Towards Intrinsic Interactive Reinforcement Learning
A New Paradigm of Interactive Learning with Brain-Computer Interfaces

BENJAMIN POOLE and MINWOO LEE, University of North Carolina at Charlotte, USA

Reinforcement learning (RL) and brain-computer interfaces (BCI) are two fields that have been growing over the past decade. Until recently, these fields have operated independently of one another. With the rising interest in human-in-the-loop (HITL) applications, RL algorithms have been adapted to account for human guidance giving rise to the sub-field of interactive reinforcement learning (IRL). Adjacently, BCI applications have been long interested in extracting intrinsic feedback from neural activity during human-computer interactions. These two ideas have set RL and BCI on a collision course for one another through the integration of BCI into the IRL framework where intrinsic feedback can be utilized to help train an agent. This intersection has created a new and emerging paradigm denoted as intrinsic IRL. To further help facilitate deeper ingratiation of BCI and IRL, we provide a tutorial and review of intrinsic IRL so far with an emphasis on its parent field of feedback-driven IRL along with discussions concerning validity, challenges, and open problems.

CCS Concepts: • General and reference → Surveys and overviews; • Computing methodologies → Reinforcement learning; • Human-centered computing → Interaction design theory, concepts and paradigms.

Additional Key Words and Phrases: brain-computer interfaces, reinforcement learning, interactive reinforcement learning, intrinsic interactive reinforcement learning, human-in-the-loop, machine learning

1 INTRODUCTION

Over the past decades, Reinforcement learning (RL) has been receiving increasing amounts of attention due to its ability to perform in a variety of environments such as Atari games [88], Go/Shogi [100], StarCraft [118], DOTA [16] and robotics [5]. Meanwhile, applications of RL have only begun to expand beyond these constrained game environments to more diverse and complex real-world environments such as chip design [86], chemical reaction optimization [133] and performing long-term recommendations [45]. To further progress towards these more complex real-world environments, greater alleviation of challenges currently facing RL (e.g., generalization, robustness, scalability, and safety) is needed [7, 27, 72, 108]. Moreover, we can expect that as the complexity of environments increases, the difficulty in alleviating these challenges will increase as well [27].

For the purpose of this paper, we broadly define known RL challenges as either an aptitude or alignment problem. Aptitude encompasses challenges concerned with being able to learn. Aptitude includes ideas such as robustness, the ability of RL to perform a task (e.g., asymptotic performance) and generalize within/between environments of similar complexity; scalability, the ability of RL to scale up to more complex environment; and aptness, the rate at which a RL algorithm can learn to solve a problem or achieve a desired performance level. Likewise, alignment encompasses challenges concerned with learning as intended [7, 27, 72]. The hypothetical paperclip agent [18] is a classic example of misalignment. The agent maximizes its goal of making paperclips in an unintended manner by turning the world into a paperclip factory. While not as extreme, RL algorithms often display lesser forms of misalignment [7, 72]. Misaligned behaviors arise from a chain reaction that starts with increasing environment complexity which leads to an increase in the complexity of reward function and in turn the difficulty of specifying the reward function (Fig. 6 in A). To address the challenges of increasing environment complexity, highly robust, scalable, apt, and aligned algorithms are needed.
Interactive reinforcement learning (IRL) is a move towards increasing RL’s aptitude and alignment capabilities by expanding the RL framework to account for human guidance [13, 73, 90, 130]. IRL can be seen as incorporating a human-in-the-loop (HITL) where human guidance acts as a form of human-to-agent knowledge transfer [2]. This transfer of prior knowledge inherently aims to alleviate aptitude and alignment challenges. A crucial concern for IRL is how to intuitively communicate human guidance from a human to an agent [61, 75]. Classically, IRL guidance has needed to be explicitly elicited by manually providing positive/negative reinforcement signals to an agent (e.g., pressing a button or key) [12, 24, 61, 63] or by providing demonstrations of how to perform a given task [92, 93]. Recently, implicitly elicited guidance communicated through sensors tracking social feedback given off by humans has become an area of focus within the IRL community, providing natural ways for communicating with agents [76]. For instance, IRL has recently seen the integration of gaze detection [131]. Gaze is used to indicate latent information about what an agent should be focusing on, allowing for an agent to understand why certain feedback or reward might have been given. Further, researchers have begun using facial expressions, gestures, and vocals as implicit forms of guidance as well [8, 76].

Independently, and in parallel, of the IRL community’s shift towards implicit feedback, brain-computer interface (BCI) researchers have followed the idea of feedback to its intrinsic source, the brain [4, 20, 28, 47, 48, 58, 82, 119, 127]. The appeal of this approach can be derived directly from the definition of a BCI: a direct line of communication from the brain to external devices. Communication elicited directly from the source (i.e., the brain) is desirable as the elicitation of brain signals is automatic. Thus, intrinsically elicited human guidance can be extracted from neural activity. A primary motivation for going to the source is the vast pool of latent information contained within the mind. However, this proves to be both a blessing and a curse as having a vast amount of neural activity concentrated in one location means potential informative signals are shrouded in noise.

This paradoxical aspect of the mind entails that BCI researchers are faced with the ever daunting Cocktail Party Problem of extracting an informative signal from a sea of noise. Even so, this has not stopped researchers from discovering informative signals connected to different cognitive processes (e.g., error processing, attention, motor activity, etc.) [10, 30, 81, 89, 129]. Although decoding of these signals is by no means a solved problem, it has been shown that it is possible to decode and utilize these signals in a wide variety of fields such as medical, smart environments, bio-security, entertainment, and robotics [26, 54, 132]. Utilizing BCI in conjunction with IRL means that the medium for communicating guidance can be done through brain signals extracted from neural activity. The combination of these two fields, BCI and IRL, has been denoted as intrinsic IRL by Kim et al. [58].

While there have been previous surveys that aimed to review IRL literature, as listed in Table 1, this survey expands on those works. The goal of this survey is to compile an informative sampling of IRL frameworks to help facilitate a deeper integration of intrinsic feedback with the IRL literature. Thus, this article provides a tutorial and review of the parents fields of BCI, RL, and IRL. A representative set of IRL works are covered to further enhance the comprehensive survey of the emerging intrinsic IRL field. To properly inform researchers, comprehensive coverage of existing intrinsic IRL works is provided with a focus on IRL motivations, various strategies for integrating intrinsic feedback, and discussion of the challenges to help establish a road map of the emerging field.

The paper is structured as follows. Section 2 provides a general overview and background for RL and BCI. Section 3 provides an overview of IRL, followed by Section 4 which provides a review foundational feedback-driven IRL works. Section 5 provides an overview of intrinsic IRL and recent works. Finally, Section 6 concludes by discussing challenges and open problems for intrinsic IRL and its parent fields of IRL and BCI.
Towards Intrinsic Interactive Reinforcement Learning

2 BACKGROUND

2.1 Reinforcement Learning

Reinforcement learning (RL) is a sub-field of machine learning interested in exploring how an agent can learn to act in an environment. Classical learning in RL entails having an agent discover mappings of actions to observations through the maximization of a numerical reinforcement signal [108]. Given the agent’s current observation or state, the agent must decide what action to take (Fig. 8 in B). After enacting the chosen action, the environment updates to a new state where a positive or negative reward is elicited. The agent learns optimal behaviors (i.e., state-to-action mappings) by maximizing the total amount of reward it receives. This means the goal or desired behavior is implicitly communicated via the elicited reward.

When the RL environment is fully observable, it can be formalized as a Markov decision process (MDP). For the sake of providing an overview, we’ll describe RL in relation to fully observable environments only. The variables required for formulating a MDP are given by the tuple \((S, A, r, p, \gamma)\). Following the notation of Sutton and Barto [108], we denote the set of all possible states (i.e., observations), action, and rewards are denoted as \(S, A, \) and \(R\). Discrete time steps are used to divide states from one another where at each time step \(t\), the agent perceives state \(S_t \in S\), takes an action \(A_t \in A(s)\) and then receives a real-valued reward \(R_{t+1} \in R\). When time \(t\) is not specified, \(s\) denotes a state, \(a\) denotes an action, \(s'\) denotes the next state and \(r\) denotes the reward received. Furthermore, The reward function \(r : S \times A \times S \rightarrow \mathbb{R}\) is denoted as \(r(s, a)\). The transition function \(p : S \times A \times S \rightarrow \mathbb{R}\) is denoted as \(p(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a)\) and articulated as the probability of transitioning to the next state \(S_{t+1}\) and receiving the reward \(R_{t+1}\) given the current state \(S_t\) and action \(A_t\). The transition function is used to model the environment’s dynamics (i.e., the probability of transitioning to another state). Algorithms can either attempt to model these dynamics (i.e., model-based) or completely forgo them (i.e., model-free). Finally, the discount factor \(\gamma \in [0, 1]\) is used to determine how the agent values future reward.

Given the formal definition of a MDP, agent learning can be defined as maximizing total discounted return \(G_t\) from the current time step:

\[
G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}.
\]  

(1)

The reward is discounted by the discount factor \(\gamma\) to account for uncertainty in the model as the sum reaches further into the future and to ensure the math remains bounded. Given Eq. (1), as \(\gamma \rightarrow 0\) the agent begins to value immediate reward over future reward. When \(\gamma\) approaches 0 the agent becomes myopic and begins to only value immediate reward. Likewise, as \(\gamma\) approaches 1 the agent becomes farsighted and begins value all future reward equally.

---

**Table 1.** Summary of IRL surveys. Each category with a ✓ indicates the review is primarily focused on said category of guidance.

| Survey | Summary | Explicit Feedback | Implicit Feedback | Intrinsic Feedback | Demonstrations |
|--------|---------|-------------------|------------------|-------------------|---------------|
| Osa et al. 2018 [93] | Reviews different learning from demonstration methodologies. | ✗ | ✗ | ✓ | ✓ |
| Zhang et al. 2019a [130] | Broad review of various IRL algorithms categorized by types of human guidance. | ✓ | ✓ | ✗ | ✓ |
| Li et al. 2019 [73] | Review of feedback IRL algorithm implementations and applications. | ✓ | ✗ | ✗ | ✗ |
| Arzate Cruz and Igarashi 2020 [13] | Review of feedback IRL algorithms, categorized by shaping methodologies, and general application design principles. | ✓ | ✓ | ✗ | ✗ |
| Najar and Chetouani 2021 [90] | Provides a taxonomy of feedback, referred to as advice, along with a review and comparison of shaping methodologies. | ✓ | ✗ | ✗ | ✗ |
| Ours | Tutorial and review of intrinsic IRL and its parent field feedback-driven IRL. | ✓ | ✗ | ✓ | ✗ |
Recall, through this maximization of the total discounted return, the agent learns behaviors represented as state-to-actions mappings. These mappings are defined by a policy $\pi : S \times A \rightarrow \mathbb{R}$ where $\pi(a|s)$ denotes the probability of taking an action in a given state under a policy $\pi$. A policy determines how the agent should behave in a given environment, and therefore is essential to the agent. There are two general methods for learning a policy in RL: generalized policy iteration and policy gradient. For readers unfamiliar with RL, these methods are further discussed in the Appendix B and by Sutton and Barto [108].

2.2 Brain-computer interfaces

Brain-computer interfaces (BCI) allow for monitoring, communicating, and translating of brain signals elicited by cognitive and sensory-motor processes. Originally, BCI applications and research were medically driven. Specifically, BCI applications focused on assisting patients with locked-in syndrome who have no other way of communicating and interacting with the world. This focus led traditional BCI applications to be directed towards control problems such as wheelchair and interface control [96, 99]. As medically focused BCI research has continued, it has begun to spill out into the non-medical realm [26]. Applications in robotics [59, 97, 124], VR and gaming [54], smart environments [70], bio-metrics and security [38], and brain-to-brain interfaces [52, 95] have all allowed for the potential of intrinsic communication to be further explored. This exploration has opened up applications not just to those with disabilities but also to ordinary people.

Most BCI application follow what is referred to as the BCI pipeline which consists of signal acquisition, pre-processing, and classification (Fig. 1). Signal acquisition is focused on recording of neural activity. Pre-processing aims to improve the already captured neural data through filtering and feature extraction algorithms. Classification involves the decoding of informative brain signals from the neural activity using machine learning techniques. A brief overview will be given for each stage where the following BCI surveys provide a more in-depth review [25, 37, 80, 94, 96, 113, 116, 132].

2.2.1 Brain-computer Interface Pipeline. The first stage of the pipe is signal acquisition. Acquisition of signals involves the monitoring and recording of neural activity. BCI acquisition is subdivided into non-invasive, invasive, and semi-invasive methods where each method is defined where the neural activity is being recorded in relation to the cortex. Non-invasive methods involve monitoring neural activity from outside the scalp. In particular, we will focus on electroencephalographs (EEG) throughout this paper. This is due to EEGs being most common non-invasive device due to their desirable properties of being safe, portable, cost efficient, and relatively easy to work with [37, 96, 113]. EEGs use electrodes that measure electrical activity given off by neurons as they exchange messages via synapses where the features are given either in the time or frequency domain (Appendix C). EEGs are characterized by their high temporal resolution but poor spatial resolution. High temporal resolution is desirable for most BCI applications as it allows for neural activity to be captured within milliseconds of occurring [81].

The major drawback for EEG devices is that they tend to suffer from low signal-to-noise ratio (SNR). Collected neural activity data is quite noisy as it consists of the accumulation of processes occurring both internally and externally.
For instance, any given brain signal is accompanied by a great deal of biological noise introduced through cognitive processes occurring within the mind (e.g., fatigue, attention, engagement, etc.) and biological artifacts (e.g., eye blinks, muscle movements). At the time of signal acquisition, external noise is also introduced through external electrical activity (e.g., power lines, electronics, etc.). Thus, separating a brain signal from all included noise is desirable although it is quite a formidable task.

The next stage of pre-processing serves to filter and extract latent features associated with a desired brain signal of interest in order to increase the SNR. To do so, two different methods are often employed: temporal filtering and feature extraction. Temporal filters aim to filter out particular specific frequency bands thought to contain noise and artifacts and not the brain signal of interest. This usually entails the application of a band-pass filter that removes frequencies outside a specified range. Feature extraction is used to extract features thought to be relevant to brain signal of interest [79]. Spatial filters are a popular example of feature extraction that aims to combine the signals from different sensors, such that the result is a new signal with a higher SNR [79, 80]. Spatial filtering algorithms tend to have a three-fold effect of feature extraction, dimensionality reduction, and artifact removal. Popular examples of spatial filtering are principle component analysis, independent component analysis, and common spatial pattern.

The last stage of classification consists of decoding brain signals from the general neural activity. Like prior stages, decoding of brain activity is a difficult problem as BCI data possess a few challenging properties. The most difficult property of BCI data is the non-stationarity that arises from the low SNR. Due to large amounts of internal/external noise and the uniqueness of each individual’s brain, BCI datasets are highly non-stationary as the same brain signal can vary greatly within and between subjects. Brain signals within-subject often even vary between trials within the same recording session [80]. Moreover, BCI datasets are often small and have a high-dimensionality causing them to suffer from the curse of dimensionality. Classification algorithms in combination with pre-processing must be able to overcome these non-trivial properties to successfully decode brain signals.

Classical machine learning approaches such as support vector machines (SVMs) and linear-discriminant analysis (LDA) have often been employed in the BCI literature. The issue with these classical algorithms is that their performance is often tied to the feature quality produced by the pre-processing stage [80]. To further address the challenges of BCI data, new methods are being explored such as deep learning (DL) [71] and Riemannian geometry (RG) [80]. One notable benefit of both DL and RG is the potential to remove the pre-processing stage, in turn allowing for training to be conducted with raw signal data. For more information, the following BCI surveys on DL [25, 132] and RG [128] have been conducted.

2.2.2 Signal Types. Brain signals are the primary currency of a BCI pipeline. The entire BCI application hinges on being able to extract and detect brain signals that can be utilized by the application. Therefore, we provide a brief overview of different types of brain signals that will be relevant below.

Event-related potentials (ERPs) are transient brain signals characterized by their unique series of positive and negative peaks that are elicited due to changes over time in scalp-recorded voltage. Elicitation of ERPs can reflect sensory, cognitive, affective, or motor processes in response to either an internal or external stimulus [81]. From a general perspective, ERPs can be seen as the brain’s automatic response to a given stimulus or event. ERPs are composed of ERP components which are associated with certain neural processes. An ERP component is characterized by its positivity or negativity at a given point in time. However, this does not mean that there is a one-to-one relationship between ERP components and the observed polarity at a given point in time [81]. This is due to the fact that many known and unknown components are occurring at the same time. Thus, a combination of the components is always observed, rather than just the component of interest.
A commonly occurring ERP component is P3 or P300 that is characterized by a large positive peak that occurs approximately 300 ms after the stimulus. The P3 component is thought to be associated with neural processes such as context updating, surprise, and attention [81]. Rare-frequent tasks (e.g., the oddball paradigm) are a classical example of P3 elicitation. These tasks entail the repeated elicitation of frequent and infrequent stimuli, where the P3 amplitude inversely scales with the rarity of the infrequent stimulus [81]. P3 is often applied to decision making or interface control tasks like the P3 speller [31].

Error-related negativity (ERN) is another ERP component that is associated with errors and occurs approximately 200 ms after a stimuli [30]. Experiments frequently use ERN to determine whether a subject has either committed or observed an erroneous event [22, 32, 97]. ERPs which contain different variations of the ERN component are referred to as error-related potentials (ErrPs). ErrPs are characterized by a negative peak (i.e., ERN or Ne) that occurs between 80-300 ms and a positive peak (i.e., Pe) that occurs between 200-500 ms [3, 30, 44, 81, 115]. ErrPs are frequently used to provide feedback to an interface or an agent regarding erroneous events [23, 99].

Lastly, it is also possible to extract cognitive and affective states (e.g., fatigue, frustration, or attention) from neural activity. Doing so entails the utilization of frequency band features where each state is associated with a different combination of bands. For instance, Myrden and Chau [89] find that attention is associated with posterior alpha band activity while frustration is associated with posterior alpha and frontal beta band activity. Classification of other cognitive states such as mental workload, stress, fatigue, emotions, and interest are also being explored [6, 10].

2.3 Notation

Table 2 summarizes the common notation that will be used throughout the paper.

| Symbol | Description |
|--------|-------------|
| \( s \) | a state |
| \( a \) | an action |
| \( t \) | discrete time step |
| \( S_t \) | state at time \( t \) |
| \( A_t \) | action at time \( t \) |
| \( \tau \) | trajectory |
| \( \pi \) | policy |
| \( A \) | advantage function |
| \( h \) | human feedback |
| \( R_{t+1} \) | reward at time \( t + 1 \) due to \( S_t \) and \( A_t \) |
| \( S_{t+1} \) | next state at time \( t + 1 \) |
| \( H_t \) | human feedback at time \( t \) |
| \( \sigma \) | trajectory segment |
| \( \pi(a|s) \) | probability of taking an action in state \( s \) under a policy \( \pi \) |
| \( \mathcal{H} \) | entropy |
| \( \hat{h} \) | modeled human feedback |

3 INTERACTIVE REINFORCEMENT LEARNING

Human-in-the-loop (HITL) systems leverage the capabilities of both humans and intelligent agents in order to create collaborative models. A classical example of such a methodology is active learning [104]. A typical active learning interaction entails a learning system that queries a human expert for help when it is unsure about a particular data sample’s label. The human then provides the system with the correct label and the system learns using the provided labels. This interactive interaction continues until the convergence or the systems performance is deemed acceptable.

To help alleviate aptitude and alignment challenges associated with the increasing environment complexity, interactive reinforcement learning (IRL) aims to apply this HITL idea to reinforcement learning. IRL incorporates HITL by substituting human-provided labels for guidance. The goal of the human guidance is to guide and accelerate the agent’s learning of desired behaviors.

Human guidance can come in a variety of forms. As depicted in Fig. 2, we separate guidance into two categories: feedback and demonstrations. Feedback entails either a quantitative or qualitative evaluation of an agent’s behaviors. Demonstrations are concerned with showing an agent how to perform a task rather than assessing an agents behaviors.
Towards Intrinsic Interactive Reinforcement Learning

Fig. 2. A breakdown of different types of human guidance where the leaf nodes represent different methodologies for how specific types of guidance can be integrated with RL algorithms.

Due to the different interpretations, learning from feedback and learning from demonstrations require their own set of methodologies for learning. Given the categories of different human guidance, we address three critical questions concerning the viability of IRL throughout this and following sections:

(i) What are the motivations for pursuing IRL?
(ii) How can guidance $h$ be integrated into an RL algorithm such that an agent learns a good policy $\pi$?
(iii) Do IRL works fulfill the motivations of IRL?

Section 3.1 aims to address critical question (3.i) by discussing the motivations for integrating human guidance. Section 3.2 and Section 3.3 aim to address critical question (3.ii) by further discussing the different types of guidance and their corresponding methodologies. Lastly, Section 4 attempts to address the critical question (3.iii) while further addressing critical question (3.ii) through exploring foundational feedback-driven IRL works. Finally, Table 3 provides definitions IRL terms used throughout this and following section.

3.1 Motivations For Human Guidance

IRL’s motivations lie in direct connection to the RL challenges of aptitude and alignment. Inherently, IRL attempts to address specific challenges of aptitude, such as sparse reward, temporal credit assignment, and sample inefficiency while broadly addressing challenges of alignment. Sparse reward arises when a RL agent does not receive reward for a long period of time or, in some cases, not at all [108]. For instance, it is common for agents to only receive a reward upon completing a task or after performing some desired behavior, both of which can take hundreds or even thousands of time steps. Even for simple tasks, sparse reward can cause difficulty in learning as agents depend on reward in order to understand what behaviors need to be learned. Tangential to sparse reward is the challenge of temporal credit assignment [107]. The problem of credit assignment arises through correctly assigning a reward to the contributing events (i.e., state-action pairs). Naively, one can assume that reward corresponds to recent events. While this may work for simple environments or environments with dense rewards, this idea tends to fail in environments with sparse rewards. This is due to occurrences where important events are far away from the elicitation of a reward. This issue clearly arises for behaviors that require the learning of sub-behaviors where a reward is only received upon achieving some overarching behavior.

The combination of these prior two issues leads to the more notable and broader challenge of sample inefficiency. In most cases, RL agents need to learn from scratch using trial-and-error search which is expensive as it entails exploring a wider verity of options where the majority of these options are likely to be ineffective. Further, the agent must do so while also dealing with any sparse reward and credit assignment challenges. This makes RL algorithms inherently sample inefficient as learning can take millions of samples before an effective solution is found even for environments humans might find to be simplistic (e.g., Atari games) [87, 88]. This indicates that scaling up to more complex real-world environments could take an unimaginable amount of samples, given learning is even possible.
Table 3. Definitions for various IRL terms.

| Term      | Definition                                                                 |
|-----------|---------------------------------------------------------------------------|
| Human guidance | Any process or act that guides an agent in its learning, allowing for a transfer of knowledge from the human to the agent. |
| Feedback  | A form of guidance that is given through quantitative or qualitative evaluation of an agent’s behaviors. |
| Explicit Feedback | Feedback conveyed through direct or manual means, such as pushing a button or natural language. |
| Implicit Feedback | Feedback conveyed through indirect and observable means such as social cues, gaze, facial expressions, body language, and vocal tone. |
| Intrinsic Feedback | Feedback conveyed through indirect and, typically, unobservable means such as brain signals, heart rate, and other biological signals. |
| Demonstrations | A form of guidance that is given through demonstrations that show an agent how to perform a task. |
| Feedback Modeling (FM) | A method where a model of the human feedback is approximated. Typically, the predicted human feedback from this model is then used to shape an agent. |
| Reward shaping (RS) | A feedback shaping methodology where feedback is used to augment or replace the agent’s reward function. |
| Value shaping (VS) | A feedback shaping methodology where feedback is used to augment or replace the agent’s value function. |
| Policy shaping (PS) | A feedback shaping methodology where feedback is used to augment or replace the agent’s policy. |
| Inverse Learning (IL) | A learning from demonstrations methodology where the agent learns a policy indirectly by recovering the latent reward function contained within the expert demonstrations. |
| Behavioral Cloning (BC) | A learning from demonstrations methodology where the agent learns a policy by using supervised learning to mimic the expert demonstrations. |

Prior IRL works have often aimed to add human guidance solely in the pursuit of increasing sample efficiency [13, 24, 63, 130]. Human guidance increases efficiency by focusing an agent’s exploration space through either demonstrating or reinforcing behaviors the human knows to be effective [12, 24, 63, 83, 93, 120]. Moreover, guidance, in particular feedback, can supplement learning by providing more frequent reinforcement signals to the agent which are often temporally closer to the important behaviors than a sparse reward [63]. It is even often case that there is no reward function at all and agent learning is solely dependent on guidance [63, 93]. Thus, guidance can interpreted as a method for indirectly transferring human priors, helping to teach the agent while also preventing it from having to learn from scratch and acting out behaviors we know to be ineffective.

In terms of alignment, IRL inherently helps in alleviating challenges such as negative side effects and reward hacking [7, 72]. Negative side effects arise when negative outcomes occur due to an agent attempting to achieve a reward. For example, a cleaning robot might disrupt its environment in the pursuit of maximizing its goal of cleaning. Similarly, rewarding hacking occurs when an agent exploits loopholes in a reward function in pursuit of efficiently maximizing reward. For instance, a cleaning robot might clean the environment and then intentionally dirty the environment so that it may continuously clean and receive a reward. Both challenges result in unintended behaviors due to our inability to correctly specify intentions through mediums such as reward functions.

Human guidance inherently targets misalignment in two ways. First, it reduces the difficulty for the human to specify their intentions as the complexity of an environment increases (Fig. 7 in A). This reduction is largely due to human guidance being a more direct, intuitive, and accessible means of communicating intention compared to a reward function which is a strict set pre-programmed rules [72]. Second, human guidance can adapt to unintended agent behaviors as human guidance is given online and posses non-stationary and policy dependent properties [83]. This differs from classical pre-programmed reward functions which are normally static during learning.

3.2 Learning from Feedback

Human feedback denoted as \( h \) entails either a quantitative or qualitative evaluation of an agent’s behaviors as depicted in Fig. 3a. We define feedback in terms of behavioral psychology where one shapes an agent by consecutively reinforcing

---

1Human feedback actually only minimizes the problem of temporal credit assignment as even humans produce short delays between an event and guidance due to biological reaction time constraints.
simple sub-behaviors such that the agent learns to perform a more complex desired behavior [39, 63]. Classically, behavioral shaping has been conducted by rewarding animals through the elicitation of either positive or negative reinforcement. This same idea can be translated to RL. The reward function acts as external reinforcement that shapes the agent’s value function (e.g., what the agent values). In turn, the agent’s value function influences its policy (i.e., how the agent behaves). This idea of shaping can be further extended to quantitative or qualitative human feedback where any augmentation of the agent’s RL framework by human feedback entails shaping. General methods for integrating human feedback into the agent’s RL framework will be referred to as shaping methodologies. The answer to the critical question (3.ii) thus lies in these different shaping methodologies. However, before discussing shaping methodologies it is imperative to define quantitative and qualitative feedback.

We define quantitative feedback as feedback that quantifies an agent’s behaviors numerically. Quantitative feedback is often given in the form of binary feedback (i.e., +1 or -1) [61, 63, 83]. Positive feedback (e.g., +1) suggests an agent’s behavior was good while negative feedback (e.g., -1) suggests an agent’s behavior was bad. Through reward aggregation, repeated elicitation of either positive or negative feedback can be summed and mapped to the corresponding state-action pair to convey the magnitude of how good or bad a behavior was [63, 83].

When it is difficult to determine the magnitude of feedback that should be given, it is possible to formulate feedback more generally [72]. We define qualitative feedback as feedback that evaluates an agent’s behaviors non-numerically (i.e., categorically) where only general feedback about an agent’s behavior is provided. While qualitative feedback can still indicate whether the agent’s behavior was either good or bad, it does not provide any explicit information about the magnitude of the agent’s behavior. Alternatively, rather than providing good or bad feedback, preference-based qualitative feedback can be given to indicate whether one behavior is preferred over another behavior [122]. In practice, preference can be given in relation to a pair of either two different states $S_1^i > S_2^i$, actions $A_1^i > A_2^i$, or trajectories $\tau^1 > \tau^2$ [122]. Fig. 3b shows how the original RL formulation is updated to account for preference-based learning with trajectories.

Additionally, it is not always possible or desirable to constantly be capturing human feedback. Thus, it can be useful to perform feedback modeling. Feedback modeling forgoes using $h$ directly and instead attempts to model the feedback as $\hat{h}$ [63, 122]. Modeling of quantitative feedback means modeling becomes a regression problem where given some state, action, or state-action pair, the magnitude of feedback is estimated. Likewise, modeling qualitative feedback becomes a classification problem where given some state, action, or state-action pair, the probability a behavior is either good or preferred is estimated. Furthermore, it can often be desirable to extract a numerical representation from the qualitative feedback by attempting to model the latent underlying quantitative value [4, 24, 119, 122, 127].
3.2.1 Shaping Methodologies. In order to integrate human feedback \( (\hat{h}) \), the implicit problem of how feedback should be interpreted needs to be addressed. The Markov Decision Process specification of an RL framework consists of three main components: reward function, value function, and policy. Thus, it is possible to interpret human feedback as influencing any one of these components such that the interpretation determines the shaping methodology. Drawing from prior literature, we extract three categories of shaping methodologies: reward shaping, value shaping, and policy shaping.

Reward shaping is a shaping methodology where the agent’s reward function \( r(s, a) \) is augmented by a supplemental reward function such that a new reward \( r' \) is produced \([91]\). In the terms of human feedback, this means interpreting feedback as akin to a human reward function. Augmentation of \( r(s, a) \) is commonly done through simple perturbation of \( r(s, a) \) by \( \hat{h}/\hat{h} \) such that

\[
r' = r + \beta h,
\]

where \( \beta \) is predefined parameter that scales the impact the human can have on an agent. While quantitative feedback can be directly utilized with reward shaping \([64, 66, 109]\), qualitative feedback requires the learning of a latent numerical function \( \hat{h} \) \([24, 82, 127]\). Moreover, when the reward function is unknown \( (r = 0) \), it is common to completely replace \( r' \) with \( \hat{h}/\hat{h} \) \([24, 65, 67, 68, 82, 110–112]\).

Similarly, value shaping which entails augmenting the value function of an agent. In the terms of human feedback, this means interpreting feedback as akin to a human value function. Augmentation of a value function is often done through perturbing \( V/Q \) by \( \hat{h}/\hat{h} \) such that

\[
Q'(s, a) = Q(s, a) + \beta h(s, a),
\]

where this equation can be adapted to use the value function \( V(s) \) as well. Once again, while quantitative feedback can be directly utilized by value shaping \([64, 66]\), qualitative feedback requires the learning of a latent numerical function \( \hat{h} \). When the value function is unknown \( (i.e., V/Q = 0) \), it is possible to replace \( V'/Q' \) with \( \hat{h}/\hat{h} \). Although value shaping is far less popular than reward shaping, it tends to perform on-par and sometimes better than reward shaping \([64, 66]\).

Finally, policy shaping entails the augmentation of the agent’s policy \([35, 83]\). Unlike prior approaches\(^3\), policy shaping has a wider variety of different approaches due to the different possibilities available for influencing a policy such as action biasing, control sharing, Advise, and COACH. For instance, action biasing uses feedback to perturb the value function only during action selection \([64, 66]\). Control sharing uses the human policy derived from feedback for action selection, given the probability \( \beta \) \([4, 64, 66, 119]\). Advise directly combines the human policy derived from feedback with the agent’s policy by multiplying the distributions together \([35]\). COACH treats quantitative feedback as a label for the agent’s policy by directly replacing the advantage function in policy gradient algorithms \([12, 83]\).

3.3 Learning from Demonstrations

Learning from demonstrations (i.e., imitation learning) allows for agents to learn complex behaviors directly from expert\(^4\) demonstrations \([93]\). The original RL formulation is updated to account for demonstrations in Fig. 4 where the agent can be thought of as observing the expert interact with the environment. As learning from demonstrations uses a different form of guidance, it utilizes two broad methodologies for learning: behavior cloning and inverse learning \([93]\).

\(^2\)Reward shaping is considered as an indirect shaping method as the reward function already shapes an agent. Thus, reward shaping first augments the reward function by the feedback and then the augmented reward \( r' \) is used to shape the agent. Direct shaping can occur in special scenarios where the value function is myopic (i.e., the reward and value function are equivalent).

\(^3\)When referring to shaping methodologies we use the term “approaches” to refer to different possibilities for implementing a given methodology.

\(^4\)The term expert is often quite vague but is often attributed but not limited to a human who has skill and prior experience in performing the desired behavior.
Behavioral cloning directly learns a task by mimicking the expert demonstrations via supervised learning [93]. Learning a policy from demonstrations involves learning the mappings for a dataset $D$ from either contexts\(^5\) to trajectories $D = \{(C_i, \tau_i)\}_{i=1}^{N}$ or states to actions $D = \{(S_i, A_i)\}_{i=1}^{N}$ [93]. Behavioral cloning formulates the policy learning as a supervised learning problem. One major downside to behavior cloning is that it is susceptible to compounding errors as a mistake in one state can lead to unseen states where agents do not know how to act.

In an attempt to overcome for the rigid policies commonly learned through behavior cloning, inverse learning learns a policy indirectly by recovering the latent reward function contained within the expert demonstrations [1, 92]. The core idea involves using a dataset that contains expert trajectories $D = \{\tau_i\}_{i=1}^{N}$ to predict the reward such that the reward function frames the expert demonstrations to appear more optimal than the agent’s current policy. A major challenge of inverse learning is that there can be a variety of reward functions that explain the same expert demonstration [92, 93]. Therefore, learning a robust, generalizable, and non-degenerate reward function is often difficult [11].

4 FEEDBACK-DRIVEN INTERACTIVE REINFORCEMENT LEARNING

In this section, we provide a tutorial and review of foundational\(^6\) feedback-driven IRL works with the motivation of further bridging the divide between the fields of IRL and BCI. To do so, we first need narrow the scope of third critical question (3.iii) to only feedback-driven IRL. This is because intrinsic IRL utilizes brain signals (i.e., intrinsic feedback) which are formulated as an alternate medium for communicating feedback. Thus, an understanding of how existing feedback-driven IRL algorithms utilize shaping methodologies is needed. Furthermore, a focused analysis on the third critical question (3.iii) concerning the validity of feedback-driven IRL is needed to highlight the successes and limitations of feedback-driven IRL algorithms. To achieve this, we provide an analysis of each work from the perspective of the RL challenges (Section 1) with a focus on aptitude due to a lack of alignment analysis found in most works.

The foundational IRL works we sample span both quantitative and qualitative feedback and all shaping methodologies. A large portion of this section is devoted to quantitative feedback as qualitative feedback has received much less attention until recently. The TAMER framework explores modeling quantitative feedback, implicitly shaping myopic and non-myopic RL agents, and addressing the human feedback credit assignment challenge [8, 64, 68, 120]. The COACH framework provides a competitive alternative to that of the TAMER framework as it explores policy shaping without any feedback modeling [12, 83]. Christiano et al. [24] explore utilizing qualitative preference-based feedback to learn a latent quantitative representation to use for reward shaping. Table 4 summarizes the sampled foundational feedback-driven IRL works with the addition of intrinsic feedback works which we will discuss in Section 5.

\(^5\)Contexts includes information about about tasks or environment setting such as point-of-view or environment layout.

\(^6\)Here we define these foundational works as works that are frequently cited in the IRL literature.
Table 4. Overview of each feedback-driven IRL works regarding the feedback medium, authors, feedback type, shaping methodology, algorithms used for shaping, testing environments, baseline algorithm used by the original paper, and improvements/degradations in aptitude and alignment performance when compared with the original baselines. A ▲ indicates improvement while a ▼ indicates degradation. The size of the triangle gives a rough approximation to the degree of improvement or degradation. The ✗ indicates environments that were used outside of the original works. Finally, purple denotes non-IRL baseline algorithms. QN=quantitative, QL=qualitative, FM=feedback modeling, RS=reward shaping, VS=value shaping, PS=policy shaping, QL=Q-learning, DQL=deep Q-learning.

| Authors & Year | Type | Shaping Methods | Algorithms | Environments | Baselines | RL Challenges |
|----------------|------|-----------------|------------|--------------|-----------|---------------|
| Knox and Stone 2009 [63] | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | SARSADA(\(\lambda\)) | ▲ ▲ ▲ |
| Knox and Stone 2010 [64, 66] | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | [63] | ▲ | ▲ | | |
| Knox and Stone 2015 [65, 66, 67] | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | [64] | ▲ | ▲ | | |
| Warnell et al. 2018 [120] | ✓ | ✓ | ✓ | ✓ | ✓ | | [63] | | | | |
| Arakawa et al. 2018 [8] | ✓ | ✓ | ✓ | ✓ | ✓ | | [120] | | | | |
| MacGlashan et al. 2017 [12] | ✓ | ✓ | ✓ | ✓ | ✓ | | [63] | | | | |
| Arumugam et al. 2019 [12] | ✓ | ✓ | ✓ | ✓ | ✓ | | [120] | | | | |
| Christiano et al. 2017 [24] | ✓ | ✓ | ✓ | ✓ | ✓ | | [87], [101] | | | | |
| Iturrate et al. 2010 [48, 47] | ✓ | ✓ | ✓ | ✓ | ✓ | | [63] | | | | |
| Kim et al. 2017 [58] | ✓ | ✓ | ✓ | ✓ | ✓ | | [63] | | | | |
| Pflüger and Grefenstette 2018 [28, 29] | ✓ | ✓ | ✓ | ✓ | ✓ | | [63] | | | | |
| Luo et al. 2018 [82] | ✓ | ✓ | ✓ | ✓ | ✓ | | [24] | | | | |
| Xu et al. 2019 [127] | ✓ | ✓ | ✓ | ✓ | ✓ | | [126], [14] | | | | |
| Akimova et al. 2020 [4] | ✓ | ✓ | ✓ | ✓ | ✓ | | [102] | | | | |
| Wang et al. 2020 [119] | ✓ | ✓ | ✓ | ✓ | ✓ | | [102] | | | | |

### 4.1 TAMER

Training an Agent Manually via Evaluative Reinforcement (TAMER) is a framework for largely focused on modeling quantitative human feedback and using value shaping with replacement to shape the agent’s learning [61–63]. TAMER goes step further by attempting to overcome the credit assignment challenge of mapping delayed human feedback to its corresponding behavior. The TAMER framework consists of three main modules: feedback modeling, action selection, and credit assignment.

TAMER models human feedback \( h : S \times A \rightarrow \{-1, 0, 1\} \) by formulating the prediction of \( \hat{h}(s, a) \) as a supervised learning problem. This formalizes TAMER’s error \( \delta_t \) at time step \( t \) as

\[
\delta_t = H_t - \hat{h}(S_t, A_t),
\]

where \( H_t \) is the target and \( \hat{h}(s, a) \) is the predicted human feedback for a given state-action pair. Learning of the parameters \( w \) for \( \hat{h} \) (also denoted as \( \hat{h}_w \)) can be done by simply minimizing the squared error of Eq. (4) using stochastic gradient descent as follows:

\[
w_{t+1} = w_t - \alpha \nabla \frac{1}{2} (H_t - \hat{h}_w(S_t, A_t))^2
\]

\[= w_t - \alpha \delta_t \nabla \hat{h}_w(S_t, A_t).\]
Towards Intrinsic Interactive Reinforcement Learning

Due to TAMER’s goal of maximizing only the human feedback, the RL agent is formulated as a MDP without reward (MDP\(\mathcal{R}\)). The MDP\(\mathcal{R}\) tuple is then given as \((\mathcal{S}, \mathcal{A}, p, \gamma)\) where \(r(s, a)\) is excluded. Action selection then follows as greedily selecting the action estimated to receive the most feedback,

\[
\pi(s) = \arg \max h(s, a). \tag{6}
\]

While no RL agent is explicitly defined, TAMER can be thought of as value shaping an implicit agent where the agent’s value function is directly replaced by \(\hat{h}\) and therefore the temporal difference update is not needed. Alternatively, TAMER’s supervised learning formulation can be interpreted as a form of myopic RL where there is no separation between feedback model and the RL agent [61]. This reformulation can be better seen by using a full MDP where both value and reward shaping with replacement are used such that Eq. (4) is derived by taking the temporal difference error (Eq. (36) in B.1) and substituting \(\hat{r}, \hat{h}\) for the value function, and setting \(\gamma = 0\).

TAMER further addresses the temporal credit assignment problem of how to map the reward, human feedback in TAMER’s case, to the contributing state-action pairs [61, 63]. Human feedback is not instantaneous, so the delay between the occurrence of state-action pairs and human feedback \(h\) needs to be accounted. Especially when the states-action pairs occur at a rate less than that of the expected human delay (i.e., many state-action pairs occur within the delay), it is difficult to find a correct mapping, so an approximation of which state-action pairs contributed to \(h\) must be made.

TAMER’s proposed solution is to estimate the human feedback delay using a probability density function (PDF) [61, 63] to compute the probability that a \(h\) can be “credited” to a given state-action pair. When an accurate PDF is used, it is possible to calculate the probability that a given state-action pair fell within the expected human delay period as

\[
c(T^s, T^e, T^h) = \int_{T^e - T^h}^{T^s - T^h} f_{\text{delay}}(x)dx, \tag{7}
\]

where \(c(T^s, T^e, T^h)\) represents the credit or probability that \(h\) belongs to a state-action pair. The PDF \(f_{\text{delay}}(x)\) is bounded by when the state-action pair began and ended with respect to when \(h\) was received\(^7\). The first bound \(T^s - T^h\) represents the difference between the time at which the state began \(T^s\) and the time at which \(h\) was received \(T^h\). The second bound \(T^e - T^h\) represents the difference between the time at which the state ended \(T^e\) and the time at which \(h\) was received \(T^h\). Although the selection of the optimal PDF remains unclear and largely unexplored [61] the authors adopt a uniform distribution that is bounded by the average human reaction time [42] to simulate the human feedback delay [61, 63].

The credit can be integrated into Eq. (4) either through the delay-weighted, individual reward [63] or delay-weighted, aggregate reward [61] methods. Delay-weighted, individual reward applies credit to \(\hat{h}(s, a)\) such that a weighted sum over \(\hat{h}(s, a)\) is computed for each label \(H_t\) as

\[
\delta_t = H_t - \sum_k c(T^s_{t-k}, T^e_{t-k}, T^h_t) \hat{h}(S_{t-k}, A_{t-k}). \tag{8}
\]

where term \(k\) allows for weighted summation using prior state-action pairs contained within a sliding window (i.e., a history of state-action pairs). Thus, only prior state-action pairs \((S_{t-k}, A_{t-k})\) are considered to contribute to the weighted sum as state-action pairs that occurred after \(t\) cannot be attributed to \(H_t\). State-action pairs are pruned from the window when they are considered be old by having a near-zero credit for all feedback [61, 63].

Delay-weighted, aggregate reward applies the credit directly to \(h\) such that a weighted sum over \(h\) is computed for each state-action pair. This weighted sum inherently incorporates all prior human feedback that corresponds to the

\(^7\)Note that the order of subtraction depends on the formulation of PDF. Some prior works integrate the PDF with negative bounds (i.e., backwards view) while others use positive bounds (i.e., forwards view). Both views are valid and are essentially the same.
current state-action pair into a single label such that the TAMER error is computed as
\[
\delta_t = \sum_k c(T^e_t, T^r_t, T^h_t) H^k_t - \hat{h}(S_t, A_t)
\]  
(9)

where term \( k \) allows for the weighted summation over future elicited feedback contained within the sliding window. Thus, for a given state-action pair \((S_t, A_t)\) only future \( H^k_t \) is considered to contribute to the weighted sum as feedback that occurred before \( t \) cannot be attributed to \((S_t, A_t)\). Feedback is pruned from the window when it is considered old by having a near-zero credit for all state-action pairs \[61\].

TAMER’s empirical results show an increase in aptitude performance when compared to traditional RL algorithms (e.g., SARSA(\(\lambda\)). This is due to an increase in aptness from having good short-term performance and an increase in scalability as limited human feedback is required for learning to occur. That being said, TAMER’s overall robustness tends to be worse than traditional RL algorithms as they have better asymptotic performance. These results seem to only apply in simplistic 2D \[61, 63\] and robotics \[69\] environments. Additional studies have shown TAMER’s overall increase in aptitude performance is quite limited as TAMER fails to learn as the complexity of environments increases \[83, 120\]. Specifically, TAMER tends to forget and fails to generalize due to receiving varying feedback in similar states \[68, 83\]. TAMER’s inability to properly handle variation in feedback is associated with its myopic qualities (i.e., supervised learning) which only allows for a rigid encoding of the current policy \[68, 83\]. TAMER is also limited by its linear modeling of \( \hat{h} \) as it is unable to properly handle high-dimensional state spaces and non-linear relationships found within more complex environments (e.g., Atari games) \[120\]. To address the innate drawbacks, new iterations of the TAMER framework have been proposed to further increase the frameworks aptitude.

4.1.1 TAMER+RL. TAMER+RL aims to exploit TAMER’s short-term performance and RL’s asymptotic performance \[61, 64, 66\] by taking advantage of both human feedback and the environmental reward. To combine the benefits of TAMER and RL, \( h \) is first modeled using TAMER’s quantitative feedback modeling where \( \hat{h}(s, a) \) is used to shape the RL algorithm using different shaping methodologies and approaches. Doing so also allows for TAMER to utilize both \( \hat{h}(s, a) \) and \( r(s, a) \) during learning. Initially proposed were eight different shaping approaches \[64\] which were later narrowed down to the following four approaches in \[66\] as follows:

- **Reward Shaping**: \( r'(s, a) = r(s, a) + \beta \hat{h}(s, a) \)
- **Q-augmentation**: \( Q'(s, a) = Q(s, a) + \beta \hat{h}(s, a) \)
- **Action biasing**: \( \pi(s) = \arg \max_a [Q(s, a) + \beta \hat{h}(s, a)] \)
- **Control sharing**: \( P(a = \arg \max_a \hat{h}(s, a)) = \min(\beta, 1) \).

Recall, Q-augmentation is a value shaping approach while action biasing and control sharing are both policy shaping approaches (as discussed in Section 3.2.1).

TAMER+RL results show an increase in robustness as it outperforms both TAMER and SARSA(\(\lambda\)) in asymptotic performance while the original increase in aptness shown by TAMER is still maintained. While not shown directly, TAMER+RL shows a potential increase in scalability as it can continue learning without human feedback as the environmental reward can still be utilized. These results are replicated roughly for all four of the aforementioned shaping approaches. In particular, findings show that shaping approaches that gently push the RL agent towards the behaviors of the TAMER agent while slowly annealing the TAMER agent’s influence (e.g., action biasing and control sharing) correlate with good performance. However, only classical benchmark environments such as cart-pole and mountain car are used for testing. While the results show an increase in aptitude over both TAMER and SARSA(\(\lambda\)),
Towards Intrinsic Interactive Reinforcement Learning

TAMER+RL targets problems where an environmental reward function is available. This differs from TAMER which targets problems where only human feedback is available.

4.1.2 VI-TAMER. VI-TAMER expands on TAMER+RL by exploring the effects of non-myopcyic learning when combining TAMER and RL [61, 65, 67, 68]. As TAMER’s supervised learning modeling can be formulated as myopic RL problem [61], this often leaves TAMER learning a rigid policy making it hard for TAMER to generalize to unseen states. The authors postulate that non-myopic learning (e.g., combining TAMER with RL) improves the aptitude due to better generalization to unseen states, however the increase in aptitude comes at a cost of increased alignment issues. To explore these effects, VI-TAMER uses a full MDP and reward shaping where $r(s, a)$ is replaced by the modeled quantitative feedback $\hat{h}(s, a)$ such that the MDP tuple turns into $(S, A, p, h, \gamma)$.

Therefore, Bellman value function equation (Eq. (35) in B.1) is updated as follows:

$$Q_\pi(s, a) = \mathbb{E}_\pi[\hat{H}_{t+1} + \gamma Q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$

where $\hat{H}_{t+1} = \hat{h}(S_{t+1}, A_{t+1})$. The temporal difference (TD) error (Eq. (36) in B.1) is in turn updated to

$$\delta_t = \hat{H}_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t).$$

Testing with varying values of $\gamma$ in the grid world and mountain car environment shows that non-myopic learning can increase the TAMER’s robustness by allowing for better generalization when the environment layout is changed. At the same time, non-myopic learning is shown to decrease overall agent alignment as issues such as reward hacking start to appear. Since human feedback tends to be positively biased [111, 112], agents are able to exploit this positive feedback and perform behavior that do not align with the human’s intentions [61, 67, 68, 83]. While these unintended behaviors can often be corrected during online training with a human, they can not be corrected when the human stops providing feedback and $\hat{h}(s, a)$ becomes static. The only way VI-TAMER is able to overcome the alignment trade-offs is by formulating the tasks as a continual task rather than an episodic task such that the agent is no longer penalized for completing task\(^8\). Further investigation into how this trade-off arises and can be alleviated when using non-linear models of human feedback and more complex environments is still needed.

4.1.3 Deep TAMER. To overcome the limitations of the linear modeling of human feedback, Warnell et al. [120] integrate deep learning [71] with deep Q-learning methods [88]. Using using a convolutional autoencoder [84] to model $\hat{h}$ allows for an end-to-end approach that uses raw images. Furthermore, pre-training with reconstruction loss [34] can be used for better sample efficiency. Once the encoder is pretrained, the weights of the fully connected neural network are fine-tuned by computing a weighted sum of TAMER’s original loss where the weight is determined by the credit assignment $c(T^s, T^h)$:

$$\mathcal{L}(\hat{H}) = \sum_k c(T^s_{t-k}, T^h_{t-k}, T^h_T)[H_t - \hat{h}(S_{t-k}, A_{t-k})]^2.$$  \hspace{1cm} (12)

The weighting of the squared error is hypothesized to bias the agent towards state-action pairs that were intended to receive $h$ [120]. Experience replay [77, 88] further allows for human feedback $h$ to be reused for learning multiple times.

Deep TAMER is tested in the Atari Bowling environment using TAMER, A3C [87], Double-DQN [114] as baselines. The average learning curve over all subjects shows a large increase in aptness and robustness due to better short-term and asymptotic performance than the baselines. Further testing shows that Deep TAMER can even outperform human experts within 15 minutes of training. However, additional studies have shown that Deep TAMER’s overall aptitude increase is limited as it struggles in more complex 3D environments [12]. Finally, the authors also reevaluate the

\(^8\)Switching to continual tasks allows the intended behavior to be considered as the future reward is no longer limited by achieving said behavior. This is in comparison to episodic tasks where achieving the intended behavior marks the end of the task such that the future reward is limited.
selection of the PDF for the credit assignment problem where it is found that the uniform distribution $U(0.28, 4)$ leads to good performance, but the gamma distribution $\text{Gamma}(2, 0.28)$ vastly underperform.

4.1.4 DQN-TAMER. Inspired by TAMER+RL, Arakawa et al. [8] combines Deep TAMER with Deep Q-networks (DQNs) [88] to take advantage of both the human feedback and environmental reward. DQN-TAMER boils down to using the Deep TAMER to model human feedback $\hat{h}$ and action biasing to further shape the DQN agent which is trained using $r(s, a)$. The action biasing for shaping is slightly changed such that $\alpha_q$ weighs $Q(s, a)$ and $\alpha_h$ weights $\hat{h}(s, a)$, however only $\alpha_h$ is annealed overtime. An action is selected in a greedy manner such that

$$\pi(s) = \arg \max_a [\alpha_q Q(s, a) + \alpha_h \hat{h}(s, a)].$$

(13)

DQN-TAMER is tested in various grid world environments using Deep TAMER and DQN as baselines. Empirical results show that DQN-TAMER maintains its increase in aptness and in some cases shows a further increase. Additionally, DQN-TAMER shows an increase in robustness as it outperforms the baselines in asymptotic performance as well. As DQN-TAMER can continue learning when human feedback stops, given an environmental reward function exists, scalability is also likely to be increased. With this in mind, DQN-TAMER targets problems where an environmental reward function is available compared to that of Deep TAMER which targets problems where only human feedback is available. Nevertheless, the overall aptitude increase DQN-TAMER remains unclear due to testing with a limited number of environments that can be considered relatively simplistic. A further caveat arises as simulated feedback and not real human feedback is used for training. Furthermore, while simulated feedback is useful for testing how different variations in feedback affect performance (e.g., delayed, sparse, etc), there is the question of whether the simulated feedback accurately simulates real human feedback. As human feedback tends to be positively biased, policy dependent, and non-stationary, simulated feedback provided via the Manhattan distance might not produce results representative of real human feedback [68, 83].

4.2 COACH

Convergent Actor-Critic by Humans (COACH) [83] takes an alternative approach to TAMER by integrating human quantitative feedback with actor-critic algorithms through policy shaping. COACH is built on the assumption that both human feedback and the advantage function (Appendix B.2) are policy dependent, which allows for human feedback to directly replace the advantage function. Here policy dependence indicates that the feedback and advantage function values for the agent’s current policy are dependent on the quality of prior observed policies.

The policy dependence of feedback is shown empirically by tracking human feedback in relation to observing a series of pre-defined good, average, or bad policies. The policy dependent assumption affords the emergence of three desirable training strategies that coincide with how human feedback is often given:

1. Diminishing Returns: There is a gradual decrease in positive feedback for good actions as the agent adopts said good actions. This is useful for decreasing the burden of how active the human trainer must be when providing feedback.

2. Differential Feedback: Feedback varies in magnitude with respect to the degree of improvement or diminishment. This allows for emphasis to be placed on important behaviors or to communicate urgency for learning.
Towards Intrinsic Interactive Reinforcement Learning

(3) Policy Dependent Shaping\(^{6}\): Positive feedback is provided for sub-optimal actions to improve the behavior and negative feedback is provided after the improvement is made. This form of feedback indicates improvement relative to the current baseline.

These three policy dependent training strategies are also claimed to be encapsulated by the advantage function. The advantage function contains the differential feedback property inherently as the advantage function is interpreted as providing information about how much better or worse the current action is compared to the previous policy \(\pi\).

The advantage function induces diminishing return due to fact that \(V_{\pi}(s)\) slowly approximates \(Q_{\pi}(s, a)\) as learning occurs such that the advantage function is ultimately driven towards \(A_{\pi}(s, a) \rightarrow 0\). Lastly, policy dependent shaping inherently arises in advantage function as more optimal actions produce positive advantage values and less optimal actions produce negative advantage values. Whether the advantage function value is positive or negative is dependent on the current policy’s relationship to prior policies. Thus, sub-optimal actions can be rewarded and then punished as more optimal actions are discovered.

Given the policy dependent nature of human feedback \(h\) and the advantage function \(A_{\pi}(s, a)\), \(h\) can be substituted in for the TD error \(\delta\) as the TD error itself is an unbiased estimate of \(A_{\pi}(s, a)\). The following equality then holds,

\[
A_{\pi}(s, a) = \delta = h
\]

such that the critic is no longer needed. COACH’s modifies the actor update (Eq. (44) in B.2) as follows

\[
\theta_{t+1} = \theta_{t} + \alpha H_{t} \nabla \log \pi_{\theta_{t}}(A_{t}|S_{t}).
\]

(15)

For credit assignment, COACH aggregates multiple human feedback values similarly to TAMER’s reward aggregation. The summation allows for variability in the magnitude of feedback. However, unlike TAMER, COACH does not factor in any weighted credit assignment for the feedback delay. Instead, COACH implements multiple eligibility traces. These short-term memory traces track the policy gradient history with a decay rate \(\lambda\) and allow for prior decisions to apply towards current feedback. Multiple traces account for different temporal decay rates, which are determined either explicitly by the human trainer or implicitly based on the value of the summed feedback. Furthermore, COACH accounts for the typical human feedback delay of 0.2 to 0.8 seconds by allowing feedback to be associated with events \(d\) steps ago. Therefore, the eligibility traces can be viewed as smoothly distributing feedback to past events.

Each individual eligibility trace vector \(e_{\lambda}\) is updated for each \(d\) previous time steps,

\[
e_{\lambda} = \lambda e_{\lambda} + \nabla \log \pi_{\theta_{t}}(A_{t-d}|S_{t-d}).
\]

(16)

An eligibility trace is selected from all possible trace vectors and the aggregated human feedback \(H'_{t}\) at time step \(t\) updates the actor’s parameters as follows:

\[
\theta_{t+1} = \theta_{t} + \alpha H'_{t} e_{\lambda}.
\]

(17)

COACH performs well in a relatively complex environment that requires a human trainer to teach certain behaviors to a TurtleBot robot using the aforementioned training strategies and compositional learning, a strategy where sub-behaviors are learned and stitched together to learn a new behavior. COACH shows an increase in robustness as it is able to learn all behaviors using all training strategies. This is unlike the baseline TAMER which is unable to learn behaviors requiring compositional learning. Likewise, COACH shows an increase in aptness as it learns all behaviors in under 2 minutes. Despite the fact that COACH proves to be a viable and potentially better alternative to the TAMER framework, it remains unclear how it performs in comparison to the other variations of TAMER such as the actor-critic TAMER (ACTAMER) \([117]\) or TAMER+RL \([64, 66]\). However, just like TAMER, COACH’s overall increase in aptitude is

\(^{6}\)MacGlashan et al. \([83]\) originally denoted this idea as policy shaping. However, we adjusted the term to differentiate between their use of policy shaping and our use of policy shaping.
limited due to its linear modeling of the policy and poor generalization in more complex environments [12]. Lastly, it is unknown to what degree COACH increases aptitude when compared to traditional RL algorithms due to the lack of baseline comparisons. The remainder of this section looks at a variation of COACH which attempts further improve upon the vanilla COACH’s shortcomings.

4.2.1 Deep COACH. Arumugam et al. [12] extend COACH by creating Deep COACH which incorporate recent deep learning and deep Q-learning trends used for more complex environments with high-dimensional state spaces. Three additional components including a convolutional autoencoder, experience replay, and an entropy regularization are integrated into the framework. As in Deep TAMER, a convolutional autoencoder enables an end-to-end control, directly taking raw images as inputs and outputting actor’s probability distribution over the action space \( \pi_\theta (a|s) \).

Modifications to the replay buffer are made by storing windows of experience (i.e., trajectories or series of transitions) where upon reception of human feedback the entire window is stored. Upon storage, a single human feedback \( h \) is mapped to the last state transition in the window. The buffer is further extended by integration of importance sampling where the importance sampling with a ratio \( \rho = \frac{\pi_\theta (A_{t-1} | S_{t-1})}{\pi_{\theta_{t-1}} (A_{t-1} | S_{t-1})} \) accounts for the discrepancy between the target policy \( \pi_\theta \) at \( t \) and the behavior policy \( \pi_{\theta_{t-1}} \) at \( t-1 \). The actor’s eligibility trace (Eq. (16)) is now updated as

\[
e_{\lambda} = \lambda e_{\lambda} + \rho \nabla \log \pi_\theta (A_{t-1} | S_{t-1}),
\]

where \( e_{\lambda} \) is updated for each transition in a window of length \( L \). Once all transitions for a window have been used to update \( e_{\lambda} \), \( h \) is then applied to \( e_{\lambda} \) such that

\[
\bar{e}_{\lambda} = \bar{e}_{\lambda} + h e_{\lambda},
\]

where \( \bar{e}_{\lambda} \) is updated for every window in a minibatch of size \( m \).

To create an agent that is more responsive to human feedback and not permanently biased by inconsistencies or natural errors in human feedback, entropy regularization is utilized [35]. The entropy regularization term is defined as \( \beta \mathbb{H}(\pi(\cdot | S_t)) \) where \( \mathbb{H}(\pi(\cdot | S_t)) \) is an entropy function that takes some action probability distribution \( \pi(\cdot | S_t) \) and \( \beta \) is the regularization scaling parameter. Entropy in actor-critic networks encourages exploration such that the agent might avoid convergence to a local minimum. The average eligibility trace \( \bar{e}_{\lambda} \) and the entropy term update the actor parameter as

\[
\theta_{t+1} = \theta_t + \alpha \left[ \frac{1}{m} \bar{e}_{\lambda} + \beta \mathbb{H}(\pi_\theta (\cdot | S_t)) \right].
\]

Deep COACH is tested in the relatively complex simulated environment of Minecraft where different behaviors must be learned (e.g., navigating to a goal and walking the perimeter of a room). Deep COACH shows an increase in aptness and robustness over the baselines Deep TAMER and COACH as it achieves greater short-term and asymptotic performance for all behaviors. Deep TAMER learns to perform all behaviors at a sub-par level while COACH fails to learn some of the behaviors. Furthermore, Deep COACH also shows potential increases in scalability as it requires significantly less feedback to learn all behaviors. However, there are two caveats to Deep COACH’s potential aptitude increase. First, due to the limited testing environments, the overall increase in aptitude is unknown. Second, the degree of the aptitude increase when compared to traditional RL algorithms is not examined and thus also unknown.

4.3 Deep Reinforcement Learning from Human Preferences

Christiano et al. [24] propose to extract a latent reward function from preference-based feedback. This method can be interpreted as combining feedback modeling with reward shaping. The feedback model learns a latent numerical representation \( \hat{h} \) using qualitative preference-based feedback where \( \hat{h} \) is interpreted as the human’s latent reward.
Towards Intrinsic Interactive Reinforcement Learning

function. In order to train a RL agent, reward shaping with replacement is employed such that $\hat{h}$ replaces $r(s, a)$. Human preference feedback $h$ is given in-relation to pair-wise trajectory segments drawn from an agent’s experiences. Informally, a trajectory segment is just a short clip from an agent interacting with an environment. Formally, a trajectory segment $\sigma$ is a short sequence of observations (i.e., states) $O_t \in O$ and actions $A_t \in A$ of length $k$ such that

$$\sigma = [(O_0, A_0), (O_1, A_1),..., (O_{k-1}, A_{k-1})] \in (O, A)^k.$$  \hfill (21)

The first trajectory segment $\sigma^1$ is said to be preferred to another $\sigma^2$ if $\sigma^1 > \sigma^2$ such that

$$[(O^1_0, A^1_0), (O^1_1, A^1_1),..., (O^1_{k-1}, A^1_{k-1})] > [(O^2_0, A^2_0), (O^2_1, A^2_1),..., (O^2_{k-1}, A^2_{k-1})].$$ \hfill (22)

Given a reward function $r : O \times A \rightarrow \mathbb{R}$, the authors make the assumption that if a trajectory segment is preferred over another, the preferred trajectory segment must have a higher total reward:

$$r(O^1_0, A^1_0) + ... + r(O^1_{k-1}, A^1_{k-1}) > r(O^2_0, A^2_0) + ... + r(O^2_{k-1}, A^2_{k-1}).$$ \hfill (23)

Since the $r(s, a)$ is unknown, it is assumed that $r(s, a)$ can be approximated by modeling a latent reward function $\hat{h}$ based on human preferences $h$, leaving only a traditional RL problem left to be solved.

The estimated probability of preferring a trajectory segment over another is modeled with the softmax function of the sum of latent rewards as

$$\hat{p}(\sigma^1 > \sigma^2) = \frac{\exp \sum \hat{h}(O^1_t, A^1_t)}{\exp \sum \hat{h}(O^1_t, A^1_t) + \exp \sum \hat{h}(O^2_t, A^2_t)}.$$ \hfill (24)

Given the human feedback label $h(\cdot)$ and the estimated preference probability $\hat{p}$, $\hat{h}$ can be learned by minimizing the cross-entropy loss:

$$L(\hat{h}) = - \sum_{(\sigma^1, \sigma^2, h) \in D} h(1) \log \hat{p}(\sigma^1 > \sigma^2) + h(2) \log \hat{p}(\sigma^2 > \sigma^1).$$ \hfill (25)

$h(\cdot)$ acts as a distribution over the pairwise trajectories where the mass is centered on the preferred trajectory segment. If both trajectories are said to be preferred then distribution mass is distributed uniformly over the trajectory segments.

In the benchmark environments of in Atari and MuJoCo environments [19], empirical results are mixed as the Advantage Actor-Critic (A2C) [87] and Trust Region Policy Optimization (TRPO) [101] baselines. Thus, suggesting a decrease in both robustness and aptness. However, scalability is vastly increased as preference feedback allows for behaviors to be learned in a significantly less amount of time, reducing the number interactions need by approximately 3 orders of magnitude. Thus, whether there is a general increase in aptitude remains largely unclear as it hinges on what is considered “sufficient” performance in each environment. A further caveat here is that some environments still require a large number of preference feedback samples (e.g., 5.5k) for learning to occur. This brings about concerns surrounding how scalable the proposed algorithm is as more preference feedback can be expected to be needed as the environment complexity increases. As comparative evaluation against other IRL algorithms has not been made, its standing among the other IRL works is still to be determined.

5 INTRINSIC INTERACTIVE REINFORCEMENT LEARNING

The point of intersection between IRL and BCI lies in the intrinsic communication of human guidance, otherwise denoted as intrinsic IRL [58]. Here IRL acts as the workhorse for agent learning while a BCI device captures informative brain signals that can be used as feedback. As intrinsic feedback is simply an alternative medium for communicating feedback, we repose the critical questions presented in Section 3 from the perspective of intrinsic feedback with
the addition of one new question. Throughout this section we attempt to readdress the following critical questions concerning the viability of intrinsic IRL:

(i) What are the motivations for pursuing intrinsic IRL?
(ii) How can intrinsic feedback $h$ be integrated into an RL algorithm such that an agent learns a good policy $\pi$?
(iii) Do intrinsic IRL works fulfill the motivations of IRL and intrinsic IRL?
(iv) What brain signals can be decoded from neural activity and used for guidance?

Section 5.1 further builds on the answer to critical question (5.i) given by Section 3.1 but now focuses on intrinsic feedback motivations. As intrinsic feedback is formulated as feedback, the answer to critical question (5.ii) is inherited from IRL. Sections 5.2 and 5.3 attempt to address critical question (5.iv) by exploring brain signals that have seen to use feedback. Lastly, Section 5.4 aims to address critical questions (5.iii) while also further addressing (5.ii) and (5.iv) by covering current approaches to intrinsic IRL.

5.1 Motivations for Intrinsic Feedback

The first motivation of intrinsic IRL arises from the intrinsic quality of brain signals in which neural activity is elicited automatically. That is, humans are free to naturally perform or observe a task without the concern of having to explicitly provide feedback. One can think of this as, for the lack of a better word, “unconsciously” eliciting guidance where the line between what is considered unconscious and conscious can be difficult to define [121]. Automatic elicitation comes with the potential of providing feedback at a faster and more consistent rate than that of explicitly elicited feedback [82, 127].

The second motivation, building further on the first motivation, arises from the fact that feedback can be provided not only while observing a task (Fig. 5a) but also while performing a task (Fig. 5b). The importance of the latter comes from an issue in IRL where it is extremely difficult to have a human perform a task while also having the human provide feedback to an agent concerning their own actions. Traditional IRL research has sought to sequentially combine learning from demonstrations and feedback such that the agent learns from a set of demonstrations and then the human provides feedback in regards to the agent’s behaviors [46]. Intrinsic IRL allows for the parallel combination of demonstrations and feedback such that humans can perform a task while providing feedback regarding their own actions. At a high level, this can be seen as a form of automatic labeling of data where the human labels their demonstrations with their intrinsic feedback, providing two interconnected sources of human guidance. To our best knowledge, the implications of having two interconnected sources of human guidance remains largely unexplored (see Section 6.6).

The third motivation comes from the vast pool of latent information contained within the brain which is the largest advantage and potentially the largest disadvantage of intrinsic IRL. This disadvantage arises due to the combination of many latent brain signals, giving rise to the famous signal processing problem referred to as the Cocktail Party Problem. Whether this advantage turns out to be a real advantage or not remains to be seen. However, our ability to identify and extract information from neural activity has been progressing over the past decade [25, 80].

5.2 Error-related Potentials as Feedback

Event-related potentials (ERPs) can be viewed as a form of short-term feedback that allows for fine-tuned feedback to be applied event-by-event [20]. In particular, ErrPs are of interest as they indicate a human has encountered an...
erroneous event. Specifically, ErrPs have seen frequent use in judging the correctness or goodness of an event or behavior [22, 28, 29, 32, 33, 47–49, 58, 78, 82, 97, 106, 127].

For instance, ErrPs have been classically used as a feedback mechanism to help correct erroneous responses made by mental typewriter applications when a wrong letter or symbol is selected by the interface [99]. Further, ErrPs have been used in correcting agent actions. Salazar-Gomez et al. [97] utilize the detection of ErrPs to correct a robot’s action when it chooses a wrong action. Likewise, Chavarriaga and Millán [22] use ErrPs to teach an agent a naive policy for a simple goal search task. When the agent makes the incorrect choice of moving away from the goal and an ErrP is detected, the probability of taking the action is decreased, otherwise the probability is increased.

Due to the use of ErrPs in judging the correctness of an action, it is not unreasonable to treat ErrPs as a proxy to a human’s subjective ground truth [58]. That is, the presence of an ErrP can be interpreted as an event being wrong. For instance, for quantitative feedback a negative (e.g., -1) signal could be provided to an agent while for qualitative feedback the categorical variable for “wrong” or “bad” could be provided to an agent [28, 48, 58]. Research on ErrPs has even attempted to detect the severity of an erroneous event, potentially providing more fine-tuned information concerning how wrong an event was [106]. Meanwhile, the absence of an ErrP can be interpreted in one of two ways: the absence of feedback or correct feedback. Firstly, the absence of an ErrP can be interpreted simply as the absence of feedback. Secondly, under an assumption similar to that of the legal principle of presumption of innocence, the absence of an ErrP can indicate what a human considers correct. For example, a quantitative positive feedback (e.g., +1) could be provided to an agent while for qualitative feedback the categorical variable for “correct” or “good” could be provided to an agent [22, 28, 47, 48, 58, 127].

Given ErrPs can be treated as an approximation to a human’s subjective ground truth, accurate detection of ErrPs is desirable as this determines the accuracy of the feedback. Poor detection of ErrPs means that inaccurate feedback will then be communicated to an agent, leading to either slowed or little learning. However, the issue of misclassification of feedback can be alleviated through the pairing of RL as it has been shown to learn even with noisy reinforcement signals [4, 48, 58, 119, 127]. Furthermore, ErrP classification has also been a long standing area of research and is always improving due to the wider need for high classification accuracies in almost all BCI applications [3, 29, 32, 33, 49, 53, 55, 56, 78, 80, 106].

5.3 Alternative Brain Signals for Feedback

Alternative brain signals such as the P3 signal or cognitive and affective states, although less frequently used, have potential to be utilized as intrinsic feedback as well. Since the use of multiple brain signals as feedback is desirable to
allow for a wider range of feedback to be given, it can be better to treat these alternate brain signals as complements to
one another rather than replacements.

The P3 event-related potential (ERP) component can be seen as a form of positive feedback as Wirth et al. [124] show
that it is possible to discriminate between a simulated robot moving towards a goal and reaching the goal using the
P3 response. This discrimination between P3 variations allows for the potential to elicit differing magnitudes correct
feedback. Wirth et al. [123] extend this idea of multi-way classification to encompass both P3 and ERN such that varying
levels of positive and negative feedback can be detected. Feedback is elicited in response to the agent’s actions of
moving towards the goal, reaching the goal, moving away from the goal, and moving off the goal. Here moving on and
off the goal elicit the greatest magnitude of either positive or negative feedback. Although classification accuracy is
quite low (slightly above levels of chance, i.e., > 25%), this work presents a promising future direction of research not
only for positive feedback that can be intrinsically elicited and not assumed but also for detecting the magnitude of
intrinsic feedback.

Chakraborti et al. [20] run a preliminary experiment that utilizes cognitive and affective states as a form of medium
to long-term feedback. The goal of this work is to utilize the detection of excitement and stress as positive and negative
rewards. Whenever the robot attempts to perform undesirable actions, the human’s stress levels increase. Likewise, as
the robot succeeds at the task the human’s excitement levels increase. The authors use reward shaping where \( r(s, a) \)
is augmented by the human intrinsic feedback \( h \). The shaped reward signal \( r'(s, a) \) is fed to a pre-trained Q-learning
algorithm. A collaborative environment where the robot and a human build a tower together is used for evaluation.
Here, the robot must learn not to use a particular block that the human has picked out within their mind and stress
is used to convey which block the robot should not use. The preliminary experiments show the robot explores new
policies which attempt to account for the human selected block although issues surrounding convergence are reported
as well.

While research surrounding alternative brain signals for feedback is less comprehensive in comparison to that of ErrP
research, it does present an intriguing area for future research that could allow for more avenues of intrinsically elicited
feedback. Expanding on types of signals that can be used for positive feedback is particularly needed as currently there
are limited alternatives for detecting short-term positive feedback outside of using the absence of ErrPs.

5.4 Current Approaches

Due to intrinsic IRL’s recent emergence, only a handful of works of have been produced so far [4, 28, 47, 48, 57, 58, 82,
119, 127]. To gain a better understanding of the approaches taken so far, we breakdown the intrinsic IRL literature into
three main approaches with respect to the how IRL is utilized: proof-of-concept, adaptive, and explorative. The goal
of most proof-of-concept works is to test the combination of BCI and IRL in different simulated and real-world toy
environments. Adaptive approaches aim to adapt existing IRL works to function with intrinsic feedback. Explorative
approaches aim to explore intrinsic feedback while also progressing the IRL field forward by proposing new or iterative
improvements to algorithms. Often, these explorative approaches introduce new ideas that aim to compensate for
intrinsic feedback’s inherently noisy signals. As before, we provide an brief analysis of each work from the perspective
of the RL challenges with a focus on aptitude. Table 4 contains the summative comparison of these intrinsic IRL works.

5.4.1 Proof of Concept. The majority of the prior intrinsic IRL works follow a naive approach where the goal is
to simply test if the combination of BCI and RL\(^{11}\) allows agents to learn in different simulated and real-world toy

\(^{11}\)This combination seems to have been done with no knowledge of IRL as a field.
environments. These works are often characterized by their use of reward shaping with replacement and use of fairly rudimentary RL algorithms (e.g., vanilla Q-learning) [21, 28, 29, 47, 48, 57, 58]. Due to the lack of proper baseline comparisons for many of these works, an analysis with regards to the RL challenges is not available.

For example, Iturrate et al. [48] utilize Q-learning in a discrete action space task where a simulated robot arm learns to select the central object given a line of five objects. If the robot selects the wrong object and an ErrP is detected, \( h = \{-0.5, -1\} \) depending on the severity of the error. If the robot performs the correct action and no ErrP is detected, \( h = +1 \). Using Q-learning and reward shaping with replacement (i.e., \( h \) is used to directly replace the \( r(s, a) \)), the robot eventually learns the correct policy of grabbing the central object.

Similarly, Kim et al. [57, 58] utilize ErrPs to teach a robotic arm to perform a predefined action when a specific hand gesture known only to the human trainer is performed. In the experiment, the robot has no prior knowledge of which predefined action belongs to which gesture, only the human trainers know the correct mapping. Thus, the robot must explore each action until the correct mapping is learned by using ErrPs which are elicited in response to the robot performing wrong actions. When an ErrP is detected due to either an incorrect action or misclassification, \( h = 0 \), otherwise \( h = +1 \). Training is done using Linear Upper Confidence Bound (LinUCB) and reward shaping with replacement until the robot converges to the human’s selected gesture-to-action policy.

Lastly, Ehrlich and Cheng [28] have a robot and human collaborate to find a common policy for selecting an object. The robot is in charge of selecting one of three objects and the human must figure out which object the robot has selected. To help the human, the robot must learn a gazing policy that maps the object it has selected to the object it is currently gazing at. If the robot learns a good gazing policy, the robot can communicate to the human which object it has selected. When the human makes a guess concerning which object the robot has selected, an interface indicates to the human whether they are right or wrong. If the human is right and no ErrP is detected, \( h = +1 \). Otherwise, if the human was wrong and an ErrP is detected, \( h = -1 \). The robot learns a gazing policy through a naive formulation of policy gradient where \( h \) is used to directly replace the \( r(s, a) \).

### 5.4.2 Adaptive

Adaptive approaches are less commonly found as they simply adapt an existing IRL work to use intrinsic feedback. Adaptive approaches tend to be less robust as most traditional IRL algorithms do not compensate for noisy feedback signals. However, adaptive approaches provide useful comparisons between intrinsic feedback and other types of feedback mediums (i.e., explicitly or implicitly conveyed feedback).

Luo et al. [82] propose adapting the preference based feedback algorithm proposed by Christiano et al. [24] with intrinsic feedback. The authors do so by using ErrPs as intrinsic feedback for determining a subject’s preference between two trajectory segments. The preference selection processes is adapted to work with intrinsic feedback such that when two trajectory segments \( \sigma^1 / \sigma^2 \) are presented, a subject must determine within their mind which trajectory segment is preferred. After observing the two segments, the interface randomly selects one. If the segment selected does not match the segment selected by the human, an ErrP is expected to be elicited. Likewise, no ErrP is expected to be elicited when there is a match. Once preference is given through the intrinsic feedback \( h \), the latent reward function \( \hat{h} \) is modeled and fed to a RL algorithm for learning as originally proposed by Christiano et al. [24].

When comparing the empirical results of intrinsic feedback against the explicitly elicited feedback using a subset of the MuJoCo testing environments, the authors find that the intrinsic feedback for all subjects shows a decrease in robustness and aptness as short-term and asymptotic performance is slightly worse. Yet, these results are still surprising as ErrP classification accuracy of 67%, meaning the reward signal is very noisy. This seems to suggest that improved accuracy could allow intrinsic feedback to perform either on par or even better than the explicit feedback. Furthermore,
scalability is suggested to be further increased as it is shown that intrinsic feedback can be captured at a faster rate than explicit feedback.

5.4.3 Explorative. An increasingly more common approach for integrating intrinsic feedback into IRL is through proposing new algorithms or iterating on previous IRL ideas. While these explorative intrinsic IRL approaches often focus on compensating for the inherently noisy properties of intrinsic feedback, they can also be adapted to use different feedback mediums. Thus, algorithms developed using intrinsic feedback can have a wider impact on the entirety of the IRL field.

Xu et al. [127] propose an IRL algorithm that aims to learn and extract a latent numerical representation from the intrinsic feedback using soft Q-learning [40]. By using qualitative feedback communicated via ErrPs, the algorithm aims to model the latent human Q-function $\hat{h}_g = Q_h$ where $\hat{h}$ is parametrized by $\theta$. Using $\hat{h}$, a soft human policy $\pi_h$ and reward function $r_h(s, a)$ are further derived. Reward shaping is finally employed where $r_h(s, a)$ is used to perturb $r(s, a)$ and the augmented reward $r'(s, a)$ is feed to any desired RL algorithm for learning.

The modeling of the soft human Q-values $\hat{h}$ is done to increase the overall resilience of the algorithm to noise in the feedback signal which is caused by misclassification of ErrPs. The resilience of soft Q-learning algorithm inherently emerges from the entropy term $\mathbb{H}(\pi(\cdot|s_t))$ which is factored directly into the reward maximization formulation. Maximization of both the future reward and uncertainty helps to encourage exploration and prevent convergence to poor local minimums that can arise due to the misclassification of intrinsic feedback $h$ [40, 127].

To estimate $\hat{h}$, ErrPs are formulated as qualitative feedback such that the detection of an ErrP indicates an agent’s behavior is either “correct” or “incorrect.” Feedback is given based on generated trajectories but can also be given for every state-action pair where the former puts less burden on the human. It is then possible to model $\hat{h}$ using neural networks by minimizing the cross-entropy using samples drawn from a replay buffer as follows:

$$L(\theta) = - \sum_{(s,a) \in D} (1 - \hat{h}) \log \pi_h(a|s) + h \log(1 - \pi_h(a|s)).$$  \hspace{1cm} (26)

where the predictions are the soft policy likelihoods of the human either taking an action $\pi_h(a|s)$ or not taking an action $1 - \pi_h(a|s)$ and the labels $h \in \{0, 1\}$ correspond to whether an ErrP was detected $h = 1$ or not $h = 0$. The human policy $\pi_h(a|s)$ is derived by using the soft policy formulation given in [40]:

$$\pi_h(a|s) = \exp [(\hat{h}(s,a) - V_h(s))/\alpha],$$

$$V_h(s) = \alpha \log \sum_a \exp(\hat{h}(s,a)/\alpha).$$  \hspace{1cm} (27)

Using Bellman equation, the human reward function $r_h(s, a)$ can be approximated by the difference between discounted next state-action value and current estimated value as follows:

$$r_h(s, a) = \gamma \max \hat{h}(s', a') - \hat{h}(s, a).$$  \hspace{1cm} (28)

Due to the limited number of labeled ErrP examples, the authors lower the variance of estimations by adding a learned baseline $t(s)$:

$$r_h(s, a) = \gamma \max[\hat{h}(s', a') + t(s')] - (\hat{h}(s, a) + t(s)).$$  \hspace{1cm} (29)

This baseline $t(s)$ is trained on both labeled and unlabeled data which allows for more information about the state transition information to be incorporated into $r_h(s, a)$. Finally, $r_h(s, a)$ is used to perturb $r(s, a)$ to create an augmented reward function

$$r'(s, a) = r(s, a) + \beta r_h(s, a),$$  \hspace{1cm} (30)

where $r'(s, a)$ is then passed to a RL algorithm for learning.
Towards Intrinsic Interactive Reinforcement Learning

For all subjects, empirical results show an increase in aptness as the algorithm converges to the optimal behavior more quickly than that of the Bayesian DQN baseline in grid world and an Atari-like environments. Scalability is likely to be increased as well due to the use of the environmental reward, the ability to use unlabeled data, and the ability to only provide feedback on certain trajectories. Furthermore, the authors compare their algorithm with a adaptive implementation of a newer IRL algorithm called FRESH [126] where similar increase in aptness and scalability can be observed. Due to the simplistic testing environments, all algorithms eventually converge to optimal performance making any increases in robustness harder to gleam. These results are made more promising as the average classification accuracy was only 73%. The overall increase in aptitude remains unknown as testing in more complex environments is needed.

Akinola et al. [4] propose an algorithm that takes a modeling approach that aims to model the human action probabilities from qualitative feedback. The integration of the modeled feedback $\hat{h}$ is achieved through policy shaping with control sharing. The proposed algorithm uses intrinsic feedback $h$ communicated via ErrPs to model the latent human policy $h = \pi_h(a|s)$. As $h$ is formulated qualitatively, $\hat{h}(s)$ outputs a probability distribution over the possible actions where the action with the highest predicted probability corresponds to the action the human is most likely to consider "good" or "correct."

A human policy $\pi_h$ is learned by modeling $\hat{h}$ while the human observes an agent interacting with the environment. Modeling of $\hat{h}$ is done by minimizing the cross-entropy loss for the samples drawn from a replay buffer. The policy $\pi_h$ of the agent is derived from $\hat{h}$ such that

$$\pi_h(s) = \arg \max_a \hat{h}(s,a). \quad (31)$$

Once $\hat{h}$ is learned, any deep RL algorithm with access to a very simple sparse reward can be used for learning the policy $\pi_{RL}$. Shaping is done through control sharing where the RL algorithm’s policy $\pi_{RL}$ is replaced by $\pi_h$ for an episode given some probability $\epsilon$ that is slowly annealed overtime.

The empirical results results from testing in a simulated 3D robotic navigation environment are mixed due to the performance being dependent on the ErrP classification accuracy for each subject. Accuracies between 60-67% showed an increase in both aptness and robustness as short-term and asymptotic performance was improved over the proximal policy optimization (PPO) baseline [102]. However, when classification accuracies are less than 60% learning did not occur. Regardless of classification accuracy, scalability is likely to be slightly increased as the algorithm can utilize a sparse or dense reward function to even when feedback is no longer available. Lack of further testing environments leaves the overall increase in aptitude unknown.

Wang et al. [119] build upon Akinola et al. [4] by adding new components that aims to further increase the aptitude of the algorithm. In particular, uncertainty is introduced to inherently address the noisy feedback signals provided by the intrinsic feedback. Thus, instead of only modeling $\hat{h}$ as the estimation of probability of actions, the proposed algorithm uses Evidential Deep Learning [103] to model $\hat{h}$ as a distribution over all such action probabilities. Meaning, the confidence of an action being considered "good" or "correct" can be measured.

Uncertainty enables active learning and a purified replay buffer to be integrated when learning $\pi_h$. Active learning allows the agent to only query for intrinsic feedback when it is not confident in the action it took such that feedback is more informative. Meanwhile, the purified replay buffer stores intrinsic feedback and state tuples only if the confidence of $\hat{h}(S_t, A_t)$ is greater than a pre-defined threshold $\epsilon$. This thresholding helps to regulate the noise in feedback such that only consistent feedback is stored. Training batches are then sampled from both the purified and basic replay

---

Two out of the seven subjects had less than 60% accuracy.
buffers. Further, $\pi_{RL}$ is further refined by using imitation learning which helps the RL algorithm to reproduce good behaviors previously performed by either $\pi_h$ or $\pi_{RL}$.

Using the same test environment from Akinola et al. [4] and a simulated robotic reaching task, empirical results results for the algorithmic changes showed larger improvements in both aptness and robustness over the PPO baseline [102] for all subjects. With the additional algorithmic changes, even classification accuracy between 55 – 60% are able to learn and show these same increases in aptness and robustness. Additionally, scalability is likely to be further increased due to active learning reducing the amount of feedback requires as the agent requests feedback only when it is uncertain.

6 Challenges and Open Problems

We discuss potential challenges and open problems for intrinsic IRL from the perspective of further establishing viability. Many of the challenges presented in this section go beyond intrinsic IRL as challenges are often inherited from the parent fields of IRL, RL, and BCI. Thus, challenges of parent fields will be included as research progression in intrinsic IRL can have implications beyond itself, affecting and being affected by research in any of the parent fields.

6.1 Alleviation of Aptitude and Alignment Challenges

Initial IRL works have shown glimpse into the potential of IRL in alleviating issues of aptitude and alignment. As alleviating such issues is a fundamental aspect of IRL, continued progress requires overcoming aptitude and alignment challenges. Nevertheless, there are still various difficulties in assessing the extent to which IRL algorithms have achieved an increase in aptitude and, even more so, alignment that need to be addressed.

First, is the lack of universal and widely accepted test environments. Many IRL works tend to use different and custom made environments where the complexity of the environments often lags far behind common benchmarks used by the RL community such as Atari or MuJoCo [19]. Even when the IRL community utilizes more widely accepted benchmarks, the breadth of testing is often limited to a single environment. This makes gleaming the "true" or overall increase in aptitude and alignment of an IRL algorithm difficult if not impossible. Thus, as done by Christiano et al. [24], adopting a variety of commonly accepted benchmarks such as the Atari 2600 suite of games can allow for the viability of IRL to be more clearly gleamed.

Second, is the lack of baseline comparisons against state-of-the-art RL algorithms [12, 83] and other IRL algorithms [24]. As increases of aptitude and alignment RL algorithms is one of the goals of IRL algorithms, proper comparisons are needed for such increases to even begin to be measured. Moveover, while IRL works tend to compare performance against at least one other IRL algorithm, it remains unclear what the current SOTA algorithm for IRL even is as no wider comparison between IRL algorithms has been conducted. The lack of comparisons to other IRL algorithms is not helped by lack reproducibility. This issue is largely due to many of the aforementioned IRL works lacking any form of open-source code [4, 8, 12, 63, 83, 119, 120, 127]. While many of these works do at least provide pseudo-code, this still leaves the replication of results and further exploration of these algorithms in an unnecessarily difficult state.

Third, is the lack of quantification measures for aptitude and alignment improvements over baselines. For the most part, aptitude gains can be observed through performance measure increases such as improved learning curves, total returns, or task completion rates. However, assessment of performance gains are typically done through mere observation rather than from a quantified measurement such as performing statistical significance tests [41]. Additionally,
it is even more difficult to assess alignment improvements simply by observing an increase in performance measures as misaligned agents can show an increase in performance measures even though they might not accomplish such an increase in an attended manner. Therefore, assessment of alignment alleviation often requires finer inspection of the agent’s behaviors and more specified evaluation metrics.

6.2 Credit Assignment

In IRL, the credit assignment problem arises when assigning delayed human feedback to contributing events [13]. While IRL inherently minimizes the original RL credit assignment problem produced by sparse reward, the problem is not fully alleviated as even human feedback contains delays. Although prior IRL works have attempted to address the credit assignment problem for human feedback, it still remains unclear which methods produce the best results or how crucial solving the problem actually is.

For instance, Knox and Stone [63] attempt to solve the credit assignment by estimating the delay between feedback and the corresponding events using a probability density function (PDF) [61, 63]. However, Warnell et al. [120] show that the selection of a PDF matters as changing the PDF can produce widely varying results. This entails that if the assumed credit assignment PDF does not match the human’s delay distribution, performance can be greatly affected. Likewise, MacGlashan et al. [83] propose using eligibility traces with reward aggregation where feedback is assigned to events that occurred between 0.2 and 0.8 seconds ago and summed. Although the approach sounds reasonable, how much it actually alleviates the credit assignment problem is unclear.

In regards to intrinsic IRL specifically, it could further reduce the credit assignment problem as, once elicited, ERP durations tend to be short and consistent [30, 81]. For instance, ErrPs typically only require a 1 second window, starting when the event occurs, for the signal to be fully captured where even shorter windows have been used [23, 58]. However, this idea only applies to once an event elicits an ERP. Take for instance real-world problems which often contain complex behaviors that require an accumulation of evidence before feedback can be provided. In these cases intrinsic feedback can still suffer from a delay in elicitation due to cognitive processing, much like its explicate feedback counter-part. Thus, even intrinsic feedback like ErrPs can only be elicited once an event has been understood as erroneous [15]. To what extent intrinsic and explicit feedback differ in this regard still remains unexplored.

A further potential downside to intrinsic feedback could be the back-to-back elicitation of one or more brain signals of interest such that the signals overlap [81, 125]. This overlapping situation has not been fully explored and could lead to further degradation in classification. This could constrain intrinsic IRL’s abilities to capture feedback elicited in quick succession.

Finally, a new potential issue that is not dealt with in the classical RL settings is future credit assignment. This problem arises when anticipatory feedback is given needs to be mapped to a future foreseen event [110]. This potential problem remains largely unexplored by the IRL community while the BCI community has begun to investigate a very similar problem specific to ErrPs referred to error expectation. This problem entails detecting and utilizing ErrPs that are elicited in response to foreseen erroneous events [15, 50].

6.3 Sample Inefficiency

While IRL has been shown to increase the rate at which learning occurs [12, 24, 61, 63, 83, 120], new challenges surrounding the efficient use of human feedback arise. Collecting human feedback is expensive, thus the amount of human feedback that can be collected is limited. Under the assumption that as the complexity of the environment increases, more feedback will be required for learning to occur; the challenge becomes efficiently learning from limited
human feedback even as complexity scales up. Prior IRL works have been able to learn from limited human feedback, yet it remains unclear how much feedback will be required as the complexity of the environments scales.

A particular sample inefficiency challenge arises when modeling feedback. A goal of modeling is to alleviate the amount of feedback that needs to be provided to the agent [72] such that the agent can learn from the modeled feedback on its own without human supervision. Thus, this leaves not only the question of how much feedback is required to train an agent as the complexity scales up, but how much feedback is required to learn a good feedback model as the complexity scales up. Ironically, developing a model could be more expensive than using the raw feedback as modeling needs to overcome the policy dependent and non-stationary nature of human feedback [83]. In essences, the true benefits and trade-offs for developing feedback models requires a much more in-depth investigation.

While intrinsic IRL might not inherently increase sample efficiency, one benefit of intrinsic IRL comes from the increased number of human feedback samples that can be captured. Luo et al. [82] suggest that more intrinsic feedback can be captured in a less amount of time than that of explicit feedback while Xu et al. [127] claim intrinsic feedback can be captured more accurately when extremely fast feedback rates are required. This raises the question concerning the importance of quantity verse quality.

6.4 Overcoming Non-stationary Brain Signal Distributions

One defining property of brain signals is that they follow a non-stationary distribution. Therefore, the distribution of brain signal can shift, causing the performance of the classification model to degrade [60, 98, 105]. This non-stationarity is largely due to constant changes in the underlying noise that is combined with the brain signal of interest. As fully separating a brain signal from the surrounding biological and external noise is not currently possible, a brain signal’s distribution is tightly coupled to the accompanying noise. Changes in cognitive and affective processes, biological artifacts, and external noise all play a role in influencing a signals distribution [85, 98]. Thus, the sensitivity to noise can shift a signal’s distributions even only after a short period of time (i.e., from trial-to-trial) [17]. The non-stationarity of brain signal distributions is only further pronounced when comparing distributions between humans as individual characteristics play a role in influencing distributions [36, 43].

A very clear example of this non-stationary distribution is found in ErrPs. In part, ErrP variations have been influenced by different cognitive processes that are elicited due to different tasks [3, 29, 49], changes in the complexity of a task [53], or changes in the way the a task is interacted with [55, 56]. For example, if an error occurs either while performing a task or while observing the same task being performed, two slightly different ErrPs are elicited [3, 55, 56]. Overcoming this challenge would lead to an improvement in the classification accuracy such that intrinsic feedback would be more accurately conveyed to agents [48, 56, 82, 127]. However, it is vastly unclear to what degree agent performance scales with classification accuracy as some works report successful learning with classification accuracies as low as 55% [119].

Nevertheless, non-stationarity has been a long on-going challenge faced by practically all BCI applications, making it an active area of research today. For example, transfer learning has been adopted to deal with these distribution shifts by using data from a source domain to either find latent features or learn parameters that capture the invariant underlying signal structure such that they can be transferred to some desired target domain [3, 29, 49, 55, 56]. The source domain serves as the initial signal distribution and the target domain serves as the shifted signal distribution. Alternatively, adaptive classification where classifiers try to continuously learn from newly collected data in order to account for the shifts in signal distribution as they occur has been applied as well [85, 105].
6.5 Integrating Multiple Types of Feedback and Mediums

Prior IRL quantitative feedback works frequently use both positive and negative feedback to provide more information to the agent [12, 61, 83, 120]. Whether providing both positive and negative rewards is necessary is still largely undetermined. There is an argument to be made for excluding positive feedback to prevent alignment challenges (e.g., reward hacking or positive circuits) [61, 64, 66]. At the same time removing positive feedback limits the human’s degrees of freedom as human trainers tend to utilize both positive and negative feedback in different ways [12, 83]. Additionally, human feedback tends to be biased towards positive feedback [12, 61, 64, 66, 111]. Thus, excluding positive feedback may reduce the amount of information available to the agent making complex tasks with more abstract goals harder to learn, although this requires further investigation. Furthermore, it remains to be seen if these potential quantitative restrictions apply to qualitative feedback as most qualitative works model a latent numerical representation that can take on either positive or negative values [24, 127].

Due to the majority of adaptive and explorative intrinsic IRL works interpreting intrinsic feedback as qualitative feedback, it is unclear if a brain signal representation of positive feedback is even needed [4, 82, 119, 127]. Regardless, intrinsic IRL works tend to be limited to a singular brain signal. Thus, integration of multi-signal feedback could prove beneficial in increasing the user’s degrees of freedom when providing intrinsic feedback. For instance, P3 could be used as a form of positive feedback while affective states could be used for eliciting long-term positive or negative feedback [20, 124]. The caveat of multi-signal feedback is that it inherently introduces multi-class classification which has already proven to be difficult for signal classification [123].

Finally, further investigation into the benefits and disadvantages between different mediums for communicating feedback could be useful as recent intrinsic IRL works have begun to show differences between explicit and intrinsic feedback. For example, intrinsic feedback has been shown to be captured at a much quicker rate [82] while the accuracy of explicit feedback has been shown to deteriorate under excessive time pressures [127]. More in-depth and thorough comparisons are needed to ensure these trends can be replicated in diverse settings. Exploration into performance comparisons between the different types of feedback mediums as well as further investigation into how best to use each type of medium or some of them collectively could prove to be beneficial directions of future research.

6.6 Integrating Multiple Types of Guidance

Given intrinsic feedback is just another medium for providing feedback, it could be beneficial to utilize other types of guidance to further increase the amount of information provided to the agent. For example, gaze tracking provides implicit guidance to the agent as a human’s gaze conveys latent information about what and where the agent should focus its attention [131]. Thus, combining gaze tracking and intrinsic feedback could help focus the agent on the aspects of the environment that are related to the human’s intrinsic feedback, thereby further accelerating learning.

Likewise, combining intrinsic feedback and demonstrations could lead to a further increase aptitude and alignment. The most naive method for doing so is the through sequential integration of algorithms where learning from expert demonstrations is performed first, followed by the fine-tuning of the agent’s behaviors using feedback. In fact, this has already begun to be explored using explicit feedback [9, 46, 74]. Moreover, parallel integration could be done where subject demonstrates the desired behaviors while providing feedback concerning their actions (Fig. 5b). This has traditionally not been possible when using explicit feedback as it is difficult to demonstrate and provide explicit feedback at the same time. In relation to this idea of parallel integration, a recent intrinsic IRL work by Batzianoulis et al. [15] explored a alternative method for combining both intrinsic feedback with demonstration by using feedback to
label trajectories where the label acts as a weight that estimates the optimality of a trajectory. These labeled trajectories are then used to learn a reward function using inverse learning. Further exploration into how combine these two interconnected sources of human guidance is likely to be a promising future direction of IRL research.

7 CONCLUSION

To facilitate a deeper integration of IRL and BCI, this paper has presented the first comprehensive tutorial and review of intrinsic IRL from the perspective that brain signals are simply an alternative medium for communicating feedback when using feedback-driven IRL. To further expand RL algorithms to more diverse real-world applications, IRL aims to help alleviate aptitude and alignment challenges faced by RL algorithms in increasingly more complex environments. Intrinsic IRL aims to further build on top of all of the alleviation of aptitude and alignment challenges by providing access to the vast pool of latent information contained within the brain and allowing for feedback to be automatically communicated. Although the overall degree to which intrinsic IRL, and IRL in general, increases an algorithm’s aptitude and alignment remains unclear, initial works do show that progress is achievable. Thus, further research into establishing not only intrinsic IRL’s viability but IRL’s viability is still needed to ensure the longevity and practical applicability of both fields.

ACKNOWLEDGMENTS

The authors would like to especially thank José del R. Millán, Brad Knox, and Peter Stone for taking the time to provide valuable feedback.

REFERENCES

[1] Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In Proceedings of the Twenty-first International Conference on Machine Learning (Banff, Alberta, Canada) (ICML ‘04). ACM Press, New York, NY, USA, 1.
[2] David Abel, John Salvatier, Andreas Stuhlmüller, and Owain Evans. 2017. Agent-Agnostic Human-in-the-Loop Reinforcement Learning. http://arxiv.org/abs/1701.04079
[3] Mohammad Abu-Alqumsan, Christoph Kapeller, Christoph Hintermüller, Christoph Guger, and Angelika Peer. 2017. Invariance and variability in interaction error-related potentials and their consequences for classification. Journal of Neural Engineering 14, 6 (Nov. 2017), 066015. https://doi.org/10.1088/1741-2552/aa8416
[4] Iretiayo Akinola, Zizhao Wang, Junyao Shi, Xiaomin He, Pawan Laphorissuth, Jingxi Xu, David Watkins-Valls, Paul Sajda, and Peter Allen. 2020. Accelerated Robot Learning via Human Brain Signals. http://arxiv.org/abs/1910.06682
[5] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litiwn, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. 2019. Solving Rubik’s Cube with a Robot Hand. http://arxiv.org/abs/1910.07113
[6] Abeer Alnafjan, Manar Hosny, Yousef Al-Ohali, and Areej Al-Walib. 2017. Review and Classification of Emotion Recognition Based on EEG Brain-Computer Interface System Research: A Systematic Review. Applied Sciences 7 (Nov. 2017), 1239. https://doi.org/10.3390/app7121239
[7] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete Problems in AI Safety. http://arxiv.org/abs/1606.06565
[8] Riku Arakawa, Susuke Kobayashi, Yuya Unno, Yuta Tsuboi, and Shin-ichi Maeda. 2018. DQN-TAMER: Human-in-the-Loop Reinforcement Learning with Intractable Feedback. http://arxiv.org/abs/1810.11748
[9] Brenna Argall, Brett Browning, and Manuela Veloso. 2007. Learning by demonstration with critique from a human teacher. In Proceedings of the ACM/IEEE international conference on Human-robot interaction (HRI ’07). Association for Computing Machinery, New York, NY, USA, 57–64. https://doi.org/10.1145/1228716.1228725
[10] Pietro Arizò, Gianluca Borghini, Gianluca Di Flumeri, Nicolò Scairola, and Fabio Babiloni. 2018. Passive BCI beyond the lab: current trends and future directions. Physiological Measurement 39, 8 (Aug. 2018), 08TR02. https://doi.org/10.1088/1361-6579/aad57e
[11] Saurabh Arora and Prashant Doshi. 2020. A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress. http://arxiv.org/abs/1806.06877
[12] Dilip Arumugam, Jun Ki Lee, Sophie Saskin, and Michael L. Littman. 2019. Deep Reinforcement Learning from Policy-Dependent Human Feedback. http://arxiv.org/abs/1902.04257
Towards Intrinsic Interactive Reinforcement Learning

[63] W. Bradley Knox and Peter Stone. 2009. Interactively shaping agents via human reinforcement: the TAMER framework. In *Proceedings of the fifth international conference on Knowledge capture (K-CAP ’09)*. Association for Computing Machinery, Redondo Beach, California, USA, 9–16. https://doi.org/10.1145/1597735.1597738

[64] W. Bradley Knox and Peter Stone. 2010. 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 (Toronto, Canada) (AAMAS ’10)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 5–12.

[65] W. Bradley Knox and Peter Stone. 2012. Reinforcement learning from human reward: Discounting in episodic tasks. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN ’12)*. IEEE, 878–885. https://doi.org/10.1109/ROMAN.2012.6343862

[66] W. Bradley Knox and Peter Stone. 2012. Reinforcement learning from simultaneous human and MDP reward. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1 (AAMAS ’12)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 475–482.

[67] W. Bradley Knox and Peter Stone. 2013. Learning non-myopically from human-generated reward. In *Proceedings of the 2013 international conference on Intelligent user interfaces (II ’13)*. Association for Computing Machinery, New York, NY, USA, 191–202. https://doi.org/10.1145/2449396.2449422

[68] W. Bradley Knox and Peter Stone. 2015. Framing reinforcement learning from human reward: Reward positivity, temporal discounting, episodicity, and performance. *Artificial Intelligence* 225 (Aug. 2015), 24–50. https://doi.org/10.1016/j.artint.2015.03.009

[69] W. Bradley Knox, Peter Stone, and Cynthia Breazeal. 2013. Training a Robot via Human Feedback: A Case Study. In *Social Robotics*. Springer International Publishing, Cham, 460–470.

[70] Nataliya Kosmyna, Franck Tarpin-Bernard, Nicolas Bonmefond, and Bertrand Rivet. 2016. Feasibility of BCI Control in a Realistic Smart Home Environment. *Frontiers in Human Neuroscience* 10 (Aug. 2016), 416. https://doi.org/10.3389/fnhum.2016.00416

[71] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (May 2015), 436–444. https://doi.org/10.1038/nature14539

[72] Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. 2018. Scalable agent alignment via reward modeling: a research direction. http://arxiv.org/abs/1811.07871

[73] Guangliang Li, Randy Gomez, Keisuke Nakamura, and Bo He. 2019. Human-Centered Reinforcement Learning: A Survey. *IEEE Transactions on Human-Machine Systems* 49, 4 (Aug. 2019), 337–349. https://doi.org/10.1109/THMS.2019.2912447

[74] Guangliang Li, Bo He, Randy Gomez, and Keisuke Nakamura. 2018. Interactive Reinforcement Learning from Demonstration and Human Evaluative Feedback. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN ’18)*. Institute of Electrical and Electronics Engineers, 1156–1162. https://doi.org/10.1109/ROMAN.2018.8525837

[75] Li, Guangliang. 2016. *Socially intelligent autonomous agents that learn from human reward*. Ph.D. Dissertation. University of Amsterdam. https://dare.uva.nl/personal/pure/en/publications/socially-intelligent-autonomous-agents-that-learn-from-human-reward(d12d11a0-04cb-4ee8-bde-0f9671e0c2b4).html

[76] Jinying Lin, Zhen Ma, Randy Gomez, Keisuke Nakamura, and Bo He. 2020. A Review on Interactive Reinforcement Learning From Human Social Feedback. *IEEE Access* 8 (2020), 120757–120765. https://doi.org/10.1109/ACCESS.2020.3006254

[77] Long-Ji Lin. 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning* 8, 3 (May 1992), 293–321. https://doi.org/10.1007/BF00992699

[78] Catarina Lopes-Dias, Andrea I. Sburlea, and Gernot R. Müller-Putz. 2019. Online asynchronous decoding of error-related potentials during the continuous control of a robot. *Scientific Reports* 9, 1 (Nov. 2019), 17596. https://doi.org/10.1038/s41598-019-51409-x

[79] Fabien Lotte. 2014. A Tutorial on EEG Signal-processing Techniques for Mental-state Recognition in Brain–Computer Interfaces. In *Guide to Brain-Computer Music Interface*, Eduardo Reck Miranda and Julian Castet (Eds.). Springer, London, 133–161. https://doi.org/10.1007/978-1-4471-6584-2_7

[80] Fabien Lotte, Laurent Bougrain, Andrezej Cichocki, Maureen Clerc, Marco Congedo, Alain Rakotomamonjy, and Florian Yger. 2018. A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update. *Journal of Neural Engineering* 15, 3 (April 2018), 031005. https://doi.org/10.1088/1741-2552/aabb2f

[81] Steven J. Luck. 2014. *An Introduction to the Event-Related Potential Technique*. MIT Press.

[82] Tian-jian Luo, Ya-chao Fan, Ji-ru Lv, and Chang-le Zhou. 2018. Deep reinforcement learning from error-related potentials via an EEG-based brain-computer interface. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM ’18)*. IEEE, 697–701. https://doi.org/10.1109/BIBM.2018.8621183

[83] James MacGlashan, Mark K. Ho, Robert Loftin, Bei Peng, David Roberts, Matthew E. Taylor, and Michael L. Littman. 2017. *Interactive Learning from Policy-Dependent Human Feedback*. http://arxiv.org/abs/1701.06049

[84] Jonathan Masci, Ueli Meier, Dan Cireșan, and Jürgen Schmidhuber. 2011. Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction. In *Artificial Neural Networks and Machine Learning – ICANN 2011* (ICANN ’11), Timo Honkela, Wlodzislaw Duch, Mark Girolami, and Samuel Kaski (Eds.). Springer, Berlin, Heidelberg, 52–59. https://doi.org/10.1007/978-3-642-21735-7_7

[85] José del R. Millán. 2004. On the need for on-line learning in brain-computer interfaces. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541) (IJCNN '04, Vol. 4)*. 2004 IEEE International Joint Conference on Neural Networks (IJCNN '04, Vol. 4). IEEE, 2877–2882 vol. 4. https://doi.org/10.1109/IJCNN.2004.1381116

[86] Azalia Mirhoseini, Anna Goldie, Mustafa Yargon, Joe Wenjie Jiang, Ibrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nabi, Jiwon Pak, Andy Tong, Kavya Srinivasa, William Hang, Emre Tuncer, Quoc V. Le, James Laudon, Richard Ho, Roger Carpenter, and Jeff Dean. 2021. A graph placement methodology for fast chip design. *Nature* 594, 7862 (June 2021), 207–212. https://doi.org/10.1038/s41586-021-03544-w
[87] Volodymyr Mnih, Adrià Puigdomènec Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. http://arxiv.org/abs/1602.01783

[88] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. Nature 518, 7540 (Feb. 2015), 529–533. https://doi.org/10.1038/nature14236

[89] Andrew Andrych and Tom Chau. 2017. A Passive EEG-BCI for Single-Trial Detection of Changes in Mental State. IEEE Transactions on Neural Systems and Rehabilitation Engineering 25, 4 (April 2017), 345–356. https://doi.org/10.1109/TNSRE.2016.2641956

[90] Anis Najar and Mohamed Chetouani. 2021. Reinforcement Learning With Human Advice: A Survey. Frontiers in Robotics and AI 8 (2021), 74. https://doi.org/10.3389/frbot.2021.584075

[91] Andrew Y. Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In Proceedings of the Sixteenth International Conference on Machine Learning (ICML ’99). Morgan Kaufmann, 278–287.

[92] Andrew Y. Ng and Stuart Russell. 2000. Algorithms for Inverse Reinforcement Learning. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML ’00). Morgan Kaufmann, 663–670.

[93] Takayuki Osa, Jony Pajarinen, Gerhard Neumann, J. Andrew Bagnell, Pieter Abbeel, and Jan Peters. 2018. An Algorithmic Perspective on Imitation Learning. https://doi.org/10.1561/2300000053

[94] Babie A. Ramadan and Athanasios V. Vasilakos. 2017. Brain computer interface: control signals review. Neurocomputing 223 (Feb. 2017), 26–44. https://doi.org/10.1016/j.neucom.2016.10.024

[95] Rajesh P. N. Rao, Andrea Stocco, Matthew Bryan, Devapratim Sarma, Tiffany M. Youngquist, Joseph Wu, and Chantel S. Prat. 2014. A Direct Brain-to-Brain Interface in Humans. PLOS ONE 9, 11 (Nov. 2014), e111332. https://doi.org/10.1371/journal.pone.0111332

[96] Mamunur Rashid, Norizam Suhaimi, Anwar P. F. Abdul Majeed, Rabiu Muazu Musa, Ahmad Fakhri Ab. Nasir, Bila Sama Bari, and Sabira Khutan. 2020. Current Status, Challenges, and Possible Solutions of EEG-Based Brain-Computer Interface: A Comprehensive Review. Frontiers in Neuro robotics 14 (2020), 25. https://doi.org/10.3389/fneur.2020.00205

[97] Andrees F. Salazar-Gomez, Joseph DelPreto, Stephanie Gill, Frank H. Guenther, and Daniela Rus. 2017. Correcting robot mistakes in real time using EEG signals. In 2017 IEEE International Conference on Robotics and Automation (ICRA ’17). sameek_erpnonstation_2015, 6570–6577. https://doi.org/10.1109/ICRA.2017.7989777

[98] Wojciech Samek and Klaus-Robert Müller. 2015. Tackling noise, artifacts and nonstationarity in BCI with robust divergences. In 2015 23rd European Signal Processing Conference (EUSIPCO ’15). IEEE, 2741–2745. https://doi.org/10.1109/EUSIPCO.2015.7362883

[99] Nico M. Schmidt, Benjamin Blankertz, and Matthias S. Treder. 2012. Online detection of error-related potentials boosts the performance of mental typewriters. BMC Neuroscience 13 (Feb. 2012), 19. https://doi.org/10.1186/1471-2202-13-19

[100] Julian Schröttweiser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. 2020. Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. http://arxiv.org/abs/1911.08265

[101] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. 2017. Trust Region Policy Optimization. http://arxiv.org/abs/1502.05477

[102] John Schulman, Filip Wolski, Prafulla Dharwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. http://arxiv.org/abs/1707.06347

[103] Murat Sensoy, Lance Kaplan, and Melih Kandemir. 2018. Evidential Deep Learning to Quantify Classification Uncertainty. http://arxiv.org/abs/1806.01758

[104] Burr Settles. 2009. Active Learning Literature Survey: Technical Report. University of Wisconsin-Madison Department of Computer Sciences. https://minds.wisconsin.edu/handle/1793/60660

[105] Pradeep Shenoy, Matthias Krauledat, Benjamin Blankertz, Rajesh P. Rao, and Klaus-Robert Müller. 2006. Towards adaptive classification for BCI. Journal of neural engineering 3, 1 (2006), R13–R23. https://doi.org/10.1088/1741-2205/3/1/R02

[106] Martin Spüler and Christian Niethammer. 2015. Error-related potentials during continuous feedback: using EEG to detect errors of different type and severity. Frontiers in Human Neuroscience 9 (2015), 155. https://doi.org/10.3389/fnhum.2015.01555

[107] Richard S. Sutton. 1984. Temporal Credit Assignment in Reinforcement Learning. Ph.D. Dissertation. University of Massachusetts Amherst. https://scholarworks.umass.edu/dissertations/AAI8410337

[108] Richard S. Sutton and Andrew G. Barto. 2018. Reinforcement Learning: An Introduction. MIT Press.

[109] Anit A. Tenorio-Gonzalez, Eduardo F. Morales, and Luis Villaseñor-Pineda. 2010. Dynamic Reward Shaping: Training a Robot by Voice. In Advances in Artificial Intelligence – IBERAMIA 2010, Angel Kuri-Morales and Guillermo R. Simari (Eds.). Vol. 6433. Springer Berlin Heidelberg, Berlin, Heidelberg, 483–492. https://doi.org/10.1007/978-3-642-16952-6_49

[110] Andrea L. Thomaz and Cynthia Breazeal. 2006. Reinforcement learning with human teachers: evidence of feedback and guidance with implications for learning performance. In Proceedings of the 21st national conference on Artificial intelligence - Volume 1 (AAAI’06). AAAI Press, Boston, Massachusetts, 1000–1005.

[111] Andrea L. Thomaz and Cynthia Breazeal. 2007. Asymmetric Interpretations of Positive and Negative Human Feedback for a Social Learning Agent. In The 16th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN ’07). IEEE, Jeju, South Korea, 720–725.
A  ILLUSTRATION OF ALIGNMENT PROBLEM IN RL AND IRL

Fig. 6. The relationship between increasing complexity of an environment and misaligned behaviors in RL.

Fig. 7. The relationship between increasing complexity of an environment and misaligned intentions for IRL.

B  REINFORCEMENT LEARNING BACKGROUND

Fig. 8. RL formalization where agent interacts with the environment at every time step $t$.

B.1 Generalized Policy Iteration

Generalized policy iteration (GPI) works by evaluating and improving an arbitrary policy $\pi$ through the ideas of policy evaluation and policy improvement [108]. Policy evaluation serves the purpose of providing insight into how well the current policy is doing by evaluating the value of a state or state-action pair. The value of state $s$ can then be given by a learned value function. Value functions works by estimating the value $s$ which corresponds to the expected discounted return when starting from $s$ and then following the policy $\pi$ thenceforth. Policy improvement serves the purpose of updating the policy based on these evaluations in order to learn a more optimal policy. Using the value function, a policy can be indirectly generated by greedily selecting the actions that lead to the states with the highest estimated
values. Given the GPI process, there are two important types of value functions to know about: state-value functions and action-value functions.

The state-value function for $\pi$ is denoted by $V_\pi(s)$ and estimates the expected discounted return of state $s$, thenceforth following policy $\pi$ [108].

$$V_\pi(s) = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \left| S_t = s \right. \right]$$  \hspace{1cm} (32)

Using the Bellman equation it is possible to rewrite Eq. (32) as a recursive relationship

$$V_\pi(s) = \mathbb{E}_\pi \left[ R_{t+1} + \gamma V_\pi(S_{t+1}) \left| S_t = s \right. \right],$$  \hspace{1cm} (33)

where $R_{t+1}$ is the most recently received reward and $\gamma V_\pi(S_{t+1})$ is the estimated value of the future discounted return when starting from the next state.

The action-value function for $\pi$ is denoted as $Q_\pi(s,a)$ and estimates the expected discounted return of state $s$ when taking action $a$, thenceforth following policy $\pi$ [108]. Note, that the values for $Q_\pi(s,a)$ are often referred to as Q-values.

$$Q_\pi(s,a) = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \left| S_t = s, A_t = a \right. \right]$$  \hspace{1cm} (34)

Once again, it is possible to rewrite Eq. (34) as a recursive relationship using the Bellman equation

$$Q_\pi(s,a) = \mathbb{E}_\pi \left[ R_{t+1} + \gamma Q_\pi(S_{t+1}, A_{t+1}) \left| S_t = s, A_t = a \right. \right],$$  \hspace{1cm} (35)

A popular partial sampling method for updating state-value or action-value functions is referred to as Temporal Difference (TD) learning. TD has two special properties that are worth noting. First, TD learning learning only needs to wait at least till the next time step to perform an update. Second, TD learning is considered a bootstrapping method as it learns a value function using the current estimate of the value function. The following formalizes the TD error using a state-value function:

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t),$$  \hspace{1cm} (36)

where same formalization applies to action-value function. The TD error can be interpreted as measuring the difference between the current state’s estimated value $V(S_t)$ and a more true estimate of the current state’s value $R_{t+1} + \gamma V(S_{t+1})$.

To learn the current state-value function $V(S_t)$ an update need only be applied using the learning rate $\alpha$ (i.e., a scaling factor) and the TD error in Eq. (36):

$$V(S_t) = V(S_t) + \alpha \left[ R_{t+1} + \gamma V(S_{t+1}) - V(S_t) \right]$$  \hspace{1cm} (37)

Further, if function approximation is used such that $V(S_t)$ is parameterized by weights $w$, the gradient of the TD error can be used to update the weights as follows

$$w_{t+1} = w_t + \alpha \delta_t \nabla V_{w_t}(S_t).$$  \hspace{1cm} (38)
Given TD can be used to update the estimates of value functions such that policy evaluation can occur, this then leaves policy improvement. Recall, that policy improvement is done by greedily selecting actions that lead to the states with the highest value. Thus, a greedy policy can be generated by selecting the action with the largest Q-value as this represents the current best estimated action.

$$\pi(s) = \arg \max_a Q(s, a) \quad (39)$$

### B.2 Policy Gradient

Policy gradient methods work by directly learning or optimizing over the policy. This is typically done by parametrizing the policy such that $\pi_\theta(a|s)$ can be optimized with respect to $\theta$. These methods are characterized by slow learning due to high variance in the gradient estimates. However, policy gradient methods work well for continuous action spaces as they do not need to optimize over all potential actions. In this section we focus particularly on the actor-critic formalization as it integrates ideas from policy gradient and generalized policy iteration.

Actor-critic algorithms can be broken into two parts: an actor and a critic. The actor determines how the agent acts by using a parameterized policy $\pi_\theta(a|s)$. Likewise, the critic evaluates the actor’s policy at each time step using a value function. The critic’s evaluation is then used to update both the value function and the policy.

Given a function approximator critic that is parametrized by the weights $w$, the update for the critic’s parameters computed exactly as in Eq (38). Similarly, given the actor is parametrized by $\theta$, the actor’s parameters are update via the policy gradient theorem [108]:

$$\theta_{t+1} = \theta_t + \alpha \nabla \log \pi_\theta (A_t | S_t) V(S_t). \quad (40)$$

To further decrease the variance of the actor’s update an advantage function $A(s, a)$ can be used to replace $V(S_t)$ such that

$$A(s, a) = Q(s, a) - V(s) \quad (41)$$

The advantage function can be interpreted as computing whether the selected action is performing better or worse than expected [108]. The actor’s parameter update can now be rewritten using the advantage function as

$$\theta_{t+1} = \theta_t + \alpha \frac{V}{\nabla} \pi_\theta (A_t | S_t) A(s, a). \quad (42)$$

Given the TD error is an unbiased estimator of the advantage function, this reduces the advantage function in Eq. (41) to simply being the TD error given in Eq. (36),

$$A(s, a) = Q(s, a) - V(s)$$
$$= r + \gamma V(s') - V(s) \quad (43)$$
$$= \delta.$$

Finally, the actor’s update can be rewritten using the TD error in place of the advantage function as

$$\theta_{t+1} = \theta_t + \alpha \delta_t \frac{V}{\nabla} \pi_\theta (A_t | S_t). \quad (44)$$
Towards Intrinsic Interactive Reinforcement Learning

(a) Time domain features
(b) Frequency domain features

Fig. 9. An example of a 1 second long random slice of neural activity given in the time and frequency domain. (a) shows the time domain features for the slice of random neural activity. (b) shows the frequency domain features for the same slice of random neural activity.

C NEURAL ACTIVITY FEATURES

Neural activity features are represented either two domains: time or frequency. Time domain features, as seen in Fig. 9a, consists of periodic measurement of the scalp voltage given in micro-volts (μV) where the sampling rate of a device determines the number of measurements taken per second. Meanwhile, frequency band features, as seen in Fig. 9b, represent the energy or power for a given frequency band. Frequency bands are broken into five discrete ranges: delta (< 4 Hz), theta (4-8 Hz), alpha (8-13 Hz), beta (13-30 Hz), and gamma (> 30 Hz) [81]. Fig. 10 visualizes what each these five bands look like when represented in the time domain. Furthermore, spectrograms (i.e., time-frequency plots) are a common way of combining both time and frequency domain features.