Should I Trust You? Incorporating Unreliable Expert Advice in Human-Agent Interaction

Tamlin Love[1], Ritesh Ajoodha[2] and Benjamin Rosman[3]

***Abstract*— A major concern in reinforcement learning, especially as it is applied to real-world and robotics problems, is that of sample-efficiency given increasingly complex problems and the difficulty of data acquisition in certain domains. To that end, many approaches incorporate external advice in the learning process in order to increase the rate at which an agent learns to solve a given problem. However, these approaches typically rely on a single reliable information source; the problem of learning with information from multiple, potentially unreliable sources is still an open question in assisted reinforcement learning. We present CLUE (Cautiously Learning with Unreliable Experts), a framework for learning single-stage decision problems with policy advice from multiple, potentially unreliable experts. We compare CLUE against an unassisted agent and an agent that naïvely follows advice, and our results show that CLUE exhibits faster convergence than an unassisted agent when advised by reliable experts, but is nevertheless robust against incorrect advice from unreliable experts.**

I. Introduction

Consider the scenario of a robot frail-care assistant, tasked with monitoring its patient and assisting in daily tasks. Suppose this robot has already learned how to optimally perform each individual task (e.g. mobility assistance, calling emergency services, dispensing medicine, etc.), but has yet to learn which tasks to perform in which situations, based on the observations it can make through its sensors (e.g. video footage, audio signal, time of day, etc.). In such a scenario, it is crucial for the robot to learn which tasks to perform for given observations, as there is a great deal of risk involved should the robot perform the wrong task. For example, if the patient has slipped and fallen, the correct response might be to call for help. If the robot does not perform these tasks, serious harm could come to the patient.

At the same time however, this type of decision-making scenario may be difficult to solve owing to its potential complexity, such as a large space of observations or a high number of available tasks. Furthermore, data acquisition can be difficult. A robot interacting with the real world may cause damage to itself or its surroundings if it executes the wrong task at the wrong time, and the robot may take a long time to learn a strategy if tasks take a long time to execute. Additionally, there may be ethical problems surrounding data acquisition, particularly where human patients are concerned. All of these factors necessitate that the robot learns as much as possible in the most sample-efficient manner.

One approach to tackling these issues is to introduce external information to the learning process [1]. For example, the robot could be advised by a human care-giver, who instructs it to perform certain tasks for certain scenarios. Given the potential complexity, it may not always be feasible to elicit all of this information before learning starts. Instead, the human advisor can advise the robot as it learns, in response to its performance. Indeed, previous work has shown that the interactive incorporation of expert advice in the learning process can improve the rate at which a reinforcement learning agent can converge to a given performance threshold, provided that said advice is correct [2].

It may be desirable to incorporate advice from multiple experts, either because a single expert does not have enough expertise to cover the full breadth of the problem, or simply because being able to incorporate more advice results in better sample efficiency [3]. For example, the robot could be assisted by a whole panel of experts composed of nurses, orderlies and other care-givers. Incorporating multiple experts introduces its own problems, however, when multiple experts offer conflicting advice for the same situation. Here the robot must decide whose advice to follow and whose to ignore. In general, expert advisers, especially humans, can give incorrect advice, either in error or through active malice [4]. Overcoming these problems is considered an open problem in the field of assisted reinforcement learning [1].

To that end, we introduce CLUE, a framework for learning single-stage decision problems (such as the example above) with policy advice from multiple, potentially unreliable experts. Our contributions include the framework itself, as well as Bayesian approaches to modelling expert reliability and pooling advice from multiple experts to facilitate decision-making. We demonstrate that CLUE, when advised by reliable experts, converges faster than an equivalent agent that does not incorporate advice, but is robust to advice given by experts that may be unreliable to some degree.

II. Background and Related Work

A. Single-Stage Decision Problems

Single-stage decision problems (SSDPs), also known as *contextual bandits* [5], are a type of reinforcement learning (RL) problem in which an agent observes some state $s \in S$ from the environment, selects some action $a \in A$ and in return receives some reward $r(s,a) \in \mathbb{R}$ from the environment. Each round of observation, decision-making and environment

All authors are with the School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg, South Africa
[1] tamlin.love1@students.wits.ac.za
[2] ritesh.ajoodha@wits.ac.za
[3] benjamin.rosman1@wits.ac.za

Code for all experiments presented in this paper can be accessed at https://anonymous.4open.science/r/CLUE_SSDP-4425

feedback is referred to as a *trial*, and each trial is independent from previous trials.

For example, the frail-care assistance example from Section I can be posed as an SSDP, with the observations made by the robot comprising the state, and each action corresponding to some task the robot is capable of performing. It is important to note in this example that each action corresponds to a high-level task rather than a low level action, such as a motor velocity or joint angle. The reward in this example could be related to the patient's well-being.

The goal of the agent is to find the optimal policy $\pi^* : S \rightarrow A$, such that $\pi^*(s) = \underset{a}{\text{argmax}}\, EU(a|s)$, where $EU(a|s)$ denotes the *expected utility* (i.e. *expected reward*) of selecting action a in state s.

A common approach to learning an optimal policy is an *action-value ε-greedy approach* [6]. The agent maintains an estimate $Q(s,a) \approx EU(a|s)$, known as the *action-value function*, and each trial either selects a random action in A with probability ε (called "*exploration*") or else selects an action $a^* = \underset{a}{\text{argmax}}\, Q(s,a)$ (called "*exploitation*"). At the end of each trial t, having observed s_t, selected a_t and received r_t, the agent updates the action-value function as follows

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha_{step}(r_t - Q(s_t,a_t)), \qquad (1)$$

where $\alpha_{step} \in (0,1]$, known as the *step size parameter*, controls the rate at which the agent learns [6].

B. Assisted Reinforcement Learning

Assisted reinforcement learning (ARL) is a framework for incorporating external information in the learning process. Many diverse methods fall under this framework, including Heuristic RL [7], RL from demonstration [8] and transfer learning in RL [9]. The most relevant of these ARL approaches to this work is *interactive RL* (IRL), in which an expert (human or software-based) provides information to the agent during the learning process [10].

IRL methods can be classified based on the type of advice the expert gives. In *reward-shaping* approaches [11],[12], the expert modifies the reward signal provided to the agent (e.g. by providing positive or negative feedback when the agent selects certain actions). In *policy-shaping* approaches [13],[14], the expert modifies the agent's policy, typically by advising an action for a given state and having this action override the agent's policy whenever that state is encountered. Both approaches are preferred for different situations and domains. For this research, we focus on policy-shaping, as state-action advice can be more easily elicited from human experts in certain domains (such as the frail-care assistance example of Section I), requires minimal similarity between the agent and expert [2], and is more robust to infrequent and inconsistent feedback [14].

Most approaches in ARL assume the advice to be coming from a single, infallible expert. However, this assumption does not always hold, especially when the expert is human [4]. Suboptimal advice could be the result of communication error, erroneous domain knowledge or a malicious expert. Furthermore, incorporating advice from multiple experts introduces the possibility of two or more experts offering contradicting advice, requiring the agent to choose which advice is more likely to be correct [3]. The problems of incorporating advice from unreliable experts and incorporating advice from multiple experts are considered open questions in ARL [1].

Several approaches deal with these problems in different ways. In a reward-shaping setting, one approach is to combine advice from multiple experts as a weighted sum of potential functions, whose weights are updated during learning [12]. In policy-shaping settings, approaches include modelling the probability C that the expert gives optimal advice using a single, static parameter for each expert [14], and by decaying the reliance the agent has on a transferred policy as learning progresses [13]. Our work differs from these approaches by focusing on policy-shaping advice in the form of state-action pairs, and on learning a model of each expert's reliability and using this model to combine this advice to calculate an optimal policy.

III. Methodology

To that end, we begin by formally defining what it means for an expert to be reliable or unreliable. In this research, the advice given by an expert e takes the form of the state-action pair $(s, a^{(e)})$. When the agent receives this advice, it knows that expert e has asserted action $a^{(e)}$ to be the optimal action for state s. If that assertion is true (i.e. $EU(a^{(e)}|s) \geq EU(a|s) \forall a \in A \setminus \{a^{(e)}\}$), the advice is said to be *correct*. If the advice given by an expert is correct for every state, the expert is said to be *reliable*. Otherwise, it is *unreliable*.

In order to make the problem of incorporating multiple, potentially unreliable experts into the SSDP learning process tractable, we introduce the following two assumptions. Firstly, we assume that, for any state in S, an expert is equally likely to give correct advice. This does not always hold for all problems. For example, in the frail-care assistance example of Section I, nurses, orderlies and other kinds of care-givers may have different areas of expertise. In order to relax this assumption, one would have to divide the state-space into domains of expertise. In general, this problem is non-trivial and so lies outside the scope of this research.

Secondly, we assume that, for any trial, an expert is equally likely to give correct advice. In general, this assumption does not hold. For example, a human expert may become more unreliable over time as they get tired, or a malicious expert may give correct advice in low-reward states and incorrect advice in high-reward states in order to sabotage the agent's learning. However, for most situations where an expert is consistent and helpful, we expect this assumption to hold.

A. CLUE

As previously stated, the aim of this research is to develop an algorithm for learning SSDPs with policy advice from multiple, potentially unreliable experts. To that end, we present **C**autiously **L**earning with **U**nreliable **E**xperts

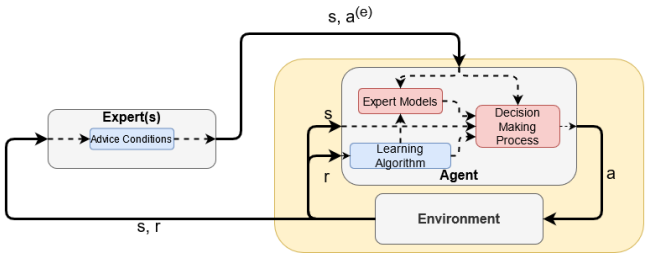

Fig. 1. A high-level overview of CLUE, showing the interactions between the environment, the agent and expert(s). Components depicted in red represent contributions made by this research.

(**CLUE**), whose high-level process is outlined below and represented in Figure 1.

The CLUE algorithm involves three actors: an environment, an agent and a panel E of one or more experts. The environment is a typical SSDP environment, as described in Section II-A. For each trial t, it samples a state s_t, accepts an action a_t from the agent and returns a reward r_t. Afterwards, each expert $e \in E$ has the chance to offer advice $(s_t, a_t^{(e)})$ to the agent, based on the agent's performance in trial t. Whether or not each expert gives advice and what advice they give is determined by each individual expert; the process used in this research is described in Section IV-A.

The agent is composed of three components, the first of which is a learning algorithm, such as the action-value ε-greedy approach described in Section II-A, which uses $\langle s_t, a_t, r_t \rangle$ to learn a policy.

The second component is a model of each expert's reliability, which is necessary in order to learn which pieces of advice to follow. At the end of each trial, after every expert has had a chance to offer advice, whatever advice the agent has received, together with the agent's own information about the environment, is used to update the model for each expert. This process is described in Sections III-A.1 and III-A.3.

The third component is a decision-making process which uses the agent's own information, the advice provided by the experts and the model of these experts to select an action for a given state. This process is described in Section III-A.2.

1) Modelling Reliability: Intuitively, we can think of an expert as being unreliable to some degree. For example, an expert that gives correct advice for 95% of trials, while still unreliable, is more reliable than an expert that always provides incorrect advice. Thus the reliability of an expert lies on a scale between always giving suboptimal advice and always giving correct advice (a reliable expert). Therefore, following [14], we model an expert's reliability $\rho \in [0,1]$, where $\rho = 0$ corresponds to an expert that always gives suboptimal advice and $\rho = 1$ corresponds to a reliable expert.

As $\rho \in [0,1]$, a natural choice of distribution to model the probability of ρ being a given value is a beta distribution $Beta_\rho[\alpha, \beta]$, where the parameters $\alpha > 0$ and $\beta > 0$ can be thought of as counts recording the number of times the expert gave correct or incorrect advice respectively.

Thus, the best estimate of the reliability of the expert is the expected value $\mathbb{E}[\rho] = \frac{\alpha}{\alpha+\beta}$.

2) Making Decisions: We now discuss the decision-making process of a CLUE agent. For any given trial t, let $E_t \subseteq E$ be the set of experts that have offered advice for state s_t in trials $[0,...,t-1]$. We note that there are three cases conditioned on $|E_t|$.

Case 1: $E_t = \varnothing$. In this case, no advice has been offered for s_t, such as will happen when $t = 0$, and thus the agent must act without any advice. The decision-making strategy employed in this research is ε-greedy exploration [6].

Case 2: $|E_t| = 1$. In this case, a single expert e has offered advice for s_t. As $\mathbb{E}[\rho^{(e)}]$ is the best estimate of the reliability of the expert, we employ it as a parameter in a similar vein to ε in ε-greedy methods [6] or ψ in the probabilistic policy reuse algorithm [13], so that, with probability $\mathbb{E}[\rho^{(e)}]$ the agent follows the advice offered by expert e, and with probability $1 - \mathbb{E}[\rho^{(e)}]$ the agent acts as in Case 1.

Case 3: $|E_t| > 1$. In this case, multiple experts have offered (potentially conflicting) advice for s_t. A simple approach might be to pick the expert with the highest value of $\mathbb{E}[\rho^{(e)}]$ and ignore all others, thus reducing this case to Case 2. However, this approach eliminates the information that could be provided by other, less reliable experts, such as information revealed by consensus among experts (the "wisdom of the crowd" [15]) or the information provided by adversarial experts (experts who are almost always wrong, thus informing the agent which actions not to take).

To take advantage of the information provided by all experts, we instead employ a Bayesian approach to calculate the probability of each action being optimal given the available advice, inspired by similar approaches in crowd-sourced data labelling [16] and potential-based reward shaping [12]. Let a^* denote the optimal action for state s_t, and $v_t^{(e)}$ denote the advice utterance given by expert e for s_t, with V_t denoting the set $\{v_t^{(e)} | e \in E_t\}$. Thus, our aim is to calculate $P(a = a^* | V_t)$ for each $a \in A$. By Bayes rule,

$$P(a = a^* | V_t) = \frac{P(V_t | a = a^*) P(a = a^*)}{\sum_{k=0}^{|A|} P(V_t | a_k = a^*) P(a_k = a^*)}. \quad (2)$$

If we assume that each expert offers advice independently, and that the prior probability of each action being optimal is uniform (a reasonable assumption in general, although some domains may allow for a more informed choice of prior), then Equation 2 reduces to

$$P(a = a^* | V_t) = \frac{\prod_{e \in E_t} P(v_t^{(e)} | a = a^*)}{\sum_{k=0}^{|A|} \prod_{e \in E_t} P(v_t^{(e)} | a_k = a^*)}. \quad (3)$$

All that remains in order to calculate $P(a = a^* | V_t)$ is to determine the value of $P(v_t^{(e)} | a = a^*)$. As discussed in Section III-A.1, $\mathbb{E}[\rho^{(e)}]$ is the probability that expert e offers correct advice for a given state. Thus, under the assumption that if an expert does not advise a correct action, they select a suboptimal action in $A \setminus \{a^*\}$ with uniform probability [17],

$$P(v_t^{(e)} | a = a^*) = \begin{cases} \mathbb{E}[\rho^{(e)}] & v_t^{(e)} \text{ advises } a \\ 1 - \mathbb{E}[\rho^{(e)}] & v_t^{(e)} \text{ does not advise } a \end{cases} \quad (4)$$

Substituting Equation 4 into Equation 3, we can calculate the probability of each action $a \in A$ being optimal, and set $a_{best} = \underset{a}{\text{argmax}}\, P(a = a^*|V_t)$. As in Case 2, we use $P(a_{best} = a^*|V_t)$ as a parameter, selecting action a_{best} with probability $P(a_{best} = a^*|V_t)$ and acting as in Case 1 otherwise. Indeed, following this procedure for $|E_t| = 1$ results in an identical process to that outlined for Case 2, and thus we need only consider cases 1 and 3.

Of course, the above formulation assumes that $\mathbb{E}[\rho^{(e)}]$ accurately models the reliability of expert e, which may not always be the case (see Section III-A.3 for examples). In particular, the over-estimation of the reliability of particularly unreliable experts may result in the over-selection of suboptimal actions. Erring on the side of caution, we introduce a threshold parameter $T \in [0,1]$, such that if $P(a_{best} = a^*|V_t) < T$, the agent acts without advice. Thus the agent only follows advice if it is sufficiently confident that it is correct.

3) Updating Reliability Estimates: Finally, we discuss how the model presented in Section III-A.1 is updated at the end of each trial. At the end of trial t, after selecting action a_t and receiving reward r_t, some subset of experts offer their advice for state s_t. The learning algorithm then updates the agent's policy using $\langle s_t, a_t, r_t \rangle$. The agent must now update the reliability estimate of each expert (if any) that offered advice this trial.

Suppose expert e advised $a^{(e)}$ for state s_t. The agent can use its own information (e.g. an action-value function) to calculate $EU(a|s_t)\ \forall a \in A$ to determine if $a^{(e)}$ is optimal. Across t trials, with expert e having advised the agent $n^{(e)}$ times, let $x^{(e)}$ denote the number of times the agent has evaluated the advice to be correct, thus making $n^{(e)} - x^{(e)}$ the number of times the advice has been evaluated to be incorrect. For ease of readability, we omit the superscript denoting expert e. In order to update the reliability estimate of the expert, we wish to set $Beta_\rho[\alpha,\beta]$ to be equal to $P(\rho|x)$, which, by Bayes rule,

$$Beta_\rho[\alpha,\beta] = P(\rho|x) = \frac{P(x|\rho)P(\rho)}{\int_0^1 P(x|\rho)P(\rho)d\rho}. \tag{5}$$

As x and $n-x$ represent the number of times correct and incorrect advice has been given respectively, a natural choice of distribution to model $P(x|\rho)$ is a binomial distribution $B_x[n,\rho]$, and consequently we model $P(\rho)$ as a beta distribution $Beta_\rho[\alpha_0,\beta_0]$, which is conjugate to a binomial distribution [18]. The parameters α_0 and β_0 can be thought of as prior counts of x and $n-x$ respectively. Substituting in the distributions and taking advantage of the conjugate distributions, Equation 5 reduces to

$$P(\rho|x) = Beta_\rho[x+\alpha_0, n-x+\beta_0], \tag{6}$$

and thus

$$\mathbb{E}[\rho] = \frac{x+\alpha_0}{n+\alpha_0+\beta_0}. \tag{7}$$

Therefore, as the agent receives more advice from expert e, it need only update $n^{(e)}$ and $x^{(e)}$ to recompute $\mathbb{E}[\rho^{(e)}]$.

A major limitation of this method is the assumption that the agent's evaluation of the expert's advice is correct, which can only be true if the agent has a sufficiently good understanding of the environment. Early in the training process, this understanding is poor, and consequently the advice evaluations will be poor. However, as the agent learns, the accuracy of these evaluations will improve. Another potential cause of poor advice evaluations could be the violation of the assumptions in Section III.

IV. Experiments

Having outlined the CLUE framework, we now present a set of experiments in a simulated environment to demonstrate that **a)** when being advised by a reliable expert, a CLUE agent converges faster than an equivalent unassisted agent, and **b)** when being advised by an unreliable expert that is likely to give incorrect advice, a CLUE agent converges asymptotically to the same threshold of performance as an equivalent unassisted agent, thereby showing that a CLUE agent can benefit from good advice, but is robust to bad advice. Such experiments in simulated environments are a necessary first step towards real-world applications such as the example given in Section I.

A. Set-Up

1) Environment: One method for simulating an SSDP environment is to use an influence diagram (ID); a probabilistic graphical model whose variables correspond to a factored representation of the state- and action-spaces and the reward signal [19]. All experiments were conducted across 100 randomly generated IDs. Each ID has 10 binary state variables ($|S| = 1024$) and 3 binary action variables ($|A| = 8$), with each ID having a different, randomly generated graph structure, utility function and set of conditional probability distributions.

2) Agents: In each experiment, we compare the performance of four agents. The *True Policy Agent* has access to a "ground truth" model of the environment and always acts optimally, thus acting as an upper bound on possible performance. The *Baseline Agent* is an action-value ε-greedy agent, as described in Section II-A, with $Q_0(s,a) = 0\ \forall s \in S, a \in A$, $\alpha_{step} = \frac{1}{k(s,a)}$, where $k(s,a) \geq 1$ is the number of times action a has been performed in state s, and ε decays from 1 to 0 at a constant rate across the first 80% of trials.

To represent existing works in ARL, which always follow the advice of a single expert, we use the *Naïve Advice Follower* (NAF), which is identical to the Baseline Agent except that it always follows any advice is receives. If more than one pieces of advice have been received for a given state, it will randomly select an expert to follow. Finally, *CLUE* uses the same base learning algorithm as the Baseline Agent, models each expert with $\alpha_0 = \beta_0 = 1$, and has a threshold parameter of $T = \frac{2}{|A|} = 0.25$.

3) Experts: All experts in these experiments are simulated. In order to simulate the potential cost of giving advice [2][20], we impose the following two conditions on when the expert can give advice. Firstly, the expert can only give advice if μ trials have elapsed since the last trial it gave advice. Secondly, in order to ensure that the expert only

gives advice if the agent is performing sufficiently poorly, the expert can only give advice if

$$\sum_{t' \leq i \leq t} \frac{EU(a_i^*|s_i) - EU(a_i|s_i)}{t - t'} \geq \gamma, \tag{8}$$

where t is the current trial, t' is the last trial advice was given, a_i^* is the optimal action for s_i, a_i is the action taken by the agent in trial i, and γ is a parameter that controls how tolerant an expert is of suboptimal behaviour [20].

In order to simulate unreliability, each expert has a true reliability $\rho_{true} \in [0,1]$. With probability ρ_{true}, the expert advises the optimal action a_t^* (retrieved from a "ground truth" model), and with probability $1 - \rho_{true}$, the expert advises a randomly selected suboptimal action from $A \setminus \{a_t^*\}$.

B. Comparison of Panels

In this set of experiments, we compare the reward obtained by each agent advised by one of several panels of experts. Rewards are obtained and plotted across 80,000 trials, averaged over the 100 random environments. For legibility, curves are smoothed with LOWESS smoothing [21].

The first experiment compares the performance of each agent with the *Single Reliable Expert*, consisting of a single expert that always offers correct advice ($\rho_{true} = 1$), thus simulating the information source assumed by most IRL approaches [1]. The second experiment compares the agent performances with the *Single Unreliable Expert*, consisting of a single expert that always offers incorrect advice ($\rho_{true} = 0$), thus simulating the worst-case scenario for traditional IRL approaches. Both experiments are plotted in Figure 2.

With the single reliable expert, both NAF and CLUE outperform the Baseline Agent, with NAF converging particularly quickly, demonstrating the power of existing ARL methods when the assumption of reliability holds. As CLUE does not assume reliability and is therefore more cautious, it does not converge as quickly, although it still is able to take advantage of the correct advice to converge faster than the Baseline Agent. A demonstration of the robustness of CLUE comes with the single unreliable expert. In this scenario, NAF exclusively follows sub-optimal advice and

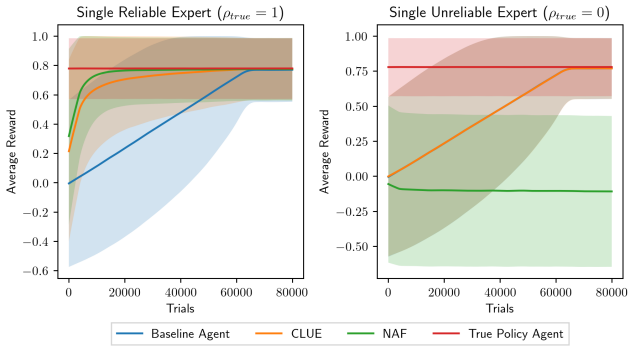

Fig. 2. A comparison of agent performance, advised by two panels, a *Single Reliable Expert* ($\rho_{true} = 1$) and a *Single Unreliable Expert* ($\rho_{true} = 0$). Note that for the single unreliable expert, the Baseline Agent and CLUE have near-identical performance.

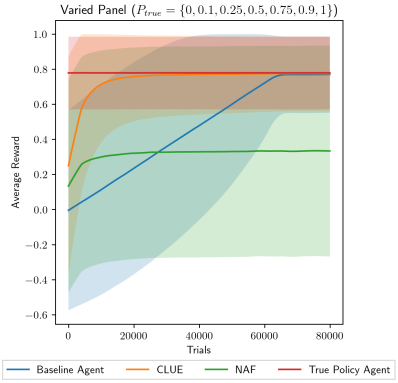

Fig. 3. A comparison of agent performance, advised by the varied panel ($P_{true} = \{0, 0.1, 0.25, 0.5, 0.75, 0.9, 1\}$).

therefore performs exceptionally poorly, failing to converge to the optimal policy. CLUE, on the other hand, correctly identifies that the advice is poor and learns to ignore it, and thus has performance almost identical to the Baseline Agent.

The next experiment compares the agents' performance with the *Varied Panel*, consisting of multiple experts of varying degrees of reliability ($P_{true} = \{0, 0.1, 0.25, 0.5, 0.75, 0.9, 1\}$), as seen in Figure 3.

Here the performance of NAF lies somewhere between the two single expert cases, as it receives a mix of advice including optimal and suboptimal actions, and cannot discern which advice is advantageous to follow. However, CLUE converges to the optimal policy even faster than it did in the case of a single reliable expert, comparable to the performance of NAF in the same case. This indicates that not only is CLUE learning to assess which experts are worth following and which are not, it is also benefiting from higher confidence in that assessment as a result of more advice collected from a wide range of experts.

C. Reliability Estimates

In order to further examine the results obtained in Section IV-B, we now compare the value of $\mathbb{E}[\rho]$ for each expert in each panel across the same 80,000 trials as in the previous experiments. As before, results are averaged over 100 runs in different randomly generated environments and the resulting plots are smoothed using LOWESS smoothing [21]. Results for the single reliable expert and single unreliable expert are presented in Figure 4, and results for the varied panel are presented in Figure 5.

For the single expert cases, the value of $\mathbb{E}[\rho]$ converges towards the correct value of ρ_{true} (1 and 0 respectively), with the final estimates being $\mathbb{E}[\rho] = 0.914$ for the single reliable expert and $\mathbb{E}[\rho] = 0.020$ for the single unreliable expert.

For the varied panel, each expert is correctly ranked according to their reliability and the value of $\mathbb{E}[\rho^{(e)}]$ for each expert e correctly converges towards the true value of $\rho_{true}^{(e)}$, even faster than the single expert cases, albeit to more conservative estimates tending away from the extremes of 1 and 0. The values to which the estimates converge, as well as the errors in these estimates, are tabulated in Table I.

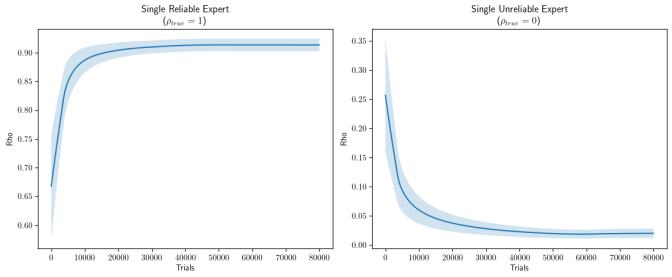

Fig. 4. The value of $\mathbb{E}[\rho]$ over time as the agent learns, advised by the single reliable expert ($\rho_{true} = 1$) and single unreliable expert ($\rho_{true} = 0$).

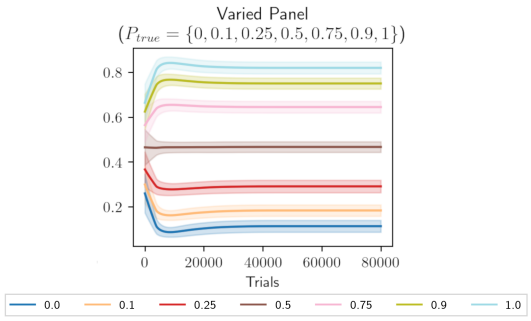

Fig. 5. The value of $\mathbb{E}[\rho]$ over time as the agent learns, advised by the varied panel ($P_{true} = \{0, 0.1, 0.25, 0.5, 0.75, 0.9, 1\}$). The legend shows the value of $\rho^{(e)}_{true}$ for each expert.

V. Conclusion

This research presents a framework for learning SSDPs with the advice of multiple, potentially unreliable experts, including Bayesian methods for estimating reliability and pooling multiple pieces of advice. Our results demonstrate that CLUE is able to retain the benefits of traditional ARL approaches when the expert is reliable, but is robust to the presence of incorrect advice. This research represents a step towards incorporating external information in real-world learning scenarios, such as robots learning from multiple domain experts. However, further research incorporating human experts and physical robot agents is required in order to determine how well our results would generalise to these settings.

TABLE I

Comparison of converged $\mathbb{E}[\rho]$ estimates for the varied panel.

ρ_{true}	**Final estimate**	**Absolute Error**	**Relative Error**
0	0.114	0.114	N/A
0.1	0.184	0.084	0.840
0.25	0.291	0.041	0.164
0.5	0.467	0.033	0.066
0.75	0.645	0.105	0.140
0.9	0.750	0.150	0.167
1	0.820	0.180	0.180

References

[1] A. Bignold, F. Cruz, M. E. Taylor, T. Brys, R. Dazeley, P. Vamplew, and C. Foale, "A conceptual framework for externally-influenced agents: An assisted reinforcement learning review," *arXiv preprint arXiv:2007.01544*, 2020.

[2] L. Torrey and M. Taylor, "Teaching on a budget: Agents advising agents in reinforcement learning," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 2013, pp. 1053–1060.

[3] C. Shelton, "Balancing multiple sources of reward in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 13, pp. 1082–1088, 2000.

[4] K. Efthymiadis, S. Devlin, and D. Kudenko, "Overcoming erroneous domain knowledge in plan-based reward shaping," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. Citeseer, 2013, pp. 1245–1246.

[5] J. Langford and T. Zhang, "The epoch-greedy algorithm for contextual multi-armed bandits," in *Proceedings of the 20th International Conference on Neural Information Processing Systems*. Citeseer, 2007, pp. 817–824.

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[7] R. A. Bianchi, C. H. Ribeiro, and A. H. Costa, "Heuristically accelerated Q–learning: a new approach to speed up reinforcement learning," in *Brazilian Symposium on Artificial Intelligence*. Springer, 2004, pp. 245–254.

[8] M. E. Taylor and S. Chernova, "Integrating human demonstration and reinforcement learning: Initial results in human-agent transfer," in *Proceedings of the Agents Learning Interactively with Human Teachers AAMAS workshop*. Citeseer, 2010, p. 23.

[9] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1633–1685, 2009.

[10] A. L. Thomaz, G. Hoffman, and C. Breazeal, "Real-time interactive reinforcement learning for robots," in *AAAI 2005 workshop on human comprehensible machine learning*, 2005.

[11] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The tamer framework," in *Proceedings of the fifth international conference on Knowledge capture*, 2009, pp. 9–16.

[12] M. Gimelfarb, S. Sanner, and C.-G. Lee, "Reinforcement learning with multiple experts: A bayesian model combination approach," *Advances in Neural Information Processing Systems*, vol. 31, pp. 9528–9538, 2018.

[13] F. Fernández and M. Veloso, "Probabilistic policy reuse in a reinforcement learning agent," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006, pp. 720–727.

[14] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz, "Policy shaping: Integrating human feedback with reinforcement learning." Georgia Institute of Technology, 2013.

[15] S. K. M. Yi, M. Steyvers, M. D. Lee, and M. J. Dry, "The wisdom of the crowd in combinatorial problems," *Cognitive science*, vol. 36, no. 3, pp. 452–470, 2012.

[16] P. Burke and R. Klein, "Confident in the crowd: Bayesian inference to improve data labelling in crowdsourcing," in *2020 International SAUPEC/RobMech/PRASA Conference*. IEEE, 2020, pp. 1–6.

[17] A. R. Masegosa and S. Moral, "An interactive approach for bayesian network learning using domain/expert knowledge," *International Journal of Approximate Reasoning*, vol. 54, no. 8, pp. 1168–1181, 2013.

[18] A. Etz, "Introduction to the concept of likelihood and its applications," *Advances in Methods and Practices in Psychological Science*, vol. 1, no. 1, pp. 60–69, 2018.

[19] R. A. Howard and J. E. Matheson, "Influence diagrams," *Decision Analysis*, vol. 2, no. 3, pp. 127–143, 2005.

[20] C. Innes and A. Lascarides, "Learning structured decision problems with unawareness," in *International Conference on Machine Learning*, 2019, pp. 2941–2950.

[21] W. S. Cleveland, "LOWESS: A program for smoothing scatterplots by robust locally weighted regression," *American Statistician*, vol. 35, no. 1, p. 54, 1981.