Fuzzy Centered Explainable Network for Reinforcement Learning

Liang Ou, Yu-Chen Chang, Yu-Kai Wang, *Member, IEEE,* Chin-Teng Lin, *Fellow, IEEE,*

***Abstract*—The Explainability of reinforcement learning (RL) models has received vast amount of interest as its applications have widened. Most existing explainable RL (xRL) models focus on improving the explainability of an agent's observations instead of the relationships between agent states and actions. This study presents a fuzzy centered explainable network (FCEN) for RL tasks to interpret the relationships between agent states and actions. The proposed FCEN leverages the interpretability of fuzzy neural networks (FNNs) to establish if-then rules and a generative model to visualize learned knowledge. Precisely, the FCEN includes if-then rules that formulate state-action mappings with human-understandable logic, such as the form "IF Input is A THEN Output is B". In addition, these rules connect with a generative model that concretizes the states into human-understandable patterns (figures). Our experimental results obtained on 4 Atari games show that the proposed FCEN can achieve a high level of performance in RL tasks and enormously boost the explainability of RL agents both globally and locally. In other words, the FCEN maintains a high-level explanation for the agent decision logic and the possibility of low-level analysis for each given observation sample. The explainability boost does not undermine reward learning performance, humans can even enhance the agent's performance with the provided explainability.**

***Index Terms*—Explainable AI, Reinforcement Learning, Fuzzy Neural Network, Generative Model.**

I. Introduction

EXPLAINABLE AI (xAI), especially explainable reinforcement learning (xRL), has attracted increasing attention from researchers in recent years. This is because the majority of current AI systems depend on complex black-box deep neural networks (DNNs), which are difficult for human beings to understand. For example, the popular GPT-3 language model[1] includes 96 attention layers and 96 heads, producing a total of 17.5 million parameters. Although such a model achieves excellent performance, its decision logic is too complicated for humans to understand. Such low transparency leads to unpredictable risks for the industry that adopts the AI system. Therefore, building a model with high explainability becomes an urgent task. This is where the fuzzy neural networks (FNNs) shine. With their intrinsic explainability, FNNs map the decision logic of an AI model into a set of if-then rules, which are similar to human logic. This paper introduces a Fuzzy Centered Explainable Network (FCEN) that provides a solution for understanding the decision-making process of RL agents.

Liang Ou, Yu-Chen Chang, Yu-Kai Wang and Chin-Teng Lin are with the School of Computer Science, University of Technology Sydney, Ultimo NSW 2007, Australia. (email: liang.ou-1@student.uts.edu.au; yucheng.chang@uts.edu.au; yukai.wang@uts.edu.au; chin-teng.lin@uts.edu.au).

We now briefly discuss the key concepts and recent works on xAI and xRL. Montavon et al. [2] defined explainability as "the collection of features of the interpretable domain, that have contributed for a given example to produce a decision (e.g., classification or regression)". In other words, xAI is used to improve human understanding of machine learning models. Many efforts have been made for xAI. According to Adadi and Berrada [3], the interpretability of a model can be classified into two aspects: global interpretability and local interpretability. While global interpretability focuses on the model structure, local interpretability targets the specific decision-making process of the model. From another perspective, Adadi and Berrada [3] classified explainable methods into intrinsic methods post hoc methods. Specifically, the intrinsic methods rely on the models' structures themselves, while the post-hoc methods utilize another tool to analyze the models after they are trained.

From the perspective of RL tasks, the explanations of an agent often focus on its black-box policy. As Puiutta [4] discussed, the current xRL methods mainly try to mimic a complex model, making them typical post hoc methods. Several several studies have established intrinsic explainable models for RL agents. One attempt by Rahimi et al. [5] views an RL model as a Tsetlin Machine, thus providing intrinsic explainability for the agent. However, this method works only with a simple grid-world environment, making it useless for an environment with complex inputs. RMIX [6] was suggested as an attention-based explainable method for visualizing the risk of a risk-sensitive RL task. This method divides the training process of an agent into two phases: the first phase learns the attention of an agent, and the second phase learns a visualization of the risk. Although this method combines both intrinsic method and post hoc methods, it only works in 2D navigation environments that cannot be generalized. This is similar to the work of [7], where the environment was not generalizable.

Some attempts have been made to establish a generalized intrinsic explainable method. Mask-A3C [8] learns the attention of an agent, while a counterfactual state explanation method [9] decomposes the reward function to analyze whether the reward has a negative or positive influence on agent performance. However, such attempts sacrifice the performance of the agent by imposing a hard restriction on the agent in the model architecture, since it forces the agent to follow some prior human knowledge to learn its behavior. To avoid this issue, in this paper, the proposed FCEN does not impose any hard restrictions on the agent but tries its best to understand its decision rules. As Fig.1 shows, the explainability of the

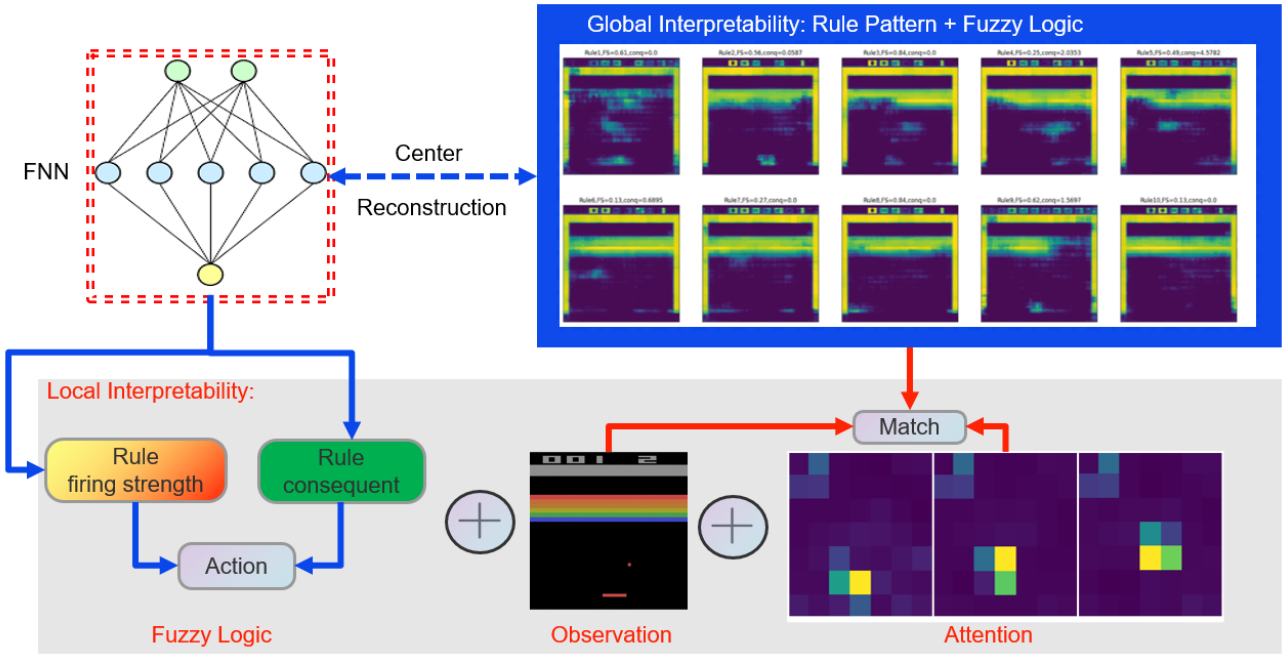


Fig. 1: The key components of the FCEN for explaining an agent's behavior. The FCEN maintains a global explanation through fuzzy rule pattern recognition and local explanations through the values of rule firing strengths and consequents.

FCEN comes from the fuzzy logic of its FNN. We visualize the learned pattern recognized by the fuzzy rules and leverage the internal attention of the FCEN model to understand the decision-making process of the agent. Such a design helps humans understand the high-level decision logic of the RL agent. In other words, at the global level, the FCEN preserves the inner pattern that can explain the agent's high-level decision rules. At the local level, for each given sample, the FCEN can explain the decision process from observation to action by following fuzzy logic. Furthermore, a human can optimize the model by reviewing the explanation given for the agent. Therefore, our contributions in this paper are as follows.

1) We propose an innovative approach (FCEN) for explaining agents' behaviors in RL tasks. The FCEN visualizes fuzzy sets so that humans can understand the intuitive rules learned by the agent.
2) The proposed FCEN is both globally and locally explainable, with a good generalization ability. Global explainability is fulfilled by fuzzy rule visualization, while local explainability is achieved by attention and the inner states of the FNN.
3) Without any hard restriction, the FCEN does not sacrifice performance in exchange for explainability.
4) The FCEN provides the possibility to improve the agent model by understanding its decision logic.

II. Related Work

This section summarizes some previous works related to the proposed FCEN.

A. Leveraging FNNs for xAI

The inherent interpretability of fuzzy inference systems (FISs) and FNNs has drawn researchers' attention with regard to explaining AI models. As discussed in Handbook on Computer Learning and Intelligence [10], the FNN is a typical evolving fuzzy systems (EFS) that can be learned on-the-fly during online processes in an incremental and mostly single-pass manner. The book also points out explainability is one of the major advantages of such systems. One study [11] in 2018 suggested that the interpretability of FNNs/FISs comes from symbols with fuzzy sets and linguistic terms that are coherent with human semantics. Many researchers have modeled AI systems directly with FNNs or FISs and learn with either swarm intelligence [12], [13] or gradient-based Q-learning [14], [15], [16], [17]. In addition, some researchers have used FNNs to explain deep models. For example, Soares et al. [18] used an FNN as a surrogate model to mimic a pretrained double deep Q-network (DDQN). To visualize the learned fuzzy rules, they proposed the MegaCould method to match fuzzy rules and principal components. Together with other similar approaches [19], [12], the high explainability of FNNs has been demonstrated. Another recent study [20] proposed a model for identifying the number of barriers needed for intrusion detection in a wireless sensor network. It demonstrated that the FNN-based method can outperform state-of-the-art models in terms of accuracy and stability, and provided interpretability of the data. However, these approaches work only on low-dimensional data inputs. If the scale of the input dimensionality is as large as the pixels of an image, both the performance and the explainability of these approaches will be undermined. Therefore, in this paper, we include a feature extractor to reduce the input dimensionality and a decoder to explain the extracted features.

B. Self-Attention for xRL

Recent research has shows that transformers [21] can achieve high performance in both the natural language processing (NLP) and image processing fields. The inherent explainability of transformers, which comes from their self-attention mechanisms, has also been utilized by researchers working with RL tasks. A. Manchin et al. [22] concatenated a convolutional neural network (CNN) with a self-attention block before the final action-critic decision network. DeepCC [23] combines long short-term memory (LSTM) with a self-attention block to produce actions. The decision transformer [24] and DTQN [25] model the sequential observations of an RL agent as the positional embeddings of a transformer. MAIRL [26] even combines self-attention adversarial inverse RL (AIRL). In this paper, we include a compact convolutional transformer (CCT) [27] as a feature extractor. Such a design helps the FCEN maintain a good level of performance while providing the possibility of viewing the agent's attention for explainability purposes.

C. Generative Models for xRL

Humans prefer concrete explanations, such as examples, rather than sequences of numbers and formulas. A generative model is one of the best tools for achieving such explanations. Many studies have explored the possibility of including a generative model for explaining RL agents. InfoRL [28] disentangles near-optimal trajectories with an InfoGAN [29] that predicts the next state. Similarly, the self-consistent trajectory autoencoder [30] uses a variational autoencoder (VAE) to predict the next state and select the policy with the best prediction. Note that these approaches all try to learn the state transition function of an RL environment, which incorporates the knowledge of the state transition function of the RL environment into the agent. However, in this paper, the role of the generative model is only to explain fuzzy sets. Therefore, we define a decoder to reconstruct the latent space as simply as possible.

D. Studies on Explaining Atari Games

There have been various challenges with previous xRL methods, notably their limited applicability in specifically defined environments, which ultimately undermines their capacity to generalize effectively. To address this, our research focuses on image-based Atari games as RL environments, which are inherently more comprehensive than simple environments like grid worlds. Previous studies have attempted to explain the actions of trained Atari game agents using methods such as the traditional occlusion-based saliency map [31] technique or the ad hoc surrogate approximation [32] and perturbation-based saliency maps [33]. However, these approaches rely on post-hoc calculations and are computationally intensive, making online analysis impractical. Intrinsic methods, which can generate explanations on the fly, are much more demanding. One such example is Counterfactual States [9], which generates counterfactual states to explain why an agent is not taking a specific action. Another example is Masked-A3C [8], which learns attention for an observation using a 1-D convolutional mask. However, these intrinsic methods impose special architecture restrictions on the agent, which can impact performance. In contrast, our proposed method does not impose hard architecture restrictions, guaranteeing the agent's performance while still being applicable in environments with low-dimensional input spaces.

III. FCEN

The proposed FCEN comprises three main components, namely a fuzzy-based decision network (FDN), a CCT feature extractor, and a CNN-based latent decoder, as illustrated in Fig.2. The FDN is capable of learning fuzzy if-then rules from the input data, thereby facilitating human users to review the decision network's decision logic, ensuring the explainability of the RL agent. The FDN is fed with a latent vector obtained from the CCT, which provides informative features and introduces attention to aid in training and explaining the RL agent. To elucidate the fuzzy rules, we utilize a decoder that reconstructs the latent variable $z \in \mathcal{Z}$ back into the input $x \in \mathcal{X}$. The subsequent section expounds on the nuances of the three components and the learning strategy of the agent.

A. FDN

In order to ensure the inherent explainability of the RL agent, our framework employs an FNN to model the actor. This approach enforces the agent to operate based on a set of fuzzy rules, which in turn guarantees transparency in the explanations provided to human users. Specifically, given m fuzzy rules, the i^{th} fuzzy rule held in the agent is expressed as:

$$\text{If } z \text{ is } M_i, \text{ Then the agent actions are } f(\mathbf{c_{ij}} z + b_{ij}), \quad (1)$$

where z a the feature extracted from x into a latent space $\mathcal{Z} \subseteq \mathbb{R}^d$. $\mathbf{c_{ij}}$ and b_{ij} are the tunable parameters of the consequence of the rule, while j is the action index. $f(\cdot)$ is an activation function that introduces nonlinearity. In this paper, we adopt rectified linear unit (ReLU) for $f(\cdot)$. The utilization of the ReLU function in the FDN can significantly enhance its explainability. This is attributed to the fact that the ReLU function eliminates negative values and retains only positive ones, leading to a more straightforward explanation of the FDN's decision-making process. This is evident in the outcomes presented in Table II, III, IV and V, where the majority of consequent action values are zero. Consequently, only the actions with non-zero values under the fired rules influence the ultimate decision-making of the FDN. A comprehensive and in-depth discussion of the explainability analysis will be expounded in section IV-B.

Given that the explainable model operates in a reinforcement learning environment that is inherently characterized by high levels of uncertainty, including incomplete or noisy observations of the environment, stochasticity in the environment dynamics, and randomness in the agent's actions, it is imperative to adopt a membership function that exhibits sufficient smoothness and flexibility. The Gaussian membership function, in this regard, is particularly well-suited to such

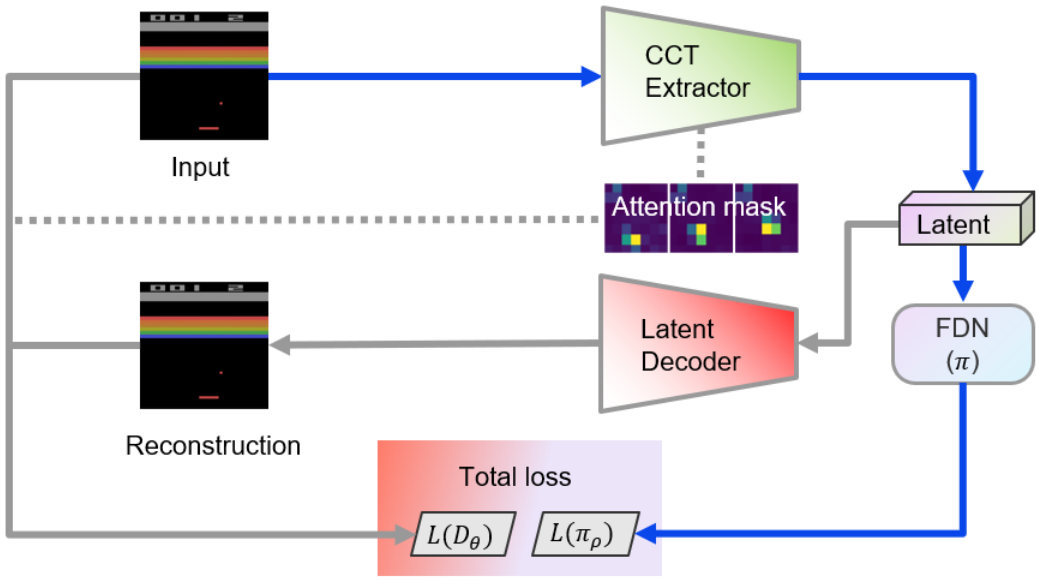


Fig. 2: The FCEN architecture integrates and interacts with three components: CCT Extractor, FDN, and latent decoder. The CCT Extractor generates a latent vector from the input frame, which is used as input to both FDN and the latent decoder. FDN produces the final action of the agent, while the latent decoder help generates human-interpretable explanations of the decision-making process.

tasks. This is primarily due to the fact that the exponential of the distance metric employed in Gaussian membership renders it less susceptible to the influence of extreme outliers. Moreover, the adjustable widths associated with the Gaussian membership offer enhanced flexibility to each set, thereby enabling the model to more effectively handle the challenges posed by the uncertain nature of the RL environment. In this article, the Gaussian membership function of each fuzzy set M_i is provided by:

$$M_i\left(z_k\right) = exp\left\{-\left(\frac{z_k - \mu_k^i}{\sigma_k^i}\right)^2\right\}, \tag{2}$$

where μ_k^i denotes the center of the fuzzy set at the k^{th} dimension and σ_k^i denotes the width of the fuzzy set at the k^{th} dimension. The membership function is deemed to be a measurement of the membership (similarity) degree between the fuzzy set center and the input latent sample. The degree of membership indicates the similarity between the fuzzy set pattern and the latent variable z. The firing strength φ_i of a fuzzy rule i is expressed as:

$$\varphi_i\left(z\right) = \mathcal{O}(M_i\left(z_k\right)), \tag{3}$$

where $\mathcal{O}(\cdot)$ is the general operation term. In practice $\mathcal{O}(\cdot)$ can be the summation, product, min, max or mean operation. In this paper, we adopt the mean value for $\mathcal{O}(\cdot)$. In other words, the firing strength of a the rule comes from its average membership value across all dimensions. By combining the firing strength of the rule and the consequent values of the rule, the final output of the FNN is expressed as

$$a_j = \sum_i^m \frac{\varphi_i(z) f(\mathbf{c_{ij}} z + b_{ij})}{\sum_i^m \varphi_i(z)}, \tag{4}$$

where $\frac{\varphi_i(z) f(\mathbf{c_{ij}} z + b_{ij})}{\sum_i^m \varphi_i(z)}$ is the output of rule i for action j, and $f(\mathbf{c_{ij}} z + b_{ij})$ is the consequent value of rule i and action j. Here, we adopt the first order of consequent, so that the consequent value is the linear combination of the latent variable z with an activation function $f(\cdot)$. The firing strengths are normalized with the denominator $\sum_i^m \varphi_i(z)$. The final output of an action a_j is derived by the combination of all rules.

Such a design helps us explain the agent's behavior with a set of fuzzy rules. Specifically, the FNN can tell us which pattern(s) the agent identifies and what value(s) the agent gives to the pattern(s). However, the meaning of a fuzzy set in a latent space is still vague to humans. Therefore, this paper includes two models to enhance the explainability of the agent: an attention-based feature extractor and a latent decoder.

B. CCT Extractor

The CCT Extractor is denoted by $E(x)$ and modeled with a CCT [27]. The purpose of adopting the CCT as the feature extractor is to enable explainability from end to end. There CCT provides several advantages.

1) The CCT includes a self-attention mechanism that enables users to analyze the agent with its inner states.
2) The self-attention mechanism can also help train the latent decoder. This part is discussed in the next section.
3) The CCT is much lighter than a vision transformer(ViT) [34], making it suitable for RL environments.

Note that the third item is critical under an RL environment. This is because the positional embedding operation in a ViT consumes a tremendous amount of memory, which leads to an out-of-memory (oom) issue when the RL optimizer calculates a gradient with thousands of observations. The architecture

TABLE I: Parameter settings for the CCT extractor

Blocks	Parameter settings
Conv block $\times 2$	Conv2d: kernel size=7, stride=2, padding=3; Maxpooling: kernel size=3, stride=2, padding=1
Transformer block $\times 1$	Number of attention heads = 3, embedded dimensionality = 96, feed forward nodes = 288

of the CCT is also designed to be shallow, where only 1 transformer block is included, followed by 2 convolutional blocks. The hyperparameters of the CCT are listed in the following table.

C. Latent Decoder

The latent decoder is another key component of the proposed network, although its architecture is very simple. The purpose of including the latent decoder is to explain the latent space $\mathcal{Z}$:

$$D_\theta : \ \mathcal{Z} \to \mathcal{X}, \ \mathcal{Z} \simeq \mathbb{R}^d, \mathcal{X} \simeq \mathbb{R}^h, \tag{5}$$

With the decoder, we can map the latent space back to the observation space and then understand the inner state of the model. Specifically, since our model keeps the learned fuzzy set inside the agent, decoding the center of the fuzzy set explains the pattern of the center of the fuzzy set:

$$D(\boldsymbol{\mu}^i) = \mathbf{p_i} \in \mathcal{X}, \tag{6}$$

where $\boldsymbol{\mu}^i$ is the center of the i^{th} rule and $\mathbf{p_i}$ is the reconstructed pattern for the i^{th} center. While this design is simple and intuitive, it can convert a black box-generated fuzzy center into a human-understandable image pattern. With the help of the latent decoder, for each input image $\mathbf{x}$, each fuzzy rule can be represented as:

$$\text{If } \mathbf{x} \text{ is pattern } \mathbf{p_i}, \text{ then the agent actions are } f(\mathbf{c_{ij}z} + b_i). \tag{7}$$

However, the obvious underlying truth is that the pattern generated by the decoder D dominates the explanation of the proposed approach. Therefore, the decoder must learned smartly so that it faithfully represents the key information of the game frames. The learning strategy of the decoder is discussed in the following section.

D. Learning Algorithm

To enable the intrinsic explainability of the proposed approach, the agent learns a policy and the explanation of the policy simultaneously. Algorithm 1 shows the workflow of the FCEN's learning process. In each iteration, the agent first collects a batch of data samples stored in its buffer. These data include the observations received by the agent, the corresponding actions that the agent has taken, and the reward received at that time. Second, the agent samples a batch of data to optimize its parameters. In the third step, the loss function is calculated following the proximal policy optimization (PPO) [35] learning scheme. The loss function has 3 components:

$$L(\pi_\rho) \ = L^p(\pi_\rho) + c_1 L^v(\pi_\rho) \ + c_2 L^{en}(\pi_\rho), \tag{8}$$

Algorithm 1: Learning Algorithm of the FCEN

Initialize: max_step, $batch\_size$, $current\_step = 0$, buffer $B = \emptyset$.
while $current\_step \leq max\_step$ **do**
1. The agent interacts with the environment for l steps, with the history recorded in $\mathbb{H}^l$. Add the history $\mathbb{H}^l$ to the buffer: $B = B \cup \mathbb{H}^l$;
2. Sample $R \subset B$;
3. Agent policy loss: $L(\pi_\rho) \ = L^p(\pi_\rho) + c_1 L^v(\pi_\rho) \ + c_2 L^{en}(\pi_\rho)$;
4. Latent reconstruction loss: $L(D_\theta) \ = \ \frac{1}{N} \sum_{o \in O} M_o^{rec} |x_o \ - \ y_o|$;
5. Total loss: $L_{all} = L(\pi_\rho) \ + L(D_\theta)$;
6. Optimize the FCEN;
7. $current\_step += rollout\_step$;
end

where $L^p(\pi_\rho)$ is the policy loss, $L^v(\pi_\rho)$ is the value loss, $L^{en}(\pi_\rho)$ is the entropy loss, and c_1 and c_2 are the coefficients of $L^v(\pi_\rho)$ and $L^{en}(\pi_\rho)$, respectively. Specifically, the policy loss is provided by:

$$L^p(\pi_\rho) = -\min(\hat{A} r_\rho, \text{clip}(r_\rho, 1-\epsilon, 1+\epsilon)\hat{A}), \tag{9}$$

where $\hat{A}$ is the advantage value and r_ρ is the probability ratio derived from the trust region policy optimization (TRPO) process [36]. The value loss is provided by

$$L^v(\pi_\rho) = \text{MSE}(G - V_\rho), \tag{10}$$

where G is the ground-truth return from the buffer and V_ρ is the predicted value. The entropy loss is provided by

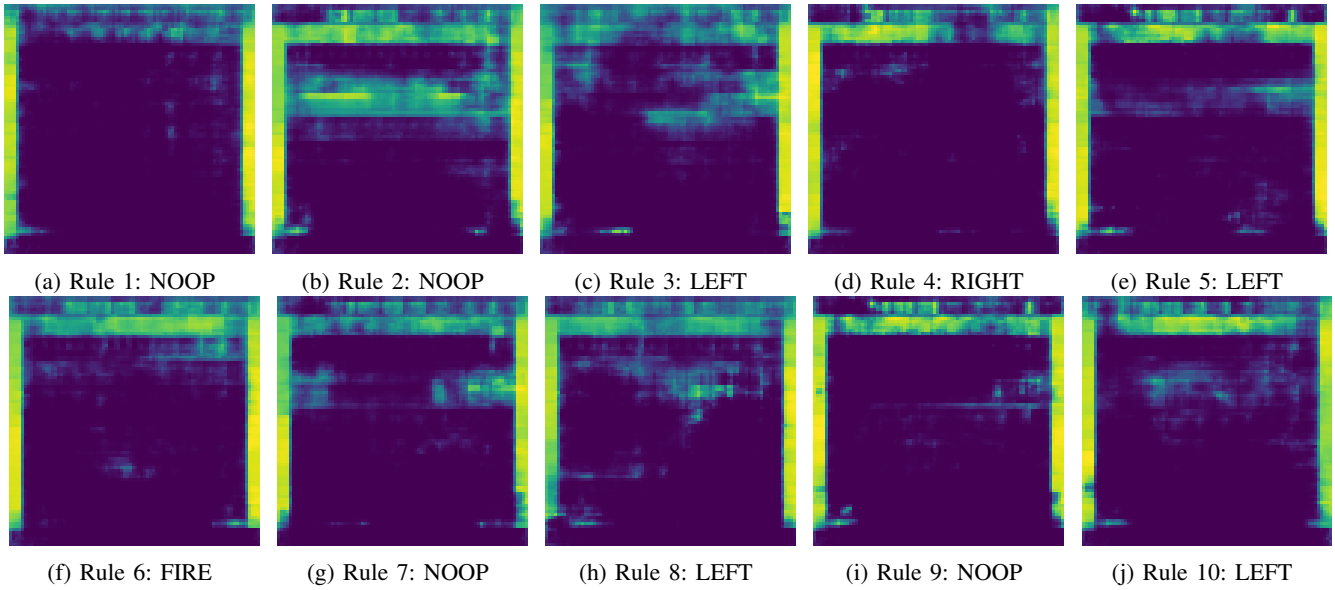
$$L^{en}(\pi_\rho) = -\hat{\mathbb{E}}[-log(P(a))], \tag{11}$$

where P(a) is the actions probability. In the fourth step, the model learns to explain the latent reconstruction via the mean squared error (MSE) loss function:

$$L(D_\theta) \ = \ \frac{1}{N} \sum_{o \in O} ||x_o \ - \ y_o||_2, \tag{12}$$

where N is the size of the image, o is one pixel of image O, x_o is the ground truth value of o and y_o is the predicted value obtained from the decoder. In our design, the gradient of the reconstruction loss can either propagate to both the CCT encoder E and the decoder D or only propagate to the reconstruction network D. An experiment shows that freezing the encoder for reconstruction has a positive effect on policy learning and does not affect the reconstruction quality of the raw image. Therefore, in this paper, we only train the decoder D, which is then considered the explainer of the latent space.

However, we find that during the reconstruction of the raw image, the network often misses some of the key factors, especially the pixels that construct the agent itself. This is because such key factors only consist of a small part of the image, so the loss function (12) tends to overweight the background, especially the fixed parts of the background. To solve this problem, we introduced a task-relevant mask that helps the decoder focus on the key factors. Our solution is



(a) Rule 1: NOOP (b) Rule 2: NOOP (c) Rule 3: LEFT (d) Rule 4: RIGHT (e) Rule 5: LEFT

(f) Rule 6: FIRE (g) Rule 7: NOOP (h) Rule 8: LEFT (i) Rule 9: NOOP (j) Rule 10: LEFT

Fig. 3: Representations of 10 rules centers learned under the Breakout environment. Each of the images is reconstructed with the decoder D, which is trained with the loss from Equation 14. These rules are explained as follows: if a rule pattern is identified, the agent takes the corresponding action.

simple and effective, and we extract the attention from the CCT extractor, normalize it and calculate its batch maximum. Then, the processed attention serves as the reconstruction mask that filters out the key part of the input image. The reconstruction mask is formulated as:

$$M^{rec} = \max_{\text{Attention}_g \in \text{Attention}_\text{B}} (\text{Norm}\,(\text{Attention}_g)), \quad (13)$$

where Attention_g represent the self-attention for image g, Attention_B is the set of all self-attention values in the batch, and the normalization method adopted in this paper is min-max normalization, which keeps the maximum attention as 1. This method downweights the background importance while retaining the key parts. Then, with such a reconstruction mask, we update the reconstruction loss function as follows:

$$L(D_\theta) \;=\; \frac{1}{N}\sum_{o\in O} M_o^{rec}||x_o \;-\; y_o||_2, \quad (14)$$

where M_o^{rec} is the mask value at position o. Then, in the 5^{th} step, we calculate the total loss,

$$L_{all} = L(\pi_\rho) + L(D_\theta), \quad (15)$$

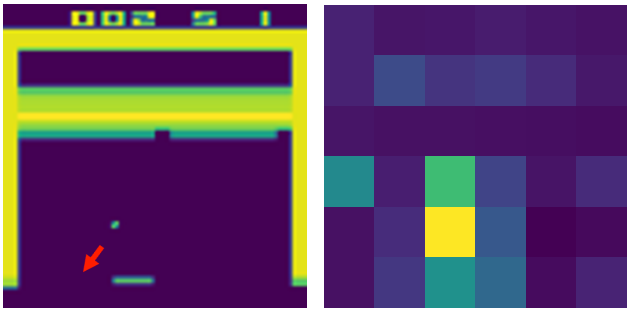
and optimize the whole network with the PPO gradient clipping technique, as described in [35]. This approach effectively constrains the magnitude of the gradient updates, preventing large changes that may otherwise destabilize the training process.

IV. Explainability Analysis and Results

We analyze the proposed FCEN under the "Breakout" environment, where the goal is to destroy bricks by moving a paddle and hitting the ball. The four possible actions in the environments are "NOOP" - doing nothing, "FIRE" - kicking the ball with the paddle, "LEFT" - moving the paddle to the left and "RIGHT" - moving the paddle to the right. The proposed FCEN is a pure intrinsic explainable method, without any additional tools or algorithms for analyzing the agent in an ad hoc manner. This section discusses both the global explainability and local explainability of the network.

A. Structurewise Global Explainability with Fuzzy Rule Visualization

Since our approach models the agent with a set of fuzzy rules, global explainability is achieved through fuzzy logic. Even without any data sample, a human can view the inner pattern of a fuzzy set to explain the high-level decision-making strategy of the agent. Lakkaraju et al. [37] claimed that humans prefer example-based explanations rather than saliency maps. By providing a rule pattern to reviewers, our approach is more similar to an example-based interpretation and thus is more friendly to human reviewers than other saliency-based methods. Fig.3 is an example of 10 fuzzy sets' explanations acquired from a pretrained agent under the Breakout environment. These images are considered the patterns of fuzzy sets, and they are matched with the features via Equations 2 and 3 to produce the firing strength of a certain rule. Each of the fuzzy set centers is listed with its highest corresponding consequent (action). By reviewing this pattern of fuzzy sets, humans can understand the rule-based logic of the agent. Specifically, using rule 8 as an example, it can be explained as follows: if the agent observes the ball in the middle-left with the paddle at the right corner, rule 8 will be fired. In such a situation, the agent prefers to go left. Note that all the rules are learned by the agent itself, and they may not always align with human intuition. However, decoding the agent's logic provides insight into the agent's policy. The human can then either learn game strategy from



(a) observation (b) attention

Fig. 4: Example of LEFT

the agent or correct the agent's wrong behavior and improve its policy.

B. Featurewised Local Explainability

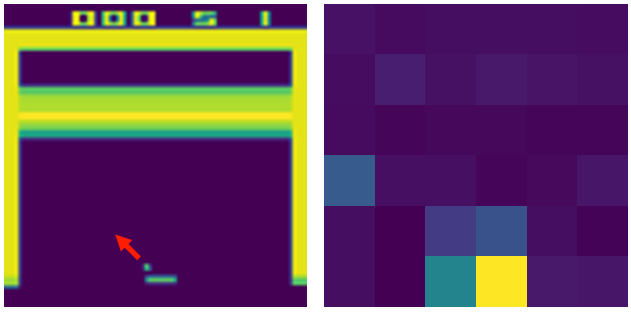
Alongside the global explainability achieved by reviewing the fuzzy set patterns, the proposed framework can provide local explanations by utilizing a single data sample. We can analyze the decision-making of an agent through fuzzy logic together with attention and rule visualization. Fig. 4(a) is an example of an input game frame, in which the ball is falling toward the bottom-left corner of the frame. The attention shown in Fig. 4(b) reveals that the CCT extractor correctly focuses on the most relevant part of the input frame, which suggests the match of the firing strength pattern. In Table II, we review the sample antecedents (firing strengths) and consequents. The top 3 most influential rules are 2, 8 and 10, with firing strengths of 0.78, 0.94 and 0.55, respectively. While rules 2 and 10 have very small consequent values, rule 8 is the rule that dominates the final decision. With the highest consequent value of 6.91, the agent decides to go left. From Equation 4, the final best action is LEFT (value of 1.628). In other words, the agent believes the middle-bottom pattern is similar to rule 8, and combined with the consequent values, the agent decides to go left.

Other action explanations are shown in Figs. 5, 6, and 7 with Table III, IV, and V.

Specifically, in Fig. 5, we can see that when the agent kicks the ball toward the top right, it instantly starts to move right by checking the brick pattern along the trajectory of the ball.

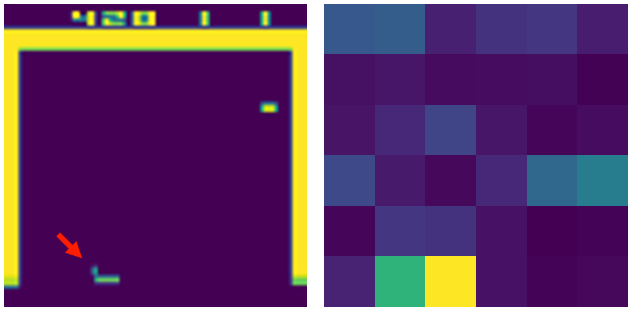
(a) observation (b) attention

Fig. 5: Example of RIGHT



(a) observation (b) attention

Fig. 6: Example of FIRE



(a) observation (b) attention

Fig. 7: Example of NOOP

Although the top 3 fired rules are 2, 3 and 7, their preferred actions differ. The final decision is produced primarily by the consequent values of rules 2 and 10. For the FIRE action in Fig. 7, the top 3 fired rules are 2, 3 and 8, while rule 8 again dominates the decision by matching the relative positions of the paddle and ball. Fig. 7 tell us that the agent stands still because of rules 3 and 4 (the consequent values of the most-fired rule (rule 2) are small). While rule 3 captures the

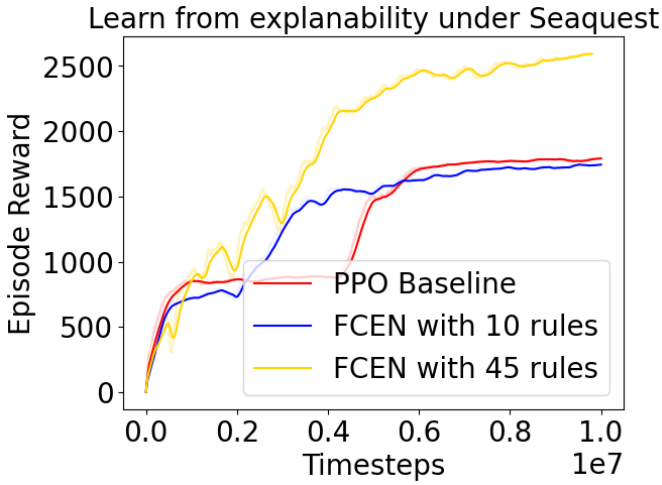


Fig. 8: Learning capacity demonstration under the "Seaquest" environment. The 10-rule agent can only maintain a similar performance to that of the PPO baseline, while the improved 45-rule agent boosts the learned reward to a much higher level.

TABLE II: This table lists the values inside the FDN when the agent takes the observation of Fig. 4(a) as input. The second row shows the firing strength values of rules, while the following 4 rows show the consequent values of the 4 actions: NOOP, FIRE, LEFT and RIGHT. The final output of the FDN is the combination of these values produced according to Equation 4.

Values	rule 1	rule 2	rule 3	rule 4	rule 5	rule 6	rule 7	rule 8	rule 9	rule 10	action
FS	0.32	**0.78**	0.7	0.38	0.11	0.34	0.19	**0.94**	0.33	**0.55**	
Conq NOOP	0	0	0	4.03	0	7.414	0	0	0	0	0.877
Conq FIRE	0	0	0	2.34	0	7.11	4.32	2.23	0.88	0	1.406
Conq RIGHT	0.06	1.16	0.07	0	0.3	0	0	0	0	0.24	0.246
Conq LEFT	0	0	0	0.33	2.74	1.76	0	**6.91**	0	0.07	**1.628**

TABLE III: This table lists the values inside the FDN when the agent takes the observation of Fig. 5(a) as input. The second row shows the firing strength values of rules, while the following 4 rows show the consequent values of the 4 actions: NOOP, FIRE, LEFT and RIGHT. The final output of the FDN is the combination of these values produced according to Equation 4.

Values	rule 1	rule 2	rule 3	rule 4	rule 5	rule 6	rule 7	rule 8	rule 9	rule 10	action
FS	0.18	**0.87**	**0.85**	0.24	0.15	0.2	**0.7**	0.62	0.42	0.46	
Conq NOOP	0	0.31	7.17	0	0	11.46	0	0	0	0	0.909
Conq FIRE	0	0	0	0	0	9.99	4.67	0	0	0	1.124
Conq RIGHT	9.24	**6.05**	0.23	0	8.8	0	0	0	0	**8.24**	**2.614**
Conq LEFT	0	0	0	4.81	0	0	3.48	0	0.57	0	0.81

TABLE IV: This table lists the values inside the FDN when the agent takes the observation of Fig. 6(a) as input. The second row shows the firing strength values of rules, while the following 4 rows show the consequent values of the 4 actions: NOOP, FIRE, LEFT and RIGHT. The final output of the produced is the combination of these values produced according to Equation 4.

Values	rule 1	rule 2	rule 3	rule 4	rule 5	rule 6	rule 7	rule 8	rule 9	rule 10	action
FS	0.29	**0.82**	**0.73**	0.4	0.11	0.24	0.3	**0.93**	0.35	0.44	
Conq NOOP	0	0		0	2.89	0	5.25	0	0	0	0.524
Conq FIRE	0.09	0	0	0	0	2.34	0	**3.52**	1.83	0	**0.979**
Conq RIGHT	0	0.15	1.44	0	0	0	0	0	0	0	0.255
Conq LEFT	0	0	0	3.19	1.1	1.34	1.64	1.74	1.09	0.44	0.957

TABLE V: This table lists the values inside the FDN when the agent takes the observation of Fig. 7(a) as input. The second row shows the firing strength values of rules, while the following 4 rows show the consequent values of the 4 actions: NOOP, FIRE, LEFT and RIGHT. The final output of the FDN is the combination of these values produced according to Equation 4.

Values	rule 1	rule 2	rule 3	rule 4	rule 5	rule 6	rule 7	rule 8	rule 9	rule 10	action
FS	0.35	**0.84**	**0.71**	**0.74**	0.3	0.45	0.14	0.33	0.15	0.68	
Conq NOOP	0	0	**5.1**	4.21	0	5.14	3.78	0	3.47	0	**2.154**
Conq FIRE	0	0	0	5.08	0	8.12	5.95	0	0	2.44	2.113
Conq RIGHT	5.63	2.83	0	0	6.45	0	0	0	0	5.03	2.071
Conq LEFT	0	0.68	1.43	1.53	0	0	2.1	2.99	0.26	1.79	1.122

TABLE VI: Local and global STD comparison between the 10-rule agentt and 45-rule agen under the "Seaquest" environment.

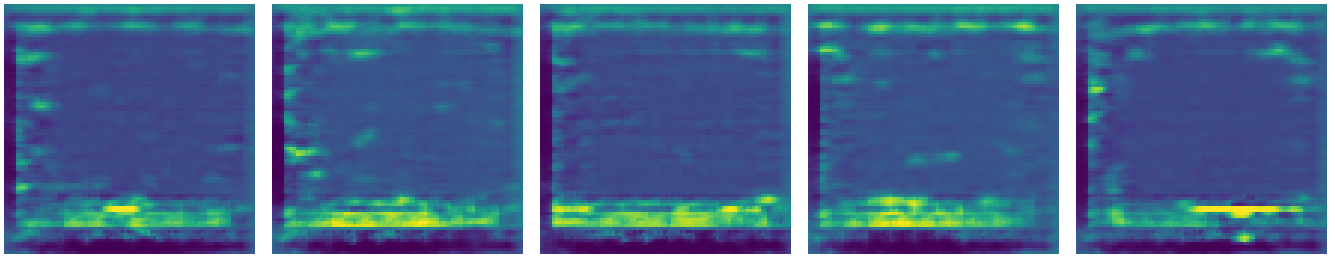
Number of rules	Pattern STD	Rule output STD(FIRE)	Rule output STD(DOWN)	Rule output STD (UPRIGHTFIRE)
10 rules	7.2891	0.864024	0.401872	0.471998
45 rules	8.5273	1.591391	2.470859	4.366544

position of standing pat, rule 4 matches the end game scenario, when few bricks are left on the screen. This is one potential local optimum we identify in the late stage of the game; when the agent almost clears every brick, it finds itself a special position where it can catch the ball by just standing still. The advantage of our explanation approach is that by understanding the decision rules of the agent, we can impose restrictions on the rules and prevent such local optima afterward.

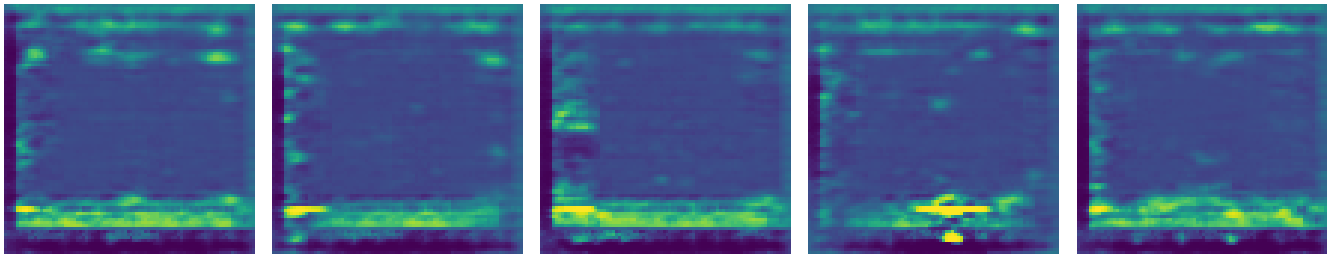
C. Learning from Explainability

We demonstrate the ability of the agent to learn from explainability by conducting experiments under the "Seaquest" environment, which is much more complicated than the "Breakout" environment. A player is asked to retrieve divers, dodging and blasting enemy subs and killer sharks. The player must monitor their oxygen tank in case it becomes emptied. The action space of "Seaquest" is also much larger than that of "Breakout", it contains 18 possible actions that cover all combinations of firing and moving directions. Here, we show a rule pattern comparison between an agent with 10 rules and an agent with 45 rules. In Fig. 9, the number of patterns limits the expression of the observation space. However, in Fig. 10, more diverse observation patterns are identified; thus, the performance of the agent is improved.

Here, we provide an example of improving the agent by understanding its decision rules. By reviewing the global explanation and local explanations of a trained agent under the "Seaquest" environment, we identify that the rules cannot

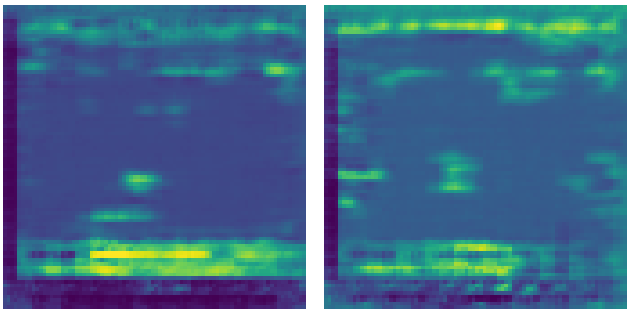


(a) Rule 1: DOWNLEFT (b) Rule 2: UPLEFTFIRE (c) Rule 3: DOWNFIRE (d) Rule 4: UPRIGHT (e) Rule 5: DOWNLEFT-FIRE



(f) Rule 6: UPFIRE (g) Rule 7: DOWNLEFT (h) Rule 8: UPLEFTFIRE (i) Rule 9: DOWNRIGHT (j) Rule 10: UPRIGHT

Fig. 9: Rule pattern reconstruction of the agent with only 10 rules.



(a) Rule 10: UPRIGHTFIRE (b) Rule 12: UPRIGHT

Fig. 10: Rule pattern reconstruction of the agent with only 45 rules. This figure shows only rules 10 and 12 for saving space, while the patterns for all rules can be found in the supporting document.

cover enough possibilities in the observation space. Fig. 9 lists the patterns identified by an agent with 10 rules: by reviewing the patterns, we find that most of the rules lean patterns in which objects are observed on the left side of the screen. However, the "Seaquest" environment contains several types of objects that affect the agent's decision, while it holds 18 possible actions for the agent to choose from. Therefore, an insufficient rule representation requires more information to be grasped in the model. In addition, by diving into the interstates of the FNN for some specific sample cases, the issue of low action value diversity is observed (Table VI). Based on such findings, we modify the agent model by increasing the number of rules to 45, yielding a significant improvement. Fig. 10 reveals that with 45 rules, the agent is able to identify more types of patterns. These include rules 10 and 12, where the submarine is placed at the center with potential enemies at the top right. Such patterns are not witnessed in Fig. 9, where only 10 rules are designed to cover the observation space. By reviewing Table VI, the diversity of the actions is also increased, and the STD of the patterns and actions are both increased significantly. From the perspective of reward learning, Fig.8 shows that the capability of the agent under 10 rules is similar to the PPO baseline, which is tremendously outperformed by the 45 rules' agent.

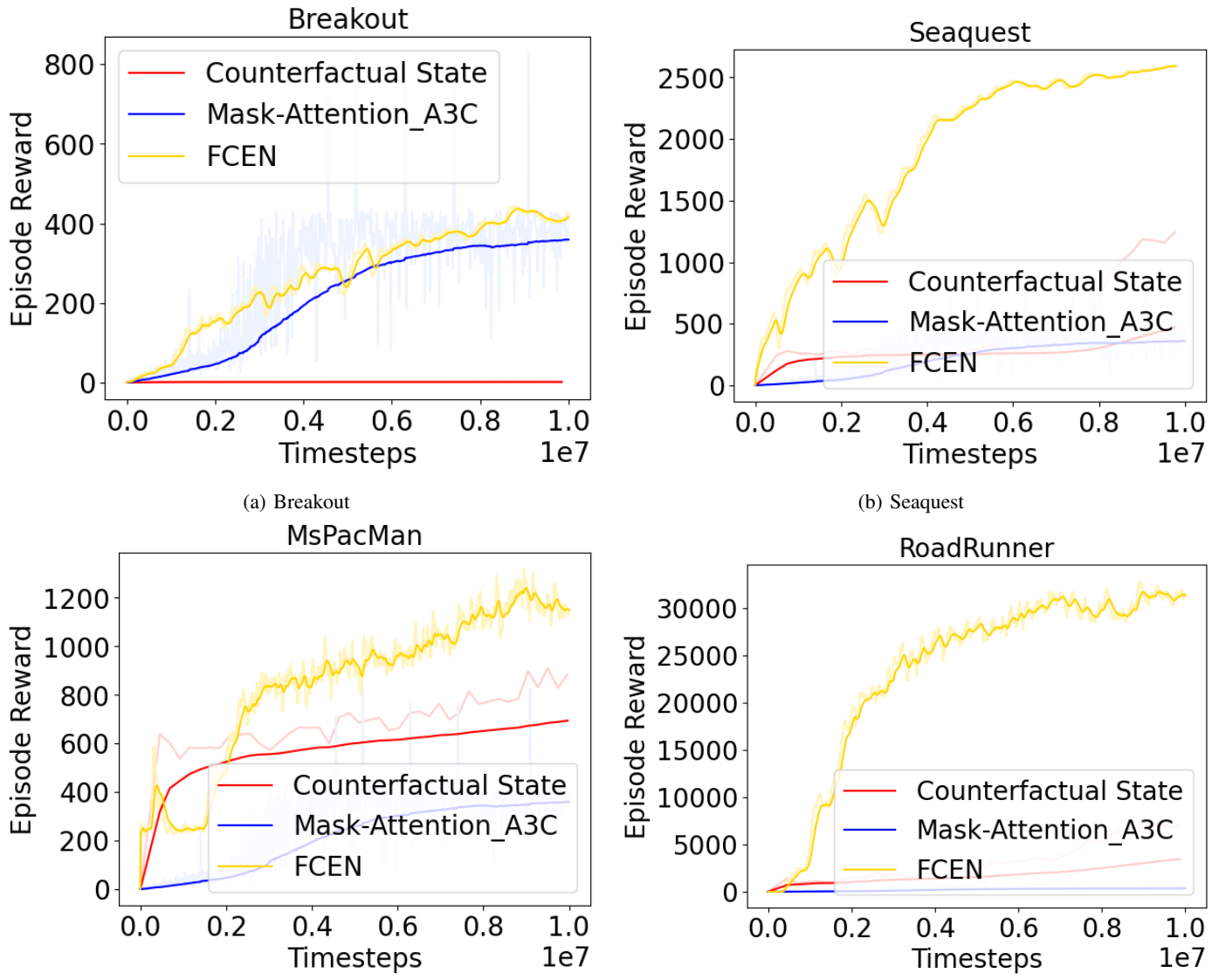
In the above discussion, we demonstrate the importance of explainability and the advantage of the proposed FCEN. Our design connects explainability with agent performance rather than balancing them as a zero-sum game. The results of the pattern comparisons and performance evaluations involving other Atari games are included in the appendices.

D. Performance Comparison

In this part, we lists the comparison results obtained by the proposed FCEN and some other xRL methods. We choose mask-attention A3C [8] and counterfactual state explanation [9] for comparison purposes. The reasons for these choices are as follows:

1) They are intrinsic methods.
2) They are recently proposed methods.
3) They can work with image-based Atari games. Note that many xRL methods work only in specifically defined environments.

This paper runs all methods under 4 Atari environments, "Breakout", "Seaquest", "Roadrunner" and "Ms. Pac-Man". The "Roadrunner" environment is as complicated as "Seaquest" with 18 possible actions included, while the "Ms. Pac-Man" environment is simple with 9 possible actions. All 4 environments offer image-like observations and a discrete action space. For training, the number of learning steps is set to 10 million for all environments. The results demonstrate that the proposed FCEN can achieve the highest reward.



(a) Breakout

(b) Seaquest

(c) Ms. Pac-Man

(d) Roadrunner

Fig. 11: Learning capability evaluations. This paper evaluates the FCEN and two intrinsic methods: mask-attention A3C and counterfactual state explanation.

V. Conclusion

In this paper, we introduce an FCEN, a fuzzy logic-centered explainable model for RL tasks. The proposed method leverages the attention mechanism of a CCT for both training and explanation. The FCEN is constructed with a set of fuzzy rules, which can be explained with fuzzy logic and a pattern decoder. The FCEN is intrinsically explainable, with global explanations embedded in the fuzzy sets and local explanations achieved through fuzzy if-then rules. Our experiment shows that the FCEN can provide both high-level and low-level explanations of agent decisions. Potentially, by understanding agent logic, human reviewers can manipulate the fuzzy rules to improve the agent's performance. This may be one of the most interesting results of this paper. In addition, the experimental results also show that our FCEN has good reward learning performance for RL in comparison with other intrinsic explainable methods. The scalability of the FCEN is tested in several image-based environments. In complex games, reconstruction quality becomes the key factor regarding the explanation of fuzzy sets. Therefore, a possible extension of this work is to improve the reconstruction quality of the fuzzy sets. Since this approach is transparent to humans, it is possible to let humans interact with the learning process of the agent by amending the learned fuzzy rules.

Acknowledgment

This work was supported in part by the Australian Research Council (ARC) under discovery grant DP210101093 and DP220100803, and the UTS Human-Centric AI Centre funding sponsored by GrapheneX (2023-2031). Research was also sponsored in part by the Australia Defence Innovation Hub under Contract No. P18-650825, Australian

Cooperative Research Centres Projects (CRC-P) Round 11 CRCPXI000007, US Office of Naval Research Global under Cooperative Agreement Number ONRG - NICOP - N62909-19-1-2058, and AFOSR – DST Australian Autonomy Initiative agreement ID10134. We also thank the NSW Defence Innovation Network and NSW State Government of Australia for financial support in part of this research through grant DINPP2019 S1-03/09 and PP21-22.03.02.

REFERENCES

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[2] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital signal processing*, vol. 73, pp. 1–15, 2018.

[3] A. Adadi and M. Berrada, "Peeking inside the black-box: a survey on explainable artificial intelligence (xai)," *IEEE access*, vol. 6, pp. 52 138–52 160, 2018.

[4] E. Puiutta and E. Veith, "Explainable reinforcement learning: A survey," in *International cross-domain conference for machine learning and knowledge extraction*. Springer, 2020, pp. 77–95.

[5] S. Rahimi Gorji, O.-C. Granmo, and M. Wiering, "Explainable reinforcement learning with the tsetlin machine," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 2021, pp. 173–187.

[6] W. Qiu, X. Wang, R. Yu, R. Wang, X. He, B. An, S. Obraztsova, and Z. Rabinovich, "Rmix: Learning risk-sensitive policies for cooperative reinforcement learning agents," *Advances in Neural Information Processing Systems*, vol. 34, pp. 23 049–23 062, 2021.

[7] M. Chen, L. Wan, C. Gou, J. Liao, and S. Wu, "Batch-constraint inverse reinforcement learning," in *Pacific Rim International Conference on Artificial Intelligence*. Springer, 2021, pp. 72–82.

[8] H. Itaya, T. Hirakawa, T. Yamashita, H. Fujiyoshi, and K. Sugiura, "Visual explanation using attention mechanism in actor-critic-based deep reinforcement learning," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–10.

[9] M. L. Olson, L. Neal, F. Li, and W.-K. Wong, "Counterfactual states for atari agents via generative deep learning," *arXiv preprint arXiv:1909.12969*, 2019.

[10] E. Lughofer, "Evolving fuzzy and neuro-fuzzy systems: Fundamentals, stability, explainability, useability, and applications," in *HANDBOOK ON COMPUTER LEARNING AND INTELLIGENCE: Volume 2: Deep Learning, Intelligent Control and Evolutionary Computation*. World Scientific, 2022, pp. 133–234.

[11] C. Mencar and J. M. Alonso, "Paving the way to explainable artificial intelligence with fuzzy modeling," in *International Workshop on Fuzzy Logic and Applications*. Springer, 2018, pp. 215–227.

[12] D. Hein, A. Hentschel, T. Runkler, and S. Udluft, "Particle swarm optimization for generating interpretable fuzzy reinforcement learning policies," *Engineering Applications of Artificial Intelligence*, vol. 65, pp. 87–98, 2017.

[13] A. Mortazavi, "Large-scale structural optimization using a fuzzy reinforced swarm intelligence algorithm," *Advances in Engineering Software*, vol. 142, p. 102790, 2020.

[14] Q. Yang, S.-J. Jang, and S.-J. Yoo, "Q-learning-based fuzzy logic for multi-objective routing algorithm in flying ad hoc networks," *Wireless Personal Communications*, vol. 113, no. 1, pp. 115–138, 2020.

[15] A. A. Khater, A. M. El-Nagar, M. El-Bardini, and N. M. El-Rabaie, "Online learning based on adaptive learning rate for a class of recurrent fuzzy neural network," *Neural Computing and Applications*, vol. 32, no. 12, pp. 8691–8710, 2020.

[16] P. Patompak, S. Jeong, I. Nilkhamhang, and N. Y. Chong, "Learning proxemics for personalized human–robot social interaction," *International Journal of Social Robotics*, vol. 12, no. 1, pp. 267–280, 2020.

[17] H. Yang and X. Xie, "An actor-critic deep reinforcement learning approach for transmission scheduling in cognitive internet of things systems," *IEEE Systems Journal*, vol. 14, no. 1, pp. 51–60, 2019.

[18] E. Soares, P. P. Angelov, B. Costa, M. P. G. Castro, S. Nageshrao, and D. Filev, "Explaining deep learning models through rule-based approximation and visualization," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 8, pp. 2399–2407, 2020.

[19] S.-C. Chu, Z.-G. Du, Y.-J. Peng, and J.-S. Pan, "Fuzzy hierarchical surrogate assists probabilistic particle swarm optimization for expensive high dimensional problem," *Knowledge-Based Systems*, vol. 220, p. 106939, 2021.

[20] P. V. de Campos Souza, E. Lughofer, and H. Rodrigues Batista, "An explainable evolving fuzzy neural network to predict the k barriers for intrusion detection using a wireless sensor network," *Sensors*, vol. 22, no. 14, p. 5446, 2022.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[22] A. Manchin, E. Abbasnejad, and A. v. d. Hengel, "Reinforcement learning with attention that works: A self-supervised approach," in *International conference on neural information processing*. Springer, 2019, pp. 223–230.

[23] F. Gao, W. Wang, M. Tan, L. Zhu, Y. Zhang, E. Fessler, L. Vermeulen, and X. Wang, "Deepcc: a novel deep learning-based framework for cancer molecular subtype classification," *Oncogenesis*, vol. 8, no. 9, pp. 1–12, 2019.

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

[25] K. Esslinger, R. Platt, and C. Amato, "Deep transformer q-networks for partially observable reinforcement learning," *arXiv preprint arXiv:2206.01078*, 2022.

[26] J. Sun, L. Yu, P. Dong, B. Lu, and B. Zhou, "Adversarial inverse reinforcement learning with self-attention dynamics model," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1880–1886, 2021.

[27] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, "Escaping the big data paradigm with compact transformers," *arXiv preprint arXiv:2104.05704*, 2021.

[28] A. Hayat, U. Singh, and V. P. Namboodiri, "Inforl: Interpretable reinforcement learning using information maximization," *arXiv preprint arXiv:1905.10404*, 2019.

[29] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," *Advances in neural information processing systems*, vol. 29, 2016.

[30] J. Co-Reyes, Y. Liu, A. Gupta, B. Eysenbach, P. Abbeel, and S. Levine, "Self-consistent trajectory autoencoder: Hierarchical reinforcement learning with trajectory embeddings," in *International conference on machine learning*. PMLR, 2018, pp. 1009–1018.

[31] 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.

[32] A. Sieusahai and M. Guzdial, "Explaining deep reinforcement learning agents in the atari domain through a surrogate model," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 17, no. 1, 2021, pp. 82–90.

[33] N. Puri, S. Verma, P. Gupta, D. Kayastha, S. Deshmukh, B. Krishnamurthy, and S. Singh, "Explain your move: Understanding agent actions using specific and relevant feature attribution," *arXiv preprint arXiv:1912.12191*, 2019.

[34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[36] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[37] H. Lakkaraju, S. H. Bach, and J. Leskovec, "Interpretable decision sets: A joint framework for description and prediction," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1675–1684.