research mainly focuses on interpreting RL, either by interpreting its intrinsic mechanisms [12], [13] or developing post-hoc interpretability [14], to increase trust. In addition to enhancing interpretability, Policy Dissection [15] proposed that incorporating humans into the inference loop (i.e., human-AI shared control) can further improve the performance and safety of agents, even in novel tasks [16]. They adopt a frequency matching method to identify one neuron in learned policy to be responsible for each kinematic attribute, then rewrite the corresponding neuron activation values to elicit agent behaviors without retraining. Nevertheless, this work suffers from several major flaws. The frequency matching is a correlation analysis, which may not be causal, thereby the identified neurons often fail to elicit desired behaviors. Besides, a single neuron may correspond to multiple attributes, leading to issues with disentanglement. For example, a neuron that closely matches the velocity attribute is also likely to affect heading, which ignores the entanglement between different attributes and makes the elicited behavior deviate from human expectations. Furthermore, the activation value is rewritten on the basis of a trial-and-error process, which is time-consuming and unstable.

In order to enhance the trustworthiness and safety of the agent, we propose a 3-step pipeline (Fig. 1) to facilitate interpretability and leverage our interpretation to achieve fine-grained human-AI shared control. **I) Interpret.** For each neuron in a trained policy, we trace its causal effect on the kinematic attributes of the agent from the macro level to the micro level, by implementing interventions on its activations. For the macro level, we can obtain an Impact Factor that quantifies the overall impact of the neuron on the kinematic attributes. Based on the observation that neurons usually display diverse impacts on different kinematic attributes, we also compute a Preference Vector to describe the detailed impact of neurons on each attribute from the micro level. A Preference Factor can be further calculated to indicate whether a neuron demonstrates a strong preference over certain sets of kinematic attributes. **II) Locate.** Since a behavior (such as jumping, backflipping, etc.) can be described by changing a subset of kinematic attributes, we are able to locate the **X-neuron** that possesses an EX-tra power over the kinematic attributes to evoke the desired behavior to achieve human-AI shared control. We locate the X-neuron following a coarse-to-fine strategy: first, narrow down the neuron search space, and then identify the optimal neuron u_{n^*} by minimizing the distribution difference. **III) Edit.** After identifying the X-neuron, instead of overwriting, we add a small perturbation to its original activation value to achieve precise control. In conclusion, we interpret neurons in trained policy, then locate X-neuron and edit its activation values to evoke desired behaviors to achieve fined-grained

*This work was done during the internship at Shanghai Artificial Intelligence Laboratory.

[1]Shenzhen International Graduate School, Tsinghua University, Shenzhen, Guangdong, China; geyh21@tsinghua.org.cn
[2]Shanghai Artificial Intelligence Laboratory, Shanghai, China; {emmaxunzhao, pangjiangmiao}@gmail.com
[3]Department of Automation, Tsinghua University, Beijing, China; mgzhao@mail.tsinghua.edu.cn
[4]Shanghai Artificial Intelligence Laboratory, Shanghai, China. dhlin@ie.cuhk.edu.hk

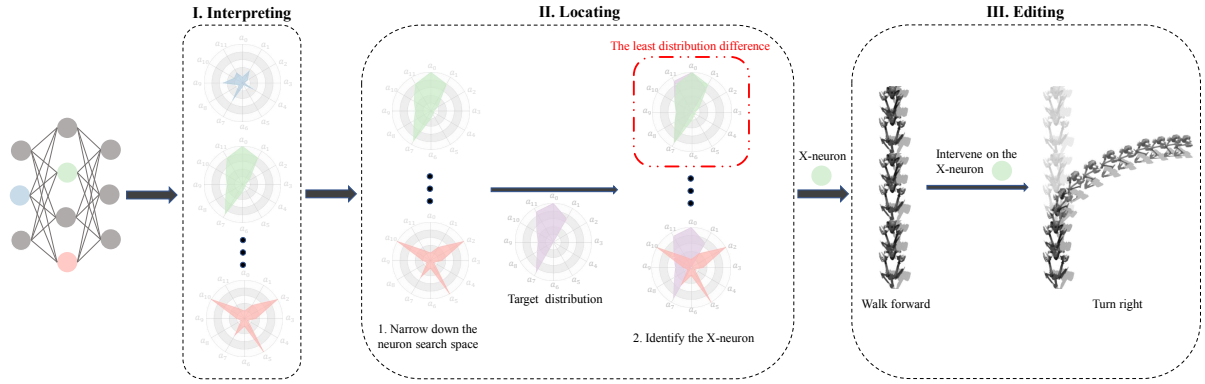

Fig. 1. The 3-step pipeline: We first 1)Interpret the causal effect of each neuron with different kinematic attributes, which is reflected by radar charts. Then we 2)Locate the optimal neuron named X-neuron by using target distribution (the purple radar chart). Finally, we 3)Edit the X-neuron to evoke the agent to behave as we expect, such as turning right. Each dimension of the radar chart corresponds to a certain kinematic attribute, reflecting the causal effect of the neuron with all attributes.

human-AI shared control, thereby making the agent more trustworthy and controllable without retraining the model.

We evaluate our method on various RL tasks, such as autonomous driving and robot locomotion. Compared to the given trained policy, our method can evoke novel behaviors and achieve zero-shot task transfer in unseen environments via editing X-neuron. For the robots simply trained to walk forward, our approach unlocks novel behaviors, ranging from gear shifting and steering to complex skills such as backflips. Compared to the qualitative control of policy dissection, our method can achieve precise and fine-grained control. Furthermore, experimental results demonstrate that our approach outperforms policy dissection and the original policy in almost all evaluation metrics.

II. METHOD

Our 3-step pipeline clarifies what neurons in a well-trained policy have learned by analyzing the hidden states, and ultimately uses neurons with different abilities to achieve fine-grained human-AI shared control.

Firstly, we **interpret** the causal effect that different neurons have varying degrees of influences on different kinematic attributes. Secondly, we **locate** the optimal neuron called X-neuron and capable of affecting attributes as we desire. Finally, we **edit** the activation value of the X-neuron to induce the agent to achieve the desired behavior.

As MLP (Multi-Layer Perceptron) is the prevalent policy network in RL tasks, we consider the policy is an MLP with N neurons. We denote each neuron and the corresponding activation value as u_n and h_n respectively, where $n \in [0, N)$. The kinematic attributes of the agent, denoted as a, is an M-dimensional vector, where the m-th attribute is denoted as a_m, and $m \in [0, M-1]$. For instance, in [17], the kinematic attributes can be described as a high-dimensional vector, each dimension of which represents a certain kinematic attribute, such as the velocity in a specific axis direction or the expected joint position of a specific joint in the next moment.

A. Interpretation

To enhance users' trust in the neural controller, we aim to improve interpretability by investigating how the neurons affect kinematic attributes from the macro level to the micro level. We impose interventions on neuron activations to trace the information flow from input to output, thereby analyzing the causal effect of neurons on kinematic attributes. For the macro level, we calculate an **Impact Factor** that quantifies the overall impact of the neuron on the the kinematic attributes. On the other hand, neurons usually display diverse influences on different kinematic attributes. Hence, from the micro level, we can compute a **Preference Vector** to illustrate the detailed impact of neurons on each kinematic attribute. A **Preference Factor** can further be derived to describe whether a neuron demonstrates a strong preference for certain kinematic attributes. For instance, in MetaDrive, an autonomous driving environment, certain neurons influence the steering significantly but only have a limited effect on the acceleration and other attributes, while some other neurons perform oppositely. This characteristic of neurons is summarized as preference.

We first roll out the given well-trained policy for S episodes, recording the sequence of the activation value of the n-th neuron and the sequence of the m-th kinematic attribute as H_n and A_m, where $H_n = \{h_n^t | 0 \le t \le T-1\}$ and $A_m = \{a_m^t | 0 \le t \le T-1\}$. As shown in Fig. 2, we select T consecutive timesteps in each episode during which the agent operates stably. We then use the same policy to interact with the environment for another S episodes. At a random moment t_0 in each episode, we apply a certain intervention i to the activation value of the neuron numbered n, and cancel this intervention at the moment t_0+1, recording the sequence $\hat{A}_m = \{\hat{a}_m^t | 0 \le t \le t_0 + w \le T-1\}$ of each kinematic attribute from moment 0 to w moments after the intervention is applied. When the n-th neuron encounters an intervention i, changes in attributes can be represented as:

$$\left[\sum_{t=t_0}^{t_0+w} \Delta\hat{a}_{n,0}^t, \ldots, \sum_{t=t_0}^{t_0+w} \Delta\hat{a}_{n,m}^t, \ldots, \sum_{t=t_0}^{t_0+w} \Delta\hat{a}_{n,M-1}^t\right]$$

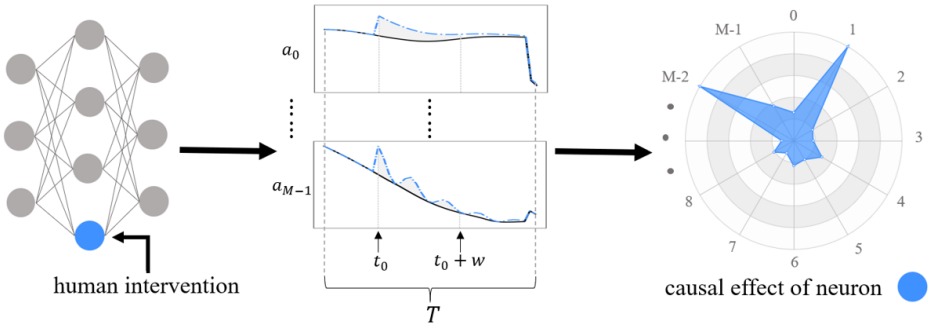

Fig. 2. Procedure of analyzing causal effects of neurons. We apply a human intervention to a single neuron and compare the sequence of kinematic attributes with the sequence under no intervention. We choose the sum of deviations over a time period w to measure the causal effect between the neuron and attributes.

where $\Delta\hat{a}^t_{n,m} = \hat{a}^t_{n,m} - a^t_m$. $\hat{a}^t_{n,m}$ is the value of the m-th kinematic attribute after applying intervention i to h_n at time t, and $t_0 \in [0, T-1)$. Considering the environment involves a lot of randomness, we repeat each episode R times and interventions are applied at different moments randomly. In addition, the intervention should be an order of magnitude larger than activation values to ensure that changes in a_m are primarily caused by interventions rather than randomness in forward dynamics. Based on neuron interventions, we obtain the causal effect of the n-th neuron's activation value h_n on the m-th kinematic attribute a_m:

$$I_{n,m} = \frac{1}{S}\frac{1}{R}\sum_{S}\sum_{R}\sum_{t=t_0}^{t_0+w}\Delta\hat{a}^t_{n,m} \tag{1}$$

Impact Factor I_n describes the overall impact of h_n on all attributes. The larger I_n is, the more obvious causal effects of the n-th neuron on all attributes are.

$$I_n = \sqrt{\sum_{m=0}^{M-1} I^2_{n,m}} \qquad (I_n \geq 0) \tag{2}$$

Preference Vector $\mathbf{P}_n$ describes the degree of preference of the n-th neuron activation value h_n on each kinematic attribute, where $P_{n,m} = I_{n,m} / \sum_{j=0}^{M-1} |I_{n,j}|$.

$$\mathbf{P}_n = [P_{n,1}, P_{n,2}, \ldots, P_{n,M-2}, P_{n,M-1}] \tag{3}$$

Preference Factor P_n describes the overall preference characteristic of h_n towards all attributes, as shown in Eq. 4. A larger P_n $(0 \leq P_n \leq 1)$ suggests that h_n is more inclined to influence one or a few a_m.

$$P_n = 1 + \frac{1}{\log M}\sum_{m=0}^{M-1} |P_{n,m}| \log |P_{n,m}| \tag{4}$$

B. Locating

Considering behaviors can be viewed as joint results of attributes, we can locate the **X-neuron** with significant causal effects on specific attributes to evoke the agent to behave as we desire by following a coarse-to-fine strategy: first, narrow down the neuron search space, and then identify the optimal neuron u_{n^*} via solving an optimization problem.

a) Narrow down the neuron search space: Given a large number of neurons, we first narrow down the neuron search space to reduce the time complexity of the subsequent optimization problem. Neurons with smaller I_n should to be excluded from the solution space because the premise of effective control is that the change in h_n can cause significant changes in kinematic attributes. A smaller I_n means that regardless of its preference for different attributes, the degree of changes in kinematic attributes is always quite limited. In addition, we will exclude neurons with smaller P_n, because these neurons have a relatively balanced impact on each attribute. Unless specifically designed, it is difficult for all attributes to change in sync with a similar magnitude in most tasks. Eq. 5 describes the set of candidate neurons, where α and β are the minimum thresholds of I_n and P_n.

$$N_{cand} = \{n | I_n > \alpha, P_n > \beta, 0 \leq n \leq N-1\} \tag{5}$$

b) Locate the X-neuron: Next, we will perform preference distribution difference minimization matching according to the desired behavior, which requires various kinematic attributes to change to different degrees and depends on the users' needs and design. For example, controlling the car's direction only requires adjusting the steering and trying not to interfere with the acceleration. The desired changes in kinematic attributes can be represented as $\Delta\acute{a}_{target} = [\Delta\acute{a}_0, \ldots, \Delta\acute{a}_m, \ldots, \Delta\acute{a}_{M-1}]$, and the $\mathbf{P}_n$ of the X-neuron is the closest among all neurons to $\Delta\acute{a}_{target}$:

$$\min \sum_{m=0}^{M-1}\left(P_{n,m} - \Delta\acute{a}_m / \sum_{j=0}^{M-1}|\Delta\acute{a}_j|\right)^2 \quad \text{s.t.} \quad n \in N_{cand}$$

Whether to understand which neuron is responsible for an unexpected behavior, or to evoke the desired behavior using a specific neuron, the corresponding X-neuron has been located successfully. On the one hand, X-neuron n^* has a suitable I_{n^*}, so when an intervention is artificially added to h_{n^*}, kinematic attributes will change significantly; on the other hand, its preference degree for each attribute also meets the user's requirements for changes in each attribute.

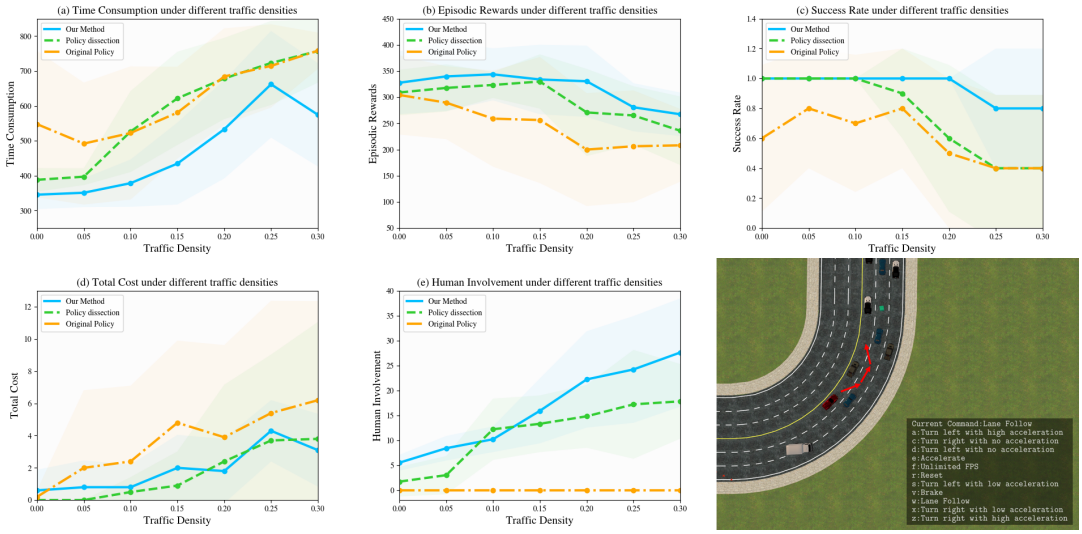

Fig. 3. We compare our method with policy dissection and the original policy without human collaboration under different traffic densities. (a)-(e) show the performances of different methods in terms of time consumption, episodic cost, human involvement, episodic rewards, and success rate under different traffic densities. Our method can get more human assistance when the agent faces hard tasks such as congested road conditions as shown in the lower right corner of the picture. At this time, our method can guide the agent to follow the red mark, while other methods cannot help the agent to pass safely and quickly.

C. Editing

After locating the X neuron u_{n^*}, we need to determine an appropriate editing value e^* to induce the agent to exhibit the desired behavior. Considering that the editing value e^* needs to be determined from the entire real number range, relying solely on manual debugging means extremely high labor costs and extremely low algorithm efficiency. Therefore, we attempt to provide a reference solution to estimate the editing value. We assume that the changes in kinematic properties caused by changes in the hidden state of neurons approach a linear relationship within a certain range. Besides, in the section II-A, we applied a certain human intervention i to h_{n^*} and obtained I_{n^*}. From these, we can derive an estimate of e^*, i.e. $\hat{e}^* = i\sqrt{\sum_{m=0}^{M-1} \Delta \acute{a}_m^2 / I_{n^*}}$.

Instead of overwriting, we add the editing value to the original h_{n^*}, which means the editing does not take over the original policy or replace its decisions but rather integrates human knowledge into the policy inference loop without re-training, enhancing the transferability and generalizability of the agent. In addition, since the neural network is nonlinear, changes in kinematic attributes often do not exhibit a linear relationship with the editing value, which leads to inevitable errors when determining the optimal editing value e^*. In our work, we performed simple fine-tuning based on $\hat{e}^*$ to determine the e^* that achieves optimal control.

III. Experiments

A. Settings

To test the generalization of our method, we evaluate it on various RL tasks ranging from autonomous driving to robot locomotion. In this study, the agents in IsaacGym [17] are only trained to walk forward, while the agents in MetaDrive are trained in a mild and obstacle-free environment with no lane-changing capability. All well-trained policies we used are available from [15] and all experiments are conducted using a GeForce GTX 1660Ti GPU.

TABLE I

Results in IsaacGym-ANYmal. P^o, P^{pd} and P^X respectively refer to the original policy, the policy based on policy dissection, and the policy based on our method.

Method	Episodic rewards	Episodic checkpoints	Success rate	Episodic mileages
P^o	6.345±1.614	0±0	0±0	4.29±0.424
P^{pd}	8.337±5.51	0.4±0.917	0.1±0.3	7.414±4.378
P^X	**19.522±5.807**	**2.6±0.8**	**0.8±0.4**	**18.913±4.365**

In IsaacGym, a quadruped robot called ANYmal that can only walk forward is required to pass a complex maze safely. Each episode is only allowed to terminate when agents crash the wall or reach the destination. Besides, the bipedal robot agent Cassie, capable of only moving forward, needs to leverage complex motor skills to perform parkour stunts.

In MetaDrive, the agent is required to reach the destination safely and quickly. We evaluate different methods in terms of time consumption, episodic rewards, success rate, total cost and human involvement. Specifically, total cost reflects security, and each collision (with obstacles or other vehicles) or driving put of the road will incur total cost +1. Human involvement refers to the number of times editing neurons in each episode. Additionally, obstacles are encountered in testing episodes with their positions generated randomly. Each episode terminates only under three conditions: out of the road, reaching the destination, and exceeding the permissible maximum timesteps. The traffic density is divided into seven levels ranging from easy to hard. The results of all metrics,

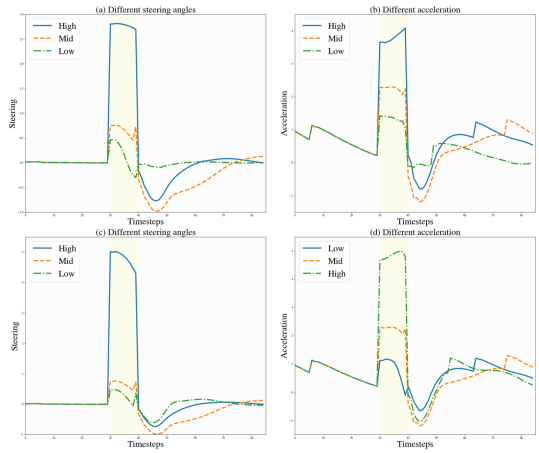

Fig. 4. Fine-grained control in MetaDrive. We impose a left-turn instruction at 50s and cancel it at 60s. The period of human intervention is indicated by the shaded area. (a)-(b) show that acceleration and steering angle maintain a positive correlation according to our requirements, and (c)-(d) show the negative correlation between these two attributes.

including those in the robot locomotion task, are obtained by averaging the outcomes of 10 test episodes. All experiments are restricted to individuals with valid driving licenses, and every participant has signed the experimental agreement and received compensation.

B. Evaluation of the X-neuron

In this section, we compare our method, locating the X-neuron and editing its original activation value to achieve human-AI shared control via interpreting neurons, with policy dissection and the original policy on both autonomous driving and robot locomotion tasks.

In the autonomous driving task, the goal of the agent is to safely reach the destination within 800 timesteps. To cope with complex and changeable road conditions, we define eight distinct behavioral commands, enabling the agent to accelerate, decelerate, and turn left and right with different levels of acceleration. Each behavior is triggered by editing the activation value of the corresponding single X-neuron. As a comparison, we also dissect the same RL policy via policy dissection and set four behavioral commands for acceleration, deceleration, and left and right turns. Fig. 3 illustrates that our approach outperforms other approaches in almost all metrics. This superiority can be attributed to the fact that our approach quantifies the impact of neurons on attributes, thereby providing more behavioral commands to choose. For example, in scenarios with high traffic density, we enable the agent to change lanes promptly with a high acceleration at appropriate instances, whereas other approaches might miss the timing due to slower turning or have no choice but to wait. Our method provides a wide range of options to allow human involvement in the policy decision-making process when necessary, ultimately enhancing the generalizability and safety of the policy.

In the robot locomotion task, the ANYmal agent, which is only trained to walk forward, is tasked with safely traversing an unseen maze containing three checkpoints (Fig. 5). Table I

TABLE II

THE SUCCESS RATE OF LOCATING X-NEURONS FOR DIFFERENT BEHAVIORS BY USING P^{pd} AND P^X $(w = 1, 2, 4, 8)$.

Method	Turn left	Turn right	Accelerate	Decelerate
P^{pd}	0.4	<u>0.8</u>	<u>0.6</u>	<u>0.4</u>
P^X $(w = 1)$	**1.0**	**1.0**	**1.0**	**0.9**
P^X $(w = 2)$	0.9	0.8	1.0	0.8
P^X $(w = 4)$	0.4	1.0	1.0	0.9
P^X $(w = 8)$	<u>0.2</u>	1.0	0.8	0.5

indicates that our method enables the agent to pass more checkpoints safely in the unseen environment compared to other methods, where P^o, P^{pd} and P^X respectively refer to the original policy, the policy based on Policy dissection, and the policy based on our method. Specifically, we locate five distinct X-neurons that can elicit various behaviors, including turning with different yaw angles and stopping. For instance, we command the ANYmal to turn left with a large yaw angle at the first checkpoint, followed by a right turn with a small yaw angle at the second checkpoint. It highlights the need to control the yaw angle in addition to basic steering, as simple turning alone is insufficient to accomplish the task. However, this poses a challenge for both the original policy and policy dissection. Overall, our method achieves zero-shot task transfer in unseen environments, greatly enhancing the transferability of agents. Furthermore, our method is applicable to a wide range of tasks, from autonomous driving to robot locomotion, demonstrating excellent generalization.

C. Fine-grained Control

In this section, we test our method in different environments, all achieving fine-grained human-AI shared control, which means our method can cater to more refined requirements.

In MetaDrive, we can control agents' behaviors via editing activation values of neurons corresponding to acceleration and steering. In Fig. 4, our method not only accommodates single turning or speed commands but also fulfills more intricate requirements. Concretely, according to users' requirements, the agent exhibits a higher acceleration when making a larger steering angle, meaning that these two attributes maintain a positive correlation (also applicable to the negative correlation). A wide array of practical choices allows operators to effectively assist agents, thereby enhancing safety and controllability in challenging environments.

In IsaacGym-ANYmal, we edit activation values of diverse X-neurons to attain multi-gear steering behavior with different yaw angles. Fig. 5(c) displays the walking trajectories of ANYmal under different gears. The larger the trajectory radius, the smaller the steering yaw angle. It should be noted that the original policy has been able to accomplish some simple tasks such as maintaining balance and walking forward. We leverage human knowledge on the foundation of the original policy's function, assisting the agent in exhibiting

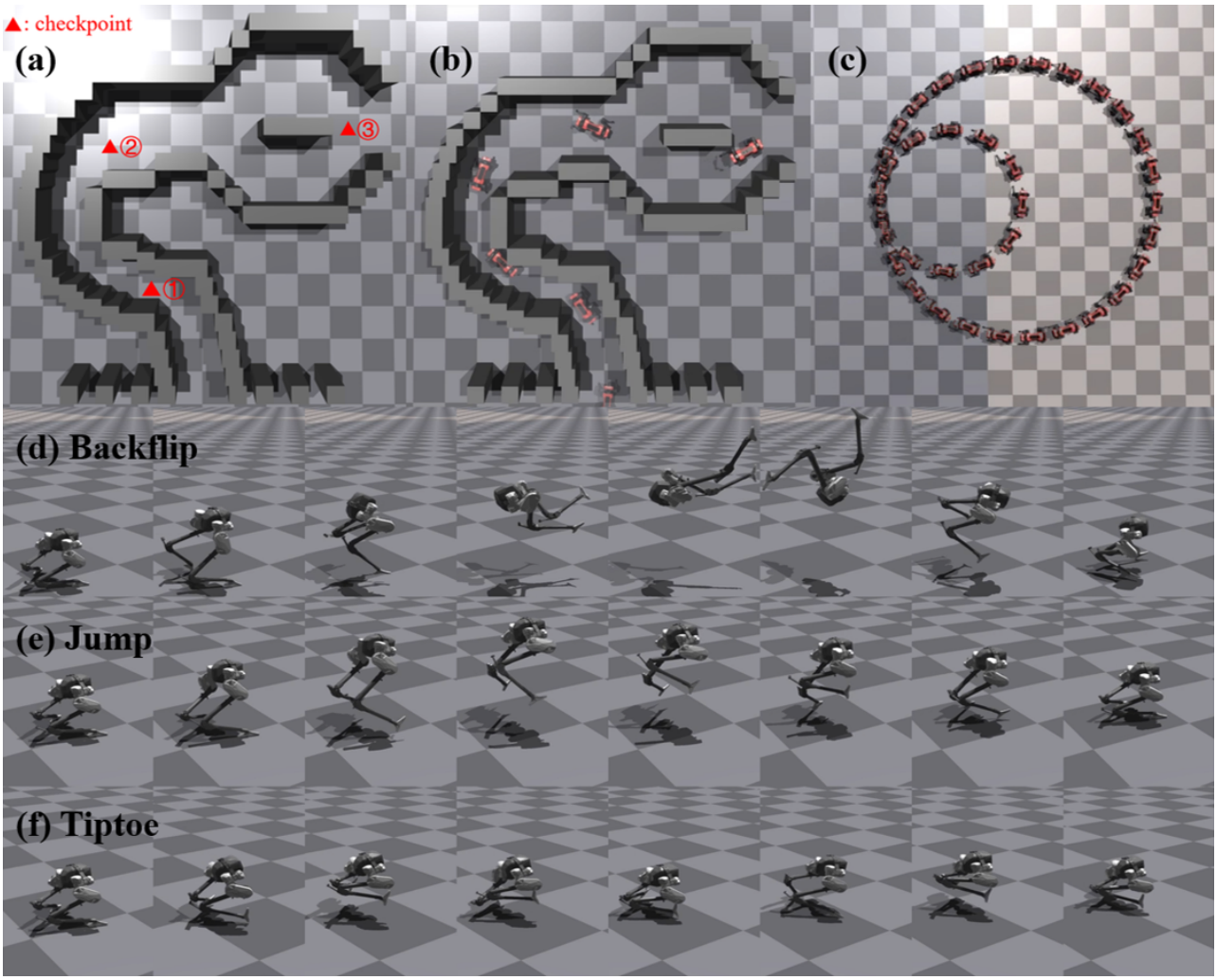

Fig. 5. Maze task and fine-grained control in ANYmal, and complex behaviors in Cassie. (a) shows a top view of the maze, including three checkpoints, which require the agent to change direction at the appropriate time. (b) shows the route that our method successfully helps ANYmal to pass through the maze. (c) shows our method achieves fine-grained control to help ANYmal steer with different yaw angles. (d)-(f) shows the biped robot Cassie performs complex behaviors via X-neurons.

behaviors that are unattainable by the original policy without the need for robot foot trajectory planning. Consequently, the novel behaviors stems from the integration of policy and human guidance. In summary, our method not only enables the agent without steering ability to perform steering but also allows for fine-grained steering control.

D. Unlock Complex Behaviors

In addition to behaviors related to speed and steering mentioned in previous experiments, our method also unlocks more complex behaviors of agents. In Fig. 5, we prompt the biped robot agent Cassie to perform various challenging behaviors such as backflipping, jumping, and tiptoeing via editing the activation values of X-neurons.

It is difficult to achieve complex behaviors solely via human knowledge, but by combining the knowledge with inherent capabilities of the original policy, the difficulty of the problem is greatly reduced. For example, we edit the X-neuron strongly correlated with the knee and ankle joints to apply explosive force, causing the robot to leap and flip backward quickly. Simultaneously, the decisions made by the original policy are dedicated to keeping the robot balanced as much as possible. With the combined effect of human knowledge and the original policy, the robot successfully executes a backflip and lands smoothly. In addition, we also hope the hip joints adduct to alleviate the interference of swinging legs on balance. Each of these behaviors is triggered by editing the activation of a single X-neuron.

E. Ablation Study

Our method is sensitive to the window length w because it directly influences results of I_n, $\mathbf{P}_n$ and P_n, which is first mentioned in Eq. 1. In this section, we investigate the influence of w on the success rate of locating X-neurons and the level of difficulty in determining editing values.

For a desired behavior, there may be multiple neurons strongly associated with it. Therefore, instead of selecting the single optimal neuron as X-neuron, we now consider the

top 10 optimal neurons as X-neurons for a specific behavior, each of which can evoke the desired behavior when edited individually.

We conduct individual tests on these neurons and discover that certain neurons are unable to evoke the desired behavior, indicating a failure in localization. We evaluate the proportion of these 10 neurons that successfully evoke the intended behavior. Furthermore, we investigate the fine-tuning cost required to determine editing values. Fig. 6 shows that there is a decreasing trend in the success rate of locating X-neurons and an increasing trend in the number of iterations (i.e. fine-tuning cost) for determining editing values, as w increases.

We are surprised to find that when w is very small, our approach shows significant improvements in both locating and editing phases: for most behaviors, the success rate of accurately locating X-neurons reaches as high as 100%, and the fine-tuning cost is also very low. We speculate that one possible reason is that changes in attributes resulting from editing h_{n^*} are primarily concentrated in the early stages of editing, and the later changes might become negligible when new editing operations arise. Table II shows that our method noticeably outperforms policy dissection in terms of the success rate of locating X-neurons for different behaviors. Since policy dissection relies entirely on manual debugging when determining editing values, labor cost is much higher than our fine-tuning cost.

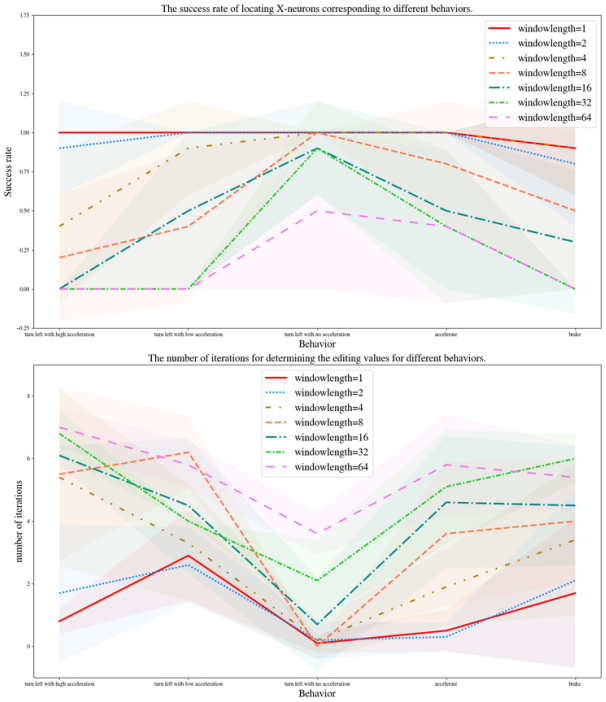

Fig. 6. The success rate of locating and the fine-tuning cost of editing under different w.

IV. RELATED WORK

Neural Network Interpretability. Researchers found abundant interpretable structures from the hidden units. For example, in the task of object classification [18] and scene classification [19], part detector units have been observed. Subsequent research succeeded in quantifying the interpretability of hidden units by aligning the feature map with certain concepts [20]. GAN Dissection locates a group of interpretable units in GANs [21], [22]. Currently, various methods have been developed to identify decisive neurons for generated images [23] or factual predictions [24] with causal intervention [25]. However, previous research mainly focuses on visual and language tasks and little work has been done to understand internal representations in Deep Reinforcement Learning.

Explainable Reinforcement Learning (XRL). Existing XRL methods [26], [27] can be divided into two types, intrinsic explainability, and post-hoc explainability. [28] proposes a method for RL that enhances interpretability through structured perception and relational reasoning. [29] focuses on using saliency maps to understand how an agent learns and executes a policy in a post-hoc way. However, current work mainly focuses on mainstream XRL methods, and few people are concerned about RL methods of leveraging human knowledge. Incorporating humans into the inference loop can greatly enhance interpretability and promote learning efficiency, which is also an important aspect of XRL.

Human-AI Shared Control. Existing research can be divided into three types. First, the human is only involved in the training period and then the well-trained agent can perform tasks without human [30], [31]. The role of humans is to provide expert demonstrations or real-time, scalar-valued feedback. Second, the human is involved in both training and test time. Human and the agent cooperate to complete the task efficiently in Atari Game [32], [33] and Robotic tasks [34]. These two types of methods exhibit superior training speed and efficacy in task completion while suffering from the high labor cost and retraining overhead. Hence, incorporating human during the testing phase is desired, but this line of work is scarce. [15] propose a frequency-based approach to locate neurons associated with corresponding kinematic attributes, and then the agent can act according to human thoughts by editing neurons. But this method doesn't delve into the causality between neurons and behavior, which leads to invalid selection of neurons. Our work is dedicated to studying causality and then realizes efficient identification of neurons.

V. CONCLUSION

We propose a 3-step pipeline to facilitate interpretability and leverage our interpretation to unlock new skills of the policy, ultimately achieving fine-grained human-AI shared control. Our work provides insight into the causal effect of neurons with kinematic attributes, and demonstrates the feasibility of evoking agents to behave as we expect via editing the X-neuron. We evaluate our method in autonomous driving and robot locomotion tasks, indicating that our method performs well in almost all metrics. Furthermore, our method stimulates agents to perform novel and challenging behaviors, which substantially improves the agent's ability to

deal with complex and unseen environments. However, one limitation of our method is that it works via editing only one single neuron. The research on multi-neurons control based on RL interpretability needs to be developed in the future.

a) Safety and Responsible AI: Our approach outperforms previous methods in almost all metrics, including safety. Specifically, we demonstrated multiple hazardous scenarios in the video demo: encountering a traffic jam ahead but failing to timely change lanes resulting in a rear-end collision, or when other vehicles change lanes, agents fail to brake and avoid them in time, leading to an accident. By editing the activation values of X-neurons, we enable the agent to promptly change lanes or brake, thereby significantly enhancing the safety of AI systems and laying the groundwork for the secure deployment in real-world scenarios in the future. Besides, it is necessary to establish licensing standards for use, like a driver's license, otherwise, it is likely to introduce potential safety risks when humans interact with AI.

ACKNOWLEDGMENT

This work is funded in part by the National Key R&D Program of China (2022ZD0160201), Shanghai Artificial Intelligence Laboratory, and STI 2030—Major Projects 2021ZD0201402.

REFERENCES

[1] S. Feng, H. Sun, X. Yan, H. Zhu, Z. Zou, S. Shen, and H. X. Liu, "Dense reinforcement learning for safety validation of autonomous vehicles," *Nature*, vol. 615, no. 7953, pp. 620–627, 2023.

[2] J. Chen, S. E. Li, and M. Tomizuka, "Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5068–5078, 2021.

[3] K. Stachowicz, D. Shah, A. Bhorkar, I. Kostrikov, and S. Levine, "Fastrlap: A system for learning high-speed driving via deep rl and autonomous practicing," *arXiv preprint arXiv:2304.09831*, 2023.

[4] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[5] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[6] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[7] P. Schegg, E. Ménager, E. Khairallah, D. Marchal, J. Dequidt, P. Preux, and C. Duriez, "Sofagym: An open platform for reinforcement learning based on soft robot simulations," *Soft Robotics*, vol. 10, no. 2, pp. 410–430, 2023.

[8] S. W. Abeyruwan, L. Graesser, D. B. D'Ambrosio, A. Singh, A. Shankar, A. Bewley, D. Jain, K. M. Choromanski, and P. R. Sanketi, "i-sim2real: Reinforcement learning of robotic policies in tight human-robot interaction loops," in *Conference on Robot Learning*. PMLR, 2023, pp. 212–224.

[9] A. Fernandez-Quilez, "Deep learning in radiology: ethics of data and on the value of algorithm transparency, interpretability and explainability," *AI and Ethics*, vol. 3, no. 1, pp. 257–265, 2023.

[10] J. Xing, T. Nagata, X. Zou, E. Neftci, and J. L. Krichmar, "Achieving efficient interpretability of reinforcement learning via policy distillation and selective input gradient regularization," *Neural Networks*, vol. 161, pp. 228–241, 2023.

[11] Q. Li, Z. Peng, L. Feng, Q. Zhang, Z. Xue, and B. Zhou, "Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning," *IEEE transactions on pattern analysis and machine intelligence*, 2022.

[12] Z. Juozapaitis, A. Koul, A. Fern, M. Erwig, and F. Doshi-Velez, "Explainable reinforcement learning via reward decomposition," in *IJCAI/ECAI Workshop on explainable artificial intelligence*, 2019.

[13] A. Raffin, A. Hill, R. Traoré, T. Lesort, N. Díaz-Rodríguez, and D. Filliat, "S-rl toolbox: Environments, datasets and evaluation metrics for state representation learning," *arXiv preprint arXiv:1809.09369*, 2018.

[14] P. Sequeira, E. Yeh, and M. T. Gervasio, "Interestingness elements for explainable reinforcement learning through introspection." in *IUI workshops*, vol. 1, 2019.

[15] Q. Li, Z. Peng, H. Wu, L. Feng, and B. Zhou, "Human-ai shared control via policy dissection," *arXiv preprint arXiv:2206.00152*, 2022.

[16] H. Liang, L. Yang, H. Cheng, W. Tu, and M. Xu, "Human-in-the-loop reinforcement learning," in *2017 Chinese Automation Congress (CAC)*. IEEE, 2017, pp. 4511–4518.

[17] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 91–100.

[18] A. Gonzalez-Garcia, D. Modolo, and V. Ferrari, "Do semantic parts emerge in convolutional neural networks?" *International Journal of Computer Vision*, vol. 126, pp. 476–494, 2018.

[19] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," *arXiv preprint arXiv:1412.6856*, 2014.

[20] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6541–6549.

[21] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Gan dissection: Visualizing and understanding generative adversarial networks," *arXiv preprint arXiv:1811.10597*, 2018.

[22] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1532–1540.

[23] Y. Xu, Y. Shen, J. Zhu, C. Yang, and B. Zhou, "Generative hierarchical features from synthesizing images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4432–4442.

[24] K. Meng, D. Bau, A. Andonian, and Y. Belinkov, "Locating and editing factual associations in gpt," *Advances in Neural Information Processing Systems*, vol. 35, pp. 17 359–17 372, 2022.

[25] K. Meng, A. S. Sharma, A. Andonian, Y. Belinkov, and D. Bau, "Mass-editing memory in a transformer," *arXiv preprint arXiv:2210.07229*, 2022.

[26] R. Dazeley, P. Vamplew, and F. Cruz, "Explainable reinforcement learning for broad-xai: a conceptual framework and survey," *Neural Computing and Applications*, pp. 1–24, 2023.

[27] G. A. Vouros, "Explainable deep reinforcement learning: state of the art and challenges," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–39, 2022.

[28] G. Cideron, M. Seurin, F. Strub, and O. Pietquin, "Self-educated language agent with hindsight experience replay for instruction following," *ArXiv*, vol. abs/1910.09451, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:204800351

[29] S. Greydanus, A. Koul, J. Dodge, and A. Fern, "Visualizing and understanding atari agents," in *International conference on machine learning*. PMLR, 2018, pp. 1792–1801.

[30] R. Zhang, F. Torabi, G. Warnell, and P. Stone, "Recent advances in leveraging human guidance for sequential decision-making tasks," *Autonomous Agents and Multi-Agent Systems*, vol. 35, no. 2, p. 31, 2021.

[31] W. Saunders, G. Sastry, A. Stuhlmueller, and O. Evans, "Trial without error: Towards safe reinforcement learning via human intervention," *arXiv preprint arXiv:1707.05173*, 2017.

[32] J. Bragg and E. Brunskill, "Fake it till you make it: Learning-compatible performance support," in *Uncertainty in artificial intelligence*. PMLR, 2020, pp. 915–924.

[33] P. Knott, M. Carroll, S. Devlin, K. Ciosek, K. Hofmann, A. D. Dragan, and R. Shah, "Evaluating the robustness of collaborative agents," *arXiv preprint arXiv:2101.05507*, 2021.

[34] H. J. Jeon, D. P. Losey, and D. Sadigh, "Shared autonomy with learned latent actions," *arXiv preprint arXiv:2005.03210*, 2020.