Explanation Augmented Feedback in Human-in-the-Loop Reinforcement Learning

Lin Guan *
CIDSE
Arizona State University
Tempe, AZ 85281
lguan9@asu.edu

Mudit Verma *
CIDSE
Arizona State University
Tempe, AZ 85281
mverma13@asu.edu

Sihang Guo
Department of Computer Science
The University of Texas at Austin
Austin, TX 78712
sguo19@utexas.edu

Ruohan Zhang
Department of Computer Science
The University of Texas at Austin
Austin, TX 78712
zharu@utexas.edu

Subbarao Kambhampati
CIDSE
Arizona State University
Tempe, AZ 85281
rao@asu.edu

Abstract

Human-in-the-loop Reinforcement Learning (HRL) aims to integrate human guidance with Reinforcement Learning (RL) algorithms to improve sample efficiency and performance. A common type of human guidance in HRL is binary evaluative "good" or "bad" feedback for queried states and actions. However, this type of learning scheme suffers from the problems of weak supervision and poor efficiency in leveraging human feedback. To address this, we present EXPAND (EXPlanation AugmeNted feeDback) which provides a visual explanation in the form of saliency maps from humans in addition to the binary feedback. EXPAND employs a state perturbation approach based on salient information in the state to augment the binary feedback. We choose five tasks, namely *Pixel-Taxi* and four *Atari* games, to evaluate this approach. We demonstrate the effectiveness of our method using two metrics: environment sample efficiency and human feedback sample efficiency. We show that our method significantly outperforms previous methods. We also analyze the results qualitatively by visualizing the agent's attention. Finally, we present an ablation study to confirm our hypothesis that augmenting binary feedback with state salient information results in a boost in performance.

1 Introduction

Deep Reinforcement Learning (DRL) algorithms have achieved many successes in solving problems with high-dimensional state and action spaces [29, 3]. However, the current state-of-the-art DRL is yet to outperform human experts in many challenging problems. One factor that limits DRL's performance in these problems is its sample (in)efficiency. It is often impractical to collect millions of training samples as required by standard DRL algorithms. One way to curb this problem is to leverage additional human guidance by following the general paradigm of Human-in-the-Loop Reinforcement Learning (HRL) [53]. This often allows the agent to achieve better performance and higher sample efficiency.

One popular form of human guidance in HRL is the binary evaluative signal on actions [18], in which humans provide a "good" or "bad" judgment for an action. This framework allows non-expert humans to provide feedback, but its sample efficiency could be further improved by asking humans to provide stronger guidance. For example, the binary feedback does not tell the agent why it made

Presented at Human And Machine in-the-Loop Evaluation and Learning Strategies Workshop, NeurIPS 2020.

a mistake. If humans can explain the “why" behind the evaluative feedback, then it is possible to leverage the explanation to further help the agent. Taking training an autonomous driving agent as a motivating example, humans can point out to the agent that the “STOP" sign is an essential signal for the right action “apply-brake." One way to convey this information is through natural language, but language comprehension is a difficult problem itself. For this type of visuomotor decision tasks, a more straightforward form of human explanation could be *saliency information*, in which humans highlight the important (salient) regions of the visual environment state. The saliency information will indicate which visual features matter the most for the decision in the current state.

Saliency maps are shown to be a powerful means of assessing the agent's internal representations in the (visual) explainable RL research [45, 50, 40, 11, 30, 35, 13]. In this work, we extend this idea to use a saliency map as a communication channel between RL agents and humans. Hence human trainers can inform the RL agents about which regions they should focus on to accomplish the given task. Note that requiring human trainers to provide explanations on their evaluations does not necessarily require them to be more adept than in the case of providing only binary feedback. Like prior approaches utilizing binary evaluations [18], we assume the human trainers have a high-level understanding of the task. In our driving agent example, the human trainers may not know things like the optimal angle of the steering wheel; however, we can expect them to be able to tell whether an observed action is good or bad, and what visual objects matter for that decision.

In this work, we present EXPAND: EXPlanation AugmeNted feeDback (the overall flow can be found at Appendix Fig. 4). EXPAND augments the conventional binary feedback with human explanations regarding features in the visual state that are relevant to the decision. We show that EXPAND agents require fewer interactions with the environment (*environment sample efficiency*) and fewer human signals (*human feedback sample efficiency*) by leveraging saliency information. To efficiently learn with saliency information, we propose a novel method that applies multiple perturbations to irrelevant regions, thereby supplementing each saliency feedback with a set of constructed perturbed states. The idea is to differentiate between relevant and irrelevant regions of the state when updating the agent's value functions: perturbations to the irrelevant regions should not significantly change the value function because they do not matter for the current decision. Different kinds of perturbation in the irrelevant regions can essentially provide more feedback samples and hence reduce the feedback sample complexity. To our knowledge, this is the first work that incorporates human explanatory information as saliency maps in the setting of learning from human evaluative feedback.

2 Related Work

Leveraging human guidance for RL has been extensively studied in the context of imitation learning [36, 15], learning from demonstration [37, 14], inverse reinforcement learning [32, 1], reward shaping [31], learning from human preference [8, 16], and learning from saliency information provided by human [51]. Surveys on these topics are provided by Zhang et al. [53] and Wirth et al. [47].

Compared to these approaches, learning from human evaluative feedback has the advantage of placing minimum demand on both the human's expertise and the trainer's ability to provide guidance (e.g. the requirements of complex and expensive equipment setup). Representative works include the TAMER framework [18, 46], and the COACH framework [28, 4]. The TAMER+RL framework extends TAMER by learning from both human evaluative feedback and environment reward signal [19, 20]. DQN-TAMER further augments TAMER+RL by utilizing deep neural networks to learn in high dimensional state space [2]. Several approaches have been proposed to increase the information gathered from human feedback, which takes into account the complexities in human feedback-providing behavior. Loftin et al. speed up learning by adapting to different feedback-providing strategies [25, 26]; the Advice framework [12, 7] treats human feedback as direct policy labels and uses a probabilistic model to learn from inconsistent feedback. Other works also consider the action execution speed [33], the confidence in predicting human feedback [48], and different levels of satisfaction indicated by human voice [43]. Although these approaches better utilize human feedback with improved modeling of human behaviors, weak supervision is still a fundamental problem in the evaluative feedback approach.

Human explanatory information has also been explored previously. The main challenge of using human explanation is to translate human's high-level linguistic representations to agent-understandable representation. As an early attempt, Thomaz et al. allow humans to give anticipatory guidance

rewards and point out the object tied to the reward [44]. However, they assume the availability of an object-oriented representation of the state. Krening et al., Yeh et al. resort to human explanatory advice in natural language [22, 49]. Still, they assume a recognition model that can understand concepts in human explanation and recognize objects. In this work, we bridge the vocabulary gap between the humans and the agent by taking human explanations in the form of saliency maps. Other works use human gaze data as human saliency information, collected with sophisticated eye trackers, to help agents with decision making in an imitation learning setting [52, 54, 17]. However, we refrain from comparing with these works since they involve humans in an offline manner. In contrast, this work is closer to Arakawa et al., Xiao et al., where human trainers are required to be more actively involved during training [2, 48]. Finally, the way we exploit human explanation via state perturbation can be viewed as a novel way of data augmentation. Data augmentation techniques are widely used in computer vision tasks [24, 39] and have recently been applied to deep RL tasks [21, 23].

3 Problem Setup

We intend to verify the hypothesis that the use of state saliency information and binary evaluative feedback can boost an RL agent's performance. We have an agent M that interacts with an environment $\mathcal{E}$ through a sequence of actions, states, and rewards. Following standard practice in deep RL, k ($k = 4$) preprocessed consecutive image observations are stacked together to form a state $s_t = [x_{t-(k-1)}, ..., x_{t-1}, x_t]$. At each time-step t, the agent can select an action from a set of all possible actions $\mathcal{A} = \{1, ..., K\}$ for the state $s_t \in \mathcal{S}$. Then the agent receives a reward $r_t \in \mathcal{R}$ from the environment $\mathcal{E}$. This sequence of interaction ends when the agent achieves its goal or when the time-step limit is reached. This formulation follows the standard Markov Decision Process framework in RL research [42]. The agent's goal is to learn a policy function π, a mapping from a state to action, that maximizes the expected return.

Additionally, we assume a human trainer who provides binary evaluative feedback $\mathcal{H} = (h_1, h_2, ...h_n)$ that conveys their assessment of the queried state-action pairs given the agent trajectory. We define the feedback as $h_t = (x_t^h, b_t^h, x_t, a_t, s_t)$, where $b_t^h \in \{-1, 1\}$ is a binary "bad" or "good" feedback provided for action a_t and $x_t^h = \{Box_1, ..., Box_m\}$ is a saliency map for the image x_t in state s_t. Box_i is a tuple (x, y, w, h) for the top left Euclidean coordinates x and y, the width w, and the height h of the bounding box annotated on the observation image x_t.

4 Method

In EXPAND, the agent learns simultaneously from both environment reward and human feedback. To learn from environment reward, we use an off-policy RL algorithm, the Deep Q-Networks (DQN) [29]. To learn from human feedback, we first interpret binary feedback as labels on the optimality of the performed actions, which is similar to the interpretation in previous works [12, 7]. Since we are using DQN as our policy network, modeling binary feedback as labels on action optimality allows us to directly link human feedback with the advantage value [5] of an action. The way we learn from human evaluative feedback will be discussed in detail in Section 4.3. To learn with explanation augmented feedback, we amplify the saliency feedback by perturbing irrelevant regions in the state and expect the Q-value approximator to be indifferent to these perturbations when computing the action values. The ideas manifest as various loss terms to update the DQN network. The train-interaction loop of EXPAND can be found in Appendix B. Within an episode, the agent interacts with the environment and stores its transition experiences. Every few episodes, it collects human feedback queried on a trajectory sampled from the most recent policy. The DQN weights are updated twice, with the usual DQN loss and then with the proposed feedback loss, in a single training step. This section covers the proposed loss terms and how we obtain perturbed states for these losses.

4.1 Learning with Environment Reward

Following DQN [29], a set of tricks is applied to stabilize training, such as experience replay, reward clipping, and soft-target network updates. The learned policy π is defined as the actions that maximize the Q-values. Since π is a deterministic policy here, according to the Bellman equation, the state value function can be defined as $V^\pi(s) = Q^\pi(s, \pi(s))$. To ensure sufficient exploration, we also use ϵ-greedy action selection mechanism, in which the probability ϵ of taking a random action is annealed

down episodically. The loss function in DQN that updates the Q-value function weights θ is:

$$L_{DQN} = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}[(r + \gamma \max_{a'} Q(s', a'; \bar{\theta}) - Q(s, a; \theta))^2] \tag{1}$$

where $\bar{\theta}$ is the target network parameters and $\mathcal{D}$ is the replay buffer that stores experience transition tuples (s, a, r, s') such that action a taken in state s brings immediate reward r and gets the agent to state s'.

4.2 Learning with Binary Evaluative Feedback

Similar to DQN's replay buffer, we maintain a feedback replay buffer $\mathcal{D}_h$, which stores the observed human feedback. Feedback signals in $\mathcal{D}_h$ are later sampled uniformly and are used to compute the feedback loss.

We use the advantage value to formulate our *advantage loss*. Advantage value is the difference between the Q-value of the action upon which the feedback was given and the Q-value of the current optimal action calculated by the neural network. Given the agent's current policy π, state s, and action a for which the feedback is given, the advantage value is defined as:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s) = Q^\pi(s, a) - Q^\pi(s, \pi(s)) \tag{2}$$

Hence, the advantage value quantifies the possible (dis)advantage the agent would have if some other action were chosen instead of the current-best. It can be viewed as the agent's judgment on the optimality of an action. Positive feedback means the human trainer expects the advantage value of the annotated action to be zero. Therefore, we define a loss function, i.e., the *advantage loss*, which forces the network to have the same judgment on the optimality of action as the human trainer. Intuitively, we penalize the policy-approximator when a marked "good" action is not chosen as the best action, or when a marked "bad" action is chosen as the best action.

For a feedback $h = (x^h, b^h, x, a, s)$, when the label is "good", i.e., $b^h = 1$, we expect the network to output a target value $\hat{A}(s, a) = 0$, so the loss can be defined as $|\hat{A}(s, a) - A^\pi(s, a)| = Q^\pi(s, \pi(s)) - Q^\pi(s, a)$. When the label is "bad", i.e., $b^h = -1$, we expect the network to output an advantage value $A^\pi(s, a) < 0$. Since here we do not have a specific target value for $A^\pi(s, a)$, we resort to the idea of large margin classification loss [34, 14], which forces $Q^\pi(s, a)$ to be at least a margin l_m lower than the Q-value of the second best action, i.e., $\max_{a' \neq a} Q^\pi(s, a')$. One advantage of such interpretation of human feedback is that it directly modifies the Q-values with the feedback information and does not require additional parameters to model human feedback.

Formally, for human feedback $h = (x^h, b^h, x, a, s)$ and the corresponding advantage value $A_{s,a} = A^\pi(s, a)$, the *advantage loss* is:

$$L_A(s, a, h) = L_A^{Good}(s, a, h) + L_A^{Bad}(s, a, h) \tag{3}$$

$$L_A^{Good}(s, a, h; b^h = 1) = \begin{cases} 0 & ; A_{s,a} = 0 \\ Q^\pi(s, \pi(s)) - Q^\pi(s, a) & ; \text{otherwise} \end{cases} \tag{4}$$

$$L_A^{Bad}(s, a, h; b^h = -1) = \begin{cases} 0 & ; A_{s,a} < 0 \\ Q^\pi(s, a) - (\max_{a' \neq a} Q^\pi(s, a') - l_m) & ; A_{s,a} = 0 \end{cases} \tag{5}$$

Note that our method is different from the COACH framework [28], another method that interprets human feedback in terms of the advantage function. In their formulation, the advantage value is exactly the advantage term in the policy gradient equation–human trainers are supposed to provide positive/negative policy-dependent feedback only when the agent performs an action better/worse than that in current policy. Thus, such a formula is restricted to on-policy policy-gradient methods. On the contrary, the advantage function in our formula aims to capture the relative utility of all the actions. Here human feedback is direct policy advice, like that in the Advice framework [12], indicating whether an action is preferable regardless of the agent's current policy. This property makes our formula a better fit for off-policy value-based methods like Q-learning.

4.3 Learning with Human Saliency Information

State saliency informs the agent about which parts of the state matter for achieving the goal. These "parts" of the state could be specific regions or objects. The intuition is that the agent must correctly identify the relevant visual features before it can make an optimal decision. Hence if the agent knows where the relevant regions are, it can figure out the optimal actions faster. In this work, each saliency map consists of a set of bounding boxes over the images, marking a region's importance in achieving the given task. We utilize this saliency information in a manner that directly affects the weights of the neural network.

It should be noted that the previous section only made use of the binary evaluations via the *advantage loss*. In this section, however, we utilize the saliency feedback along with the binary feedback. As a data augmentation technique, we first describe how multiple perturbations are applied to an image. Then we introduce the loss terms that EXPAND uses. We propose two additional loss terms, namely the *policy invariant loss* and the *value invariant loss* that make use of human explanation along with binary evaluation.

Data augmentation through perturbation: We here propose a novel data augmentation technique, in order to exploit extra information from the human saliency map to further improve sample efficiency. The key idea here is that, ideally, states with perturbation in irrelevant regions should have the same Q-values as the original states. Hence the manipulations to irrelevant regions should not affect the agent's policy. A standard way of performing such perturbation is to apply Gaussian perturbations over irrelevant regions [11], essentially blurring them out and motivating the agent to focus more on the clear relevant regions. Consider a feedback $h = (x^h, b^h, x, a, s)$. We need to convert state s into some states s_t^r with perturbations on relevant regions and some other states $\tilde{s}_t^r$ with perturbations on irrelevant regions. Let $\mathbb{M}(x, i, j)$ denote a mask over relevant regions, where $x(i, j)$ denotes the pixel at index (i, j) for image x. We then have:

$$\mathbb{M}(x, i, j) = \begin{cases} 1 & \text{if (i, j) lies in Box, } \exists \text{ Box} \in b^h \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$\phi(x, M, i, j) = x \odot (1 - \mathbb{M}(x, i, j)) + G(x, \sigma_G) \odot \mathbb{M}(x, i, j) \tag{7}$$

Then we can perturb pixel (i, j) in image observation x according to mask $\mathbb{M}$ using a function ϕ defined as above, where $\odot$ is the Hadamard Product and function $G(x, \sigma_G)$ is the Gaussian blur of the observation x. Hence, we can get an image with perturbed relevant regions (x^r) with mask $\mathbb{M}$ and perturbed irrelevant regions ($\tilde{x}^r$) with mask $\neg\mathbb{M}$.

Policy invariant loss: The intuition behind the *policy invariant loss* is that under a set of perturbations over irrelevant regions in human explanation, the action marked "good" is always good and the action marked "bad" is always bad. This means the agent's judgment on the optimality of action should not change under perturbations over irrelevant regions in the human saliency map. Thus, this loss is essentially the *advantage loss* calculated over states with perturbed irrelevant regions. As defined in the previous section, we use Gaussian perturbations of various filter sizes and variances (see Appendix D for detailed settings) to obtain a number of states with perturbed irrelevant regions. The g^{th} perturbation is denoted by $\tilde{s}^{rg}$ with the feedback $h = (x^h, b^h, x, a, s)$. Formally, if we have g number of perturbations for a single state s in feedback h, the *policy invariant loss* is defined as:

$$L_P = \frac{1}{g}(\sum_g L_A(\tilde{s}^{rg}, a, h)) \tag{8}$$

Value invariant loss: In previous definitions of the loss function, we utilized the difference in judgments on the optimality of actions between the agent and the humans. We also note that, for humans, these states with perturbations on irrelevant regions are not "seen" differently from the original state. Therefore, the Q-values of these perturbed states should be similar to those of the original state. Hence, we define a mean squared error loss term over the two Q-values as our *value invariant loss*:

$$L_V = \frac{1}{g}\sum_{i=1}^{g}\frac{1}{|\mathcal{A}|}\sum_{a\in\mathcal{A}}(Q^\pi(s, a) - Q^\pi(\tilde{s}^{ri}, a))^2 \tag{9}$$

where $\mathcal{A}$ is the action set and g is the number of perturbations.

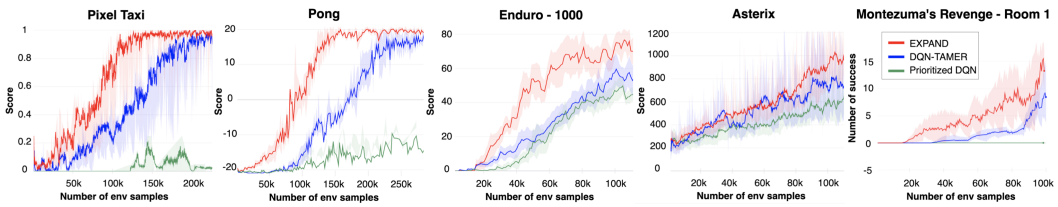

Figure 1: The learning curves of EXPAND and the baselines. The solid lines show the mean score over at least 3 random seeds. The shaded regions represent the standard error. In *Pixel Taxi*, the score is a running average over the last 20 rollouts. In *MR-1*, we counted the total number of times that the agent managed to solve the first room within 1000 steps. Our method (EXPAND, in red) outperforms the baselines in all the tasks.

Combining Feedback Losses: We linearly combine the three losses to obtain the overall feedback loss:

$$L_F = \lambda_A L_A + \lambda_P L_P + \lambda_V L_V \tag{10}$$

where λ_A, λ_P and λ_V are the weights of *advantage loss*, *policy invariant loss* and *value invariant loss* respectively. The agent is trained with L_{DQN} as well as L_F. As an extreme case, the value invariant loss can go down to zero when the policy network assigns an identical Q-value to actions in all the states, which will hurt the performance. To prevent this we give a smaller weight to value invariant loss: $\lambda_V = 0.1$, $\lambda_P = 1.0$, $\lambda_A = 1.0$. We note that these losses improve human feedback sample efficiency and environment sample efficiency in different ways: value invariant loss imposes a stricter latent space similarity, whereas policy invariant loss helps differentiate between different actions.

5 Experimental Evaluation

To evaluate EXPAND, we conducted experiments on five tasks: *Pixel Taxi* and four *Atari* games. This section introduces the tasks and different metrics we use to compare EXPAND with other baselines. Our baselines include Prioritized DQN [38] and an HRL algorithm DQN-TAMER [2] that simultaneously learns from environment reward and human binary feedback. We also compare to three ablated versions of EXPAND with one loss term at a time: EXPAND-Advantage (λ_A=1.0, λ_P=0, λ_V=0), EXPAND-Value-Invariant (λ_A=1.0, λ_P=0, λ_V=0.1) and EXPAND-Policy-Invariant (λ_A=1.0, λ_P=1.0, λ_V=0). The goal of experimenting with different variants of EXPAND is to determine the effect of each loss term on the learning performance. Hence, this serves as an ablation study.

In all our experiments, our algorithm and the baselines employ the same DQN network architecture adopted by [29], which has three convolutional layers following by one fully-connected layer. The same set of hyperparameters is used to train the models. Details on the architecture and hyperparameters can be found in Appendix A. We use multi-step returns and the prioritized experience replay mechanism [38] both in EXPAND and in the baselines. Following the standard preprocessing procedure, each frame is converted from RGB format to grayscale and is resized to 84×84. The input to the networks is normalized to the range of [0, 1]. During training, we start with an ϵ-greedy policy ($\epsilon = 1.0$) and reduce ϵ by a factor of λ_ϵ at the end of each episode until it reaches 0.01. The reported results are averaged over at least 3 random seeds.

Inspired by the Taxi domain that is widely used in RL research [10], we design a task named *Pixel Taxi* (see Appendix Fig. 4). This is a grid-world setup where the agent, as a taxi, occupies one grid cell at a time, and passengers denoted by different colored dots occupy separate grid cells in the world. Our taxi agent's goal is to pick up the correct passenger and reach the destination cell. For the agent to learn the target passenger instead of simply remembering the "locations", we randomize the passengers' positions at the beginning of each episode. A reward in this task is given only when the taxi drops off the correct passenger at the destination cell. In addition, we also choose four *Atari* games [6] with default settings: *Pong*, *Asterix*, *Enduro*, and *Montezuma's Revenge*. Original *Enduro* and *Montezuma's Revenge* can be infinitely long and can make human training impractical.

Therefore, in *Enduro*, our goal is to teach the agent to overtake as many cars as possible within 1000 environment steps (an episode); hence we denote this task as *Enduro-1000*. In *Montezuma's Revenge*, we train the agent to solve the first room within 1000 environment steps per episode; hence we denote this task as *MR-1*.

We evaluate our work on two fronts. The principal role of humans in the RL process is to improve the environment sample efficiency; hence this serves as our primary indicator of success. Also, one of the major claims of this work is that augmenting binary feedback with human explanation in the form of saliency maps can improve human feedback sample efficiency; hence this is our second metric.

5.1 Obtaining Feedback

The steps for obtaining human feedback are described as follows (see Appendix B): in every N_f ($N_f = 4$) episodes, we sample one trajectory and query the user for binary evaluative feedback as well as saliency information. Although actively querying humans could be preferable, the goal of this work is to demonstrate how human explanation can augment binary feedback. Following the setting of our baseline DQN-TAMER, we allow humans to "watch" the queried trajectories and provide feedback at will. We use explanation information from human trainers for *Asterix*, *Enduro*, and *Montezuma's Revenge*. For *Pixel Taxi* and *Pong*, we use a synthetic oracle, since oracles for these two tasks are relatively easy to construct.

In *Asterix*, *Enduro*, and *MR-1*, we evaluated EXPAND and DQN-TAMER with human trainers. The human interface we used to collect human feedback and explanation is similar to a video player (see Appendix G), with which the trainers can start/pause the replay of the queried trajectory and provide explanations by directly drawing bounding boxes on the screen. The trainers can also adjust the "video" frame rate based on their needs. Similar to Deep-TAMER [46], we also apply each received feedback and explanation to frames that are displayed between 2 and 0.2 seconds before the feedback occurred – we assume that within this 1.8-second window, the salient regions/objects should be the same. Thus, we can use an off-the-shelf object tracking method, like the CSRT tracker [27], to locate the highlighted regions/objects in previous frames. Our experiments show this temporal consistency assumption works well in practice. In addition to backward tracking, we also keep track of the highlighted regions/objects and perform object detection (e.g., automatically detect and annotate the cars ahead in *Enduro*) in succeeding frames. The tracking and detection tricks greatly reduce the number of times human trainers need to annotate the images, and in our experiments, we found that human trainers could provide their feedback and explanation with ease.

To perform a systematic analysis and get additional insights, we conduct a larger set of experiments that include 5 runs of all algorithms, the ablation study, and runs with different numbers of perturbations on *Pixel Taxi* and *Pong* with a synthetic oracle. A synthetic oracle allows us to run a large number of experiments and give consistent feedback across different runs, providing fair and systematic comparisons between different approaches [12, 2]. In *Atari Pong*, we use a trained model from the *Atari Zoo* [41]; in *Pixel Taxi*, we use a self-trained DQN model. To annotate "relevant" regions on the image observation, we use hard-coded models to highlight the taxi-cell, the destination-cell, and the target-passenger in *Taxi*, as well as the two paddles and the ball in *Atari-Pong*.

5.2 Results

Sample efficiency : Fig. 1 compares the environment sample efficiency as well as performance between EXPAND (in red), DQN-TAMER (in blue), and Prioritized DQN (in green) in all tasks. We restrict the human experiments–*Enduro-1000*, *Asterix*, and *Montezuma's Revenge MR-1*–to 100k environment samples, which corresponds to around 10k human feedback and approximately one-hour interaction time with a human trainer. In *Enduro-1000* and *Asterix*, EXPAND achieves a score that is over 30% higher than DQN-TAMER at the end of the training. In *MR-1*, which is a hard-exploration domain with sparse reward, we counted the number of times that the learning agent solved *MR-1*. Without sophisticated exploration strategies, EXPAND managed to solve *MR-1* 15 times on average, which is twice as many as DQN-TAMER. As expected, Prioritized DQN could not solve the task even a single time within the 100k env. steps for *MR-1* and was worse in all other tasks. This result demonstrates that EXPAND performs better than our baselines with limited environment samples.

In *Pixel Taxi* and *Pong*, we do not restrict the number of feedbacks from the oracle. EXPAND learns a near-optimal policy in just 130k and 140k environment samples, whereas DQN-TAMER takes

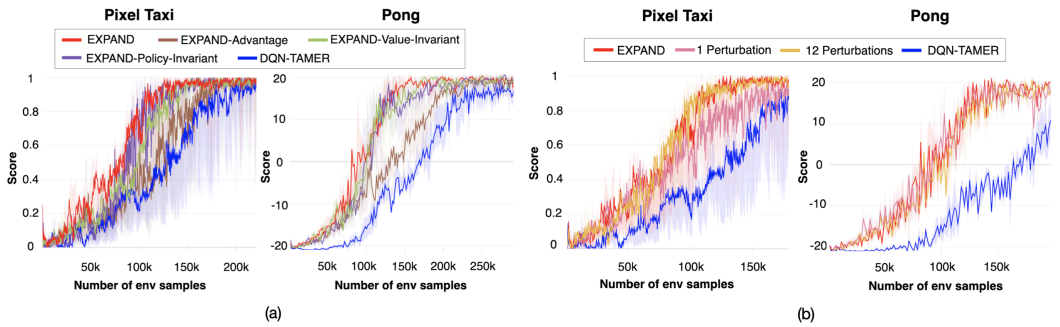

Figure 2: (a) Ablation experiments analyzing the effects of each individual loss terms on the performance. The results verify that the combination of all loss terms leads to the best performance. (b) Learning curves of the variants of EXPAND with different number of perturbations on *Pixel Taxi* (left) and *Pong* (right). EXPAND (in red) uses 5 perturbations.

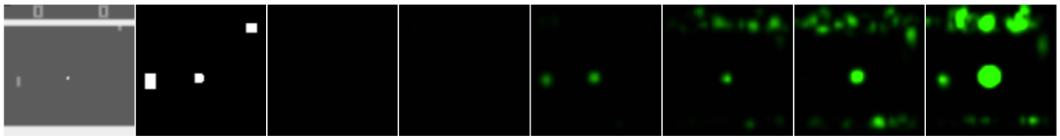

Figure 3: EXPAND agents gradually learned to attend to regions indicated by the human during training. From left to right: game frame (Pong), human saliency map, and the agent's attention maps [11] at 5k, 10k, 20k, 40k, 80k, and 300k training samples. Examples for other four tasks can be found in Appendix F.

over 200k and 250k samples to reach a similar performance. This is an over 35% improvement in environment sample efficiency. Regarding human feedback efficiency, in *Pixel Taxi* and *Pong*, EXPAND requires about 30k and 35k binary and saliency feedback pairs, respectively, from the oracle to learn a near-optimal policy. In contrast, DQN-TAMER requires more than 50k feedbacks in both tasks. As seen from our study, we note that the marginal cost of asking for saliency information is low (owing to off-the-shelf object detection and tracking), making EXPAND a promising method on the front of human feedback sample efficiency. Finally, the performance of Prioritized-DQN was poor in both tasks.

Ablation study : Fig. 2 shows that all four variants of EXPAND (EXPAND, and EXPAND with individual loss terms) eventually converged to near-optimal policies. We note that EXPAND-Policy-Invariant (in purple) and EXPAND-Value-Invariant (in light green) perform significantly better than EXPAND-Advantage, highlighting that each saliency loss alone provides significant improvements over the baseline. Finally, combining these losses (EXPAND) boosts this performance even further, indicating that collecting saliency information in addition to binary evaluations is a better approach.

Varying number of perturbations : We experimented with the number of perturbations required in each state to get the best performance. Figure 2 shows a comparison when the number of perturbations is varied among $\{1, 5, 12\}$ for *Pixel Taxi* and *Pong* using a synthetic oracle. The plots suggest that increasing perturbations only evoke slight performance gains, and therefore setting the number of perturbations to 5 for EXPAND is apt.

Visualizing agent attention : To verify that the agents have successfully learned to pay attention to the critical regions highlighted by human saliency maps, we visualize the attention maps of trained EXPAND agents with a method commonly used to provide visual interpretations of deep RL agents [11]. The algorithm takes an input image I and applies a Gaussian filter to a pixel location (i, j) to blur the image. This manipulation adds spatial uncertainty to the surrounding region and produces a perturbed image $\Phi(I, i, j)$. A saliency score for this pixel (i, j) can be defined as how much the blurred image changes the policy π [11]. The results for Pong can be found in Fig. 3, where EXPAND learns to focus on the important objects (the ball and the opponent) as early as 20k environment steps. This result provides another insight on why EXPAND performs better as object representation is a critical cognitive capacity lacking from current deep RL agents [9].

6 Conclusion & Future Work

In this work, we presented a novel method to integrate human explanation–as saliency maps on image observations–with their binary evaluations of agents' actions in a Human-in-the-Loop RL paradigm. We show that our proposed method, EXPAND, outperforms previous methods in environment sample efficiency, and shows promising results for human feedback efficiency. We also see that supplementing human saliency feedback with perturbed irrelevant regions is helpful when multiple such perturbations are used. Our results show that leveraging information about relevant parts of the image, for the task at hand, indeed helps the agent perform better. We currently treat saliency feedback as human advice; however, the advice in the form of natural language interaction would be an improvement. Moreover, we note that we have restricted "perturbations" to be Gaussian blurs to the state image. Future work can experiment with different types of perturbations and even domain-dependent perturbations that involve object manipulation.

References

[1] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

[2] Riku Arakawa, Sosuke Kobayashi, Yuya Unno, Yuta Tsuboi, and Shin-ichi Maeda. Dqn-tamer: Human-in-the-loop reinforcement learning with intractable feedback. *arXiv preprint arXiv:1810.11748*, 2018.

[3] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[4] Dilip Arumugam, Jun Ki Lee, Sophie Saskin, and Michael L Littman. Deep reinforcement learning from policy-dependent human feedback. *arXiv preprint arXiv:1902.04257*, 2019.

[5] Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

[6] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[7] Thomas Cederborg, Ishaan Grover, Charles L Isbell, and Andrea L Thomaz. Policy shaping with human teachers. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[8] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pages 4299–4307, 2017.

[9] Guy Davidson and Brenden M Lake. Investigating simple object representations in model-free deep reinforcement learning. *arXiv preprint arXiv:2002.06703*, 2020.

[10] Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of artificial intelligence research*, 13:227–303, 2000.

[11] Sam Greydanus, Anurag Koul, Jonathan Dodge, and Alan Fern. Visualizing and understanding atari agents. *arXiv preprint arXiv:1711.00138*, 2017.

[12] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in neural information processing systems*, pages 2625–2633, 2013.

[13] Piyush Gupta, Nikaash Puri, Sukriti Verma, Sameer Singh, Dhruv Kayastha, Shripad Deshmukh, and Balaji Krishnamurthy. Explain your move: Understanding agent actions using focused feature saliency. *arXiv preprint arXiv:1912.12191*, 2019.

[14] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4565–4573. Curran Associates, Inc., 2016.

[16] Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari. In *Advances in neural information processing systems*, pages 8011–8023, 2018.

[17] Heecheol Kim, Yoshiyuki Ohmura, and Yasuo Kuniyoshi. Using human gaze to improve robustness against irrelevant objects in robot manipulation tasks. *IEEE Robotics and Automation Letters*, 2020.

[18] W Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, pages 9–16, 2009.

[19] W Bradley Knox and Peter Stone. Combining manual feedback with subsequent mdp reward signals for reinforcement learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 5–12. Citeseer, 2010.

[20] W Bradley Knox and Peter Stone. Reinforcement learning from simultaneous human and mdp reward. In *AAMAS*, pages 475–482, 2012.

[21] Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

[22] Samantha Krening, Brent Harrison, Karen M Feigh, Charles Lee Isbell, Mark Riedl, and Andrea Thomaz. Learning from explanations using sentiment and advice in rl. *IEEE Transactions on Cognitive and Developmental Systems*, 9(1):44–55, 2016.

[23] Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020.

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] Robert Loftin, Bei Peng, James MacGlashan, Michael L Littman, Matthew E Taylor, Jeff Huang, and David L Roberts. Learning something from nothing: Leveraging implicit human feedback strategies. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pages 607–612. IEEE, 2014.

[26] Robert Loftin, Bei Peng, James MacGlashan, Michael L Littman, Matthew E Taylor, Jeff Huang, and David L Roberts. Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning. *Autonomous agents and multi-agent systems*, 30(1):30–59, 2016.

[27] Alan Lukezic, Tomas Vojir, Luka Cehovin Zajc, Jiri Matas, and Matej Kristan. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6309–6318, 2017.

[28] James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2285–2294. JMLR. org, 2017.

[29] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[30] Alexander Mott, Daniel Zoran, Mike Chrzanowski, Daan Wierstra, and Danilo Jimenez Rezende. Towards interpretable reinforcement learning using attention augmented agents. In *Advances in Neural Information Processing Systems*, pages 12329–12338, 2019.

[31] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287, 1999.

[32] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[33] Bei Peng, James MacGlashan, Robert Loftin, Michael L Littman, David L Roberts, and Matthew E Taylor. A need for speed: Adapting agent action speed to improve task learning from non-expert humans. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*, 2016.

[34] Bilal Piot, Matthieu Geist, and Olivier Pietquin. Boosted bellman residual minimization handling expert demonstrations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 549–564. Springer, 2014.

[35] Nikaash Puri, Sukriti Verma, Piyush Gupta, Dhruv Kayastha, Shripad Deshmukh, Balaji Krishnamurthy, and Sameer Singh. Explain your move: Understanding agent actions using specific and relevant feature attribution. In *International Conference on Learning Representations*, 2019.

[36] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[37] Stefan Schaal. Learning from demonstration. In *Advances in neural information processing systems*, pages 1040–1046, 1997.

[38] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR 2016 : International Conference on Learning Representations 2016*, 2016.

[39] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

[40] Ivan Sorokin, Alexey Seleznev, Mikhail Pavlov, Aleksandr Fedorov, and Anastasiia Ignateva. Deep attention recurrent q-network. *arXiv preprint arXiv:1512.01693*, 2015.

[41] Felipe Petroski Such, Vashisht Madhavan, Rosanne Liu, Rui Wang, Pablo Samuel Castro, Yulun Li, Jiale Zhi, Ludwig Schubert, Marc G. Bellemare, Jeff Clune, and Joel Lehman. An atari model zoo for analyzing, visualizing, and comparing deep reinforcement learning agents. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 3260–3267, 2019.

[42] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[43] Ana C Tenorio-Gonzalez, Eduardo F Morales, and Luis Villaseñor-Pineda. Dynamic reward shaping: training a robot by voice. In *Ibero-American conference on artificial intelligence*, pages 483–492. Springer, 2010.

[44] Andrea Lockerd Thomaz, Cynthia Breazeal, et al. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Aaai*, volume 6, pages 1000–1005. Boston, MA, 2006.

[45] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.

[46] Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, and Peter Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[47] Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *The Journal of Machine Learning Research*, 18(1):4945–4990, 2017.

[48] Baicen Xiao, Qifan Lu, Bhaskar Ramasubramanian, Andrew Clark, Linda Bushnell, and Radha Poovendran. Fresh: Interactive reward shaping in high-dimensional state spaces using human feedback. *arXiv preprint arXiv:2001.06781*, 2020.

[49] Eric Yeh, Melinda Gervasio, Daniel Sanchez, Matthew Crossley, and Karen Myers. Bridging the gap: Converting human advice into imagined examples. *Advances in Cognitive Systems*, 6: 1168–1176, 2018.

[50] Tom Zahavy, Nir Ben-Zrihem, and Shie Mannor. Graying the black box: Understanding dqns. In *International Conference on Machine Learning*, pages 1899–1908, 2016.

[51] R Zhang, A Saran, B Liu, Y Zhu, S Guo, S Niekum, D Ballard, and M Hayhoe. Human gaze assisted artificial intelligence: A review. In *International Joint Conference on Artificial Intelligence*, 2020.

[52] Ruohan Zhang, Zhuode Liu, Luxin Zhang, Jake A Whritner, Karl S Muller, Mary M Hayhoe, and Dana H Ballard. Agil: Learning attention from human for visuomotor tasks. In *Proceedings of the european conference on computer vision (eccv)*, pages 663–679, 2018.

[53] Ruohan Zhang, Faraz Torabi, Lin Guan, Dana H. Ballard, and Peter Stone. Leveraging human guidance for deep reinforcement learning tasks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 6339–6346. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcaai.2019/884.

[54] Ruohan Zhang, Calen Walshe, Zhuode Liu, Lin Guan, Karl S Muller, Jake A Whritner, Luxin Zhang, Mary M Hayhoe, and Dana H Ballard. Atari-head: Atari human eye-tracking and demonstration dataset. *arXiv preprint arXiv:1903.06754*, 2019.

Appendix

A. Hyperparameters

- Convolutional channels per layer: [32, 64, 64]
- Convolutional kernel sizes per layer: [8, 4, 3]
- Convolutional strides per layer: [4, 2, 1]
- Convolutional padding per layer: [0, 0, 0]
- Fully connected layer hidden units: [512, number of actions]
- Update interval: 4
- Discount factor: 0.99
- Replay buffer size: 50,000
- Batch size: 64
- Feedback buffer size: 50,000 in *Atari* games, 10,000 in *Pixel Taxi* [1]
- Feedback batch size: 64 in *Atari* games, 32 in *Pixel Taxi*
- Learning Rate: 0.0001
- Optimizer: Adam
- Prioritized replay exponent $\alpha = 0.6$
- Prioritized replay importance sampling exponent $\beta = 0.4$
- *Advantage loss* margin $l_m = 0.05$
- Rewards: clip to [-1, 1]
- Multi-step returns: $n = 5$
- ϵ episodic decay factor λ_ϵ: 0.99 in *Pixel Taxi*, 0.9 in *Atari* games

B. Train - Interaction Loop

Algorithm 1 Train - Interaction Loop

Result: Trained DQN agent M
Input: DQN agent M with randomly-initialized weights θ, replay buffer $\mathcal{D}$, human feedback buffer $\mathcal{D}_h$, feedback frequency N_f, total episodes N_e, maximum number of environment steps per episode T, update interval b
Begin
for $i = 1$ **to** N_e **do**
  for $t = 1$ **to** T **do**
    Observe state s
    Sample action from current DQN policy π with ϵ-greedy, observe reward r and next state s$'$ and store (s, a, r, s$'$) in $\mathcal{D}$
    if t mod $b == 0$ **then**
      Sample a mini-batch of transitions from $\mathcal{D}$ with prioritization
      Compute the DQN loss L_{DQN} and update θ over L_{DQN}
      Sample a mini-batch of human feedback from $\mathcal{D}_h$
      Compute the feedback loss L_F and update θ over L_F
    end if
  end for
  if i mod $N_f == 0$ **then**
    Obtain the last trajectory τ from $\mathcal{D}$
    Query τ to obtain feedback $\mathcal{H}_i$
    Append $\mathcal{H}_i$ to buffer $\mathcal{D}_h$
  end if
end for
End

[1] We use a smaller feedback buffer size for *Pixel Taxi* because the trajectory length in *Pixel Taxi* is much smaller than that in *Atari* games, so less human feedback data are collected per query.

C. Overall Flow of EXPAND

Figure 4 shows the the overall flow of EXPAND.

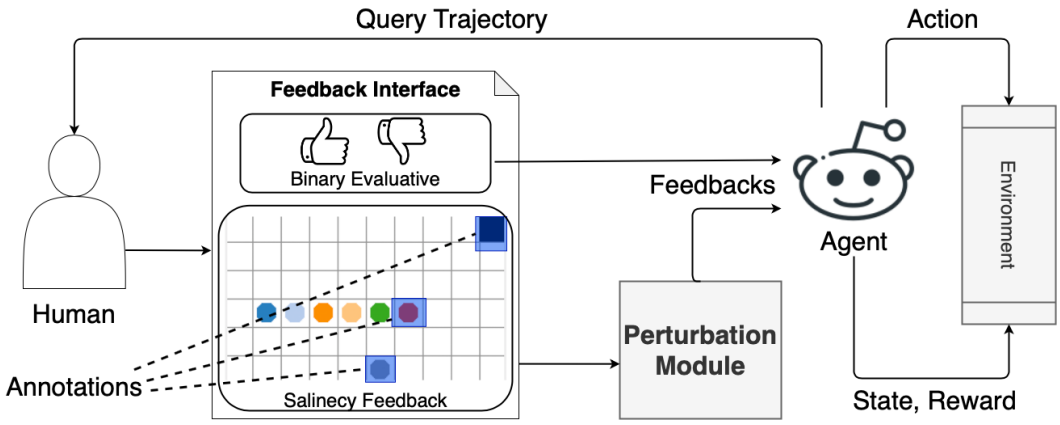

Figure 4: Overview of EXPAND. The agent queries the human with a sampled trajectory for binary evaluative feedback on the action and saliency annotation on the state. The Perturbation Module supplements the saliency explanation by perturbing irrelevant regions. The agent then consumes the integrated feedback and updates its parameters. This loop continues until the agent is trained with feedback queried every N_f episodes. The domain shown for "Salinecy Feedback" is *Pixel Taxi*.

D. Settings of Gaussian Perturbation

- 1 Perturbation:
  - filter size: 5, σ: 5
- 5 Perturbations:
  - filter size: 5, σ: 2
  - filter size: 5, σ: 5
  - filter size: 5, σ: 10
  - filter size: 11, σ: 5
  - filter size: 11, σ: 10
- 12 Perturbations:
  - filter size: 5, σ: 2
  - filter size: 5, σ: 5
  - filter size: 5, σ: 10
  - filter size: 7, σ: 3
  - filter size: 7, σ: 5
  - filter size: 7, σ: 10
  - filter size: 9, σ: 3
  - filter size: 9, σ: 5
  - filter size: 9, σ: 10
  - filter size: 11, σ: 3
  - filter size: 11, σ: 5
  - filter size: 11, σ: 10

E. Examples of Perturbation

Fig. 5 shows examples of different perturbations on irrelevant regions in image observations.

F. Additional Saliency Maps

Fig 6 shows how EXPAND gradually learns to attend to regions deemed relevant by the human during training.

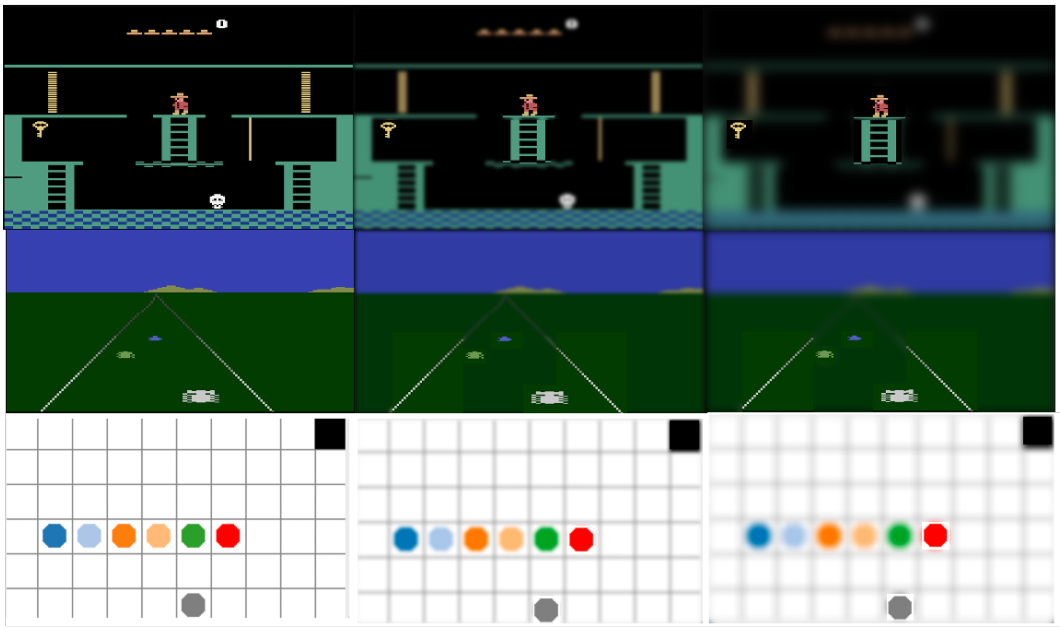

Figure 5: Examples of different perturbations on irrelevant regions in an image observation in the *Montezuma's Revenge MR-1*, *Enduro-1000* and *Pixel Taxi* environments. For each environment, the left image is the original observation. The remainder are observations with different perturbations on irrelevant regions, keeping relevant regions unchanged. Relevant regions are the player, the ladder beneath and key in *MR-1*; the player, other cars and lane boundary in *Enduro-1000*; and, the taxi-agent (the gray cell), the passenger to pick up (the red cell) and the destination (the black cell in *Pixel Taxi*.

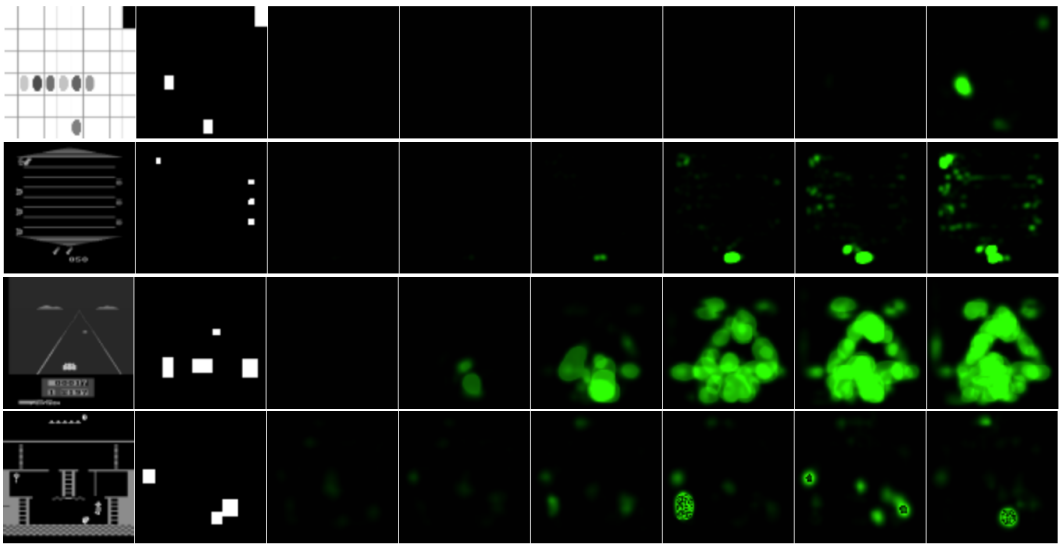

Figure 6: Each row is dedicated for one task. Starting from top to bottom, shown here are tasks *Pixel Taxi*, *Asterix*, *Enduro-1000* and *Montezuma's Revenge Room 1 (MR-1)*. Visualization for *Pong* can be found in the main paper. From left to right, we have the image observation (grayscaled), human saliency map (white regions were given as bounding boxes during training), the agent's attention maps [11] at 5k, 10k, 20k, 40k, 80k, and 300k training samples.

G. Human Interface

Fig 7 shows the web-interface used to conduct the human experiment. Before the experiment began, the trainers were briefed about the usage of this interface. The trainers were asked to provide their

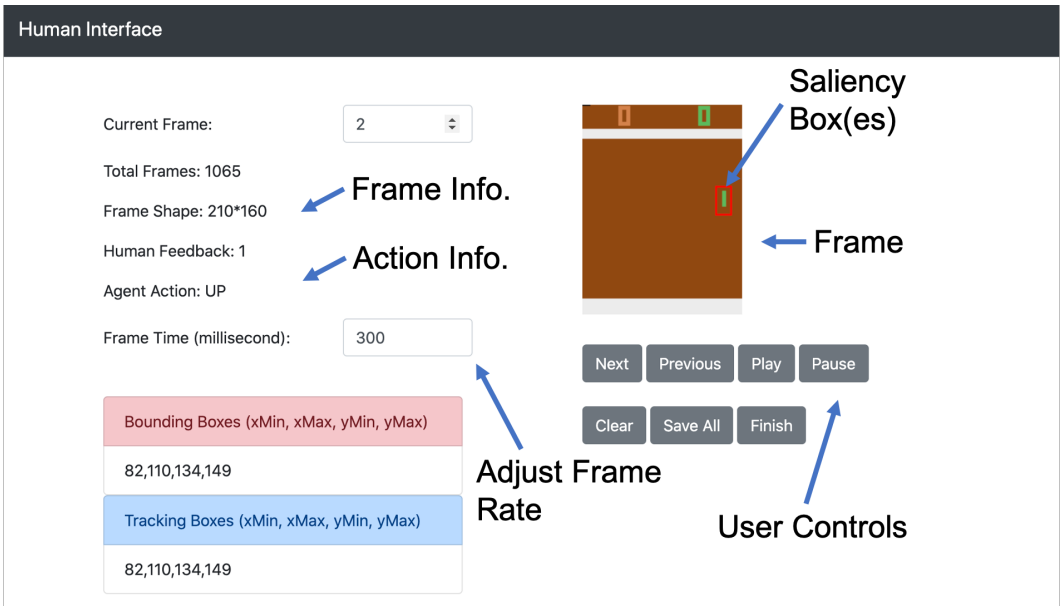

Figure 7: Web Interface to collect saliency and binary feedback from human trainers. This example uses a frame from *Pong*.

consent to use any information they provide for any analysis and experiments. The feedback interface, Fig 7, is divided into two columns, the information pane (left) and the control pane. (right). On the information pane, users were able to adjust the frame rate at which they would want to view the "video" made using the state transitions. Users could also view bounding-box related information, such as which ones were the drawn boxes and the boxes suggested by the implemented object tracker. On the right pane, users could view the frame upon which they would give their saliency feedback. Users could see the agent's current action in the information pane. For their ease, we provided controls like *Pause*, *Play*, skip to *Next* Frame, go to *Previous* Frame, *Clear* all bounding boxes on the frame, *Save* feedback and finally *Finish* the current feedback session. To provide a binary feedback on agents' actions, users were required to press keyboard keys "A" for a good-feedback, "S" for a bad-feedback, and "D" for no feedback. They could provide saliency explanations via clicking on at the required position on the image frame and dragging to create a rectangular selection.