Tirl: enriching actor-critic RL with non-expert human teachers and a trust model

Felix Rutard, Olivier Sigaud, Mohamed Chetouani

TIRL: Enriching Actor-Critic RL with non-expert human teachers and a Trust Model

Félix Rutard, Olivier Sigaud and Mohamed Chetouani

Abstract— **Reinforcement learning (RL) algorithms have been demonstrated to be very attractive tools to train agents to achieve sequential tasks. However, these algorithms require too many training data to converge to be efficiently applied to physical robots. By using a human teacher, the learning process can be made faster and more robust, but the overall performance heavily depends on the quality and availability of teacher demonstrations or instructions. In particular, when these teaching signals are inadequate, the agent may fail to learn an optimal policy. In this paper, we introduce a trust-based interactive task learning approach. We propose an RL architecture able to learn both from environment rewards and from various sparse teaching signals provided by non-expert teachers, using an actor-critic agent, a human model and a trust model. We evaluate the performance of this architecture on 4 different setups using a maze environment with different simulated teachers and show that the benefits of the trust model.**

I. INTRODUCTION

Various studies have shown that Reinforcement Learning (RL) can be used to train a robot to perform various tasks in autonomy [13], [2], [1], [15] in labs, but the corresponding algorithms generally suffer from a very low sample efficiency which render them impractical for deployment in real-world applications. This limitation can be addressed by a human partner in charge of accelerating the learning process by providing adequate demonstrations, instructions or feedback. In such situations where humans and robots share the same social and task spaces, identifying factors for efficient, safe and successful interactions is challenging. Trust is one of such factors and has mainly be considered as the human willingness to cooperate with the robot [17]. It is usually considered as a critical factor for establishing and maintaining effective relationships with robots [11]. In [8], trust in the robot has been considered as a main indicator of acceptance and measured by the participants' conformation to the robot's answers to questions on functional and social tasks.

The main focus of such works was the human willingness to cooperate and interact with a robot. In this paper, we propose to reverse the question by considering situations where human and robot are engaged in an interactive task learning scenario, and in case of non-optimal teaching, we investigate mechanisms allowing a robot to efficiently learn the task by considering a human task trust assessment. We propose to estimate this trust by assessing how the human teacher guides the robot during interactive task learning.

Sorbonne Université, CNRS, Institut des Systèmes Intelligents et de Robotique, ISIR, F-75005 Paris, France

II. RELATED WORK

As described in [4], Interactive Robot Learning deals with models and methodologies allowing a human teacher to guide the learning process of the robot by providing it teaching signals [26]. Usual teaching signals include instructions [10], [20] advice [9], demonstrations [3], guidance [23], [18] and evaluative feedback [14], [18]. A standard assumption in Interactive Robot Learning is that teaching signals are provided by experts. A distinguishing feature of our work is that our system can handle erroneous teaching signals. In that respect, the most closely related works are [18], [6], [28] and more recently [16].

To deal with this more difficult concern, our key contribution in this paper is to endow the agent with a trust model towards its teachers.

A taxonomy of trust related errors and mitigation strategies is proposed in [27]. In the literature, most of the works about trust in human-robot interaction focus on the human towards robot trust point of view, in sharp contrast with our approach. For instance, in [22], the authors investigated the effects of robot's error, task type and personality on cooperation and trust. Similarly, in [21], the authors also investigate the effects of robot error on human trust with a specific focus on time-critical situations. Few different works studied trust from a different perspective, such as [24] in which the robot uses surface cues to estimate trustworthiness of the human it interacts with. Other approaches develop methods in which enable the agent to use its own confidence to decide when to ask a human to help it by providing teaching signals [7].

In our study, the definition of the trust of the agent towards the teacher is closely correlated to the expertise level estimation of the teacher by the agent. A similar work studied the estimation of user's expertise level by a robot [5]. The authors proposed a method to estimate the user's expertise level relying on a Bayesian inference framework updating incrementally the human level of expertise model processed using a Boltzmann policy. Our approach requires fewer computation and can be incrementally performed within the interaction loop.

Finally, like us the authors of [19] propose a mechanism where the agent can choose which human to listen to when they are not equally trustworthy.

III. MODEL

In this section we describe TIRL (Trust-based Interactive Reinforcement Learning), our approach to endow a reinforcement learning agent with the capability to make profit of non expert teaching signals.

A. Overview

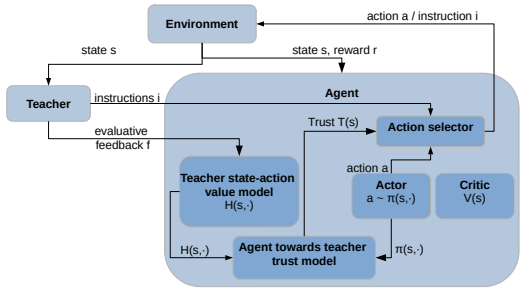


Fig. 1: Architecture overview: the agent can estimate trust in human teaching signals during interactive task learning

The architecture depicted in Figure 1 is composed of two blocks. First, an actor-critic agent learns from rewards a stochastic policy represented as a distribution. Second, the agent learns one or multiple models of non-expert teacher(s) who provide sparse teaching signals as instructions and evaluative feedback. In this paper, we address the problem of selecting an action using either the suggestion of the agent's actor or the teacher guidance signal. For this purpose, we introduce a trust value estimation in human guidance. A key design choice in our architecture is that evaluative feedback is always used to update the trust model through the teacher state-action value, whereas instructions are only followed if the trust of the agent towards the teacher is high enough. The trust model estimates a value of trust of the agent towards the teacher based on the difference between the actor's probability distribution over actions and the probability distribution over actions computed by the teacher state-action value. The teacher state-action value estimates how good the action is in the state according to the teacher, based on the provided evaluative feedback. A specific value propagation mechanism is necessary here to deal with sparse teaching signals.

In this architecture we used a fixed size experience replay buffer as described in [12] and a mini-batch training mode for the update of all architecture estimators in order to improve training efficiency. The indice j used in the following pseudo-codes describing the architecture refers to the j^{th} transition of the mini-batch transitions sampled randomly from the replay buffer.

See Algorithm 1 for the description of the complete algorithm.

Algorithm 1 Trust-based Interactive Reinforcement Learning

Inputs: actor $\pi(s)$, critic $V(s)$,
teacher state-action value estimator $H(s,a)$
initialized trust estimator $T(s)$,
for *steps* t *in* $\{1,2,3,...k\}$ **do**
  for each teacher **do**
    if $instruction$ is available **then**
      $i \leftarrow instruction$
    end if
  end for
  Select action:
  $a \leftarrow action\_selection(i, \pi, ifr, T_{thr})$
  Play action a and observe (s', r)
  for each teacher **do**
    if $evaluative\ feedback$ is available **then**
      $f \leftarrow evaluative\ feedback$
    end if
  end for
  Add transition to replay buffer:
  $(s_{prev}, a_{prev}, f_{prev}, s, a, r, s')$
  Store current episode for future transition storage:
  $s_{prev} \leftarrow s$, $a_{prev} \leftarrow a$, $f_{prev} \leftarrow f$, $s \leftarrow s'$
  Sample mini-batch of size N in replay buffer
  Update critic V and actor π from mini-batch
  for each teacher **do**
    Update teacher state-action value estimator H from mini-batch
    Update trust estimator T from mini-batch
  end for
end for

B. Reinforcement Learning agent

We consider a Markov Decision Process (MDP) $(\rho, \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma)$ where ρ is the initial state distribution, $\mathcal{S}$ the state space, $\mathcal{A}$ the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ the transition function, $R : \mathcal{S} \rightarrow \mathbb{R}$ the reward function and $\gamma \leq 1$ is a discount factor. The RL agent uses an Actor-Critic algorithm as described in [25]. The critic is defined as an estimated state value function $\hat{V}(s_t)$ which estimates the true value function $V(s_t) = \mathbb{E}_{\pi_t}(J_t | S_t = s)$ defined as the expected long-term reward return while following policy π_t. The long-term reward return is defined by $J_t = \sum_{k=0}^{End} {\gamma_{agent}}^k r_{t+k+1}$ with $k = End$ corresponding to the end of episode and r_{t+k+1} corresponding to the reward from the environment received at step $t+k+1$.
The critic is updated using:
$\hat{V}(s_t) \leftarrow \hat{V}(s_t) + \alpha_{critic}\, \delta_{agent_t}$ where δ_{agent_t} is the temporal difference (TD) error $\delta_{agent_t} = r_t + \gamma_{agent}\, \hat{V}(s'_t) - \hat{V}(s_t)$.
The actor $\pi(s,a)_t = Pr(a_t = a | s_t = s) = \frac{e^{\frac{p(s,a)}{\tau}}}{\sum_k e^{\frac{p(s,k)}{\tau}}}$ is defined as a probability distribution over action obtained from a softmax function with temperature τ on state-action preferences $p(s,a)$. The temperature parameter τ controls the exploration behavior of the agent: with low values the agent is more greedy and with high values of τ it explores a lot. State-action preferences $p(s,a)$ are initialized at 0 and are updated using the previously processed TD error δ_{agent_t} as $p(s_t, a_t) \leftarrow p(s_t, a_t) + \alpha_{actor}\, \delta_{agent_t}$ with α_{actor} the update parameter of the actor. We call π^* the optimal policy defined such that $V_{\pi^*}(s) = \max_{\pi} V_\pi(s) \,|\, \forall s$ and we define the optimal path as the set of states crossed by the optimal policy starting

from the initial state. See Algorithm 2 for the complete actor-critic update.

Algorithm 2 Actor π and critic $\hat{V}$ update

Inputs: mini-batch of transitions of size N, actor π, critic $\hat{V}$, actor and critic learning rates α_{actor} and α_{critic}, agent discount factor γ_{agent}
for each transition j in mini-batch **do**
Compute TD error with critic V:
$\delta_{agent_j} = r_j + \gamma_{agent}\,\hat{V}(s'_j) - \hat{V}(s_j)$
Update Critic $\hat{V}(s_j) \leftarrow \hat{V}(s_j) + \alpha_{critic}\,\delta_{agent_j}$
Update $p(s_j, a_j) \leftarrow p(s_j, a_j) + \alpha_{actor}\,\delta_{agent_j}$
Process Actor $\pi(s,\cdot) = \frac{e^{\frac{p(s,\cdot)}{\tau}}}{\sum_k e^{\frac{p(s,k)}{\tau}}}$
end for

C. Non-expert human teacher

We consider a non-expert human teacher who can provide two types of teaching signals: evaluative feedback and instructions. Evaluative feedback is a scalar provided after one or multiple agent transitions, which can take values in $\{-1, 0, 1\}$. An instruction is a human guidance signal toward an action associated with the current agent's state. These signals come with a *sparsity rate*: a sparsity of 0 corresponds to a silent teaching signal and a sparsity of 1 corresponds to the teacher providing it at each time step.

An evaluative feedback is erroneous if it is positive when the agent is not following the optimal policy and negative when the agent behaves optimally, whereas an instruction is erroneous if it provides actions different from the optimal policy. A human teacher who never provides erroneous teaching signals is said *expert*.

The agent is not aware of the sparsity rates and the details of the erroneous zone (if any) of the teacher.

D. Teacher state-action value estimator

A teacher can provide evaluative feedback for a given transition or for a set of transitions and the agent does not know which transition(s) the evaluative feedback is about. This problem is known as the temporal credit assignment problem and has already been tackled by the RL community using temporal difference (TD) learning.

The teacher state-action value estimates the expected value corresponding to the evaluative feedback return starting from state $s_t = s$ and playing action $a_t = a$ while following the agent's policy π_t. To perform this estimation, we define a specific discount factor $\gamma_{teacher} \in [0, 1]$ which accounts for how much the agent should back-propagate the evaluative feedback signal in time. Put differently, the parameter $\gamma_{teacher}$ controls the credit decay time horizon of evaluative feedback for teacher state-action value, thus we need to set this $\gamma_{teacher}$ to fit the assumed credit decay time horizon when the teacher provides an evaluative feedback.

The teacher state-action value $\hat{H}(s, a)$ is defined as the estimate of the expected evaluative feedback return starting from state s and playing action a: $\hat{H}(s, a) = \mathbb{E}(F_t | S_t = s, A_t = a)$. It uses the teacher evaluative feedback return $F_t = \sum_{k=0}^{End} {\gamma_{teacher}}^k\, f_{t+k+1}$ with f_{t+k+1} corresponding to the evaluative feedback provided by the teacher at step $t+k+1$.

$\hat{H}(s, a)$ is updated using Bellman equation with the SAFSA algorithm. The SAFSA algorithm comes from the SARSA algorithm with evaluative feedback (F) replacing the reward (R). The SARSA algorithm commonly used in RL is defined as $\hat{Q}(s, a) \leftarrow \hat{Q}(s, a) + \alpha\,(r_t + \gamma\,\hat{Q}(s', a') - \hat{Q}(s, a))$ with $\hat{Q}$ corresponding to the state-action value, s corresponding to the current state, a corresponding to the current action, s' corresponding to the future state, a' corresponding to the future action, γ corresponding to the discount factor and α corresponding to a learning rate.

We modified the SARSA algorithm using $\hat{H}(s, a)$ instead of $\hat{Q}(s, a)$ and the teacher evaluative feedback f_t instead of environment reward r_t and obtained $\hat{H}(s, a) \leftarrow \hat{H}(s, a) + \alpha_{teacher}\,(f_t + \gamma_{teacher}\,\hat{H}(s', a') - \hat{H}(s, a))$.

The SARSA algorithm tends to update the state-action value relatively to the current policy while the update rule in Q-Learning tends to update the state-action value relatively to the greedy policy. Since $\hat{H}(s, a)$ represents the teacher value according to its own policy, we chose to use the SARSA update rule rather than the Q-learning update rule.

In our implementation, instead of using the current state, action and teacher evaluative feedback (s, a, f_t) and future state and action (s', a'), we used previous state, action and feedback $(s_{prev}, a_{prev}, f_{prev})$ and current state and action (s, a). This is due to SAFSA algorithm which needs two transitions in order to obtain every component of the update rule $(s_{prev}, a_{prev}, f_{prev}, s, a)$. See Algorithm 3 for the update of $\hat{H}$.

Algorithm 3 Teacher state-action value estimator $\hat{H}$ update

Inputs: mini-batch of size N, actor π, critic V, $\hat{H}$ update coefficient $\alpha_{teacher}$
For each transition of the mini-batch update $\hat{H}$:
for *steps* j *in* $\{1, 2, ...N\}$ **do**
Compute TD error with SAFSA:
$\delta_{H_j} = f_{prev_j} + \gamma_{teacher}\,\hat{H}(s_j, a_j) - \hat{H}(s_{prev_j}, a_{prev_j})$
Update $\hat{H}(s_{prev_j}, a_{prev_j})$:
$\hat{H}(s_{prev_j}, a_{prev_j}) \leftarrow \hat{H}(s_{prev_j}, a_{prev_j}) + \alpha_{teacher}\,\delta_{H_j}$
end for

As a result, the replay buffer stores any previous evaluative feedback teaching signal provided by the teacher f_{prev} as well as previous (s_{prev}, a_{prev}, s) and current (s, a, s', r) agent transitions in order to update both actor-critic and the teacher state-action value estimator.

E. Trust estimator

With a trust model, the agent can filter teaching signals by evaluating the teacher considering the agent's state. By listening to teacher's instructions when the trust model is high and ignoring teacher's instructions when the trust is

low, we leverage teacher's knowledge to train faster while we avoid too much wastes of time or dead-ends when the teacher is wrong. Our trust value is obtained by comparing two discrete distributions over the action space: the stochastic actor's distribution $\pi(s,\cdot)$ and teacher state-action value distribution $H_{dist}(s,\cdot)$, with

$$H_{dist}(s,\cdot) = \frac{e^{\hat{H}(s,\cdot)}}{\sum_k e^{\hat{H}(s,k)}} \quad (1)$$

for a given state s. This comparison is performed using a Bhattacharyya distance D_B between $\pi(s,\cdot)$ and $H_{dist}(s,\cdot)$: $D_B(H_{dist}(s,\cdot),\pi(s,\cdot))$. We want the trust value to be 1 if both distributions are identical and 0 if they are completely different. Since $D_B(H_{dist}(s,\cdot),\pi(s,\cdot))$ is defined on $[0,+\infty[$ with 0 in the case of identical distributions, we compute the trust model using a exponentiated negative Bhattacharyya distance:

$$T(s) = e^{-D_B(H_{dist}(s,\cdot),\pi(s,\cdot))} = \sum_k \sqrt{H_{dist}(s,k)\,\pi(s,k)}.$$

We want to update this trust model only if $H(s,\cdot)$ and $\pi(s,\cdot)$ are accurate. Otherwise, we may process trust based on the Bhattacharyya distance between two random distributions. Consequently, the trust model $T(s)$ is initialized at 1 (agent starts fully confident in the teacher) and updated if both the agent and teacher TD errors associated to the state are below thresholds $\delta_{agent_{thr}}$ and $\delta_{teacher_{thr}}$. See Algorithm 4 for details.

Algorithm 4 Trust estimator T update

Inputs: mini-batch of size N
For each transition of the mini-batch update T:
for *steps* j *in* $\{1,2,...N\}$ **do**
  Compute $H_{dist}(s_j,\cdot)$ using (1)
  if $\delta_{agent}(s_j) \leq \delta_{agent_{thr}}$ and $\delta_{teacher}(s_j) \leq \delta_{teacher_{thr}}$ **then**
    $T(s_j) = \sum_k \sqrt{H_{dist}(s_j,k)\,\pi(s_j,k)}$
  end if
end for

F. Action selection

During training, we want the agent to filter teacher's instructions depending on the trust value. At each time step, if an instruction was provided by the teacher and if trust is above a threshold T_{thr}, the agent follows the teacher's instruction. Otherwise, the agent plays the action processed by its actor.

This strategy could also be employed in case of multiple teachers, where we need to choose which teacher to learn from. If only one teacher provides an instruction and the trust $T(s)$ associated to the current state for this teacher is greater than the trust threshold T_{thr}, we select this instruction. Otherwise, if multiple teachers provide instructions, we apply the following heuristic: we select a teacher based on the trust attributed to each teacher who provided instructions and whose trust value associated to the current state is greater than the trust threshold ($T(s) > T_{thr}$). We call this subset of teachers *validated teachers* VT. Hence we apply a softmax function to the trust vector $T(s)$ of each validated teacher and we sample a teacher from this probability vector K, the selected instruction is the instruction provided by the selected teacher. See Algorithm 5 for multiple teacher instruction selection.

Algorithm 5 Multiple teacher instruction selection (MTIS)

Inputs: teacher vector $Teach$, instructions vector $I[teacher]$, trust vector T,
Filter valid teachers:
$VT = Teach(T > T_{thr})$;
if $VT = \emptyset$ **then**
  return $valid = False$; $instruction = None$
else
  Apply softmax to trust: $K = softmax(V)$
  Sample $selectedTeacher$ in VT with probability K
  return $valid = True$;
    $instruction = I[selectedTeacher]$
end if

To control the balance between agent exploration and following teacher's instructions, we use a stochastic instruction following policy parameterized by ifr, the instruction following ratio. At each time step, if an instruction was provided by the teacher, we sample a value from a uniform distribution $\mathcal{U}(0,1)$ and if this value is below the instruction following ratio and the trust value is greater than the trust threshold, the agent plays the instruction. Otherwise, the agent plays the action suggested by its actor. See Algorithm 6 for action selection.

Algorithm 6 Action selection

Inputs: Actor π, state s, trust threshold T_{thr}, instruction-following ratio ifr, instruction or (teachers vector $Teach$, instructions vector $I[Teach]$, trust vector T) if multiple teachers
if multiple teachers **then**
  $instruction, valid = MTIS(Teach, I[Teach], T)$
else
  $valid = T(s) \geq T_{thr}$
end if
Sample x from uniform distribution $\mathcal{U}(0,1)$
if $instruction$ is available and $valid$ and $x \leq ifr$ **then**
  $a \leftarrow instruction$
else
  $a \sim \pi(s,\cdot)$
end if

IV. EXPERIMENTS

A. Environment

We use a maze with 60×60 cells, each action leads to a reward $r = \frac{-0.1}{(maze_{size})^2}$. The agent receives a reward $r = 1$ for reaching the terminal state. An episode is considered finished with success if the agent reaches terminal state within $(maze_{size})^2$ and is considered finished with failure if the agent reaches terminal after more than $(maze_{size})^2$ or if the number of steps of the episode exceeds $100\,(maze_{size})^2$ steps. Training is considered finished if the agent has succeeded 100 episodes in a row. The initial state is the top left corner of the maze and the rewarding state is the bottom right corner of the maze.

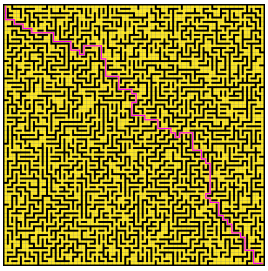


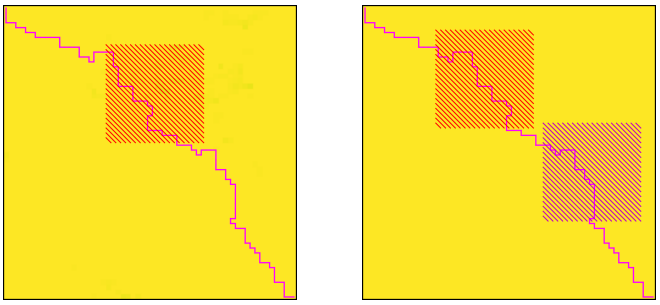
Fig. 2: Maze with optimal policy path in magenta

B. Experimental setup

In our experiments, we bind to the teacher an erroneous zone in which he or she provides erroneous teaching signals. In the maze environment the erroneous zone is defined by a geometry, which is either a square (associated with its position in the maze) or a diagonal zone, and a size, which is the edge size for the square zone, and a strip width for the diagonal zone.

In order to test our architecture we study the training performance (time to complete episode versus the episode index) in four different experimental setups:

- **Exp 1**: Agent trained with an expert teacher (no erroneous zone) with different sparsity rates from 0.0 (no teaching signals provided by the teacher) to 1.0 (teaching signals provided at each step);
- **Exp 2**: Agent trained with an expert teacher (no erroneous zone) then trained with a non-expert teacher (square erroneous zone of edge size 10 and including some optimal path's states) with different sparsity rates (from 0 to 1, with the same sparsity rates for both teachers);
- **Exp 3**: Agent trained with a non-expert teacher (square erroneous zone of edge size 10 and including some optimal path's states, figure 3) with different sparsity rates (from 0 to 1);
- **Exp 4**: Agent trained with two non-expert teachers sharing the same erroneous zone shape and size but not the same position (square erroneous zone of edge size 10 and including some states from the optimal path, figure 3) with different sparsity (from 0 to 1, with the same sparsity rates for both teachers).



(a) Err. zone in Exp. 2&3 (b) Erroneous zones in Exp. 4

Fig. 3: Erroneous zones and optimal policy path in magenta

C. Hyper-parameters and settings

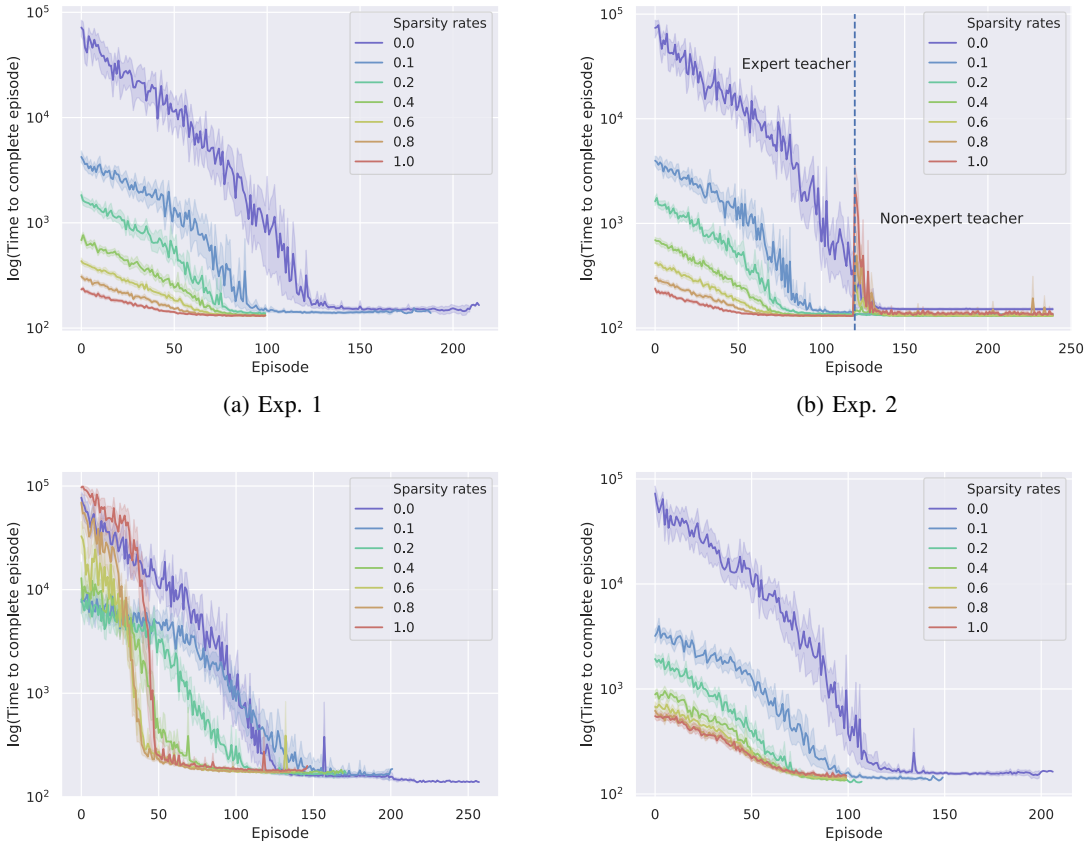
The mini-batch size N and the experience replay buffer size R have been arbitrarily set to values 10 and 5000 respectively. We begin by setting the actor-critic parameters τ, α_{actor} and α_{critic} which is done by training the agent without any teaching signals. The environment discount factor γ has been set to 0.99, a standard value in RL. The agent exploration temperature τ, actor and critic learning rates α_{actor} and α_{critic} have been set using a grid search method trying to maximize the performance. We then set teacher related parameters $\alpha_{teacher}$ and $\gamma_{teacher}$. To set $\alpha_{teacher}$, we start with a $\alpha_{teacher}$ and increase it until learning stops converging. We apply this method for different sparsity rates and we keep the lowest acceptable value of $\alpha_{teacher}$. Setting $\gamma_{teacher}$, $\delta_{teacher}$ and δ_{actor} is performed in a similar way using again different sparsity values. Finally we set T_{thr}, using a grid search for different configurations of teachers. The settings used in this study are described in Table I below.

TABLE I: Hyper-parameters of the study

Hyper-parameter name	value
Maze size	60
Mini-batch size N	10
Experience replay buffer size R	5000
Instruction-following ratio ifr	0.6
Trust threshold T_{thr}	0.6
Environment discount factor γ	0.99
Teacher discount factor $\gamma_{teacher}$	0.75
Actor learning rate α_{actor}	6
Critic learning rate α_{critic}	0.6
Teacher learning rate $\alpha_{teacher}$	0.7
Agent TD error threshold $\delta_{agent_{thr}}$	10^{-4}
Teacher TD error threshold $\delta_{teacher_{thr}}$	10^{-3}
Exploration temperature τ	2.5

V. RESULTS

All experiments have been conducted using 20 different seeds for each set of sparsity rates. Figure 4 shows the time to complete an episode in log scale versus the episode index in the different settings. The lower this time, the better the performance. Results are presented for each sparsity rate as a mean represented by a line and trust interval of 95% represented by a shadow band area around the line.



(a) Exp. 1

(b) Exp. 2

(c) Exp. 3

(d) Exp. 4

Fig. 4: Time to complete episode in log scale versus the episodes index

A. *First experiment*

In Exp. 1, the teacher is expert, providing no erroneous teaching signal. Figure 4a shows that every sparsity rate condition converged to a near-identical value. However, the higher the sparsity the faster the training and the more robust the training. Indeed the higher the sparsity rate, the narrower the confidence interval band.

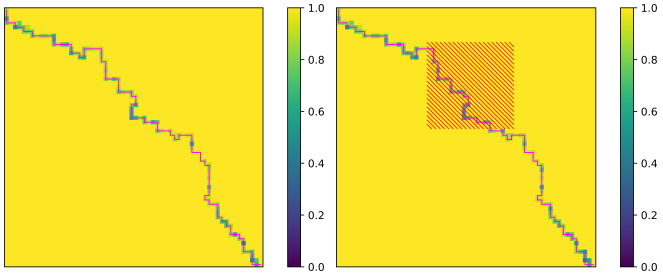
B. *Second experiment*

In Exp. 2, the teacher is expert for the first 120 episodes, but not for the 120 last ones. Figure 4b shows that for the 120 first episodes, results are identical to the first experiment, since we use the same training conditions. However we identify a performance drop at the beginning of the 120 last episodes of the experiment which vanishes in a few dozen episodes. The higher the sparsity rates, the bigger is the performance drop and the slower it vanishes. This can be explained by the trust value being very high in every state for the first 120 episodes since the first teacher was expert, and the agent taking more time to lose trust after episode 120 when the teaching signal is more sparse. When the signal is not sparse, the agent successfully attributes in a dozen episodes a low trust value to the teacher erroneous zone states and improves its performance using its own policy in the erroneous states which leads to a performance increase.

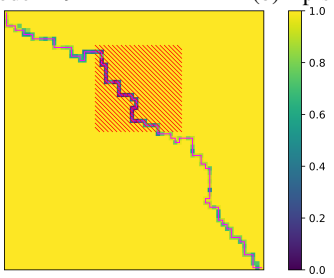
Figure 5a shows the trust map of the teacher just before swapping between expert and non-expert teachers ($step = 119$). As expected, the trust is high. Figure 5b shows the trust map just after the swap and we observe the trust in the teacher erroneous zone has not been reduced since the agent still had not enough time to discover the teacher erroneous zone. Figure 5c shows the last episode ($step = 239$) trust map on which we can identify very low trust attributed to states crossed by the agent's policy which overlap the erroneous zone of the teacher.

C. *Third experiment*

In Exp. 3, the teacher is not expert. Figure 4c shows that the higher the sparsity rate, the lower the performance at the beginning of the training (expected for sparsity rate 0.0, that is no teaching signal). Besides, the higher the sparsity rate, the higher the convergence speed except for sparsity 1.0 which is a little slower than sparsity rate 0.8. For high sparsity rate, the teacher provides way more erroneous signal than for lower sparsity rates, but the agent faster identifies the teacher's erroneous zone and so to learns faster. However, a high sparsity rate induces wastes of time or dead-ends at the beginning of training, because the agent ignores the teacher's



(a) Episode 119 (b) Episode 120



(c) Last episode

Fig. 5: Trust maps in Exp. 2. The color scale shows trust towards the teacher along the trajectory.

erroneous zone. We identify an optimum at sparsity $rate = 0.6$ in Figure 4c.

Figure 6 shows the last episode trust map, we notice very low trust has only been attributed to states crossed by the agent's policy which overlap the erroneous zone of the teacher. We also observe the agent policy is not optimal. Indeed, during training, the agent has been misled by the teacher in the teacher erroneous space. Thus the agent spent a lot of time trapped in this zone without being rewarded. This phenomenon led the agent to estimate expected reward in states overlapping teacher's erroneous zone to be very low. As a consequence, the agent's policy avoided the teacher's erroneous zone.

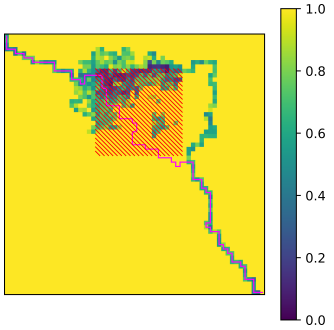
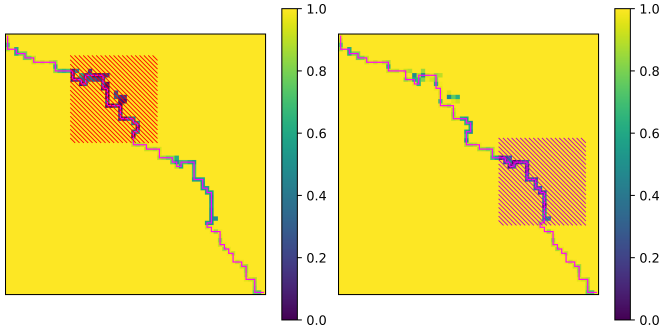


Fig. 6: Trust maps of last episode in Exp. 3

D. Fourth experiment

In Exp. 4, there are two non-expert teachers. Figure 4d shows that the higher the sparsity, the faster and more robust the training (the higher sparsity rate, the narrower the trust band). The trade-off between the low performance at the beginning of training and the high convergence speed we encountered during Exp. 3 is not present anymore. This can be attributed to the complementary nature of the erroneous zones of both teachers. While one teacher is providing erroneous teaching signal, the other one is providing correct teaching signal. The trust mechanism helps the agent to ignore the erroneous teacher depending on its state. This demonstrates how useful the trust mechanism is when using multiple teachers.

Figure 7a shows the last episode trust map of the first teacher. We notice very low trust has only been attributed to states crossed by the agent's policy which overlap the erroneous zone of the first teacher. Figure 7b shows the last episode trust map of the second teacher. We also notice very low trust in states crossed by the agent's policy which overlap the erroneous zone of the second teacher. This shows that the agent correctly estimates and uses the trust mechanism when training with multiple teachers.



(a) First teacher trust map (b) Second teacher trust map

Fig. 7: Trust maps in Exp. 4

VI. CONCLUSIONS

In this work, we proposed a new architecture where non expert teachers can guide a reinforcement learning agent with various teaching signals. This architecture relies on two important components: the teacher state-action value estimator which allows to handle sparse evaluative feedback signal, and the agent trust model towards teacher. With this model, the agent can identify zones in the state space where teaching signals are wrong. Along four experiments, we obtained faster and more robust training for every combination of teacher configurations and for different values of teaching signals sparsity rates. In the fourth experiment, we demonstrated the trust model can leverage multiple teachers with complementary erroneous zone.

An immediate extension for future work consists in adding more types of teaching signals such as demonstrations. Besides, in this work, the interpretation of evaluative feedback and instructions was given, but we could incorporate an interpretation learning process by using the methodology of the TICS architecture [18]. Finally, we implemented and tested TIRL using a discrete environment to facilitate the interpretation of our results focused on trust, but our work could easily be extended to continuous state and action environments using state-of-the-art deep reinforcement learning algorithms.

REFERENCES

[1] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[2] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*, 2018.

[3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[4] J. Broekens and M. Chetouani. Towards transparent robot learning through tdrl-based emotional expressions. *IEEE Transactions on Affective Computing*, pages 1–1, 2019.

[5] P. Carreno-Medrano, A. Dahiya, S. L. Smith, and D. Kulić. Incremental estimation of users' expertise level. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–8, 2019.

[6] C. Celemin, J. Ruiz-Del-Solar, and J. Kober. A fast hybrid reinforcement learning framework with human corrective feedback. *Auton. Robots*, 43(5):1173–1186, June 2019.

[7] S. Chernova and M. Veloso. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research*, 34:1–25, 2009.

[8] I. Gaudiello, E. Zibetti, S. Lefort, M. Chetouani, and S. Ivaldi. Trust as indicator of robot functional and social acceptance. an experimental study on user conformation to icub answers. *Computers in Human Behavior*, 61:633 – 655, 2016.

[9] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2625–2633. Curran Associates, Inc., 2013.

[10] J. Grizou, M. Lopes, and P. Y. Oudeyer. Robot learning simultaneously a task and how to interpret human instructions. In *2013 IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pages 1–8, Aug 2013.

[11] P. A. Hancock, D. R. Billings, K. E. Schaefer, J. Y. C. Chen, E. J. de Visser, and R. Parasuraman. A meta-analysis of factors affecting trust in human-robot interaction. *Human Factors*, 53(5):517–527, 2011.

[12] L. ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. In *Machine Learning*, pages 293–321, 1992.

[13] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Qt-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[14] W. B. Knox, P. Stone, and C. Breazeal. *Training a Robot via Human Feedback: A Case Study*, volume 8239 of *Lecture Notes in Computer Science*, book section 46, pages 460–470. Springer International Publishing, 2013.

[15] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *Int. J. Rob. Res.*, 32(11):1238–1274, Sept. 2013.

[16] M. Kwon, E. Biyik, A. Talati, K. Bhasin, D. P. Losey, and D. Sadigh. When humans aren't optimal: Robots that collaborate with risk-aware humans. *arXiv preprint arXiv:2001.04377*, 2020.

[17] J. J. Lee, W. B. Knox, J. Baumann, C. Breazeal, and D. DeSteno. Computationally modeling interpersonal trust. *Frontiers in Psychology*, 4, 2013.

[18] A. Najar, O. Sigaud, and M. Chetouani. Training a robot with evaluative feedback and unlabeled guidance signals. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 261–266, Aug 2016.

[19] S. M. Nguyen and P.-Y. Oudeyer. Active choice of teachers, learning strategies and goals for a socially guided intrinsic motivation learner. *Paladyn*, 3(3):136–146, 2012.

[20] V. Paléologue, J. Martin, A. K. Pandey, A. Coninx, and M. Chetouani. Semantic-based interaction for teaching robot behavior compositions. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 50–55, Aug 2017.

[21] P. Robinette, A. M. Howard, and A. R. Wagner. Effect of robot performance on human–robot trust in time-critical situations. *IEEE Transactions on Human-Machine Systems*, 47(4):425–436, 2017.

[22] M. Salem, G. Lakatos, F. Amirabdollahian, and K. Dautenhahn. Would you trust a (faulty) robot? effects of error, task type and personality on human-robot cooperation and trust. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI'15, pages 141–148, New York, NY, USA, 2015. Association for Computing Machinery.

[23] H. B. Suay and S. Chernova. Effect of human guidance and state space size on interactive reinforcement learning. In *2011 RO-MAN*, pages 1–6, July 2011.

[24] V. Surendran and A. R. Wagner. Your robot is watching: Using surface cues to evaluate the trustworthiness of human actions. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–8, 2019.

[25] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[26] A. L. Thomaz and C. Breazeal. Teachable characters: User studies, design principles, and learning performance. In *Intelligent Virtual Agents*, pages 395–406. Springer, 2006.

[27] S. Tolmeijer, A. Weiss, M. Hanheide, F. Lindner, T. M. Powers, C. Dixon, and M. L. Tielman. Taxonomy of trust-relevant failures and mitigation strategies. In *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, pages 3–12, 2020.

[28] Z. Zhu, K. Lin, B. Dai, and J. Zhou. Learning sparse rewarded tasks from sub-optimal demonstrations, 2020.