University of Kentucky
UKnowledge

Theses and Dissertations--Computer Science

Computer Science

2023

The BASIL technique: Bias Adaptive Statistical Inference Learning Agents for Learning from Human Feedback

Jonathan Indigo Watson
University of Kentucky, jonathan.indigo.watson@protonmail.com
Author ORCID Identifier:
https://orcid.org/0009-0000-2101-6237

Digital Object Identifier: https://doi.org/10.13023/etd.2023.162

Right click to open a feedback form in a new tab to let us know how this document benefits you.

Recommended Citation

Watson, Jonathan Indigo, "The BASIL technique: Bias Adaptive Statistical Inference Learning Agents for Learning from Human Feedback" (2023). *Theses and Dissertations--Computer Science*. 134.
https://uknowledge.uky.edu/cs_etds/134

This Doctoral Dissertation is brought to you for free and open access by the Computer Science at UKnowledge. It has been accepted for inclusion in Theses and Dissertations--Computer Science by an authorized administrator of UKnowledge. For more information, please contact UKnowledge@lsv.uky.edu.

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student’s advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student’s thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Jonathan Indigo Watson, Student

Brent Harrison, Major Professor

Simone Silvestri, Director of Graduate Studies

The BASIL technique: Bias Adaptive Statistical Inference Learning Agents for Learning from Human Feedback

DISSERTATION

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the College of Engineering at the University of Kentucky

By
Jonathan Indigo Watson
Lexington, Kentucky

Director: Dr. Brent Harrison, Professor of Computer Science
Lexington, Kentucky 2023

Copyright© Jonathan Indigo Watson 2023

ABSTRACT OF DISSERTATION

The BASIL technique: Bias Adaptive Statistical Inference Learning Agents for Learning from Human Feedback

We introduce a novel approach for learning behaviors using human-provided feedback that is subject to systematic bias. Our method, known as BASIL , models the feedback signal as a combination of a heuristic evaluation of an action's utility and a probabilistically-drawn bias value, characterized by unknown parameters. We present both the general framework for our technique and specific algorithms for biases drawn from a normal distribution. We evaluate our approach across various environments and tasks, comparing it to interactive and non-interactive machine learning methods, including deep learning techniques, using human trainers and a synthetic oracle with feedback distorted to varying degrees. We demonstrate that our algorithm can rapidly learn even in the presence of normally distributed bias, which other methods struggle with, while also exhibiting some resistance to other types of distortion.

KEYWORDS: Machine Learning, Reinforcement Learning, Interactive Machine Learning, Interactive Reinforcement Learning, IML, IRL, RL, ML, BASIL, TAMER, Bias, Statistical Inference, Tetris, Bowling, Atari, Agent presentation

Author's signature: Jonathan Indigo Watson

Date: April 25, 2023

The BASIL technique: Bias Adaptive Statistical Inference Learning Agents for Learning from Human Feedback

By
Jonathan Indigo Watson

Director of Dissertation: Brent Harrison

Director of Graduate Studies: Simone Silvestri

Date: April 25, 2023

Dedicated to Wallace Watson Sr., Sarah Watkins, and Terry Simmons for their support which has been unending despite their passing.

ACKNOWLEDGMENTS

We wish to acknowledge the late J. D. Hunter for providing the world with Matplotlib[18] which we used to produce the graphs in this paper.

We wish to acknowledge the many authors and developers who produced and contributed to Keras[9], Open A.I. Gym[3], SciPy[33], NumPy.[17], and all those who contributed to the Python programming language and it's many core libraries, upon which this work has been built.

We wish to acknowledge Angelica Watson for her aid, support, insight, wisdom, and encouragement without which this would never have been written.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

Today, machine learning plays an ever increasing role in the development of autonomous agents. Fields from game design to industrial control systems require virtual agents that react to an observed system state in a manner consistent with some objective, such as task completion, safety, or inducing player immersion. While the very intuitive and traditional approach to the creation of such agents involves the hand coding of complicated sets of ad-hoc rules and this approach can produce good results, it is immensely time consuming and becomes unfeasible as the complexity of the system (number of actions available at one time, number of different possible states, etc.) continues to grow, particularly for small teams of programmers. This approach may also require an immense amount of domain/behavioral expertise, which programmers may not necessarily have. It is also possible that the information a programmer needs to hand code ad-hoc rules does not exist as it regards human preferences or that the only information available is that which is self-reported, which may be suspect for a variety of reasons or only relevant to one individual. For example: a game designer may be required to create a virtual character such that the character plays a hand of poker while conveying the quality of their hand to the opponent (i.e. the character is easy to read); however, most game designers cannot be reasonably expected to possess body language expertise.

A solution to this problem is to use sophisticated computational techniques, such as machine learning, to learn these behaviors. One of the major approaches to machine learning is reinforcement learning [31] where the agent learns behavior from a reward signal and trial-and-error exploration of its environment. In many traditional approaches to reinforcement learning, the reward signal comes from the agent's environment, possibly interpreted by a heuristic function. An agent will, more or less, begin by behaving randomly but will receive a reward from the environment about the quality of its behaviors. Over time, the agent will adjust its behavior to favor actions that give it the most long-term, positive reward. This approach has been shown to be able to teach virtual agents very complex behaviors in environments where hand-coded rules that achieve good performance are elusive, while techniques for evaluating performance are easy to obtain. For example, consider an agent learning to play an arcade style game. While it may be unclear how to specify behavior rules that result in optimal behavior, it is easy to specify a performance measure to optimize in the game's score. For other tasks, such as creating believable or likeable virtual characters, creating an optimization or scoring function to evaluate the actions taken by the virtual agent can be just as difficult as hand-coding the action rules themselves. While classic reinforcement learning techniques rely on environmental scoring or utility functions to learn optimal behaviors, interactive reinforcement learning (IRL) allows a human trainer to provide a reward signal either in addition to or in replacement of an environmentally produced signal [8]. This enables a human trainer to provide direct input on how the agent learns and what behaviors that the agent will learn. Often, this results in more robust scoring functions that can fill in gaps that could exist in purely environmental scoring functions, and this can also result in enabling agents to more quickly learn desired behaviors. Knox and Stone even showed that without access to an environmental reward signal, an agent learning from a human trainer could very quickly reach a high level of performance within a small number of training episodes[23].

However, the introduction of a human trainer is not without drawbacks. By utilizing human trainers, researchers introduce human factors that could affect how agents learn in ways that we do not completely understand. Examples of these factors include:

- aspects of the human instructors such as teaching style and understanding of the problem
- aspects of the agent's presentation to the human instructors such as a picture representing the agent or the name an agent is referred to by
- the interaction thereof, such as bias regarding the apparent gender of the agent

Humans also have limited time to devote to training tasks and the rate at which they can provide feedback is also limited. Understanding these human factors is important for setting up training in a manner that facilitates learning of the desired behaviors. If researchers and programmers can understand human biases, that understanding should enable them to take better advantage of human feedback. However, the scope of human biases is quite vast and understudied with few techniques available to take into account those biases.

In this body of work we have set ourselves the following goals/tasks:

- Increase our understanding of the ways in which factors, such as agent presentation, can influence feedback given to interactive reinforcement learning algorithms
- Develop a technique and algorithms for interactive reinforcement learning agents that account for bias in feedback given by human trainers
- Combine our techniques for bias with techniques to address variance and inconsistency in feedback given by human trainers

1.1 Understanding Human Bias

This body of work focuses on developing techniques that take into account human bias in regards to the feedback that human trainers provide to interactive reinforcement learning agents. As human bias has been demonstrated to effect the feedback provided in numerous ways, from withholding feedback to being overly positive or negative in the feedback given, the development of a technique to handle this bias within the provided feedback requires a modicum of understanding of that bias and the means to counteract that bias within that feedback such that a learning algorithm can learn the correct policy to complete its task or goals. However, developing a full understanding of human bias as it applies to interactive learning schemes is a task beyond any one body of work. In pursuit of developing a technique to take into account human bias in an IRL environment, we first sought to increase our understanding of the effects of human bias on feedback provided by conducting an experiment to study human bias, as to fulfill our first research task. Most existing work on determining the extent of human bias on feedback given is done with respect to anthropomorphization of the agent presented to human trainers, whether this be by representing the agent as a puppy or as a robotic entity among many other forms of representation. However, there are a number of understudied factors when it comes to an agent's representation that will bring human bias into play. One of these understudied factors in the field of virtual learning agents is how the presented gender of the agent effects the quality and consistency of feedback that a human trainer will provide. The effects of an agent's gender representation are almost unavoidable as the mere act of applying a name to the agent, generally the first step in anthropomorphizing it as was done by Waytz, Heafner, and Epley[36], will typically imply a gender identity in most Human languages and cultures. Even if a unisex name is chosen, applying a gender is, in itself, a basic technique for anthropomorphization and anthropomorphizing an agent is usually desired in many commercial applications[36]. Therefore, we set out to explore how human factors involving the perceived gender of an agent can affect how humans train said agents. This experiment in understanding bias in human trainers presented human trainers with the task of providing feedback to a virtual agent navigating a grid world environment in a browser-based game where the agent was alternatively represented by a female pixel art representation with English female pronouns named Alice, versus a presentation of a male pixel art representation with English male pronouns named William. We also surveyed the instructors for a number of demographic qualities to study the effect of those qualities on the feedback given. Specifically, we sought to answer the following questions:

1. Is there a statistically significant difference in the feedback given to an agent when presented with the female representation versus the male one across all human subjects and of the states presented in the environment?

2. Qualitatively, how do the feedback histograms differ for feedback given to the female representation of the agent versus the male one?

3. What demographic characteristics of human instructors have a statistically significant effect on feedback given to the agents?

4. In particular what effect does the instructor's reported gender have on feedback given to the agent?

5. Is there a cross effect between instructor gender and the agent's presented gender?

By answering these questions, we can gain a greater understanding of the role that perceived agent gender plays in training virtual characters. In this body of work, we present the first foray into exploring this understudied effect. By studying the effects of an agent's gendered representation on human trainer provided feedback, we will complete one of our research goals.

1.2 Bias Adaptive Statistical Inference Learning (BASIL) Agents for Learning from Human Feedback

This work addresses the issue of human bias in the feedback provided to interactive reinforcement learning algorithms by developing a technique that accounts for that bias, as per our second research goal. Existing techniques are limited in handling human factors in discrete feedback, which also limits the sorts of problems for which they are suited to address. As far as we know, no existing technique handles human bias in numerically valued reward schemes for interactive reinforcement learning agents. However, we argue that when provided the means to communicate more detailed numerical rewards, human trainers can provide an agent with a heuristic estimate of the utility of taking a given action in a given state. By using a numerical value for the reward scheme we can numerically counteract the bias using statistical techniques, such that we can obtain less biased reward values. Utilization of numerical values allows us to model that numerical human feedback as being separable into an assessment of the utility of taking a given action in a given state and a biasing value, which is drawn from a probabilistic distribution. Therefore, in this body of work we present a novel technique for separating bias from the reward signal.

Our approach is inspired by the ISABL algorithm by Loftin et.all [25] and it is worth taking a moment to discuss the similarities and differences between the algorithm we present and the ISABL algorithm. Both algorithms share a similar structure and use expectation maximization to compute the best possible action based on imperfect human feedback. However, there are some key differences. In ISABL, feedback is assumed to have discrete values of positive, negative, or zero and the nature of the human factors it wishes to address is limited to the meaning of silence. Finally, it's use of expectation maximization is done to directly determine the human trainer's target policy. The limitations of this are: ISABL requires the human to have one single on policy action for any state; it does not allow for situations in which the human considers any of several paths as valid; and it does not allow for situations in which the human trainer considers one action to be 'good' and another action to be 'better'. In ISABL's model a human should only reward the best possible action, however, for more complicated tasks a human trainer may not be able to determine the best possible action, but may still provide a useful evaluation of actions taken heuristically.

The technique which we present is a general technique that is widely applicable to a variety of interactive machine learning environments in which one may encounter bias in provided feedback from human trainers. This technique separates the utility of evaluation and biasing values from a history of reward signals when given the form of the biasing distribution, but without knowing the parameters of the distribution using expectation maximization (EM)[10] to compute a maximum likelihood estimate of the utility assessments. Additionally, we provide an algorithm utilizing this technique for a normal distribution and test it against several other algorithms under a variety of biasing effects and permutations of parameters using simulated human trainers. The algorithm/technique we present, by contrast, takes numerical feedback and does not address the meaning of silence, instead it seeks to address the issue of bias in the feedback signal. To accomplish this, our technique does expectation maximization not to compute the human trainer's policy, but to compute the human trainer's evaluation of the utility of each action and state for which they have been provided feedback. Our technique treats the human feedback signal as being comprised of the following: 1) a heuristic assessment of the utility provided by taking an action that is a measure of both the actions immediate benefit and its potential for long-term benefit, and 2) a biasing factor that depends upon the human trainer and aspects of the agent's presentation to the human trainer. For this technique, we assume the value of the biasing factor attached to any given feedback instance is drawn probabilistically from a distribution determined by unknown parameters.

The goal of our technique is to learn the underlying heuristic evaluations present in the feedback to obtain better performance on the problem where a human trainer's feedback signal is provided but is highly biased. This technique is the Bias Adaptive Statistical Inference Learning algorithm, referred to from here onward as the BASIL algorithm or BASIL technique. The BASIL technique provides a means to account for human bias in provided feedback and is applicable to a variety of tasks and environments, as we will demonstrate below. To test the BASIL algorithm it was compared with other interactive reinforcement learning algorithms, primarily SABL, ISABL, and TAMER. The BASIL technique was also hybridized with the TAMER algorithm, producing the BASILTAMER algorithm. The goal of our experiments utilizing the application of the BASIL technique with the TAMER algorithm was to determine if adding normally distributed bias to the feedback signal given to the classical TAMER and BASILTAMER algorithms would result in the classical TAMER algorithm failing to learn the desired behavior encoded in the feedback signal while not preventing the BASILTAMER algorithm from learning the desired behavior. We present our findings below.

Adaptive Variance and Silence BASILTAMER Extensions

Addressing bias is only a part of dealing with human factors in feedback to interactive reinforcement learning agents. In order to make the most of the feedback given by human trainers, the variance and inconsistency in the feedback also needs to be addressed, which also accomplishes our third goal in regards to this research. Inconsistencies in human feedback can take various forms such as: feedback withholding; consistently small or neutral feedback values; and variance of feedback values around their mean. Existing work by Loftin et. al. [25] addresses the problem of interpreting feedback withholding, however the technique they presented is limited to discrete feedback as their work assumes that feedback is defined only as positive or negative and it does not have a numerical value. Furthermore, the algorithms they present are extremely limited in the number of problems that they can feasibly be applied to without significant modifications as they are designed and tested in domains with very small state spaces. It is also unfortunate that with human trainers variance in the feedback given is, generally, an inevitability. In its simplest forms, variance is the human trainer being inconsistent with the feedback given and providing different feedback values for the same situation. For example: taking a particular action in a particular state such as placing a block in a Tetris game in the same spot and orientation when the field is empty, might get a reward value of 5 from a trainer once and a value of 6 at another time. This problem gets worse when there are multiple human trainers, which becomes increasingly necessary for larger scale tasks. Additionally, this variance does not always mean that the human trainer made a mistake. Depending upon the feature set provided to an algorithm, human trainers may be able to see differences in two states that look identical to the algorithm. In our Tetris example, perhaps the human trainer is aware of what block is to be placed next while this information was not provided to the algorithm so as to maintain the Markov Decision Principle. This inconsistency can easily lead to confusion as to what the optimal action to take is for a given state and can cause an agent's behavior to become erratic as it incorporates new feedback instances. Therefore, we have built upon the BASIL technique to develop three extensions of the BASILTAMER algorithm that handle, alternatively, variance and silence in addition to the normal capabilities of the BASILTAMER algorithm.

The BASIL technique is already designed to calculate the observed variance in the feedback received, particularly of distributions, such as the normal distribution where it is a defining parameter. The first extension we developed was for the purpose of addressing variance in feedback provided by dynamically adjusting the agent's learning rate. Full details for this algorithm will be described within our Methods sections.

For addressing silence in feedback signals from human trainers, we have developed two extensions for the BASILTAMER algorithm. The first of these takes the extremely simple approach of treating silence as though it were an explicitly given feedback value of zero, which we refer to as the Zero Silence Extension. We also developed a more sophisticated extension to address silence. This last extension, which we refer to as the Average Silence Extension, calculates the average expected feedback as given by BASILTAMER's neural network for each observed instance of silence. The algorithm uses the expected feedback values to calculate an average expected feedback for silence and uses this at each EM step as the feedback value for each instance of silence in the observation history. This is used in computing the next phase of the neural network as the EM loop continues until it converges. The algorithms for the Zero Silence and Average Silence Extensions can be found below.

In this work we compare these Extensions with the default BASILTAMER and unmodified TAMER in the Tetris environment we utilize for our BASILTAMER experiments.

1.3 BASIL and Deep Neural Networks

In addition to the basic BASIL technique we examined the application of the BASIL technique within the framework of Deep Neural Networks, which, once more, addresses our secondary goal for this research. This involved applying the BASIL technique to the Deep TAMER algorithm. Deep Learning generally refers to machine learning performed with Deep Neural Networks which can benefit from interactive reinforcement learning techniques. However, some related work may also fall under this label. In general terms, Deep Neural Networks are defined as Artificial Neural Networks with enough layers that the task of updating weights of early layers in the network in response to error signals becomes non-trivial. Therefore, in order to expand the scope of our work to Deep Learning tasks, we choose to apply the BASIL technique to Deep TAMER [34]. Deep TAMER is an expansion of the original TAMER algorithm that incorporates a Deep Neural Network in place of the single layer preceptron used by the original TAMER algorithm.

To apply the BASIL technique to a Deep Neural Network, we applied the technique to Deep TAMER and utilized it within an ATARI bowling environment as was done by Warnell et. all [34] as they presented in their research for the development of Deep TAMER. A secondary goal of this experiment was to further verify the observations of Bartneck, Reichenback, and Carpenter [2] of how agent presentation between different grades of anthropomorphism produced different feedback when using images of the same robotic avatars as they did in their experiment. To incorporate the BASIL technique into the Deep TAMER algorithm we had to use it only with the latter fully connected, online trained portion of the Deep Neural Network. We refer to this adaptation as the Deep BASILTAMER algorithm or as Deep BASILTAMER.

Multi-Circumstance Aware Deep BASILTAMER

When the phrase bias is used, it is often used to describe a difference that occurs between two samples. It is therefore worthwhile to consider using the BASIL technique for multiple feedback sources, whether these sources are different human trainers, or different presentations of the agent to human trainers, or any other factor which might alter the way feedback is given. We have developed and tested a modification of our Deep BASILTAMER algorithm to be multi-trainer aware for this sort of case, which addresses our second and third goal simultaneously. We refer to this as Multi-Circumstance Aware BASIL. We have tested Multi-Circumstance Aware BASIL on the feedback given by human trainers who were presented with varying presentations of the agent using data obtained from our preceding Deep BASILTAMER experiments. State and actions pairs combined with feedback sourced from the AIBO and P.K.D. sessions of the BASIL DeepTAMER experiments were used to train models using this modification offline. In these experiments we compared Deep TAMER, Deep BASILTAMER, and Multi-Circumstance Aware Deep BASILTAMER. As described above, for these experiments Multi-Circumstance Aware Deep BASILTAMER is aware that there are two feedback profiles and it is also aware of what data is from which profile. It utilized this awareness to perform expectation maximization on the two data sets separately in order to generate two different sets of bias distribution parameters and it then modifies new feedback based on which parameter set applies to the data. The primary goal of these experiments was to determine if Multi-Circumstance Aware Deep BASILTAMER, which is specifically adapted for handling multiple biasing conditions as we have put forth, performs substantially better than unmodified Deep BASILTAMER, and Deep TAMER without BASIL when feedback for each algorithm is produced under multiple biasing conditions.

Chapter 2 Background and Related Work

In this section we provide an explanation of reinforcement learning and interactive reinforcement learning for those unfamiliar with them. We will then focus on general work in IRL, in particular the TAMER algorithm which has become a standard for IRL algorithms, followed by reviewing existing work into bias in human provided feedback. Finally, we will discuss the related work in Social Sciences that seek to study the manifestation of human gender bias which bears relevance to our first experimental study to see how a gendered representation of an agent affects the feedback it receives.

2.1 Reinforcement Learning and Interactive Reinforcement Learning

Our work primarily concerns reinforcement learning with an emphasis on the problems associated with the addition of a human trainer. Reinforcement learning is a technique that is used to solve a *Markov decision process* (MDP). A MDP is a tuple $M = \langle S, A, T, R, \gamma \rangle$ where S is the set of world states, A is the set of agent actions, T is a transition function $T : S \times A \rightarrow P(S)$, R is a reward function $R : S \times A \rightarrow \mathbb{R}$, and γ is a discount factor $0 \leq \gamma \leq 1$.

The result of reinforcement learning is a *policy* $\pi : S \rightarrow A$, which defines which actions should be taken in each state in order to maximize expected future reward. The reinforcement learning problems we consider have the addition of a human provided feedback signal. This is typically known as interactive reinforcement learning. This feedback signal is typically not included in the cumulative reward total used to evaluate an agent's performance, but may assist the agent in learning.

Interactive reinforcement learning is not the only approach to interactive machine learning. An example of another approach is apprenticeship learning where instead of learning from human feedback given in response to the agent's actions, the agent learns from one or more examples of good performance from human trainers. This type of learning is also referred to as imitation learning or learning from demonstration. An example of this approach is inverse reinforcement learning by Abbeel and Ng [1]. The idea of this approach is to, from a demonstration by a human expert, attempt to learn a reward function that produces the policy that the expert acted in accordance to. To do this, Abbeel and Ng defined the concept of the feature expectation of the policy, which is the summation of all the features of all the states visited when following given policy discounted by an exponentially decaying factor. In practical computation, the sum can be ended after a finite number of steps because the discount factor will have rendered the contribution of further steps minute. With that defined algorithm for computing, a reward function is as follows:

- Start with a random policy.
- Compute the feature expectation of that policy.

- Compute the set of weights that maximizes the normal of the weights such that the normal does not exceed one, that minimizes the difference in the weighted feature expectation of all tried policies and the expert's demonstrated policy. If that difference is within our error tolerance, we are done and the reward function we seek is a weighted sum of the features weighed by the weights we just computed. If we are not, compute the policy that maximizes the reward function determined by the weights we have just computed and add that policy to our list of tried policies before returning to step 2.

In practice, there are optimizations to avoid the costly prospect of finding the new optimal weights at each step however, they do not change the inherit nature of the algorithm.

While alternative approaches to interactive machine learning, such as apprenticeship learning, do not necessarily suffer from the same sorts of bias as interactive reinforcement learning, they still are affected by complex human factors and may have other limitations. For example, apprenticeship learning techniques on their own do not generally exceed the performance of the human trainer while interactive reinforcement learning techniques can as they rely only on the human's ability to critique performance and not to perform similarly.

2.2 Existing General Work in Interactive Machine Learning

The recent past has seen a growing body of work in the area of agents learning from human feedback. Tomaz and Breazal [32] conducted an experiment where human trainers helped a virtual agent to learn to perform a simulated cake baking task. Among other findings, they observed that human trainers gave rewards in anticipation of good actions and not just for immediate positive outcomes, and that the feedback from human trainers corresponds more to an action's utility than the immediate reward it provided.

Knox and Stone [23] provided the TAMER algorithm which they showed learned very quickly to perform moderately well in the complicated task of the game Tetris. The TAMER algorithm remains a standard among interactive reinforcement learning algorithms. This algorithm works very similarly to a single layered neural network or perceptron [26]. It takes as input at each training step a vector of features describing the difference between the before and after states of some action. What features are used for a given problem is a task specific issue that an engineer using the algorithm must address for their task. This need to engineer appropriate features for any new task undertaken is one of the major limitations of the TAMER algorithm. At each learning step, the TAMER algorithm, adjusts the weight associated with each feature of the feature vector it considers proportionally to the change in the feature a singular human feedback signal and a learning rate meta parameter. In order to choose an action when given a state, the TAMER algorithm is assumed to be able to predict the results of its action in terms of the feature vector. For each possible action the algorithm might take, it computes the resulting change in the feature vector and calculates the weighted sum of these changes to "score" each possible action. It, then, simply selects the highest scoring action. The issues presented for the TAMER algorithm by needing a human engineered feature set are made worse by the need for these features to be fairly sophisticated assessments of the task state. This is because the single layered model used by the TAMER algorithm has no ability to consider features in more than a linear combination. For example: it cannot consider a high value of one feature as good only in the presence of the high value of a second feature. The relatively recent developments in deep neural networks led to a solution in the work of Warnell et. all [34] which presented Deep TAMER. Deep TAMER replaces the simple single layer perceptron of the original TAMER algorithm with a deep neural network. In Warnell et. all's work the deep neural network is specifically a convolutional neural network followed by several fully connected layers, however, this is some what task specific and the term 'Deep TAMER' could be applied to any modification of the TAMER algorithm to use a deep neural network in place of a single layer perceptron.

The COBOT agent in work by Isbel et.all [19] was a chat bot employed in the online community LambdaMOO and it was trained by human feedback to promote and engage in useful, or at least entertaining, discussions in the community chat rooms. The COBOT agent's reinforcement learning algorithm was very simple. From the perspective of the RL algorithm, there were nine possible actions for COBOT to take when it might choose to act, which it did every few minutes. One of these actions was to do nothing, the rest can be broadly broken up into broad categories. The first category are topic starters where it attempted to initiate a conversation by pulling on the most recent online version of the Boston Globe, the second are role call actions where COBOT attempted to take a role call of people tired of a given subject by picking either a noun or a verb phrase out of recent posts, the third was a social commentary routine that relied on a statistical database of social interaction, and the fourth was a social introduction routine. For state features, COBOT considered which users were active and a vector of social statistics on the conversation. For a feedback signal, the users had a variety of phrases which were assigned different numerical feedback values. After selecting an action, COBOT would consider all reward signals until it's next action to apply to its most recent action taken in the state to which it took it and directly updated this q-value. When it came time to select an action, COBOT would simply pick probabilisticly according to a distribution based on the q-values of each possible action. While COBOT did not directly attempt to address human biases in feedback to interactive reinforcement learning algorithms, Isabel et. all did observe reliability issues in human feedback. Notably, they observed a tendency to give less feedback over time. Our work contributes to this growing body of work by presenting a general technique to account for human bias when utilizing interactive machine learning techniques.

2.3 Existing Work in Human Bias in Interactive Machine Learning

There has been some work on how human factors can influence feedback given by a human trainer such as the experiments performed by Bartneck, Reichenback, and Carpenter's[2], where they studied the effects of agent presentation on human interaction with an agent, although, they did not conduct their research with regards to their agent's gender. Instead, the researchers had their agents represented by robots with differing degrees of anthropomorphism. The first of these is a distinctly robotic humanoid called Tron-X with a somewhat human, although entirely hairless, face and visibly robotic torso. The second, the PKD model by Hanson Robotics, was a much more human-like robot modeled after legendary science fiction author, Philip K. Dick, and is human enough looking to pass a very cursory inspection. The third was a dog-shaped robot, AIBO, from Sony. The researchers performed experiments, both where the robots were physically situated at computers in the same room as the human experimental subjects, and where the robots were represented only by photographs. They found that the human experimental subjects were more likely to praise the agent when it was represented by the PKD robot than when represented by the Tron-X robot, despite these agents performing the same task and using identical algorithms. They also found that the human experimental subjects were both more likely to praise and less likely to punish the agent when it was represented by the robotic dog, AIBO. They found that subjects often overestimated the frequency of punishments they had given as well as underestimating the frequency of praise given. Subjects when questioned even stated that they did not want to punish AIBO because they found the robot dog to be "very cute". They found that the difference in presentation had a profound impact on the way that the human instructors gave feedback, despite there being no difference in the way that any of these agents behaved.

Lehdonvirta et. all[24] found that a player's avatar's gender had an effect on the way the player both sought out and received help in a massively multi-player online game irrespective of the player's actual gender, which strongly suggests that a virtual agent's presented gender will have an effect on feedback it receives.

In Cakmak, Chao, and Thomaz[4], a few different models for human interaction with a learning agent were proposed, and groundwork for future research in interactive machine learning was laid. Their experiment utilized a "waist up" humanoid robot with the, typically male given, name of Simon. Their experiment is also notable for surveying the humans who interacted with Simon for both quantitative and qualitative information about their experience with Simon in each of three different training modes. The researchers found that human teachers perceived Simon as being intelligent and doing more than just passively listening to their instructions.

Research by Knox et.all[22] investigated differences in feedback provided by human trainers when those human trainers believed that they were providing feedback to a live agent or critiquing a completed performance, and found little difference between the ways humans provided feedback in those scenarios. They also found that human trainers can decrease the consistency of feedback they give over time, however, having the agent make mistakes encouraged the human trainer to provide more feedback. The inconsistency of feedback given over time leads us to the work of Griffith et. all [16] which produced the Advise algorithm that interprets human feedback as a discrete communication that reflects the policy the human trainer was attempting to teach, which could be inconsistent and infrequent.

The work that bears the greatest similarity to our own is by Loftin et.all [25] which attempts to allow for learning of desired behavior even when the human trainer does not provide explicit feedback by using Bayesian techniques to weigh silence as positive or negative depending upon the human trainer's teaching strategy. They found that aspects of the trainers had a significant impact on the feedback they supplied. In their experiments, they utilized a population of computer savvy university students and another population of professional dog trainers. They found, among other things, that the professional dog trainers had a very strong bias towards positive feedback only. The practice of only providing positive feedback is an often recommended practice among dog trainers when training actual dogs, which suggests that the dog trainers treated the virtual dog in the experiment the same way as they would treat an actual dog. They also presented two algorithms: SABL, which is explicitly told the parameters governing the trainer's teaching strategy, and ISABl, which infers the parameters of the human trainer's teaching strategy from their feedback history using maximum likelihood estimation via the Expectation Maximization algorithm [10]. However, these algorithms assume that feedback is given only in a discrete form of positive, negative, or zero; and that for any given state human trainers should consider only one on policy action as worthy of receiving reward. This results in creating significant room for the development of algorithms that account for these human factors while also maintaining the more precise communication abilities of a numerical reward signal. Both Advise and the SABL algorithms limit themselves to discrete feedback. We present a technique that handles the numerical bias that occurs in numerical feedback.

Another technique for removing error from human feedback to an interactive machine learning agent is REPaIR by Faulkner et. all. [20] REPaIR assumes that a reward signal from a human may be noisy and contain errors. It attempts to learn how these errors occur in the feedback from the human and then adjust or ignore erroneous feedback. These are all things that REPaIR has in common with our technique, however, where REPaIR differs is in the method used to determine how the feedback is erroneous and the resulting limitations. Unlike our work, REPaIR requires a non-erroneous reward signal, such as an environmental reward signal, which REPaIR uses to evaluate the feedback signal from the human trainer to determine where it is unreliable. This frees REPaIR from one limitation of our work, which is that our technique requires that its user chose a model for the bias signal with some degree of accuracy. However, it requires an unbiased environmental reward, limiting REPaIR. Furthermore, using such an environmental reward to clean the human feedback signal may re-introduce some of the issues of learning from an environmental reward, such as a large degree of separation between a critical action and feedback, ultimately resulting from that action. This should not be taken to imply that REPaIR offers no benefits over learning from just the environmental reward. However, a programmer should consider what environmental reward functions are available when choosing between REPaIR, our work, or other algorithms for cleaning up erroneous human feedback. An avenue for future study would be the synthesis of techniques utilized in REPaIR and those from our work. Unfortunately, due to how recently REPaIR has been made available, we have not had the chance to incorporate such work into this body of research; nor to evaluate the techniques comparatively, given their different limitations.

Cannon and Anwar [6] developed another technique for improving the ways in which human interactive reinforcement learning algorithms incorporate feedback. In particular, they studied using human attention to guide agent exploration. Exploration is an aspect of reinforcement learning currently neglected by many interactive reinforcement learning algorithms, such as Deep TAMER. They applied their technique of model calibration to Deep TAMER in the domain of Montezuma's Revenge, an infamously difficult Atari video game available for machine learning research as part of the Open AI Gym package. Montezuma's Revenge is particularly difficult for machine learning algorithms due to a high degree of separation, not only between tasks and rewards, but also between tasks and the reasons for carrying out the sub task; such as the need to pick up items for use in other rooms. In their study, Cannon and Anwar utilized uncertainty estimates to direct agent exploration and studied the use of language as a feedback mechanism. They attempted to use language feedback to provide supervised attention. They used language feedback to generate saliency maps of their visual input to guide exploration, however, they saw no significant improvement or worsened performance on Montezuma's Revenge. After testing variations limiting the salient information that was passed to the model, they eventually settled on a model calibration technique where they used a Bayesian neural network to provide uncertainty estimates for each of the agent's available actions. These calibrations were then utilized with an upper confidence bound styled algorithm to choose actions. This technique was able to take advantage of the language feedback and saliency maps that had previously been ineffective in improving agent performance. This combination of techniques resulted in an agent that successfully explored and reached a much higher number of game states than the Deep TAMER baseline to which they compared their work to. While the topic of exploration in interactive reinforcement learning was not the primary focus of our own work, it is an issue that we encountered. We did not incorporate calibrated models into our research due to the recent nature of Cannon and Anwar's work. However, studying interaction and compatibility between their techniques and our own presents an interesting avenue for future research.

A number of these studies show that the presentation of a learning agent has an effect on the way that humans provide feedback to that agent, which raises many questions regarding agent presentation and inherent human bias when it comes to providing feedback. While many of these works sought to understand how the agent's appearance effected the feedback given and attempted to account for the human factors that affect feedback being provided, most did not address or study how the agent's perceived gender had an effect on the feedback given. Furthermore, few have sought to account for human bias, though some have attempted to correct for erroneous feedback. Therefore, while there is research in this field on how the appearance of an agent affects the amount of feedback given and possibly the quality of feedback given, there is currently a lack of research as to how the presented gender of an agent affects feedback given by a human subject. And while there exists some research on accounting for human training bias as it regards providing feedback to an agent regardless of gender or appearance, all existing techniques have their own limitations regarding either the sorts of errors they can account for or the nature of the problems upon which they can be utilized. So, while any new technique will, undoubtedly, have its own limitations, there is currently a significant space for new techniques with differing limitations from the existing techniques. As a result of the lack of research into the effects of human bias regarding the presented gender of an agent, we have taken the first steps to explore whether a statically relevant bias exists. We have also focused our efforts on accounting for human bias as it regards interactive reinforcement machine learning.

2.4 Related Work in the Social Sciences

Because of the psychological and sociological nature of the effects we studied, regarding the gendered appearance of computer agents, work in fields other than machine learning is relevant in order to understand how the perceived gender of an agent could affect the feedback it receives from a human trainer. Straus and Stewart[29] performed a nationally representative study on corporal punishment by American Parents and found, among other things, that frequency and severity of punishment were higher for boys. Similarly, Shaw and Braden[27] found a weak effect of race on the use of corporal punishment in a school setting in America, and a stronger effect of gender, with African-American students more likely to receive punishment than white students and with male students more likely to receive corporal punishment than female students. In further research conducted by Younger and Warrington [37], it was discovered that teachers gave more positive attention and support to girls, with teachers prepared to be more lenient and tolerant with girls than with boys, identifying that it was the teachers' attitudes towards the gender of the students that was driving the discrepancies. And Gregory[15] found that school aged males accounted for 81.6% of incidents of physical disciple, resulting in boys being four times more likely to be stuck by school administrators and teachers than girls, and that males were more likely to be suspended. However, this trend does not stop with school aged children. Starr[28] found that among federal cases in the United States, males received, on average, 63% more prison time for the same crime as a female. Starr also found that males 'faced a modestly, but significantly higher, probability of a charge before a district judge' during the filing stage after an arrest. Starr also found that female offenders were also more likely to be given the option of alternatives to a prison sentence, such as probation or fines for the same crime. Even the entertainment industry has this trend, as discovered by Downs and Gowan[11] who found that prime-time television programs when analyzed for frequencies of positive reinforcement and punishment among performers based on age and sex found that males were more likely to receive punishment and adult males would receive more punishment overall. These works collectively suggest a hypothesis that an agent that presents as male will be likely to receive more punishment, and punishment of a greater intensity, than one that presents as female; although it is unclear from this previous research how anthropomorphic the agent's presentation would need to be to invoke such an effect. A limitation of the sociological research is the potential for the behavior of the subject studied to vary with gender. Although many of these papers use statistical techniques to account for such an effect, the possibility must be considered that the differences in feedback reflect primarily differences in behavior and not a bias. This issue can be addressed by our research because it presents a situation in which the behavior of the subjects receiving feedback is entirely identical and all statistical differences in feedback given must result from instructor bias.

It is worth noting that there are other methods for humans to teach agents beyond giving feedback which include providing demonstrations such as in the work of Cakmak and Lopes [5] or providing a curriculum as in the work by Khan, Zhu, and Mutlu [21]. These methods are affected by human factors in different ways and will need different techniques to address them. However they can be combined with teaching from human feedback, and, in those cases, our technique can be used.

Chapter 3 Algorithms

In the following sections, we present our algorithms for handling bias in human trainer provided feedback.

Our first entry is our general BASIL technique. We then provide the BASIL-TAMER algorithm which is the result of applying the BASIL technique the IRL algorithm TAMER. We also provide algorithms for our Adaptive Variance, Zero Silence, and Average Silence BASILTAMER Extensions. We then provide our Deep Learning algorithm for Deep Neural Networks, the Deep BASILTAMER Algorithm. Finally, we provide our Multi-Circumstance Aware Deep BASILTAMER Algorithm, which is an adaptation of the BASILTAMER Algorithm designed to incorporate multiple bias distributions into the BASIL technique.

3.1 The BASIL technique

Our efforts to complete our goal of developing a technique and algorithms for interactive reinforcement learning agents that account for bias in feedback given by human trainers begins with the Bias Adaptive Statistical Inference Learning (BASIL) technique, which is a general technique for handling bias in human provided feedback. At a high level, the BASIL algorithm can be described as follows. We first must determine how to model the bias distribution. Here, we assume that bias is being drawn from some distribution that can be modeled. It is an important limitation of the technique that the distribution chosen must be one with a computable maximum likelihood function. Second, we collect some number of instances of feedback from the human trainer recording both the feedback value and the combination of state and action it was given in response to. Third, use the estimation maximization algorithm by Dempster, Laird, Rubin [10] to compute a maximum likelihood estimate of the human trainer's heuristic utility evaluations using the feedback history collected. At a high level this consists of making an initial guess for the human's heuristic utility evaluations, from that initial guess calculating the maximum likely values for the parameters of the bias distribution, and, then, from the new bias distribution parameters, calculating a new guess for the human trainer's heuristic utility evaluation. This EM step repeats until values converge within acceptable tolerances for your application. Finally, the computed estimate of the human's utility function can be used either to directly determine the agent's policy, such as in the case of the algorithm we present later, or to be combined with other data, such as environmental reward, to determine policy.

BASIL Algorithm for Normal Distributions

For the purpose of our experiments with the BASIL technique, we used the BASIL technique assuming a normal distribution for bias to develop a simple BASIL algorithm that computes a new estimate of the human trainer's utility function after each feedback instance and determines the agent's policy directly from the current estimated utility function. A more detailed view of the BASIL algorithm is shown as Algorithm 1.

```
Algorithm 1: A simple BASIL algorithm for normally distributed bias
  Result: Learned Policy $\lambda$
  $\lambda \leftarrow randomPolicy(), h \leftarrow \langle\rangle, t \leftarrow 0, \mathrm{Q}(s'a') \leftarrow 0$ for all $s'a' \in S, A$
  while Learning has not terminated do
      $s_t \leftarrow observeState()$
      $a_t \leftarrow \lambda(s_t)$
      $applyAction(a_t)$
      $f_t \leftarrow getFeedback()$
      $h \leftarrow \langle h_0, ...h_{t-1}, (s_t, a_t, f_t)\rangle$
      Q' $\leftarrow expectationMaximizationStep(\mathrm{Q}, h)$
      while $\sum_{s',a' \in S,A}$ | $Q'(s', a') - Q(s', a')$ | ¿ tolerance do
          Q $\leftarrow$ Q'
          Q' $\leftarrow expectationMaximizationStep(\mathrm{Q}, h)$
      end
      for s' $\in S$ do
          $\lambda(s') \leftarrow argmax_{a' \in A}\mathrm{Q}(s', a')$
      end
      $t \leftarrow t + 1$
  end
```

```
Algorithm 2: expectationMaximizationStep()
  Input: Q,$h$
  Result: Update to the estimate of the utility function Q'
  diff $\leftarrow 0, c \leftarrow 0, \mu \leftarrow 0$, Q'$(s'a') \leftarrow 0$ for all $s'a' \in S, A$
  for (s,a,f) $\in h$ do
      diff $\leftarrow$ diff $+ f - \mathrm{Q}(s, a)$
      $c \leftarrow c + 1$
  end
  $\mu \leftarrow \frac{\text{diff}}{c}$
  for (s',a') $\ni$ (s',a',f) $\in h$ do
      $f' \leftarrow \mathrm{avg}(f \ni (s', a', f) \in \mathrm{h})$
      Q'$(s'a') \leftarrow f - \mu$
  end
  return Q'
```

Our simple BASIL Algorithm, (Algorithm 1), begins by initializing a random policy, an empty history, and q-values of zero for all state action pairs in our state and action spaces. The q-values, in this algorithm, are related to the q-values used in q-learning.[35] However, in our algorithm, the q-values are the human trainers' estimates of the "utility" of taking an action in a state, rather than being calculated from the immediate environmental reward. After this initialization, the algorithm begins a loop that lasts throughout the training process. In this loop, first the current state is observed. Next, the current policy is consulted to determine what action to take in the observed state. The policy here consists of a mapping of states and actions to take in them. Then the action is executed within the state, and then feedback is collected. In the formulation, given here, feedback is assumed to always be given, due to the discrete non-real-time nature of our experimental environment. However, it is a trivial matter to modify the algorithm to return to the beginning of the loop if feedback is not given. Next, the history is updated by appending the the state observed, action taken, and feedback received as a triple. Next, we begin a looping process where EM is used to update our q-value estimates. In this process, we first calculate the difference between the feedback actually received and our estimate of the q-value for the state observed and action take within each element of our history. We then calculate the mean of these differences and modify our q-value estimation for each previously observed state action and action pair by subtracting the measured mean from the previous q-value estimation for those pairs. This results in a new estimated q-function. Then we measure the total change between our new and old estimations of the q-function by summing the absolute values of the difference between their estimations for each state and action. Finally, we check if the total change is less than the user provided tolerance. If it is not, we repeat the EM loop. Once the difference between successive versions falls below the user defined tolerance, the most recent q-value estimate is left as the estimation that will be used going forward. Finally, a new policy is computed by finding for each state the action that will result in the highest estimated q-value, and this action is set as the on-policy action for the observed state. Once the new policy is computed, we return to the beginning of the main loop and repeat the process until training is complete.

The expectation maximization step in Algorithm 2 is the only part of the algorithm that depends upon the bias values being normally distributed, by modifying this sub-routine (seen in Algorithm 2) the algorithm can be adapted to other types of distributions. Using a normal distribution makes the expectation maximization step relatively simple because the symmetry and other characteristics of a normal distribution allows the calculation to be simplified such that it does not require integration.

The algorithm, as presented, does not consider the trade-off between exploration and exploitation as is the case with many interactive machine learning algorithms. The reason for this is that the inclusion of a human trainer, who can react dynamically, allows the agent to always exploit, that is to chose the action it currently believes to the be the best course of action, while allowing the human trainer to encourage exploration via their feedback signal. In our experiments, the BASIL algorithm and the other interactive reinforcement learning algorithms are given the benefit of an ϵ-greedy shell agent that, during the training episodes, probabilistically chooses between the action chosen by the underlying algorithm and a random action. This is done because the simulated human trainers used in our experiments, naturally, cannot be as dynamic as real human trainers.

3.2 BASILTAMER Algorithm

While the general form of the BASIL technique we have already presented fulfills our secondary goal at a theoretical level, in practice the implementation presented so far has several limitations in its applicability and scope. Notably, the first BASIL algorithm we present (Algorithm 1) assumes a state space that can be fully enumerated and one where it is reasonable to observe and receive feedback for taking every possible action in each possible state that we intend our learned policy to cover. Furthermore, it has no mechanism to transfer learning between similar states. Additionally, even where the algorithm can be applied, the method of directly storing q-values and on-policy actions can be extremely resource intensive in terms of both computation time and memory. While these limitations are not unique to our algorithm, we feel it is necessary to expand the BASIL technique so that it can be used in problems with large state spaces as this category includes most real world problems, and many other domains of interest. Therefore in order to more fully complete our research, we have developed more practical algorithms using the BASIL technique. The first of these is a modification of the TAMER algorithm to incorporate BASIL. Here we present the BASILTAMER Algorithm as Algorithm 3.

The classical TAMER algorithm uses a vector of weight values that correspond to the feature set chosen for a given problem. This weight vector when multiplied with the vector of features for a particular state in the problem is intended to give the value of the state as estimated by TAMER's human trainer. To learn the values of the weights, TAMER first initializes its weights to all zeros and then enters a loop where it first selects an action, according to its current weight vector, and then performs this action. If the human trainer then gives it feedback, it updates its weights by calculating the utility estimate for both the state preceding the action and the state resulting from it, taking the difference of these and comparing this difference to the feedback given by the human trainer. Assuming that these two values are not the same, TAMER then updates each weight by multiplying together the error, the difference in feature values between the two states that correspond to the weight for each weight and a learning rate hyper-parameter, and adding the result of this multiplication to the weight whose associated feature produced it. After updating its weights, TAMER simply continues the loop until training is terminated. TAMER performs action selection by using its current weights to predict the difference in utility between its current state and the state resulting from each action and choosing the action that results in the greatest positive or least negative difference. TAMER does assume the ability to accurately predict the feature vectors of states resulting from taking each action available to it at all times.

For our BASILTAMER algorithm with a normal distribution we make several additions. First, we address the parameters of a normal distribution. Mathematically, a normal distribution is parameterized on its median μ and variance σ , however, variance, in this case, corresponds solely to noise in the feedback signal and, as such, cannot be helpfully addressed by BASIL techniques, as the only counter to noise is

Algorithm 3: The BASILTAMER Algorithm

```
Input: $\alpha, update\_interval, tolerance$
$t \leftarrow 0$
$\mu \leftarrow 0$
$h \leftarrow \langle\rangle$
$steps\_since\_update \leftarrow 0$
$\vec{w} \leftarrow \vec{0}$
$\vec{f_{t-2}} \leftarrow \vec{0}$
$\vec{f_{t-1}} \leftarrow \vec{0}$
$a \leftarrow ChooseAction(s_t, \vec{w})$
$TakeAction(a)$
while $Learning\ has\ not\ terminated$ do
    $t \leftarrow t + 1$
    $steps\_since\_update \leftarrow steps\_since\_update + 1$
    if $steps\_since\_update \geq update\_interval$ then
        $\mu \leftarrow \mu\_Update(\mu, \vec{w}, h, tolerance)$
    if $t \geq 2$ then
        $r_{t-2} \leftarrow getHumanFeedback()$
        $h \leftarrow \langle h_0, ...h_{t-1}, (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2})\rangle$
        if $r_{t-2} \neq 0$ then
            $\vec{w} \leftarrow UpdateWeights(r_{t-2} - \mu, \vec{f_{t-1}}, \vec{f_{t-2}}, \vec{w}, \alpha)$
    $a \leftarrow ChooseAction(s_t, \vec{w})$
    $TakeAction(a)$
    $s_t \leftarrow GetState()$
    $vecf_{t-2} \leftarrow vecf_{t-1}$
    $vecf_{t-1} \leftarrow getFeatureVec(s_t)$
end
```

Algorithm 4: $\mu\_Update() for the BASILTAMER Algorithm$

```
Input: $\mu, \vec{w}, h, tolerance$
$ne\vec{w}\_w \leftarrow \vec{0}$
while $|ne\vec{w}\_w - \vec{w}| \geq tolerance$ do
    $\vec{w} \leftarrow ne\vec{w}\_w$
    $differences \leftarrow \langle(\vec{f_{t-1}} \times \vec{w} - \vec{f_{t-2}} \times \vec{w} - r_{t-2}) for (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h\rangle$
    $\mu \leftarrow mean(differences)$
    $adjusted\_history \leftarrow \langle(\vec{f_{t-1}} - \vec{f_{t-2}}, r_{t-2} - \mu) for (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h\rangle$
    $ne\vec{w}\_w \leftarrow LinearLeastSquaresFit(adjusted\_history)$
end
return $\mu$
```

more data and a slower learning rate. Therefore, for simplicity and efficiency, this BASILTAMER algorithm is concerned only with the median of the bias. It should be noted that this does not necessarily hold for other non-normal distributions with more complicated variance parameters. The median is represented by a variable μ initially set to zero. The TAMER algorithm is then allowed to perform its main loop for several iterations with only two changes from the classical TAMER algorithm. When receiving feedback from the human trainer the value of the feedback is adjusted by subtraction of μ before being used to update the weight vector, and the history of unmodified feedback values, feature vectors, and actions taken is stored for future use. However, after a number of iterations determined by its update interval hyper-parameter, the value of the median is recalculated using Expectation Maximization and, simultaneously, the weight vector is updated by the same process. To do this Expectation Maximization step, which is the $\mu\_Update()$ step in the pseudo-code, a new weight vector is first set to zero, then a loop is entered. In this loop the new weights are used to calculate the predicted feedback for each action the agent has taken so far in its learning history, in the same manner that TAMER normally predicts rewards. Then these predictions are compared to the actual feedback for each action. The average difference between predictions and the corresponding historical rewards is then set as the new bias median. A new weight vector is then calculated by using the newly calculated bias median to produce a modified version of the history of actions and feedback values, where the feedback values are modified by having the median bias subtracted from them. This modified history is then used to create the new weight vector. Because TAMER uses a linear model for the human trainer's utility, the new weight vector can be calculated by performing a least squared error fit for the features to the modified feedback. This process of using the weight vector to calculate the median bias and then using the median bias to calculate a new weight vector is then repeated until it converges such that the difference between the weight vectors calculated in consecutive steps is within the predetermined tolerance of the implementation.

The $\mu\_Update()$ step and the adjustment of the feedback received before each weight update are the only parts of the algorithm that depends upon the bias values being normally distributed. By modifying these, the algorithm can be adapted to other types of distributions. Using a normal distribution makes the Expectation Maximization step relatively simple because the symmetry and other characteristics of a normal distribution allows the calculation to be simplified such that it does not require integration and the variance parameter can be safely ignored.

Adaptive Variance BASILTAMER Extension

To address the third goal of this research, we begin by addressing variance in human trainer provided feedback with three extensions of the BASILTAMER algorithm. The first of these is the Adaptive Variance BASILTAMER Extension which can be seen in Algorithm 5 and Algorithm 6.

The Adaptive Variance BASILTAMER Extension relies on dynamically adjusting the algorithm's learning rate to account for variance. This is possible due to the

Algorithm 5: The Adaptive Variance BASILTAMER Algorithm

```
Input: $\alpha, \beta, update\_interval, tolerance$
$t \leftarrow 0$
$\mu \leftarrow 0$
$h \leftarrow \langle\rangle$
$steps\_since\_update \leftarrow 0$
$\vec{w} \leftarrow \vec{0}$
$\vec{f_{t-2}} \leftarrow \vec{0}$
$\vec{f_{t-1}} \leftarrow \vec{0}$
$a \leftarrow ChooseAction(s_t, \vec{w})$
$TakeAction(a)$
while $Learning\ has\ not\ terminated$ do
    $t \leftarrow t + 1$
    $steps\_since\_update \leftarrow steps\_since\_update + 1$
    if $steps\_since\_update \geq update\_interval$ then
        $\mu, \sigma \leftarrow \mu\_Update(\mu, \vec{w}, h, tolerance)$
    if $t \geq 2$ then
        $r_{t-2} \leftarrow getHumanFeedback()$
        $h \leftarrow \langle h_0, ...h_{t-1}, (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2})\rangle$
        if $r_{t-2} \neq 0$ then
            $\vec{w} \leftarrow UpdateWeights(r_{t-2} - \mu, \vec{f_{t-1}}, \vec{f_{t-2}}, \vec{w}, \alpha \times e^{\frac{\sigma}{\beta^2}})$
    $a \leftarrow ChooseAction(s_t, \vec{w})$
    $TakeAction(a)$
    $s_t \leftarrow GetState()$
    $vecf_{t-2} \leftarrow vecf_{t-1}$
    $vecf_{t-1} \leftarrow getFeatureVec(s_t)$
end
```

Algorithm 6: $\mu\_Update() for Adaptive Variance BASILTAMER$

```
Input: $\mu, \vec{w}, h, tolerance$
$new\vec{\_}w \leftarrow \vec{0}$
while $|new\vec{\_}w - \vec{w}| \geq tolerance$ do
    $\vec{w} \leftarrow new\vec{\_}w$
    $differences \leftarrow \langle(\vec{f_{t-1}} \times \vec{w} - \vec{f_{t-2}} \times \vec{w} - r_{t-2}) for (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h\rangle$
    $\mu \leftarrow mean(differences)$
    $\sigma \leftarrow standard_deviation(differences)$
    $adjusted\_history \leftarrow \langle(\vec{f_{t-1}} - \vec{f_{t-2}}, r_{t-2} - \mu) for (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h\rangle$
    $new\vec{\_}w \leftarrow LinearLeastSquaresFit(adjusted\_history)$
end
return$\mu, \sigma$
```

fact that the BASILTAMER algorithm's learning rate hyper-parameter controls the impact that each feedback instance has on the agent's model of the utility function. This means that if the learning rate is lowered, then the agent can better handle higher levels of variance in the feedback that it receives. However, if that learning rate is lowered more than it needs to be to account for the variance within the provided feedback, the result will be a dramatic increase in the time it takes the algorithm to learn the correct policy. Therefore, in order to combat variance in the human feedback provided to algorithms such as BASILTAMER, we dynamically adjust the learning rate based upon the observed variance. To adjust the learning rate we use a Gaussian function with the observed variance of the feedback, as given by BASIL, serving as the input to the function. To control the rate at which the learning rate is decayed with increased variance, we introduce an additional hyper-parameter, which we represent as beta, that controls the variance of the Gaussian function.

Zero Silence BASILTAMER Extension

Silence, or feedback withholding, can result in the agent not learning the correct policy to accomplish its task or goal, or in completing the task/goal but not optimally as the trainer did not provide sufficient training for the agent to learn the on-policy behavior.

The next two extensions of the BASILTAMER Algorithm address silence, another aspect of human trainer feedback inconsistency that we seek to address as part our third goal for this research. This first of these extensions to address the issue of silence is the Zero Silence BASILTAMER Extension which can be seen in Algorithm 7. When it comes to the issue of silence in human provided feedback to an IRL agent, the algorithms developed by Loftin et. all [25] address this issue, but do so only with significant limitation. These limitations are that the algorithms, SABL and ISABL, in provided feedback, work assume that feedback is defined only as positive or negative. These algorithms also assume that there are no numerical values for the feedback provided. Therefore, to address the issue of silence in the context of numerically valued feedback, we have developed the Zero Silence and Average Silence BASILTAMER Extensions.

The Zero Silence Extension assumes that the lack of feedback from a human trainer means that the feedback signal that would have been provided has a numerical value of zero. This approach is simple and limited, however, it is intuitive. The most obvious limitation is that the feedback that "should" have been given may not necessarily be precisely zero. However, if the feedback that the human trainer would have given is close to zero, then this answer will be approximately correct and it can then be further corrected by the regular workings of the BASILTAMER algorithm. It is expected that the majority of instances of silence would be cases where the feedback given would have been relatively neutral, as it is generally expected that where the human trainer would give a strongly non-neutral feedback they are unlikely to withhold it.

Algorithm 7: The Zero Silence BASILTAMER Algorithm

```
Input: $\alpha, update\_interval, tolerance$
$t \leftarrow 0$
$\mu \leftarrow 0$
$h \leftarrow \langle\rangle$
$steps\_since\_update \leftarrow 0$
$\vec{w} \leftarrow \vec{0}$
$\vec{f_{t-2}} \leftarrow \vec{0}$
$\vec{f_{t-1}} \leftarrow \vec{0}$
$a \leftarrow ChooseAction(s_t, \vec{w})$
$TakeAction(a)$
while Learning has not terminated do
    $t \leftarrow t + 1$
    $steps\_since\_update \leftarrow steps\_since\_update + 1$
    if $steps\_since\_update \geq update\_interval$ then
        $\mu \leftarrow \mu\_Update(\mu, \vec{w}, h, tolerance)$
    if $t \geq 2$ then
        $r_{t-2} \leftarrow getHumanFeedback()$
        if $r_{t-2} = None$ then
            $r_{t-2} \leftarrow 0$
        $h \leftarrow \langle h_0, ...h_{t-1}, (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2})\rangle$
        $\vec{w} \leftarrow UpdateWeights(r_{t-2} - \mu, \vec{f_{t-1}}, \vec{f_{t-2}}, \vec{w}, \alpha)$
    $a \leftarrow ChooseAction(s_t, \vec{w})$
    $TakeAction(a)$
    $s_t \leftarrow GetState()$
    $vecf_{t-2} \leftarrow vecf_{t-1}$
    $vecf_{t-1} \leftarrow getFeatureVec(s_t)$
end
```

Average Silence BASILTAMER Extension

The Average Silence Extension, in contrast to the Zero Silence Extension, is a more sophisticated algorithm for addressing the issue of silence in human trainer provided feedback. By calculating the average expected feedback values extracted from the BASILTAMER neural network we can provide the algorithm with values for the missing feedback instances. These estimated values are then used in the EM process to generate following estimations as the end process heads towards convergence. This involves modification of both the main body of the algorithm and the *μ_update*. The modifications to the main body consists of initializing the avg_silence to zero at the beginning and replacing feedback values of ”none” with the avg_silence after the ”none” feedback value is recorded in the history set, but before the feedback value is utilized to update the weight vector. This means that the ”none” value is what is recorded in the history set for the use of the *μ_update*. The *μ_update* first differs

Algorithm 8: The Average Silence BASILTAMER Algorithm

```
**Input:** $\alpha, update\_interval, tolerance$
$t \leftarrow 0$
$\mu \leftarrow 0$
$avg\_silence \leftarrow 0$
$h \leftarrow \langle\rangle$
$steps\_since\_update \leftarrow 0$
$\vec{w} \leftarrow \vec{0}$
$\vec{f_{t-2}} \leftarrow \vec{0}$
$\vec{f_{t-1}} \leftarrow \vec{0}$
$a \leftarrow ChooseAction(s_t, \vec{w})$
$TakeAction(a)$
while Learning has not terminated do
    $t \leftarrow t + 1$
    $steps\_since\_update \leftarrow steps\_since\_update + 1$
    if $steps\_since\_update \geq update\_interval$ then
        $\mu, avg\_silence \leftarrow \mu\_Update(\mu, \vec{w}, h, tolerance)$
    if $t \geq 2$ then
        $r_{t-2} \leftarrow getHumanFeedback()$
        $h \leftarrow \langle h_0, ...h_{t-1}, (\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2})\rangle$
        if $r_{t-2} = None$ then
            $r_{t-2} \leftarrow avg\_silence$
        $\vec{w} \leftarrow UpdateWeights(r_{t-2} - \mu, \vec{f_{t-1}}, \vec{f_{t-2}}, \vec{w}, \alpha)$
    $a \leftarrow ChooseAction(s_t, \vec{w})$
    $TakeAction(a)$
    $s_t \leftarrow GetState()$
    $vecf_{t-2} \leftarrow vecf_{t-1}$
    $vecf_{t-1} \leftarrow getFeatureVec(s_t)$
end
```

Our third extension, the Average Silence Extension, also addresses the issue of silence in human trainer feedback. While the Zero Silence Extension does meet our goal of addressing silence in provided feedback, it does so simplistically, and may not be sufficient, depending upon the work being done, to address silence. Therefore, we present the Average Silence BASILTAMER Algorithm in Algorithm 8.

```
Algorithm 9: $\mu\_Update() for Average Silence BASILTAMER$
  Input: $\mu, \vec{w}, h, tolerance$
  $new\vec{\_w} \leftarrow \vec{0}$
  while $|new\vec{\_w} - \vec{w}| \geq tolerance$ do
      $\vec{w} \leftarrow new\vec{\_w}$
      $differences \leftarrow \langle(\vec{f_{t-1}} \times \vec{w} - \vec{f_{t-2}} \times \vec{w} - r_{t-2}) for(\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h : r_{t-2} \neq None\rangle$
      $silence\_values \leftarrow \langle(\vec{f_{t-1}} \times \vec{w} - \vec{f_{t-2}} \times \vec{w}) for(\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h : r_{t-2} = None\rangle$
      $\mu \leftarrow mean(differences)$
      $avg\_silence \leftarrow mean(silence\_values)$
      $adjusted\_history \leftarrow \langle(\vec{f_{t-1}} - \vec{f_{t-2}}, r_{t-2} - \mu) for(\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h : r_{t-2} \neq None\rangle \cup \langle(\vec{f_{t-1}} - \vec{f_{t-2}}, avg\_silence - \mu) for(\vec{f_{t-1}}, \vec{f_{t-2}}, a, r_{t-2}) \in h : r_{t-2} = None\rangle$
      $new\vec{\_w} \leftarrow LinearLeastSquaresFit(adjusted\_history)$
  end
  return $\mu, avg\_silence$
```

where the set of differences is calculated. Only elements of the history where r_{t-2} is not "none" are used in the calculation. Elements where r_{t-2} is "none" are used to calculate a separate set of silence values as the products of the weight vector and change in factors for that element of the history. The avg_silence is then, simply, the mean of the silence values. The final change is that when the new weight vector is calculated the error value is computed as normal for elements of the history where the feedback is not "none", however for elements of the history where the feedback is "none" the avg_silence is utilized in place of the feedback. The sets resulting from these two cases are then unionized and this union is used to perform the linear least square fit to calculate the new weights.

3.3 Deep BASILTAMER Algorithm

Deep Learning is a subset of machine learning which results in the production of applications and services that improve automation, as well as performing a variety of tasks, such as data analysis, by utilizing very large data sets and it relies on complex and layer heavy neural networks to process and identify that data which can include an image, sound, or even texts. Many modern technologies are the result of Deep Learning, meaning that Deep Learning lies behind many everyday products and services, as well as emerging technologies, such as self-driving cars, voice-enabled remotes or menus, object recognition, text processing, image processing and virtual assistants, just to name a few. Therefore, developing the BASIL technique into a Deep Neural Network is necessary as the field of Deep Learning is becoming more common and prevalent. This algorithm, Algorithm 10 and Algorithm 11, was developed for the propose of applying the BASIL technique to Deep Neural Networks in conjunction with Deep TAMER to facilitate Deep Learning in accordance with our second goal for this research. However, in order to develop the Deep BASILTAMER algorithm we must first incorporate TAMER into a Deep Neural Network.

Adapting BASIL to Deep TAMER

The addition of the BASIL technique to the Deep TAMER algorithm does not substantially change the majority of the algorithm however we will walk through the pseudo-code in full for unfamiliar readers. Those substantially familiar with the original Deep TAMER algorithm will find most of the changes in the ”DDN Updates” and ”BASIL Updates” subsections.

Deep BASILTAMER: Variable and Subroutine Meanings

$'i'$ is a counter variable for the number of iterations of the main loop. $'j'$ is a counter variable for the number of feedback instances processed. $'k'$ is a counter variable for the number of stochastic gradient descent updates preformed on the neural network. $'\mu'$ is the current estimate of the the mean bias in the feedback provided by the human trainer. $'D'$ is a record of tuples of all feedback instances and each set of state, action, and timestamps that the feedback might apply to by the weighting scheme. s_i is an observation of the state in cycle $'i'$ represented by the two most recent graphical representations at that point. a_i is the action selected to take in cycle $'i'$. $\hat{H}_k(s_i, a)$ is the results of feeding i-the state and action into the k-th version of the neural net which is an estimation of the human's utility function by which they judge agent actions. y is a feedback instance consisting of a feedback value h and time stamp t^f. y_j is the j-th feedback instance. Similarly D_j is the subset of D that corresponds to y_j. D_{adj} is a subset of D such as D_j that has been adjusted such that the feedback value of each feedback instance has had the current μ subtracted from them.$w(x, y_j)$ is the importance weighting function and as used by Warnell et al. is a uniform continuous distribution from 0.2 to 4 seconds and gives value 0 for timestamps outside this range.

Action Selection

Action selection is done by simply taking the ArgMax of the available actions for the present state. This has one significant difference from classic TAMER. In classic TAMER, the weight vector is used as an estimate of the human evaluation of the state and the ArgMax calculation requires predicting the resultant state of each action such a that before and after states can be processed by the wait vector and the difference of the resulting values can be taken. In Warnell et. all's Deep TAMER formulation this is avoided by treating the human trainer's evaluation as being of a state and action pair rather than just as a state.[34] This is analogous to the difference between a Q value and a utility value in traditional reinforcement learning. We maintain Warnell et. all's technique in our action selection.

Handling Real Time Feedback

In order to have the agent play the game in real time while the human trainer provides feedback, the problem of assigning feedback instances to corresponding state action pairs had to be addressed. In Deep TAMER, this was achieved with importance weights that were used to distribute the weight of feedback instances among state action pairs that might have resulted in that feedback. Warnell et. all found that a continuous uniform distribution that assigned equal weights to all time stamps that occur between 0.2 and 4 seconds after a state action pair and 0 weight elsewhere performed best for them. We kept this formulation and therefore associated each feedback instance to all state action pairs such that the feedback will have fallen within the time interval.

DNN Updates

To update the Deep Neural Network, stochastic gradient descent (SGD) is used. SGD is an algorithm used to update the weights of a Neural Network that makes incremental updates by approximating the gradient of the hyper-surface of the error in the Neural Network's prediction as a function of its weights and takes an incremental step in the direction of the gradient and, by doing so, seeks a local minimum of the error function. Theoretically, with an infinitesimal incremental step size, SGD will find the global minimum. Adding BASIL makes only one substantial change to this process which is that before feedback values are used to perform an SGD update, they are adjusted by having the current estimated mean bias subtracted from them. In order to make optimal use of its computational time the algorithm does not only perform SGD updates when receiving new feedback, but instead performs the SGD updates at regular intervals through use of an update interval hyper parameter with sampling from all previous state actions and feedback triples used to perform these incremental updates.

Deep BASILTAMER Updates

The primary addition to the algorithm to incorporate the BASIL technique is the process by which the current estimate of the mean bias contained in the human feedback is updated. This process is performed on a regular interval of the number of feedback instances received as determined by our hyper parameter. In order to perform this update, Expectation Maximization is done between the last fully connected part of the Neural Network, the estimated parameters of the bias distribution, and the history of feedback state and action sets observed. This consists of an iterative process where a new mean bias value is calculated as the average difference between the human feedback predicted by the current Neural Network for a state action pair and what was actually observed. This new mean value is then used to compute a new back end of the Neural Network by finding a best fit for the observed feedback instances and state action pairs after correcting for the current mean bias estimate using the method previously described. This new Neural Network is then used to compute a new mean bias value and then the process repeats until the differences between the Neural Networks of consecutive generations is within a pre-defined tolerance. Note that this iterative process is performed with two copies of the Neural Network and does not change the weight values of the Neural Network as utilized in the main algorithm. Rather, only the median bias estimate is changed which, in turn, is utilized to adjust the feedback values used to update the Neural Network. This done to prevent wild jumps in the agent's policy, which could confuse and frustrate a human trainer. The algorithm for the *μ_update* can be seen in Algorithm 11.

Algorithm 10: The Deep BASILTAMER Algorithm

```
Input: pre-initialized $\hat{H}_0$, step size $\nu$, buffer update interval $\beta$, bias update
       interval $\epsilon$, tolerance $\theta$
$i \leftarrow 0$
$j \leftarrow 0$
$k \leftarrow 0$
$\mu \leftarrow 0$
$D \leftarrow \langle\rangle$
while Learning has not terminated do
    $i \leftarrow i + 1$
    $s_i \leftarrow observe\_state()$
    $a_i \leftarrow \arg max_a \hat{H}_k(s_i, a)$
    $TakeAction(a_i)$
    $x_i \leftarrow (s_i, a_i, t_i, t_{i+1})$
    if new feedback $y = (h, t^f)$ then
        $j \leftarrow j + 1$
        $y_j \leftarrow y$
        $D_j \leftarrow \langle(x, y_j)|w(x, y_j) \neq 0\rangle$
        $D \leftarrow D \cup D_j$
        $D_{adj} \leftarrow \langle(x, y_j - \mu) for (x, y_j) \in D_j\rangle$
        $\hat{H_{k+1}} \leftarrow SGD\_update(\hat{H}_k, D_{adj})$
        $k \leftarrow k + 1$
        if $mod(j, \epsilon) = 0$ then
            $\mu \leftarrow \mu\_Update(\mu, \hat{H}_k, \hat{H}_0, D, \theta)$
    if $mod(i, \beta) = 0$ then
        $D_s \leftarrow sample(D)$
        $D_{adj} \leftarrow \langle(x, y_j - \mu) for (x, y_j) \in D_s\rangle$
        $\hat{H_{k+1}} \leftarrow SGD\_update(\hat{H}_k, D_{adj})$
        $k \leftarrow k + 1$
end
```

Multi-Circumstance Aware Deep BASILTAMER

Typically, in experiments, IRL agents learn from one human trainer, who trains them to fully accomplish the task. However, as the domain in which the IRL agent learns

```
Algorithm 11: $\mu\_Update() for Deep BASILTAMER$
    Input: $\mu, \hat{H}_k, \hat{H}_0, D, \theta$
    $CNN, FCNN \leftarrow split(\hat{H}_0)$
    $old\_hatH_k \leftarrow \hat{H}_k$
    $new\_hatH_k \leftarrow \hat{H}_0$
    while $|new\_hatH_k - old\_hatH_k| \geq \theta$ do
        $old\_hatH_k \leftarrow new\_hatH_k$
        $differences \leftarrow \langle (new\_hatH_k(s_i, a_i) - y_j) for((s_i, a_i, t_i, t_{i+1}), y_j) \in D \rangle$
        $\mu \leftarrow mean(differences)$
        $D_{adj} \leftarrow \langle (x, y_j - \mu) for(x, y_j) \in D \rangle$
        $new\_hatH_k \leftarrow concatenate(CNN, QuadraticLeastSquaresFit(D_{adj}))$
    end
    return$\mu$
```

to complete its task increase in size and complexity, the data the one human trainer can provide will become insufficient. To apply IRL techniques to larger, real world domains it becomes increasingly necessary to integrate data from multiple human trainers to provide a sufficient body of feedback for the agent to learn the task. Additionally, the ability to take feedback from multiple trainers gives us the ability to utilize a large sample size of feedback data to compensate when domain expertise cannot be easily sourced. This, however, introduces multiple bias distributions into the feedback that a Deep Learning algorithm relies on to develop its policy. Therefore, the most likely solution to account for these multiple bias distributions is to incorporate them into the BASIL technique. The simplest way to incorporate multiple bias distributions into the BASIL technique is in a piece wise manner that treats each distribution entirely separately. We present here an algorithm for Multi-Circumstance Aware Deep BASILTAMER. (See Algorithm 12 and Algorithm 13.) We cover only the modified parts of the algorithm in this section and refer the reader to previous sections for a detailed explanation of the rest of the algorithm.

Multi-Circumstance Aware Deep BASILTAMER Variable and Subroutine Meanings

$'N_\mu'$ is the number of separate bias distributions being considered. $'\mu_l'$ is the current estimate of the the mean bias in the feedback provided by the human trainer under biasing conditions assigned index l which will be a unique integer from 1 to $'N_\mu'$. $'c'$ is an additional component of a feedback instance which gives the index number of the set of biasing conditions under which the feedback was given, and c_j is this component of the j-th feedback instance. n is a counter variable for the $\mu\_Update()$ subroutine which is used to have that subroutine work separately on the parameters for the bias distribution under each set of biasing conditions being considered.

Algorithm 12: Multi-Circumstance Aware Deep BASILTAMER Algorithm

```
Input: pre-initialized $\hat{H_0}$, step size $\nu$, buffer update interval $\beta$, bias update
       interval $\epsilon$, tolerance $\theta$, Number of separate feedback conditions $N_\mu$
$i \leftarrow 0$
$j \leftarrow 0$
$k \leftarrow 0$
$\mu_1, \mu_2, \dots \mu_{N_\mu} \leftarrow 0, 0, \dots 0$
$D \leftarrow \langle\rangle$
while Learning has not terminated do
    $i \leftarrow i + 1$
    $s_i \leftarrow observe\_state()$
    $a_i \leftarrow \arg\ max_a\ \hat{H}_k(s_i, a)$
    $TakeAction(a_i)$
    $x_i \leftarrow (s_i, a_i, t_i, t_{i+1})$
    if new feedback $y = (h, t^f, c)$ then
        $j \leftarrow j + 1$
        $y_j \leftarrow y$
        $D_j \leftarrow \langle(x, y_j)|w(x, y_j) \neq 0\rangle$
        $D \leftarrow D \cup D_j$
        $D_{adj} \leftarrow \langle(x, h_j - \mu_c) for (x, (h_j, t^f_j, c_j) \in D_j\rangle$
        $\hat{H_{k+1}} \leftarrow SGD\_update(\hat{H}_k, D_{adj})$
        $k \leftarrow k + 1$
        if $mod(j, \epsilon) = 0$ then
            $\mu_1, \mu_2, \dots \mu_{N_\mu} \leftarrow \mu\_Update(N_\mu, \mu_1, \mu_2, \dots \mu_{N_\mu}, \hat{H}_k, \hat{H}_0, D, \theta)$
    if $mod(i, \beta) = 0$ then
        $D_s \leftarrow sample(D)$
        $D_{adj} \leftarrow \langle(x, h_j - \mu_c) for (x, (h_j, t^f_j, c_j) \in D_s\rangle$
        $\hat{H_{k+1}} \leftarrow SGD\_update(\hat{H}_k, D_{adj})$
        $k \leftarrow k + 1$
end
```

Algorithm 13: $\mu\_Update()$ for Multi-Circumstance Aware Deep BASIL-TAMER

```
Input: $N_\mu, \mu_1, \mu_2, \dots \mu_{N_\mu}, \hat{H}_k, \hat{H}_0, D, \theta$
$n \leftarrow 0$
while $n < N_\mu$ do
    $n \leftarrow n + 1$
    $CNN, FCNN \leftarrow split(\hat{H}_0)$
    $old\_hatH_k \leftarrow \hat{H}_k$
    $new\_hatH_k \leftarrow \hat{H}_0$
    while $|new\_hatH_k - old\_hatH_k| \geq \theta$ do
        $old\_hatH_k \leftarrow new\_hatH_k$
        $differences \leftarrow \langle (new\_hatH_k(s_i, a_i) - y_j) for((s_i, a_i, t_i, t_{i+1}), y_j) \in D \rangle$
        $\mu_n \leftarrow mean(differences)$
        $D_{adj} \leftarrow \langle (x, h_j - \mu_n) for(x, (h_j, t_j^f, c_j) \in D | c_j = n \rangle$
        $new\_hatH_k \leftarrow concatenate(CNN, QuadraticLeastSquaresFit(D_{adj}))$
    end
end
return$\mu_1, \mu_2, \dots \mu_{N_\mu}$
```

Multi-Circumstance Aware Deep BASILTAMER DNN Updates

The only change to the DNN update process is that before feedback values are adjusted by having the current estimated mean bias for the set of conditions that correspond to the those which the feedback was given under subtracted from them. Instead of one universal mean for all feedback instances.

Multi-Circumstance Aware Deep BASILTAMER Updates

The primary change to the algorithm to incorporate the multiple bias distributions is to the $\mu\_Update()$ subroutine. This process is modified to be preformed entirely separately for each set of biasing conditions in turn. For each set of biasing conditions only feedback instances which have been given under that condition set are treated as extant, and the calculation of the iterative version of the back-end of the DNN in each iteration is preformed only with adjusted feedback instances from that set. Once convergence is reached for every and all sets of biasing conditions under which we have so far received feedback the subroutine returns the newly calculated means for each distribution.

Chapter 4 Methods: Experiments in Bias and the BASIL Technique

In the following sections, we present our experiments in human bias. The first experiment, Understanding Bias in Human Trainers, focuses on our first research goal of increasing our understanding of the ways in which factors, such as agent presentation, can influence feedback given to interactive reinforcement learning algorithms. By focusing on developing the first foray, we hope to discover if a learning agent's presented gender affects how the human trainer will provide feedback, i.e. if a gendered representation induces bias.

As part of completing our secondary goal for this research, our following experiment, the BASIL technique in a Gridworld, tests our technique to determine if it does, indeed, account for human bias in provided feedback to an IRL agent.

Our third experiment, BASILTAMER, continues building upon our secondary goal. It incorporates the BASIL technique into the TAMER algorithm to determine if our technique when applied to an IRL algorithm results in the agent not only learning the desired policy, but if it learns the desired policy quickly, and perhaps even more quickly, than the algorithm does so without the technique's application, or if the algorithm without the technique's application fails to learn the desired policy due to the bias within the feedback provided.

For our BASILTAMER algorithm, we developed three extensions: the Adaptive Variance, Zero Silence, and Average Silence BASILTAMER Extensions. These extensions seek to address our third goal for this research of combining our techniques for bias with techniques to address variance and inconsistency in feedback given by human trainers. These algorithm extensions were tested to determine if the techniques we had developed for handling inconsistencies in human provided feedback would improve the performance of the BASILTAMER algorithm. We compare these algorithms performance to those of BASILTAMER and TAMER.

Continuing with completing the second goal of our research, we address the addition of the BASIL technique to Deep TAMER. By applying the BASIL technique to Deep TAMER we seek to determine if the BASIL technique, when applied in the setting of a Deep Neural Network, results in significant performance improvements as human trainer bias is accounted for in the provided feedback.

Finally, in our last experiment, Multi-Circumstance Aware Deep BASILTAMER, we complete our research by developing our Deep BASILTAMER algorithm to draw upon multiple instances of human provided feedback which seeks to not only address human bias, but also the inconsistencies of variance and silence within that provided feedback. This addresses our third research goal as it concerns Deep Neural Networks.

4.1 Understanding Bias in Human Trainers

To better understand bias in human trainers, we chose to examine the effect of gendered representations of agents on human feedback. Our goal was to determine if such representations were subject to bias despite the human trainer being aware of the fact that the agent is not biological in nature. In this experiment, we examined human trainer provided feedback to a virtual agent navigating a grid world environment in a browser-based game. In this game, the agent was alternatively represented by a female pixel art character utilizing English female pronouns, named Alice, and by a male pixel art character utilizing English male pronouns, named William.

We also surveyed the human instructors for a number of demographic qualities to study their effect on the feedback given as part of the process of exploring the effects of an agent's gender on the feedback given in the process of interactive machine learning as certain qualities may result in a propensity for certain biases. The survey focused on collecting data to aid the researchers in attempting to answer the following questions:

1. Is there a statistically significant difference in the feedback given to an agent when presented with the female representation versus the male one across all human subjects and of the states presented in the environment?

2. Qualitatively, how do the feedback histograms differ for feedback given to the female representation of the agent versus the male one?

3. What demographic characteristics of human instructors have a statistically significant effect on feedback given to the agents?

4. In particular what effect does the instructor's reported gender have on feedback given to the agent?

5. Is there a cross effect between instructor gender and the agent's presented gender?

By answering these questions, we have gained a greater understanding of the role that perceived agent gender plays in training virtual characters, which satisfies our first research goal. We also briefly discuss how this knowledge can be used to improve how humans train agents.

Understanding Bias in Human Trainers: Task and Environment

In our experiments on the effects of presented gender, agents must navigate a virtual environment consisting of a grid of square tiles, which may be either passable, impassible, lethal hazards, or the goal. The agent's state consists of integer coordinates for it's grid tile and a facing direction which may be north, south, west, or east. The starting state is (0, 2, east). Valid, although not necessarily reachable states, include any combination of facing direction and grid coordinates corresponding to a passable, hazardous, or goal tile.

In any passable tile, the agent has the following available actions available actions:

1. Advance, which will move it onto the tile it is immediately facing, provided that the tile is not impassible.

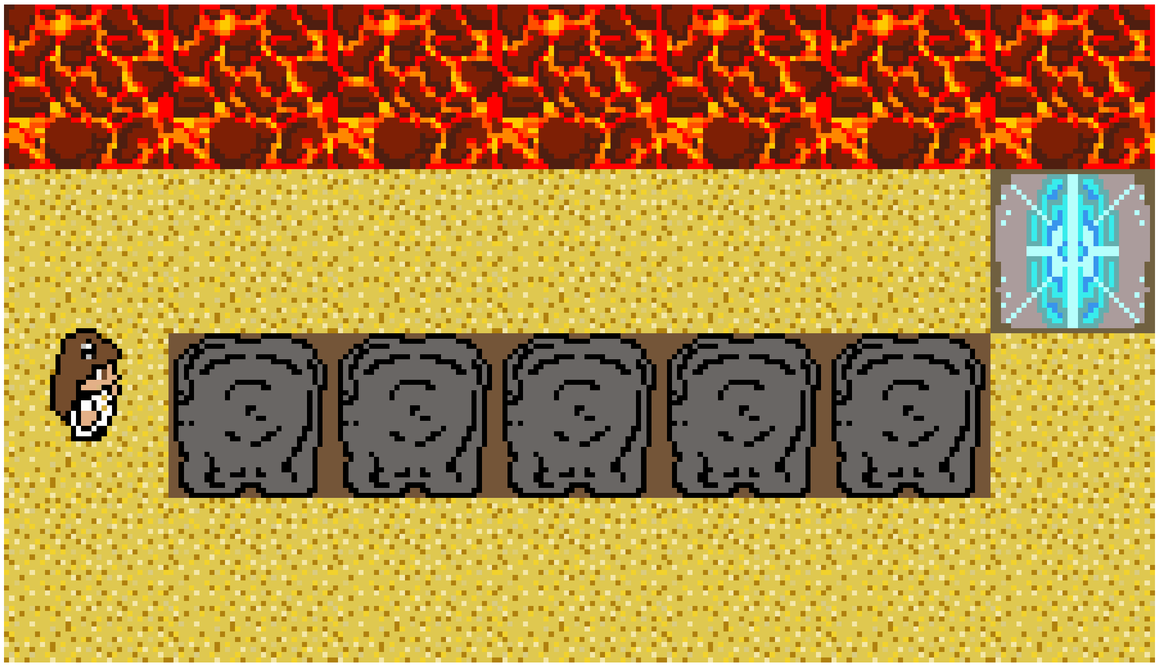


Figure 4.1: A visual representation of the grid navigation test environment. in the starting state of (0, 2, east). Sand textures represent fully passable tiles. Large stones represent impassible tiles. Molten rock represents hazard tiles that end the episode when entered and provide large negative environmental rewards, and the portal texture represents the goal tile that also ends the episode when entered and provide large positive environmental rewards. All tiles out of frame are impassible.

2. Make no change to the current state, which is the action taken when the agent is facing an impassible tile.
3. Turn left, which will rotate the agent's facing position one step counterclockwise.
4. Turn right, which will rotate the agent's facing position one step clockwise.

When the agent enters a hazardous or goal tile, the episode is considered complete and therefore no action can be taken in states that correspond to these tile types. Entering hazardous tiles results in a strong penalty (-1.0) for entering them in addition to ending the episode. Entering into a goal tile, by contrast, gives the agent a large reward (1.0) before ending the episode. All other actions that do not result in entering either a hazard or a goal tile give a very small penalty (-0.01). This is a common practise in reinforcement learning problems and is done to encourage the efficiency of agents, as well as to prevent learning being slowed by overly cautious agents. The particular grid configuration used in our experiment consists of a rectangular area bordered by impassible tiles with two paths to the one goal tile from the starting state (See Figure 4.1). This is an example of a popular reinforcement learning problem often referred to as the *cliff walker problem* [30]. The path to the north is shorter, but passes by hazardous tiles that a curious agent may venture into, while the lower path is long, but is bordered only by impassible tiles. This type of problem is difficult for traditional reinforcement learning agents because exploration of the optimal path is dis-incentivised by the possibility of exploring into hazardous states. This problem is also very suitable for our work because it has two paths to the goal of unequal efficiency and represents a simple case of a problem with good and better options for an agent to take. The maximum cumulative reward an agent can receive for an evaluation episode is 0.92 which corresponds to taking the upper path with no extraneous actions. Taking the lower path with no extraneous actions yields a cumulative reward of 0.89

Understanding Bias in Human Trainers: Testing for Bias Between Gendered Agent Representations

To evaluate the effect that perceived gender has on how humans train virtual agents, we conducted an evaluation of the feedback human trainers provided between two subjects with one primary independent variable. That variable was the representation of the virtual agent being trained. The representations provided to the human trainers are in figure 4.2. In this course of this research, human trainers were asked to watch a series of simulated learning episodes and to provide online feedback to the agent to help it learn to achieve the desired goal. This feedback was recorded and then analyzed with respect to the research questions posed earlier in this body of work.

Understanding Bias in Human Trainers: Human Trainer Populations

This experiment used 140 human subjects that completed the full study, and 13 additional subjects that partially completed the study. They were recruited via online postings and from a university student population. These two populations formed the vast majority of the subject pool. The online population was recruited by posting online in the popular web forum Reddit. Specifically, participants were drawn from a sub-reddit (a forum dedicated to a specialized topic) devoted to machine learning, who participated without compensation. The other was a population of sociology students who were recruited by their professors to participate voluntarily, also without compensation. The remaining portion of the subject pool was recruited by online posting on the social media platform Facebook. An attempt was made to utilize snowball sampling with the Reddit and Facebook populations, however, we believe that very few members of either population were recruited in this matter. Populations were kept separate by the use of different instances of the test environment located at different URLs that were only given in specific postings. This allowed us to draw conclusions about each individual population. The Reddit population reported as being universally non-female in the demographics part of the study while the sociology students reported as being an overwhelming majority of females. The sociology student population reported as being overwhelmingly American in both current residence and the primary part of their childhood, and the Reddit population reported as being a majority American as well.

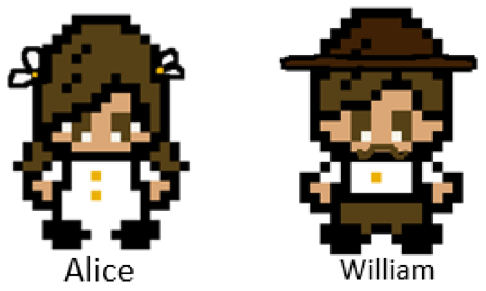


Figure 4.2: Alice and William. Alice, the female agent, was designed to possess features commonly associated in American culture with the female gender. William, the male agent, was designed to possess features commonly associated in America with the male gender.

Understanding Bias in Human Trainers: Agent Representation

As mentioned previously, the primary independent variable in this study is the presentation of the agent that each participant is tasked to train. To fit the overall artistic representation of the test environment, we used two agents that were represented with pixel art sprites (see Figure 4.2). Each sprite is meant to possess features that are commonly associated with Western perceptions of male and female genders. The male agent, who is referred to as William, is depicted as having short hair, a hat, pants, and a mustache. Each of these qualities is commonly associated with the male gender. Conversely, the female agent, referred to as Alice, is depicted as having long hair, a dress, and hair ribbons.

While it has been found that different colors are often associated with different genders [14] [13], we chose to use the same neutral color scheme for both agents. The reason that we chose to do this was to avoid any potential priming effects that could coincide with different color schemes[12]. Though color is often associated with gender, it is possible that different colors could evoke certain emotions or predispose the human trainer towards different types of behavior. Using a neutral color scheme mitigates these potential confounding factors.

In addition to the agent's visual representation, we also use the agent's name or the appropriate gendered pronouns when referring to the agent in text. All textual references to William, for example, uses the agent's name or male pronouns such as *he, him,* or *his*. Conversely, when referencing Alice we use female pronouns such as *she, her*, or *hers*.

Understanding Bias in Human Trainers: Protocols

As mentioned previously, subjects were given URLs to visit in order to participate in the study. Upon visiting the given URL, participants would be asked to confirm that they are over 18, legally capable of giving consent and to confirm their consent to participate in the study. They would then be assigned, at random, to a male or female agent presentation (see Figure 4.2). They were then presented with a short tutorial screen that consisted of an example of the test interface, an example action to give feedback on, and instructions on how to give the agent feedback (see Figure 4.3).

The interface displayed the game world utilizing simple pixel art with a different texture for each tile type:

- passable sand where the agent may pass through
- goal tiles where the agent must reach to end the game successfully
- lava/hazardous tiles where the agent may enter, but upon doing so the agent is considered to have been 'killed' and is returned to the start of the map
- impassible tiles which the agent cannot enter and will have no change in state if it tries to enter that tile

The map is treated as being bordered by impassible tiles. A text prompt at the top of the interface instructed the human subject to provide rewards and punishments to the agent to help guide it towards the goal tile. The human subject could provide feedback in the form of punishment and reward signals that ranged from negative one to one. The human subjects had the opportunity to provide feedback after every action being shown both the before and after states of that action. The interface allowed the human subjects to either directly type a feedback value they wished to provide, or to quickly click one of several buttons to assign a default feedback value of either 0, 0.3, 0.6, 0.9, -0.3, -0.6, or -0.9. These values were labeled: *No Feedback*, *Mild Reward*, *Significant Reward*, *Major Reward*, *Mild Punishment*, *Significant Punishment*, and *Major Punishment*, respectively. The actual feedback numbers were displayed beside the label of each of these buttons. The reason that multiple representations of feedback were used was to better mitigate the effect that human perceptions of reward or punishment had on their training. Including numeric values helps the human trainer assign value to the text descriptions, and the text descriptions help the trainer understand what the numeric values mean.

Each human subject provided feedback on two full learning episodes consisting of the agent taking actions until it either entered lava or reached the goal. The actions taken by the agent were predetermined. During the first episode, it would take the upper path, eventually turning and walking into lava. During the second episode, it would take the slightly longer lower path and eventually make it to the goal. Both runs contained extraneous turn actions and attempts to move into impassible tiles and the second run contained a section where the agent turned around and headed back towards its start for a short while before eventually "correcting itself" and heading towards the goal. This was done to better simulate training an interactive reinforcement learning agent. Since reinforcement learning is based on trial-and-error learning, to accurately simulate the training process we have to simulate the agent "randomly" exploring the environment by adding in extraneous actions. In total, each agent took 37 actions across two simulated learning episodes.

Human subjects were not explicitly informed at this point that the actions were pre-determined, but they were given a count of the number of actions left to evaluate from which they may infer that the actions were pre-determined, this is a not a concern as Cederborg [7] found that there is no difference between human feedback given to live action versus recorded agents. This forms an example of a *Wizard of Oz* experiment where actual machine learning techniques are replaced with an illusion of learning or interactive behavior. These experiments are a common technique for Interactive Machine Learning that allows multiple human subjects to face consistent conditions or otherwise negate undesired effects of randomness.

After providing feedback on each action that the agents take and completing the two full runs, participants were informed that the agent's actions were pre-determined and then were asked to complete a demographic questionnaire. The questions asked are omitted for space. No default answers were assigned for any of these questions, and participants were not required to answer all questions in order to proceed.

These questions were chosen because we believed that they corresponded to demographic information that could have a significant effect on the way human instructors provided feedback to the agent. However, our sampling population was too small and too homogeneous to allow meaningful analysis on most demographics.

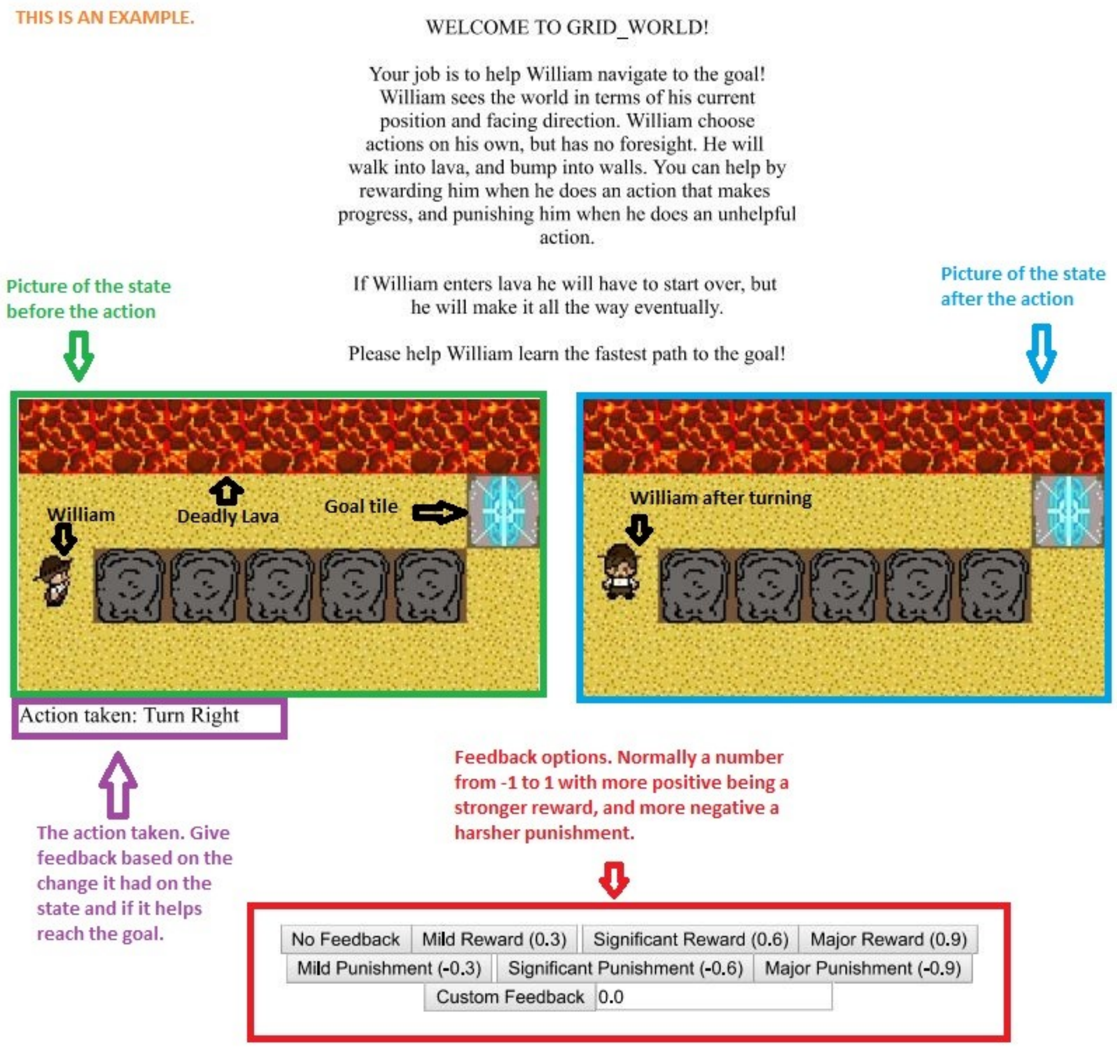
After completing the questionnaire, the subject was left with a message thanking them for their participation. In order to increase participation in the study, we also invited participants to invite friends and family members to participate in the study.

4.2 The BASIL Technique in a Gridworld

To meet our second goal of developing a technique and algorithms for IRL agents that account for bias in human trainer feedback, not only must we develop them, we must test them to ensure that they complete the task for which they were developed. In order to test our BASIL algorithm utilizing a normal distributions, we performed a series of experiments comparing the cumulative environmental reward the BASIL algorithm received to that received by other interactive machine learning algorithms and a q-learner which learned from an environmental reward. The interactive machine learning algorithms received feedback from simulated humans which included statistically generated bias. The statistically generated bias included both a normal bias, such as that which our BASIL algorithm is designed to handle, and instances of human silence, such as the SABL algorithm is designed to handle. The BASIL algorithm was not expected to out perform all other algorithms in all conditions, rather it was expected to maintained reasonable performance when the human feedback is heavily biased, such that other interactive machine learning algorithms cannot learn or find their learning heavily impacted.

The BASIL Technique in a Gridworld: MDP Environment

In our Gridworld environment, the state space consists of the coordinates of each passable tile paired with each possible facing direction. The action space consists of turning a quarter turn in either direction, and attempting to advance one tile


THIS IS AN EXAMPLE.
Picture of the state before the action
Picture of the state after the action
William
Deadly Lava
Goal tile
William after turning
The action taken. Give feedback based on the change it had on the state and if it helps reach the goal.
Feedback options. Normally a number from -1 to 1 with more positive being a stronger reward, and more negative a harsher punishment.
No Feedback
Mild Reward (0.3)
Significant Reward (0.6)
Major Reward (0.9)
Mild Punishment (-0.3)
Significant Punishment (-0.6)
Major Punishment (-0.9)
Custom Feedback
0.0

WELCOME TO GRID_WORLD!

Your job is to help William navigate to the goal! William sees the world in terms of his current position and facing direction. William choose actions on his own, but has no foresight. He will walk into lava, and bump into walls. You can help by rewarding him when he does an action that makes progress, and punishing him when he does an unhelpful action.

If William enters lava he will have to start over, but he will make it all the way eventually.

Please help William learn the fastest path to the goal!

Action taken: Turn Right

Figure 4.3: Example given to human subjects for the agent representation of William. The example indicates William's current state and the state that William will be in after he takes the action. The example also indicates to the human subject what action William will take, as well as what feedback options are available to give William. The human subject also receives text explaining the grid world game to the human subject.

in the current facing direction. The transition function consists of deterministic transitions of states where the facing direction is modified by the turning actions and the advancing action may result in changes to the occupied tile depending upon what type of tile would be entered, as previously described. The reward function is described above and within the Simulated Human Trainers section below.

The BASIL Technique in a Gridworld: Protocols

For our Gridworld experiments, all of the agents were placed in a ϵ-greedy 'shell' with ϵ set to 0.05. During evaluation episodes, learning does not occur and all actions were taken according to the agents' policies. Both training and evaluation episodes were limited to 100 steps.

An evaluation episode was preformed after each training episode to test the current quality of the learned policy. Each agent received 250 training episodes. To counteract the non-determinism introduced by the ϵ-greedy shell, each algorithm was instantiated with 20 independent agents that were trained separately, and the average of the cumulative rewards for each agent for each evaluation episode was taken to obtain the score for each algorithm as a function of the number of training episodes undergone.

The BASIL Technique in a Gridworld: Simulated Human Trainers

We used simulated human trainers in our experiments. We choose to use simulated human trainers because this allowed us to both know and to set the ground-truth for the feedback and bias provided in our experiments. This makes our experiments significantly more reliable and reproducible then they could be using live humans. This is a common practice in interactive reinforcement learning work. The simulated human trainers used in our experiments provided feedback in two parts that were combined before being given to the agents. The first part was the utility of the action taken as evaluated by a heuristic function. The second part was a bias value drawn from a normal distribution. Additionally, the simulated human trainer would withhold feedback during some experiments according to parameters determining separate rates for withholding positive and negative feedback. The designation of feedback as positive or negative for the purposes of withholding happened based solely on the utility value and did not account for bias.

In total 4 parameters determined the feedback to be given. These parameters were: the mean of the normal distribution; the variance of the normal distribution; the rate at which feedback was given when the utility for an action was positive; and the rate at which feedback was given when the utility for an action was negative. The utility of an action was determined by the following rules checked in order until one applied:

1. Actions which caused the agent to enter a hazard have value -1.0

2. Actions which caused the agent to enter the goal have value 1.0

3. Actions which caused the agent to have no change in state have value -0.6

4. In the starting tile turning to face south had a value of 0.3
5. Advancing from the stating tile while facing south had a value of 0.3
6. Turning to face east while one tile below the starting tile had a value of 0.6
7. Returning to the start state had a value of -0.3
8. Any action that results in facing west had a value of -0.3
9. Any action that resulted in less total moves being needed to reach the goal while not resulting in less total moves being needed to reach a hazard had a value of 0.9
10. Any action that resulted in less total moves being needed to reach the goal but also resulting in less total moves being needed to reach a hazard had a value of 0.6
11. Any action that resulted in less total moves being needed to reach a hazard while not resulting in less total moves being needed to reach the goal had a value of -0.9
12. Any action that did not fall under any of the above rules had a value of -0.3

The result of these rules was that actions that take the agent to the goal on either path would receive positive feedback, but the initial steps for the upper path would receive stronger positive feedback then the initial steps for the lower path. Approaching the hazards while following the upper path was rewarded, but turning to face them unnecessarily was strongly punished. Counter productive or wasteful actions were also punished. For our experiments the mean for the normal distribution took values of 2.0, 0.5, 0.0, -0.5, and -2.0. The variance of the normal distribution took values of 0.0, 0.3, and 2.0. The rates at which feedback was given when the utility value was positive or negative each independently took values of 1.0, 0.8, 0.5, and 0.1. Because we tested over a hundred permutations of parameters we will be unable to discuss all of these experiments in our results section, therefore we will limit our discussion to illustrative results.

The BASIL Technique in a Gridworld: Algorithms Tested

SABL As the algorithm most similar to our work, of which we are aware, we chose the SABL algorithm by Loftin et.al. [25] as one of our points of comparison. We also tested feedback withholding of the type SABL expects in addition to normally distributed bias as expected by our BASIL algorithm. We choose to use SABL over ISABL because we can provide the algorithm with the exact value of the rate at which feedback is withheld. And by supplying these values to SABL, we have a stronger baseline than ISABL could provide. Our implementation was based on source code kindly provided by Loftin et.al.[25] from their experiments. To adapt it to our task, our implementation treated any instance of positive feedback as an instance of reward and any negative feedback as an instance of punishment for its calculations.

Both forms of the SABL algorithm require the user to provide an estimation of the error rate, that is the rate at which positive feedback is given when negative feedback is appropriate and vice versa. The SABL algorithms assume that this rate is the same for both cases and that it is known to the users. For our experiments, calculating this value is often infeasible because it depended upon the actions taken by the agent. It is also important to note that the value is non-zero, as zero values can lead to the algorithm breaking down if any mis-assignment of feedback does occur.

Lastly, it should be noted that under the SABL algorithm's assumptions any instance where multiple actions receive positive reward for the same state would be considered an error. For all of these reasons, we parameterized the SABL algorithm with an error rate of 0.2 for all experiments.

TAMER For our second algorithm to compare against, we used the TAMER algorithm by Knox and Stone [23]. We chose it because it has been a reliable baseline for interactive machine learning that has done well on a number of machine reinforcement learning tasks. To apply it to our task, it was necessary to develop a feature set for TAMER to use. TAMER weighs actions by considering the change they produce in the feature set, using weights and sums of these changes in a linear fashion. It is therefore important that is has access to features that directly and individually correspond to desirable or undesirable changes in the environment. For our experiments we provided TAMER with 8 features. The first two were the X and Y coordinates of the current state; the next four represented the facing direction of the current state in a 1-hot fashion; and the last two were the minimum number of actions needed to reach a hazard and the goal respectively. It is the last two that are particularly important to TAMER's performance. In particular, learning to minimize the actions needed to reach the goal is virtually equivalent to learning the optimal policy. These features are very domain specific, but because of it's linear nature, without highly engineered features, TAMER performs poorly. One other consideration that was required for TAMER in our experiment was its learning rate. The learning rate determines how much TAMER updates its weights when it receives feedback. For our experiments, we used a constant learning rate of 0.1.

Environmental Q-Learner For our final baseline of comparison we used standard Q-Learning agents[35]. These agents learn only from environmental reward. They make no use of the feedback signal provided by the simulated human trainers. We included these agents as a baseline to show that, even with inconsistent and biased human trainers, it is possible to achieve better performance than using an environmental reward alone in many cases. The only significant parameters for the Q-Learner were the learning rate and discount factor. The learning rate determines how significant the updates it makes to its internal model each time it receives feedback and the discount factor determines how much it discounts possible long term rewards versus immediate rewards. We chose 0.1 for both of these parameters in all of our experiments.

4.3 BASILTAMER

The BASIL technique fulfills our secondary goal of developing a technique to handle bias in human provided feedback. However, we still must test practical applications of the technique that are more complex in nature, if we are to more fully complete this goal. For this, we utilize the TAMER algorithm by Knox and Stone.[23] We chose to integrate the BASIL technique into the TAMER algorithm because the TAMER algorithm has become a standard among IRL algorithms. The addition of the BASIL technique to the TAMER algorithm results in what we refer to as BASILTAMER. The goal of our BASILTAMER experiment was to determine if adding normally distributed bias to the feedback signal given to classical TAMER and BASILTAMER would result in the classical TAMER algorithm failing to learn the desired behavior encoded in the feedback signal while not preventing the BASIL TAMER algorithm from learning the desired behavior. To determine the performance of the algorithms, the cumulative reward of each algorithm in an episode as a function of the number of learning episodes previously experienced was used. The utilization of a cumulative reward is a standard metric for these kinds of evaluations in reinforcement learning. To compare these algorithm the game of Tetris was utilized. In the Tetris environment, one point of environmental reward was given for each line cleared and, thus, the cumulative reward for each episode was equivalent to the total number of lines cleared in that episode.

BASILTAMER: Task and Environment

The task chosen to test our BASILTAMER algorithm was a version of the game "Tetris", modified to be suitable for reinforcement learning via conversion into a Markov Decision process. This task was chosen because it was the same task utilized in the original presentation of TAMER. [23] In brief, Tetris is a game where the player is presented with a series of tetrominos, which are connected shapes of four blocks. The player's job is to rotate and move the tetrominos until they contact a placed tetromino or the bottom of the play field, at which point they lock in place. When a row is completely filled, the blocks in that row disappear and blocks in all higher rows fall by one level. While various renditions of the game use different point scoring schemes, in general, the goal is to clear as many rows as possible.

In reinforcement learning friendly formulations of the game, the player's options for actions to take consists of each possible valid placement of the currently active tetromino. While placing the active tetromino, the player has no knowledge of upcoming tetrominos which results in the problem being an MDP. For our experiment, both classic TAMER and BASILTAMER, utilized a vector of features that described the current game state and those vector of features were pre-calculated for them. These features were: the heights of each column; the height differences between each pair of adjacent columns; the height of the highest column; the total number of holes in the play field, that is empty squares with full squares above them; and a fixed weight for computational purposes. This feature set has been used successfully in many works including the original presentation of TAMER. [23]

BASILTAMER: MDP Environment

As described elsewhere for our BASILTAMER experiments, we use a version of the game of Tetris that has been modified to be an Markov Decision Process. In this MDP, the state space consists of every possible Tetris grid of settled blocks, without regard to color, permutated by each tetromino to be placed within the grid. The action space consists of placing the tetromino within each possible, valid, final location and orientation within the grid. A valid, final location is any location that can be reached through the normal moves in human playable Tetris that is supported on at least one square. The reward function is the number of lines cleared by taking the action in the state. The transition function maps states and actions to the grid that results from placing the tetromino in the chosen location after full rows are removed and higher rows are dropped, then it pairs that grid with each possible next tetromino to place stochasticly.

BASILTAMER: Protocols

An evaluation episode was preformed after each training episode to test the current quality of the learned policy. Each agent received 20 training episodes. In the bias free case it took under 6 episodes for both algorithms to match the oracle's performance, and this result is similar to what was observed in Knox and Stone's original publication of the TAMER algorithm. [23] During evaluation episodes, learning does not occur and all actions are taken according to the agents' policies. Both training and evaluation episodes were limited to 3000 steps. To counteract the non-determinism inherent in Tetris, each algorithm was instantiated with 30 independent agents that were trained separately, and the average of the cumulative rewards for each agent for each evaluation episode was taken to obtain the score for each algorithm as a function of the number of training episodes undergone.

BASILTAMER: Hyper-parameters

Both algorithms used α values of 0.01, and tolerances of 0.000001 for all experiments. The BASIL TAMER algorithm had two additional Hyper-parameters. It used a μ update interval of 20 steps for all experiments, and the expectation maximization step was limited to 3000 iterations, although we believe this was never reached in any of the experiments.

BASILTAMER: Simulated Human Trainers

We used simulated human trainers in our BASILTAMER experiments. We choose to use simulated human trainers for the same reasons as mentioned above in the previous section. Being able to know and set the ground-truth for the feedback and bias provided in our experiments makes our experiments significantly more reliable and reproducible then they could be using live humans. This is a common practice in interactive reinforcement learning work. The simulated human trainers used in our experiments provided feedback in two parts that were combined before being given to the agents. The first part was the utility of the action taken as evaluated by a heuristic function. The second part was a bias value drawn from a normal distribution. The utility of taking an action for a particular state was determined by the weighted sum of the difference of feature vectors for the state that preceded the action and the resulting state. The weights used are included in the source code for our experiments. These weights were such that an agent adhering to them exactly when deciding between actions will average 66.3 lines cleared as measured over 600 episodes. This is comparable to the peak performance for TAMER learning live humans as observed by Knox and Stone. [23] These weights were found via the use of an evolutionary algorithm. Two parameters determined the bias applied to the feedback given: the mean of the normal distribution of the bias and the variance of the normal distribution. For our experiments, the mean for the normal distribution took values of 80, 20.0, 5.0, 0.0, -5.0, -20 and -80.0. The variance of the normal distribution took values of 0.0, 3.0, and 30.0.

4.4 BASILTAMER Extensions

The BASIL technique is already well set up to calculate the observed variance in the feedback received, particularly of distributions, such as the normal distribution where variance is a parameter of the distribution function. However, the technique, on its own, does nothing to address the observed variance, nor does it address silence from the human trainers, in this instance silence refers to an explicit lack of feedback with regards to a specific action as opposed to a small or neutral feedback value. With human trainers, variance in the feedback given is, generally, an inevitability. In its simplest forms, variance is the human trainer being inconsistent with the feedback given and providing different feedback values for the same situation. As stated previously, this problem becomes more difficult to account for when there are multiple human trainers, which is increasingly necessary for larger scale tasks. Additionally, this variance does not always mean that the human trainer made a mistake. Depending upon the feature set provided to an algorithm, human trainers may be able to see differences in two states that look identical to the algorithm. Inconsistency due to variance in feedback values provided to the agent, as well as a lack of feedback (silence), can cause the agent's behavior to become erratic as it attempts to incorporate the new feedback instances or the result can be the agent appearing to "quit" attempting to learn the correct policy as is flounders due to not receiving enough feedback to develop a correct policy. To account for variance and silence in human trainer provided feedback, we have developed three extensions: the Adaptive Variance, Zero Silence, and Average Silence Extensions. Accounting for these inconsistencies, in regards to BASILTAMER, will address our tertiary goal for this research.

The primary goal of these experiments was to determine if the techniques we had developed for handling inconsistencies in human feedback would result in an improved performance, in the case where the human provided feedback was subject to variance or silence, when compared to the same algorithm using only BASILTAMER and TAMER, respectively.

BASILTAMER Extensions: Task and Environment

The task chosen to test our BASILTAMER Extensions was the same version of the game "Tetris", modified to be suitable for reinforcement learning via conversion into a Markov Decision process, that we utilized for our BASILTAMER experiment. We chose this environment because the framework we had previously developed for it was highly robust and adaptable. This allowed us to easily control all aspects of the feedback given to the learning agents including: bias, silence, and variance.

BASILTAMER Extensions: Protocols

Evaluation episodes for the BASILTAMER Extension experiments were performed similarly to how they were performed for BASILTAMER without the addition of these extensions. We reduced the number of episode that each experiment trained for to 10, as we had observed in the BASILTAMER experiments that this was sufficient to see full learning of a correct policy within the environment. We also reduced the number of independent instances of each algorithm to 10 samples, because we found that this was sufficient for developing reasonably smooth learning curves.

For the variance experiments, we tested all permutations of average biases equal to 0.0 and 5.0; as well as variance equal to 0.0 and 1.0. For the silence experiments, we tested the same average biases, and tested rates of silence conditions of withholding: no feedback; half of all positive feedback; all positive feedback; half of all negative feedback; and all negative feedback. However, certain combinations of bias and feedback withholding would result in no or very little feedback being given to any agent due to the feedback tending towards the all positive or all negative based upon the bias. These permutations are, therefore, excluded from our testing and include the 5.0 bias value with any degree of positive feedback withholding and the 0.0 bias value with any degree of negative feedback withholding. This results from the simulated human trainers for this experiment and environment primarily giving feedback in the low negative range before bias, meaning the ground truth feedback values are within this range. This, in turn, is the result of using TAMER algorithms with an environment and feature set that utilizes primarily values that should be minimized for optimal play, such as the heights of each column.

BASILTAMER Extensions: Simulated Human Trainers

We used simulated human trainers in our BASILTAMER Extension experiments, similarly to how we used them in our BASILTAMER experiments and we chose to use simulated human trainers for the same reasons that we chose to utilize simulated trainers in those experiments.

4.5 Deep BASILTAMER

We now focus on implementing the BASIL technique into a Deep Neural Network. Incorporating BASIL into this complex deep learning algorithm will complete our secondary goal for this research and is the most complex application of the BASIL technique that we present in this body of work. To apply our technique within a Deep Neural Network, we adapted the BASIL technique to Deep TAMER. We refer to this algorithm as Deep BASILTAMER.

The primary goal of this experiment was to determine if adding normally distributed bias correction to the Deep TAMER algorithm via the BASIL technique would result in significant performance improvement over the unmodified Deep TAMER algorithm, as well as the exploratory epsilon-greedy Deep TAMER algorithm, when applied to feedback from a live human trainer. An additional goal of our experiment was to determine if adding artificial normally distributed bias to the feedback provided produced a significant difference in performance among the versions of the algorithm when compared to instances where raw feedback was given, especially when compared to each other. We also sought to further verify the observations of Bartneck, Reichenback, and Carpenter [2] of how agent presentation between different grades of anthropomorphism produced different feedback when using images of the same robotic avatars. We also sought to determine how well the algorithms performed as measured by the cumulative reward of each algorithm in an episode as a function of time spent learning.

Deep BASILTAMER: Adapting BASIL to Deep TAMER

There are a few challenges that come with incorporating TAMER into a Deep Neural Network. The main challenge is that the learning steps in a Deep Neural Network are computationally expensive and performing this learning in an online manner, as is necessitous for interactive machine learning, is feasible only on extremely powerful hardware. In order to combat this issue, Deep TAMER subdivides its Deep Neural Network into two distinct parts. The first part is a Convolutional Neural Network that is trained entirely offline as an auto-encoder. The output of this Convolutional Neural Network is then fed into a fully connected Neural Network consisting of a few layers. In the case of the Atari Bowling task utilized by Warnell et. all, [34] two fully connected layers were used. It is the second fully connected portion of the Neural Network, exclusively, that is trained during interaction with the human trainer. To perform real time training, Deep TAMER assigns each feedback instance to all state action pairs such that the feedback falls into a pre-defined feedback window for the action. All instances of feedback are then stored in a buffer and learning is performed both on each instance of feedback as it is received and on instances of feedback sampled from the buffer at a fixed rate.

At predetermined intervals of the training updates, Expectation Maximization is performed between the fully connected portion of the Neural Network, which serves as a function mapping feature sets produced by the auto-encoded Convolutional Network to human feedback values, and the Bias Distribution, defined by unknown parameters. This is analogous to how Expectation Maximization is performed between the weight vector and the Bias Distribution in non-Deep BASILTAMER. Because the fully connected, online trained portion consists of more than one layer, the least square linear approximation used in the non-Deep BASILTAMER Expectation Maximization step to produce the weight vector at each iterative step could not be applied directly.

We originally foresaw two possible methods for replacing the linear approximation. The first was to use a polynomial approximation. This approximation was seen as potentially viable because of the relatively few number of layers involved and, if it was viable, it would have been the preferred method to obtain the weight vector because it would have produced an exact optimal answer for each iterative step. However, this approach proved to be too computationally expensive for online training with the computational hardware available to us. The second option we foresaw for this work would have required approximating the layers of the Neural Network under the hypothetical bias that would be expected for each agent. The most straightforward means to perform this approximation would have been to train a copy of the fully connected layers of the Neural Network on a Bias Corrected sampling of previously observed feedback instances. For this option, as long as sufficient training was performed, it should have produced workable results. However, this option also proved to be too computationally expensive and for both of these options it would have required adjusting the interval to lengths that we considered to be unacceptable for the agent to maintain real time training.

Ultimately, we discovered a third option for implementing the Expectation Maximization step. During the Expectation Maximization update, all expect the last layers of the neural network were treated as fixed and not subject to change throughout the iterations of Expectation Maximization. The last layers were the only layers which changed with each iterative update and each consisted of one independent vector per possible action. Each vector is updated by an independent linear approximation using the current bias parameter estimation. These vectors are then utilized to generate the next estimation of the bias parameter. This iterative process continues until convergence is achieved, at which point, the convergent values for the bias parameter are used to adjust future feedback, including instances of feedback used in re-sampling of previous observations.

Intuitively, this approach seemed viable due to our use of a normal distribution for the assumed bias and for the architecture of a Deep TAMER Neural Network. Our intuition is that the auto-encoder portion of the Neural Network, which is fully trained and formed without regard to what is and is not a good move in the game of Atari Bowling, serves as a feature extractor. Therefore, the first of the two online layers serves to learn which combination of features are relevant to learning the task of achieving a high score in Atari Bowling. Because the bias is expected to be drawn from a normal distribution, accounting for bias is a matter of adjusting feedback by the mean of that distribution. With this method, we may treat the bias as existing entirely in the last layer of the human's internal reward module that we are attempting to learn. Our results bear out the correctness of this approach. And this approach is computationally viable with the hardware available to us. It should be noted that our success with this approach may rely on the use of a normal distribution for modelling human trainer bias. With a more complex bias model, such as one where different actions are biased independently, this shortcut may not be viable. Furthermore, using such models with a Deep Neural Network may require either significant computational resources or the application/development of useful mathematical tricks. The addition, we discovered that the application of the BASIL technique to the Deep TAMER algorithm does not substantially change the majority of the algorithm.

In the course of conducting this research, we discovered that there were issues in the Atari bowling environment we utilized due to Deep TAMER having no mechanism to drive exploration. We are unclear if this was a problem that was also experienced by Warnell et. all [34] or if the difficulty arose from the method by which our human instructors provided feedback to the Deep TAMER agents as we could not find details as to the interface used by Warnell et. all.

In particular, there are a handful of ways in the bowling environment in which an agent's policy may prevent it from making any progress whatsoever. The most straightforward of these is when the agent uses the wait action while still holding the ball. If this is the action dictated by the agent's policy and the agent acts entirely on policy, then there is no mechanism for the agent to attempt other actions and to receive feedback upon them.

A very similar problem can arise when the agent has positioned it's avatar near the top or bottom of the allowed area and then continues to move the agent up or down, respectively, resulting in no change in the game state and no exploration of alternative actions. In theory, this situation can be corrected by the human trainer providing sufficient negative feedback on the repeated actions to change the agent's policy. However, this may take a large number of feedback instances depending upon the agent's learning rate, which is a hyper parameter that was not given in the original Deep TAMER research by Warnell et. all, and must be configured for a particular programmer's problem. Additionally, the use of the BASIL technique compounds this issue, because the situation can arise where Deep BASILTAMER has a current policy dictating an action which does not change the game state as described above and then performs an update of its estimation of the average bias, which can result in all incoming feedback being adjusted such that it is not possible for incoming feedback to change the agent's policy without exploring other actions. This is a consequence of the need to only make periodic updates to bias estimation computations due to the intense computative resources required for an update.

Conversely, we observed that Deep TAMER without the benefit of exploration or a technique such as BASIL, can potentially be stymied by a human trainer who is reluctant to give negative feedback, even in situations where the agent is "stuck". This is not a problem reported by Warnell et. all in their work on Deep TAMER and did not likely occur with their presentation and user interface. However, we did receive reports of this situation occurring with the AIBO representation of the agent, which is in line with the findings of Bartneck et. all [2], and our expectations that human trainers would be reluctant to give negative feedback to it. Therefore, we added an exploratory element to the Deep BASILTAMER algorithm and we performed our test against two variations of Deep TAMER.

The first algorithm was an unmodified Deep TAMER, while the second was an epsilon-greedy Deep TAMER. The epsilon-greedy module used by both Deep BASILTAMER and the augmented baseline consists of a simple additional computation when the agent selects a move. When selecting a move, the agent generates a random number between zero and one. If this number falls below epsilon, which is itself a number between zero and one that is selected as a hyper-parameter of the agent, then the agent selects its move randomly from the available actions. However, if the number is above or equal to epsilon, the agent will select the on policy action, as is normal for Deep TAMER. We used an epsilon of 0.2 for our epsilon-greedy Deep TAMER and our Deep BASILTAMER agents.

It is worth noting that epsilon-greedy action selection is only performed while learning. When evaluating a currently learned policy, such as in our evaluation of the data later in this work, actions are always selected on policy and resulting non-update loops are dealt with by terminating the game after a maximum of five thousand steps. For our model, one step is one execution of the EMV.step function of the open A.I. Gym implementation of Atari Bowling.

Like Warnell et all[34] we did rely on wall clock time measures of training. Human trainers were instructed to limit training sessions to roughly fifteen minutes. Each algorithm was trained with each representation by human trainers in three separate training sessions. The policies learned in the course of those training sessions were snap-shotted at each instance of human feedback with a maximum of one snapshot per second. These snapshots were then tested within five games of Atari bowling to generate an average score for each algorithm at each feedback point in each training session. These average scores were then utilized to determine how well each algorithm performed in learning an effective policy to achieve a high score in Atari Bowling.

Deep BASILTAMER: Task and Environment

The task we utilized to test our Deep BASILTAMER algorithm was Atari Bowling with a graphical representation. (See Figure 4.4) This task was chosen because it is the same task utilized in the original presentation of Deep TAMER. [34] They chose this task because it is one that even state of the art deep learning algorithms have struggled with, and they were able to not only out preform those algorithms but, also, out performed most of the human trainers themselves.

In the environment, players are presented with a visual representation at each frame and can chose between taking no action and taking one of three actions consisting of Up, Down, and Bowl. See figures 4.6, 4.7, and 4.8 for the virtual representations of the agents. The environment can be seen in figure 4.5. Each of these actions does different things depending on whether or not the ball has yet been bowled. Before bowling, Up and Down adjust the position from which to launch the ball from, and Bowl sends the ball down the track. After Bowl is selected further uses of Bowl are equivalent to no action until moving on to the next ball. The first instance of Up or Down used after bowling the ball result in applying spin in the corresponding direction, and subsequent uses of either are equivalent to no action until moving on to the next ball. The game is then scored and played like common ten pin bowling

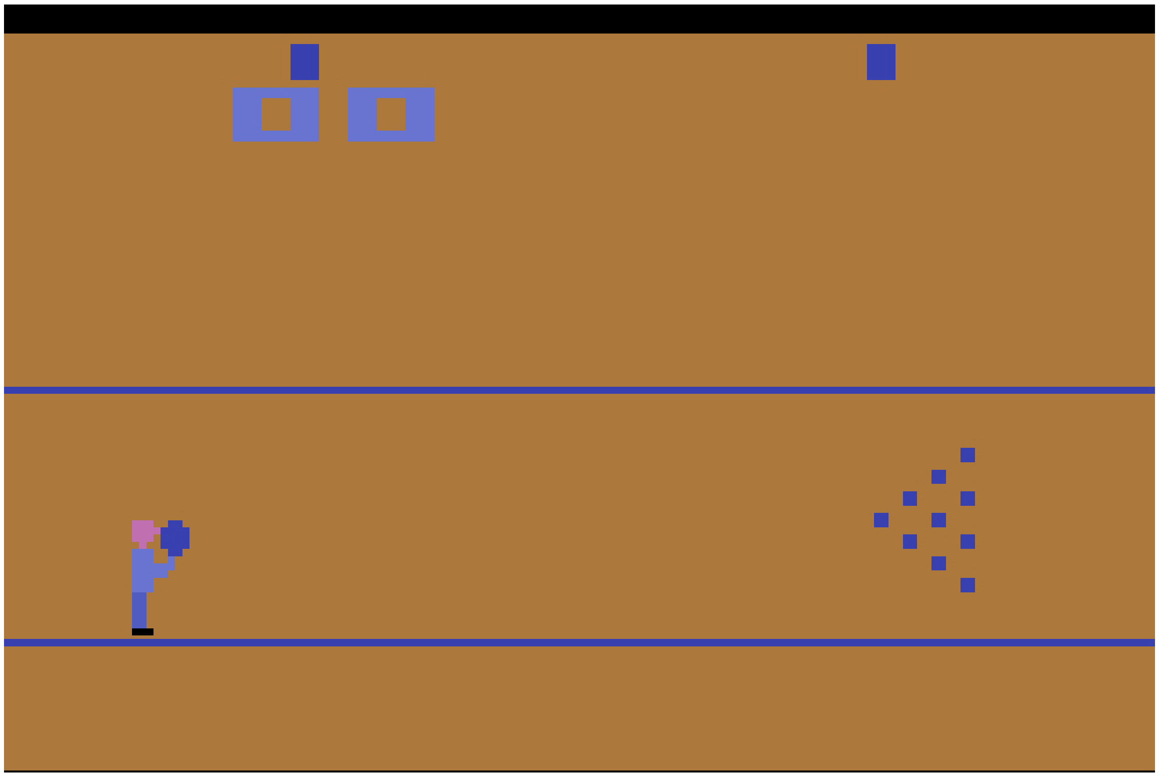
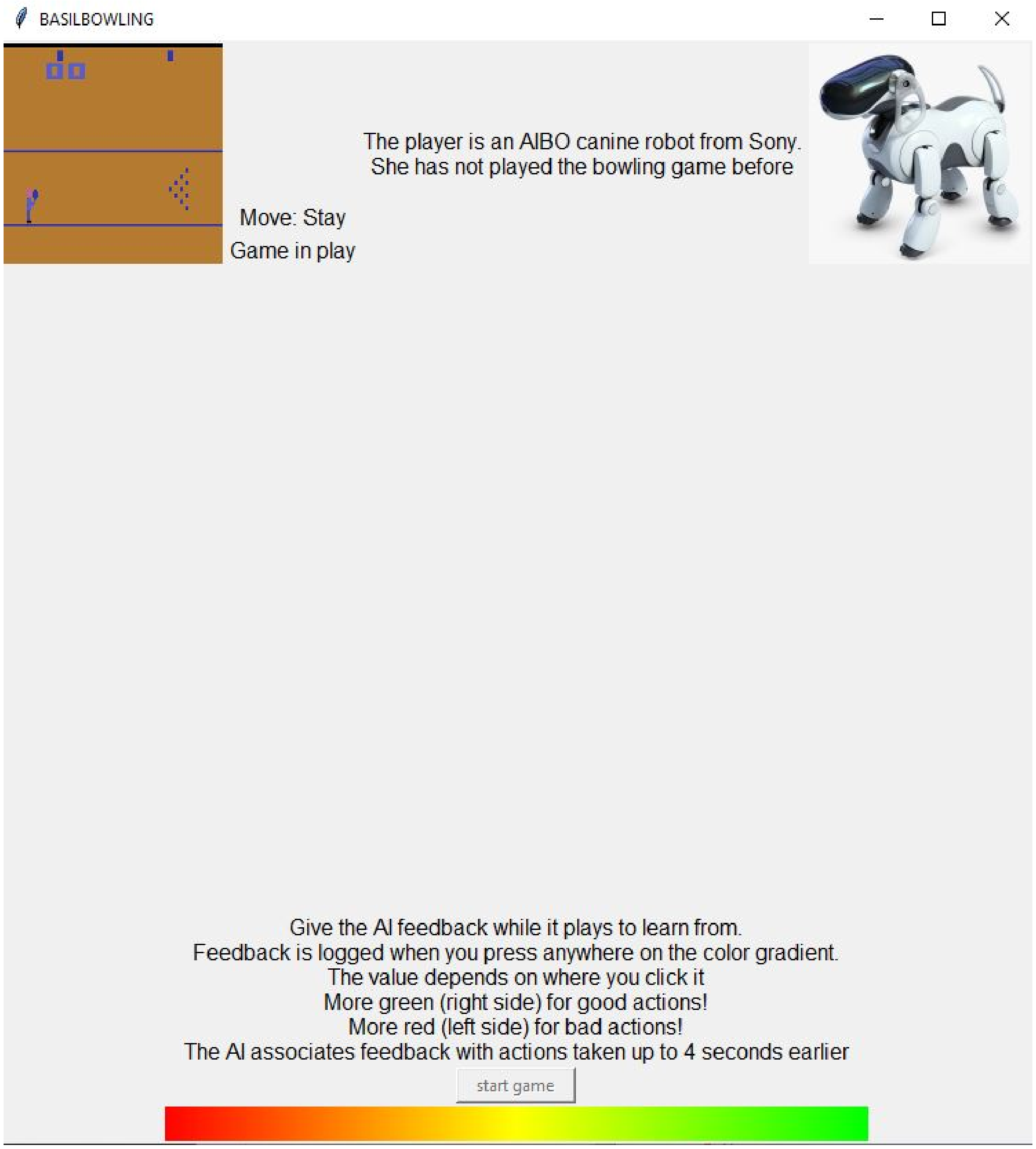


Figure 4.4: A visual representation of the Atari bowling environment.

except that no extra balls are allowed after the tenth frame, therefore the maximum score possible for a game is 270. Like Warnell et all, we used the two most recent graphical frames in 160 x 160 grey scale as our state representation and set the rate to approximately 20 frames per second.

Deep BASILTAMER: MDP Environment

The state space for the Atari Bowling environment is visual. It consists of two consecutive frames from the bowling games. The action space consists of moving the player character up or down on the field, bowling, and taking no action. If used in a state where the ball has been tossed but spin has not been applied, the up and down functions apply spin to the ball instead. The transition function is highly non-deterministic as the Atari Bowling game is played in real time, and consists of transitioning to the state which the environment has reached since our last observation. While the player character holds the ball, the only changes in the state will be those directly resulting from the agent's actions, however, once the ball is tossed, the state can be expected to change between observations, even if no action has been taken. The reward function consists in the change to the agent's score in the game following the normal rules for bowling as described above.



The player is an AIBO canine robot from Sony.
She has not played the bowling game before

Give the AI feedback while it plays to learn from.
Feedback is logged when you press anywhere on the color gradient.
The value depends on where you click it
More green (right side) for good actions!
More red (left side) for bad actions!
The AI associates feedback with actions taken up to 4 seconds earlier

Figure 4.5: The application interface presented to our Human Trainers with the Sony AIBO presentation of the agent.

Deep BASILTAMER: Protocols

An evaluation episode consists of a game of Atari Bowling played according to the policy learned by an agent in a snapshot of its training. During evaluation episodes, learning does not occur and all actions are taken according to the agents' policies. Each of our algorithms being tested, the Deep TAMER baselines, with and without epsilon-greedy exploration, and Deep BASILTAMER with epsilon-greedy exploration, was used during nine training sessions; three with each visual representation. Each of these training sessions, was allowed to run just over fifteen minutes. These training sessions were performed by volunteers drawn from the researchers' social circles. Each trainer was given a practice training session before the recorded training session, during which they were allowed to question one of the researchers as much as they desired. Snapshots of each algorithms' learned policy were saved whenever the human trainer gave the agent feedback with a maximum of one snapshot per second. These snapshots are the ones utilized to perform the evaluation episodes analyzed in our results section later in this body of work. Additionally, to maximize the use we could obtain from our limited pool of human researchers, without adding to their labor or commitment, we recorded the feedback instances and matching state action pairs that occurred during each human trainer's session training the agent.

The offline training of the algorithms consists of performing SGD with the feedback action and state triples recorded during the online training session and offline trained agents of this sort are used to produced our results that analyze the difference between agents with and without artificial biasing. This sort of offline trained agent is also used in our later sections to produce their results in the Atari Bowling environment.

In total 27 training sessions, each lasting at least 15 minutes, with a live, human trainer were conducted. Each of the three robots used in the research by Bartneck et. al.[2] were used as the basis of one agent representation. A presentation consisted of the image of the robot and a short phrase describing the robot as the player the agent was training. For the AIBO presentation, the description provided was: "The player is an AIBO canine robot from SONY. She has not played the game before". For the Tron-X robot, the description was: "The player is the Tron-X robot from Festo AG. It has not played the bowling game before". The Philip K. Dick representation's description was: "The player is Phil. A replica of sci-fi author Philip K. Dick from Hanson Robotics. He has not played the bowling game before". These pictures and descriptions were presented in the UI as pictured in Figure 4.5.

To ensure we observed training sessions under different biases, these representations were paired with different instructions, given verbally, to the human trainers during their pre-training practice sessions. Human trainers paired with the AIBO presentation were instructed to "be patient" and to "reinforce actions you want her to take again". Human trainers paired with the Tron-X representation were instructed to "give negative feedback when they felt it was necessary" and to "remember that it is a machine and does not have feelings to hurt nor any purpose beyond playing Atari Bowling as well as it can". Those paired with the Philip K. Dick representation were instructed to "follow your instincts when giving feedback" and that "both positive and negative feedback can help Phil learn".

Each representation was combined with each of the three algorithms being tested: Deep TAMER, Deep TAMER with epsilon-greedy exploration, and BASIL Deep TAMER with epsilon-greedy exploration; to create nine distinct experimental testing conditions. Each of these conditions was assigned randomly to three human trainers, who completed the experiment.

During the experiment, human trainers gave the agent feedback by clicking on a colored rectangle located at the bottom of the UI, as seen in Figure 4.5. The value of the feedback is determined by the horizontal position at which the human trainer clicks, with negative values on the left side of the bar and positive values on the right side of the bar. These values range from -10.0 on the far left edge and 10.0 on the far right edge. This range was determined by pre-experiment testing to provide sufficient weights to new training instances relative to the Neural Network training rates which are 0.1 in all cases, and our starting weights which were generated using a Keras truncated normal initializer with a mean of 0.1. This feedback method was chosen because it allowed the human trainer to rapidly give an agent feedback following a relevant action. This feedback method is imprecise, however, it proved sufficient for our needs. We could not find sufficient details in Warnell et. all [34] to replicate their user feedback interface nor their feedback methodology. While our methodology was sufficient for the experiments that we wished to conduct, deficiencies in our methodology may be a contributing reason to the significantly lower performance we saw for all of our agents compared to that observed by Warnell et. al., which we will discuss further in a our analysis of this experiment.

4.6 Multi-Circumstance Aware Deep BASILTAMER

So far, we have largely addressed handling bias in interactive reinforcement learning, but one particular circumstance still bears addressing. When the phrase bias is used, it is often used to describe a difference that occurs between two samples. It is therefore worthwhile to discuss using the BASIL technique for multiple feedback sources, whether these sources are different human trainers, or different presentations of the agent to human trainers, or any other factor which might alter the way feedback is given. When using such distinct sources of feedback, the simplest way to adapt the technique is to treat each body of feedback as having its own parameters for its bias distribution, but being offset from the same utility evaluations of state action pairs. In the expectation maximization step, the agent should calculate the distribution parameters for each body of feedback separately, and then, while calculating new estimates of the utility of each state action pair, use feedback from all bodies stripped of bias according to the current estimates of the bias distribution for the appropriate feedback body. We refer to this as Multi-Circumstance Aware BASIL. And, when utilized with Deep TAMER, we refer to this technique as Multi-Circumstance Aware Deep BASILTAMER. For this body of work, we focused on Multi-Circumstance Aware Deep BASILTAMER. To that end, we needed to determine if Multi-Circumstance Aware Deep BASILTAMER, which is specifically adapted for handling multiple biasing conditions as we have put forth, performs substantially better than unmodified

Deep BASILTAMER, and Deep TAMER without BASIL, when feedback for each algorithm is produced under multiple biasing conditions.

To determine the performance of the algorithms, the cumulative reward of each algorithm in an episode as a function of the number of learning episodes previously experienced was, again, used.

The application of a Multi-Circumstance Aware BASIL technique to Deep TAMER satisfies our secondary research goal. By being "aware" of feedback provided by other human trainers silence and variance can be accounted for.

Multi-Circumstance Aware Deep BASILTAMER: Task and Environment

The task we used in these experiments was the Atari bowling with graphical representation as was used in our Deep BASILTAMER experiments. We refer the reader to the task and environment subsection of the Deep BASILTAMER experiments section of this paper for a full description on that environment and the results of that experiment.

Data obtained from our Deep BASILTAMER experiment was used for this experiment. State and actions pairs combined with feedback sourced from the AIBO and P.K.D. sessions of the Deep BASILTAMER experiments were used to train models using this modification offline.

Multi-Circumstance Aware Deep BASILTAMER: Protocols

This experiment utilized data obtained from our Deep BASILTAMER experiments. The data sets utilized were for the AIBO and P.K.D. avatars of the algorithms. In offline training sessions, Multi-Circumstance Aware Deep BASILTAMER was implemented and tested against the baselines of Deep TAMER with epsilon greedy exploration and Deep BASILTAMER.

Due to our findings in our previous experiments, we have chosen to utilize data from only two of the three presentations that we previously used. The reason we chose to utilize these data sets is because of how similarly our human trainers provided feedback to the P.K.D. and Tron-X avatars in this task environment.

The PKD Robot is an animatronic model of Science Fiction Author Philip K. Dick and represents a higher point on the scale of anthropomorphism as compared to the other presentations. The last avatar is the zoomorphic robot AIBO from Sony, which has a canine like form and provides a radically different presentation for contrast.

To perform the training, the training data from our AIBO samples and PKD samples were paired randomly to create three feedback sets, each of which consisted of the feedback received partnered with all game state and action tuples that fell within the relevance window. This is similar to how Deep TAMER algorithms resample previously given feedback during their training. Each algorithm being trained was run through simulated time-steps, during which feedback instances from either the AIBO sample or PKD sample being utilized were provided to the algorithm when it hit the appropriate second count. In this manner, each algorithm learned from both sessions in its training set as if they had both occurred simultaneously, although

Figure 4.6: The Tron-X robot from Festo AG. [2]

the states and actions taken were the ones that appeared in the original training session from which the data is sourced and not the actions which the algorithm being trained would have necessarily taken at that moment of its training. This approach to offline learning is similar to learning from demonstration but, differs in that the demonstration is provided by very similar algorithms engaged in active learning and the feedback provided is that received by those learning algorithms, whereas learning from demonstration uses either an environmental reward or makes the assumption that all or most of the demonstrated actions are on policy for the desired policy to be learned. For each tested algorithm, three sample feedback histories were created by selecting, at random, an AIBO training session and a PKD training session from those recorded in our Deep BASILTAMER experiments. Each of those histories was then used to train one instance of the algorithm and at each time-step where an evaluation was to be performed, five games were played by the algorithm with its current policy to produce an average score for that instance of the algorithm at that time-stamp. The cumulative reward curves of each algorithm are then interpolated and averaged to produce the cumulative reward function for the algorithm as a whole, in the same manner as we utilized in our earlier Deep BASILTAMER experiments.

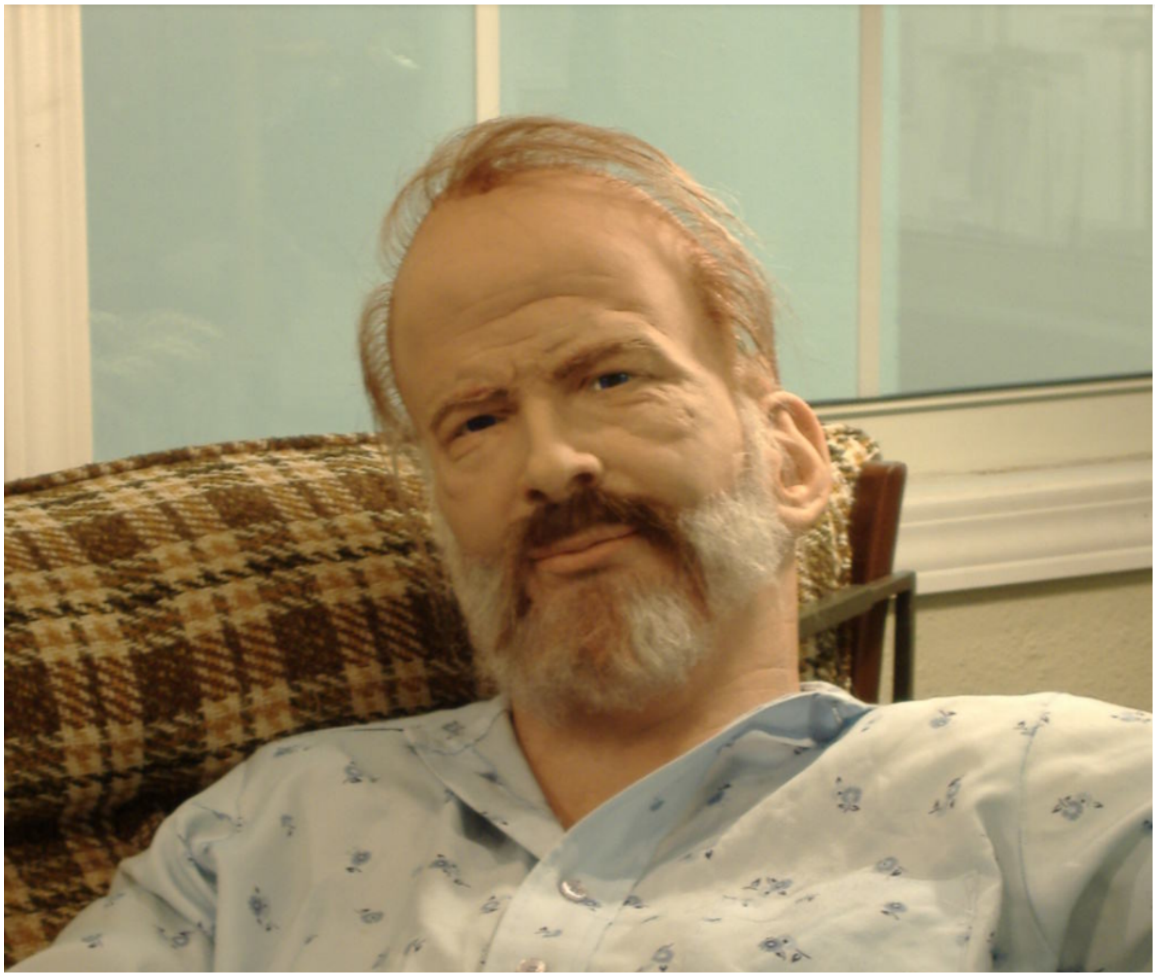


Figure 4.7: The PKD robot from Hanson Robotics.[2]

Figure 4.8: The AIBO robot from Sony.[2]

Chapter 5 Results

In the following sections, we present the results of our experiments. Our first section deals with Understanding Bias in Human Trainers where we studied the effects of an agent's gendered representation on the feedback provided by human trainers. Next, we present our findings on how well the BASIL technique performs within the domain of a Gridworld. The results of our third experiment, which incorporated the BASIL technique into the TAMER algorithm and compared the performance of the resulting BASILTAMER algorithm to TAMER within the domain of Tetris, are presented. We also present our findings for the Extensions of the BASILTAMER algorithm. The Deep BASILTAMER experimental results, where we applied the BASIL technique to the Deep TAMER algorithm are found below. Finally, we present our findings for the Multi-Circumstance Aware Deep BASILTAMER.

5.1 Understanding Bias in Human Trainers: Results

To understand the bias in human trainers in reference to the gendered representation of an agent, we took the provided feedback values and grouped them into histograms evenly spaced about the default feedback values given. In practice there was only a single instance of a subject using the custom feedback option, so, this process was very natural. This histogram data formed from participants that trained Alice is then compared using a Chi-squared test against the histogram data formed from participants that trained William to determine the probability that two histograms come from the same distribution. Using this technique, we answered the research questions posed earlier. We present each of these answers in greater detail below.

Research Question 1: Statistical Significance

First, we wanted to know if there was a statistically significant difference in the feedback given to an agent when presented as the female representation versus the male representation across all human subjects and states presented in the environment. The Chi-squared test that we ran shows that the feedback distributions between both experimental groups were significantly different from one another (p=0.00015, $\tilde{\chi}^2 = 26.94839$). This indicates that there is a strong likelihood that both histograms were generated by different underlying processes, which provides strong support that there was a difference in the way that each group trained their agents. Due to the way this study was designed, this implies that the presentation of the representation had an effect on how participants gave provided feedback to the agent.

Research Question 2: Qualitative Differences

The second research question addressed involves a qualitative analysis of the feedback histograms for people training the female agent versus the male agent. When presented as Alice, major rewards made up a greater percentage of its reward feedback instances whereas the William representation was slightly more likely to receive the mid-level significant reward and significantly more likely to receive the minor reward. That being said, the overall percentage of feedback instances that are rewards is very close for both agent representations, 61% for Alice and 63% for William.

In terms of punishment, the William representation was more likely to receive both the major and mild punishments, while the Alice representation was more likely to receive the middling significant punishment. Both agent representations received punishment as 34% of all feedback. The *No feedback* category made up a larger percent of the feedback distribution for Alice versus William, 5% versus 3% respectively (See Table 5.1).

Research Question 3: Demographic Effects

Answering the third research question involved analyzing if any demographic characteristics of human instructors could have had a statistically significant effect on the feedback that agents received. We preformed Chi-squared tests for direct effect of demographic characteristics where the samples could be partitioned into binary subgroups along demographic lines. The only demographic effect for which our analysis has found a statically significant effect is the instructor's reported gender (p=0.00011, $\tilde{\chi}^2 = 27.67627$). A very small number of subjects choose to either not report their gender identity or reported options other then male or female, they are not included in this analysis due to insufficient representation in our samples. It is important to note that we believe it is likely that other demographic information does have an influence on how human trainers provide feedback. For example training behaviors could be affected by the subject's experience or inexperience with providing training to siblings or children; such as the case with the dog trainers' bias discovered in Loftin, MacGlashan, et all.[25] We were, unfortunately, unable to analyze these effects due to the size and homogeneity of our subject population. For example, our participants were largely American and heterosexual, meaning that it was impossible for us to determine how nationality or sexual orientation could potentially affect training behaviors due to lack of data.

Research Question 4: The effect of reported gender

Seeing that reported gender had a significant effect on feedback tendencies, we sought to contextualize these results. The most notable difference we found was that the difference in feedback given by female instructors versus that given by male instructors is strikingly similar to the overall difference in feedback given to the Alice representation versus the William one. For example, female instructors preferred the major reward whereas male instructors were more likely to give the mild reward. Male instructors were also more willing to provide the major punishment whereas female instructors preferred the less significant ones. Female instructors were also more likely to give no feedback compared to male instructors (see Table 5.2).

Group cond.	Num. of samples	0.9 RV.	0.6 RV.	0.3 RV.	No RV	-0.3 RV	-0.6 RV	-0.9 RV
Alice	3102	28%	19%	14%	5%	13%	12%	9%
William	2038	24%	20%	18%	3%	13%	11%	10%

Table 5.1: Histographic data for all feedback instances of Alice versus William representations respectively. Note the discrepancy in the overall number of feedback instances for Alice versus William. This is a result of a quirk in the random number generator used to select an agent representation and is still completely uncorrelated with any attribute of the human subject. It is unfortunate as the statistical significance of our results is limited by the size of the smaller sample, but the assignment of agent representation was still done at random, just with unequal weights on the representations.

Group cond.	Num. of samples	0.9 RV.	0.6 RV.	0.3 RV.	No RV	-0.3 RV	-0.6 RV	-0.9 RV
Female Subject	3103	28%	19%	14%	5%	13%	12%	9%
Male Subject	1860	25%	19%	19%	4%	12%	10%	10%

Table 5.2: Histographic data for all feedback instances provided by human subjects, Female versus Male. This table represents all populations merged together. Despite our populations turning out to be heavily segregated by gender, the trends remained the same even if we only limit it to the sociology student population or the machine learning reddit population. However, the machine learning reddit population is small enough for the effects to fall below statistical significance. For that reason, we present the data with all populations included without concern for other effects correlated with gender among our population.

Research Question 5: Potential Cross Effects

The behavior outlined in research question 4 indicates that there may be a cross effect between instructor gender and the agent's presented gender. Such as in group bias where human subjects would give more positive feedback to representations of their own gender identity.

However, we found that this was not the case. To determine this, we compared feedback distributions between the cases where gender of the agent's representation and the instructor's reported gender matched verse cases where they did not. We found that there was no significant difference in these feedback distributions according to a Chi-squared test, which indicates that there is no in-group bias in feedback with regards to gender.

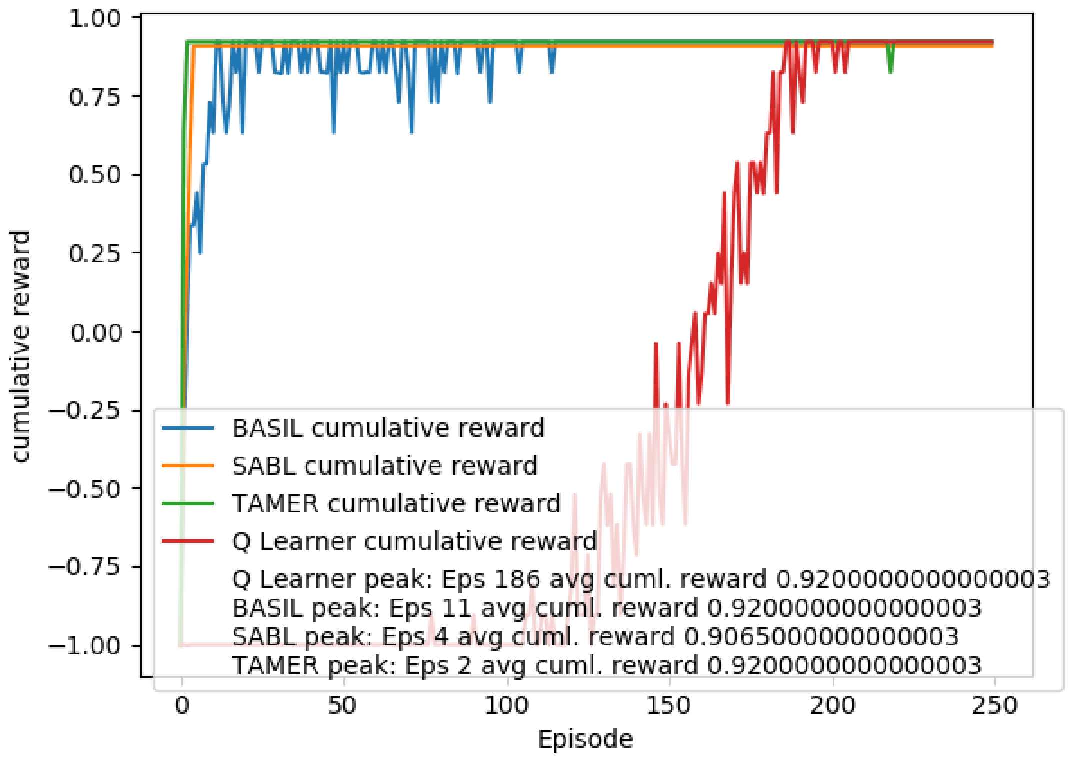


Figure 5.1: Cumulative Reward for each algorithm when feedback contains no bias, and feedback is always given.

5.2 The BASIL Technique in a Gridworld: Results

As mentioned previously, we tested on hundreds of permutations of the parameters controlling the simulated human trainer's feedback and we will limit our analysis here to illustrative examples to maintain readability.

The BASIL Technique in a Gridworld: Perfectly Reliable Feedback

As can be seen in Figure 5.1, when the simulated human trainers gave perfectly reliable feedback, TAMER preforms the best of all tested algorithms, needing only 2 episodes to learn the optimal policy for all agents. SABL needs only 4 episodes to reach it's peak, however because SABL does not distinguish between levels of reward, both the upper and lower paths are equal to it, and SABL agents received an average cumulative reward of 0.9065 rather then the value of 0.92 it would have received if all agents learned the optimal path. The BASIL algorithm takes 11 episodes to reach a peak where all agents take the optimal path and actually suffers a degree of instability for over 100 episodes. The BASIL algorithm does eventually fully stabilize, but this experiment illustrates that correcting for bias comes at a cost.

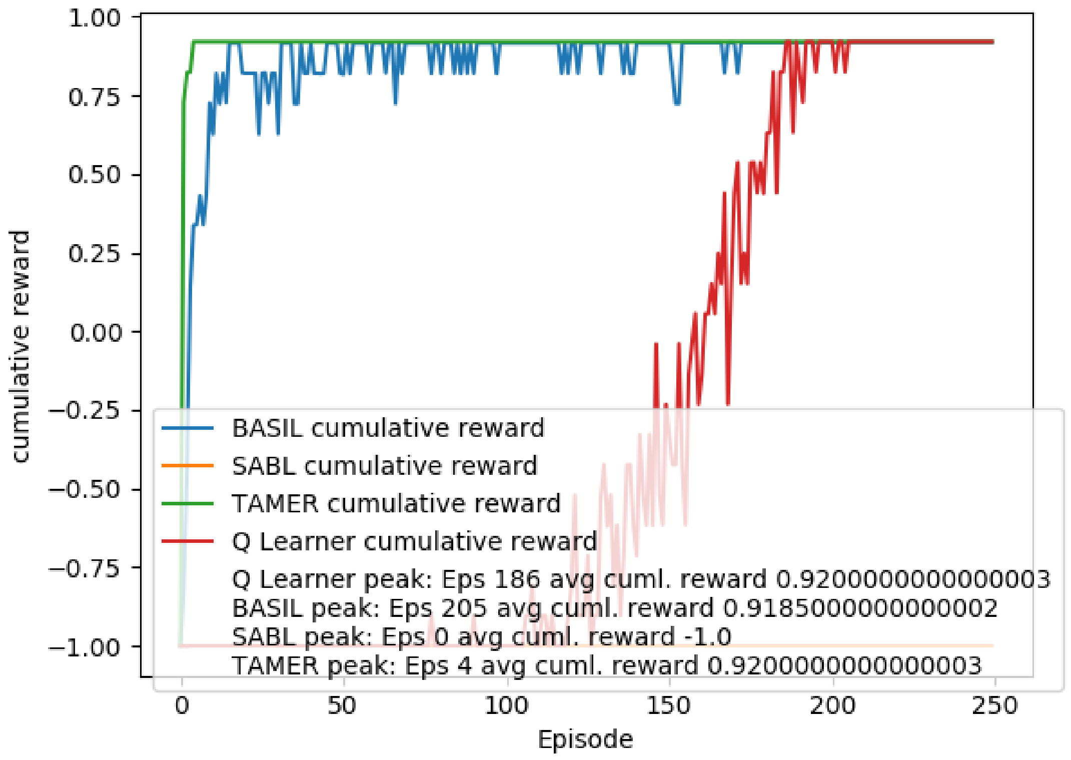


Figure 5.2: Cumulative Reward for each algorithm when feedback contains moderate positive bias $\mu = 0.5$ with moderate variance $\sigma = 0.3$, and feedback is always given.

The BASIL Technique in a Gridworld: Normally Biased Feedback

Adding a moderate bias has different effects depending on the direction of the bias as can be seen in Figures 5.2 and 5.3. Even a moderate positive bias is enough to prevent SABL from ever learning either path because some time wasting actions now receive a small positive reward on average. TAMER on the other hand copes with a moderate positive bias easily as it still learns to reduce the distance to the goal and avoid the hazards. A moderate negative bias actually helps SABL as it makes starting the lower path non-rewarding, but, in this case, it causes TAMER to never stabilize over the course of 250 episodes. In both cases, the BASIL algorithm preforms well. For the small positive bias, TAMER still out preforms the BASIL algorithm slightly which we believe is because of the very strong feature set it has access to, and how this small positive bias is not enough change to what weights are attached to what features. However with an equivalent negative bias, the BASIL algorithm maintains it's performance while TAMER falls significantly below it. The BASIL algorithm and SABL preform very similarly on the small negative bias, with a slight advantage to SABL, but as we discussed above even the small positive bias causes SABL to fail entirely, while the BASIL algorithm maintains performance.

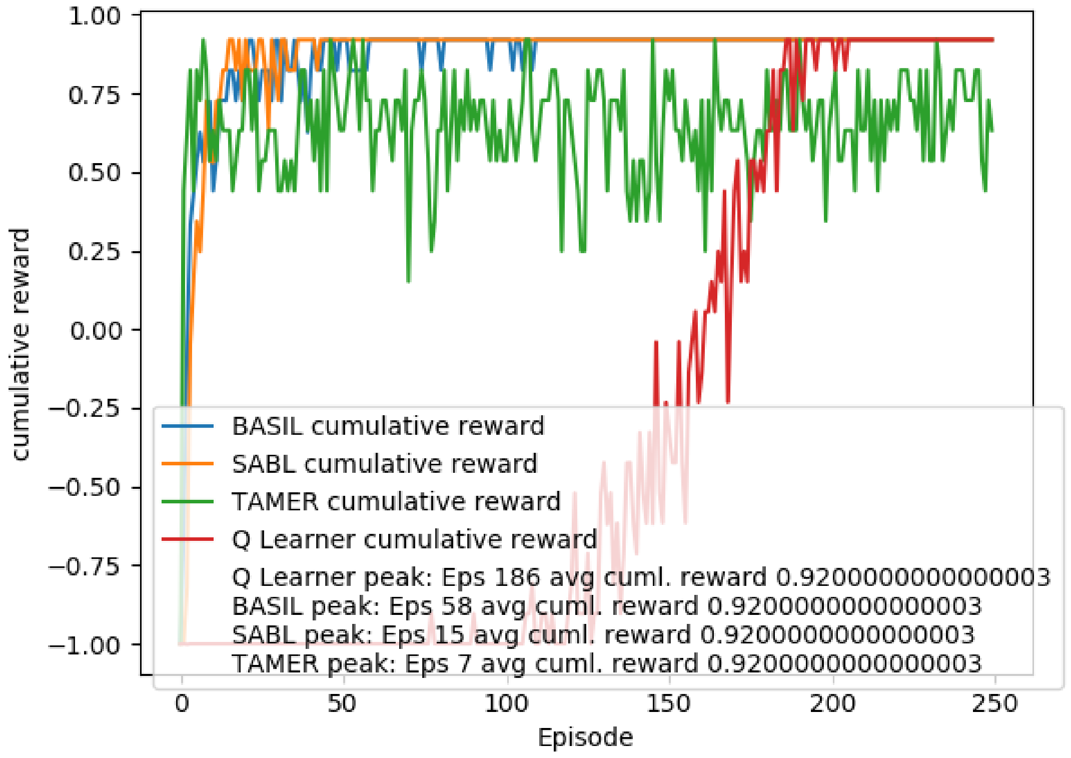


Figure 5.3: Cumulative Reward for each algorithm when feedback contains moderate negative bias $\mu = -0.5$ with moderate variance $\sigma = 0.3$, and feedback is always given.

Introducing a large bias (2.0) in either direction leaves BASIL as the only interactive reinforcement learning algorithm to preform well. See Figures 5.4 and 5.5. With a large positive bias (Figure 5.4), TAMER learns self destructive behavior as it now receives rewards for actions that take it closer to hazards. This causes it to fail entirely, even though it coped well with smaller positive bias. SABL, which coped well with small negative bias, fails under a large negative bias as it receives punishment for all actions and can not distinguish between grades of punishment. Of the three interactive reinforcement learning algorithms tested, the BASIL algorithm is the only algorithm which maintains performance for bias values in both directions or for large bias in one direction.

The BASIL Technique in a Gridworld: Inconsistently Provided Feedback

Withholding feedback has different results depending on whether it is positive or negative feedback that is withheld. See Figures 5.6 and 5.7. Neither case impacts SABL as it was designed to handle this type of feedback distortion. Withholding most negative feedback hinders the BASIL algorithm's learning significantly and causes it to develop instability on this problem, and slows TAMER down some (Figure 4.6).

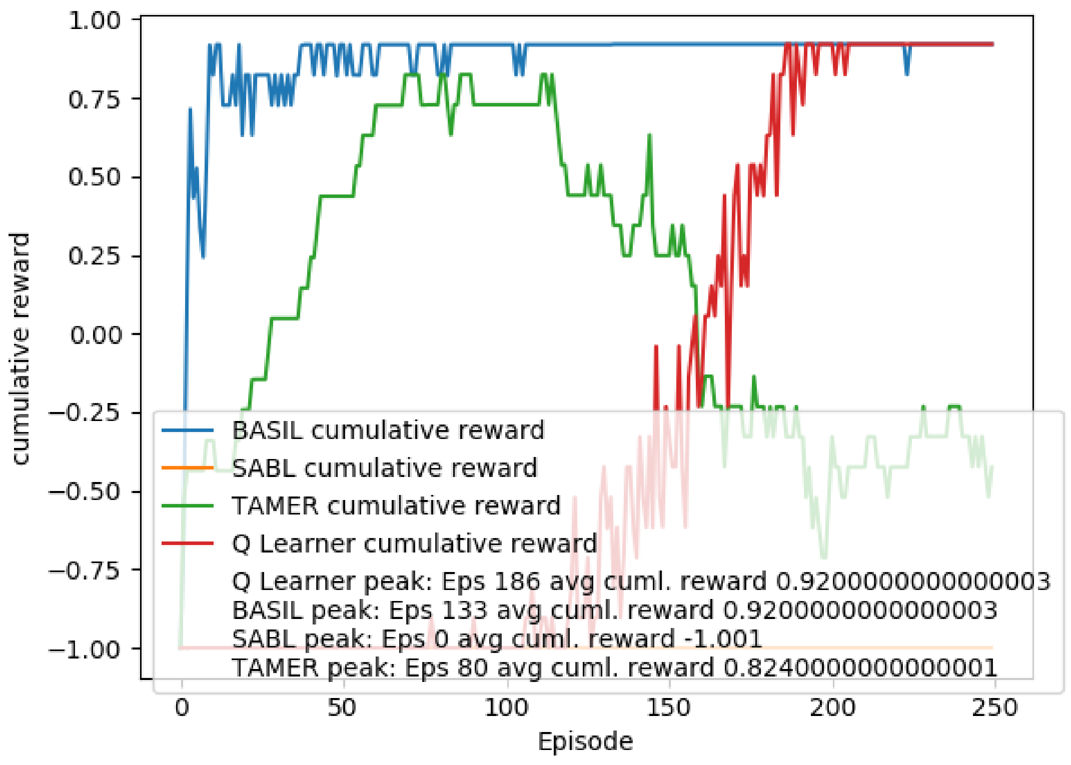


Figure 5.4: Cumulative Reward for each algorithm when feedback contains large negative bias $\mu = +2.0$ with moderate variance $\sigma = 0.3$, and feedback is always given.

Withholding most positive feedback has much less effect on the BASIL algorithm's stability but does cause it to have difficulty distinguishing between the upper and lower paths. TAMER has much more difficulty with this scenario, and suffers significant instability with no sign of recovery (Figure 5.7).

The BASIL Technique in a Gridworld: High Variance in Feedback

Having very high variance $\sigma = 2.0$ in the feedback signal is a problem for all the Interactive Reinforcement Learning algorithms we tested, but was more of an issue for TAMER then the others as can be seen in Figure 5.8. We speculate this is because the BASIL algorithm innately averages past feedback to get it's estimate of action utility, and SABL is unconcerned with exact feedback values. Even at this high level of noise in the feedback signal, the BASIL algorithm and SABL are both able to learn behavior far faster then the environmental Q-Learner, that does not use the simulated human feedback, when this noise is the only distortion given. Combining the high variance with other distortions each algorithm was not built for does cause their performance to fall below that of the environmental Q-Learner.

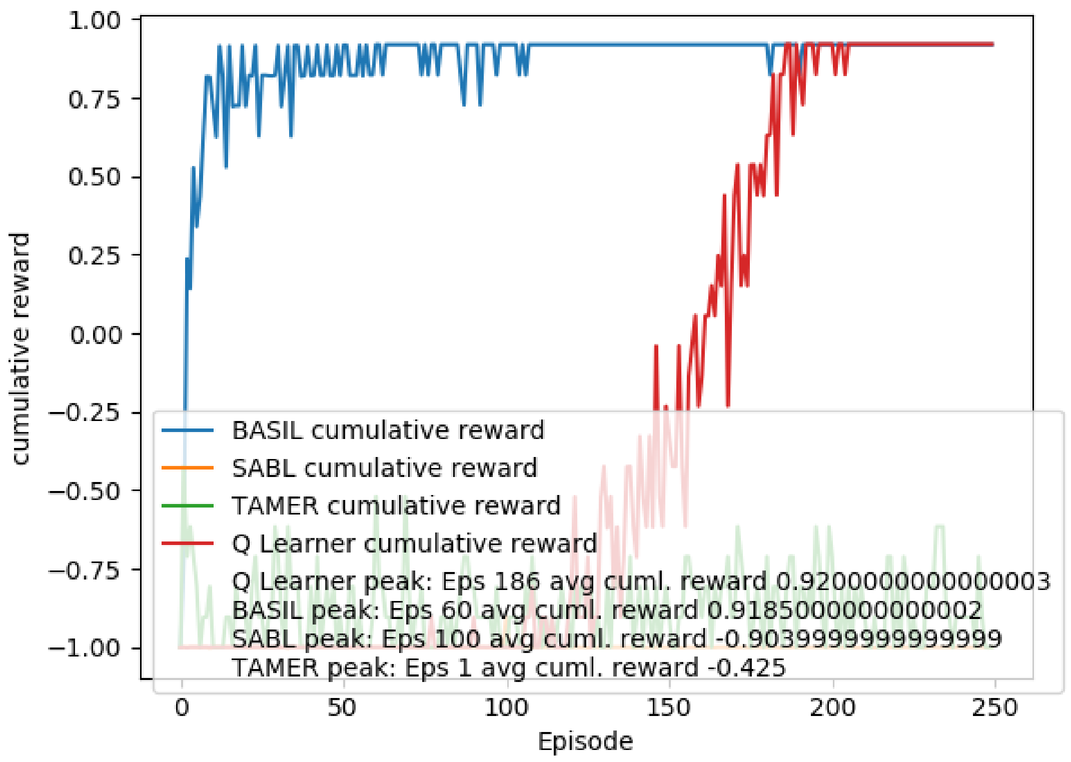


Figure 5.5: Cumulative Reward for each algorithm when feedback contains large negative bias $\mu = -2.0$ with moderate variance $\sigma = 0.3$, and feedback is always given.

The BASIL Technique in a Gridworld: Mixed Feedback Distortions

Mixing multiple types of distortion has complex effects. Both the BASIL algorithm and SABL account for one type of effect and tend to respond as they do when only the type of distortion they don't account for is present. TAMER suffers the worst of both effects. For example, when a moderate positive bias $\mu = 0.5$ is combined with withholding most feedback that would be positive before bias is applied (Figure 5.9), the BASIL algorithm does fairly well, just as it does when feedback is given the same way without bias. In this scenario SABL does terribly as the bias is enough to make it waste all of its steps. TAMER does poorly, worse in fact then it preforms with either distortions individually. As discussed above, adding very high variance to other factors is enough to severely hinder the performance of all Interactive Reinforcement learning algorithms.

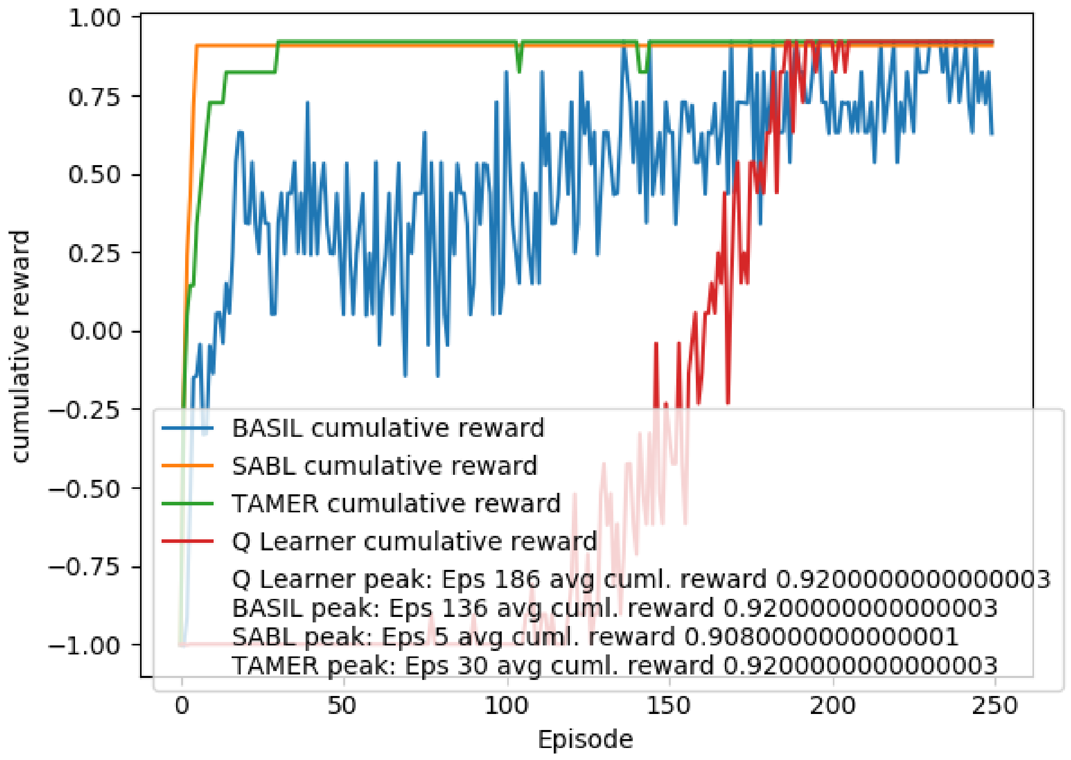


Figure 5.6: Cumulative Reward for each algorithm when feedback contains no bias, and positive feedback is always given, but negative feedback is only given with probability 0.1.

5.3 BASILTAMER: Results

As mentioned previously, we tested on 21 permutations of the parameters controlling the simulated human trainer's feedback and due to space constraints, we will have to limit our analysis here to illustrative examples. Averages of the performance of each algorithm under each permutation of basis for the last 10 episodes of training are provided in Table 5.3. Our full results for all experiments are available in supplemental files.

BASILTAMER: Perfectly Reliable Feedback

As can be seen in Figure 5.10, when the simulated human trainers give perfectly reliable feedback with no bias both TAMER and BASIL TAMER algorithms preform very well and match the oracle weights performance before the 5th episode.

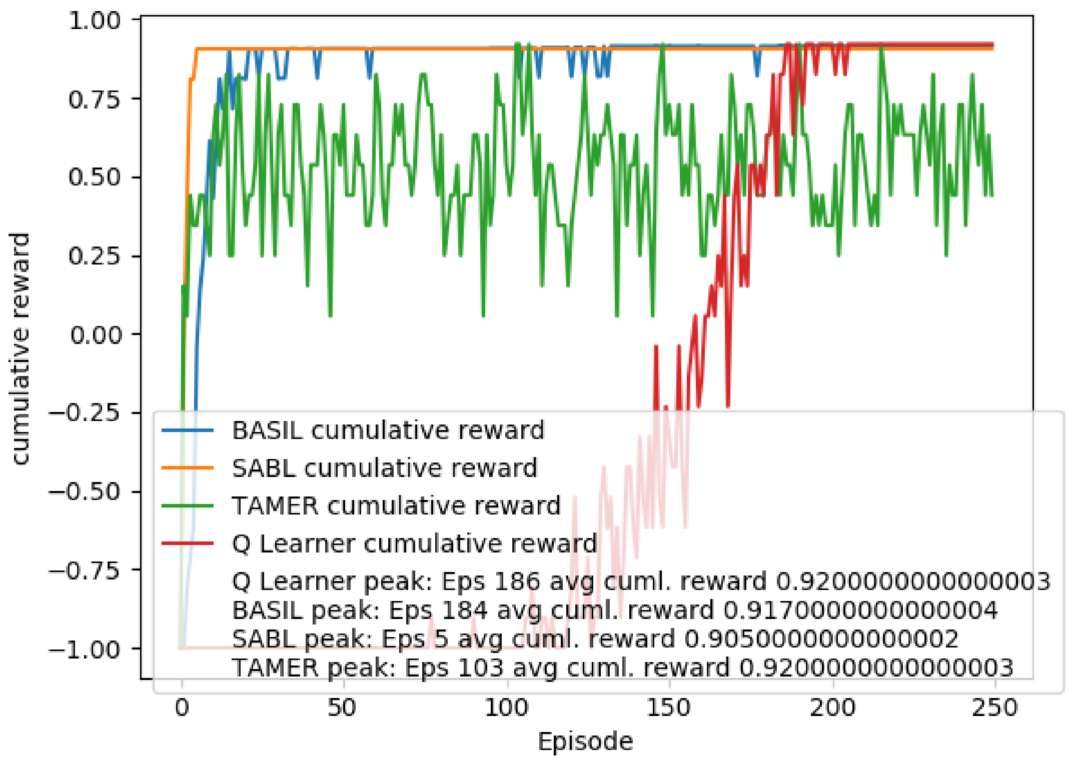


Figure 5.7: Cumulative Reward for each algorithm when feedback contains no bias, and negative feedback is always given, but positive feedback is only given with probability 0.1.

BASILTAMER: Normally Biased Feedback

Adding a moderate positive bias such as 5.0 causes classic TAMER to completely fail as can be seen in Figure 5.11. However, it is worth mentioning that for this problem classic TAMER did better with negative bias and, as can be seen in Table 5.10, maintained a decent performance at -5.0 median bias value although its performance was still negatively impacted with more extreme negative bias having greater consequences. We believe the difference in performance for negative verse positive bias can be attributed to the feature set which is such that all features in it should generally be minimized as opposed to maximized and therefore feedback corresponding high performance such as that given by the oracle will be almost entirely negative. BASIL TAMER maintains its performance at all bias values even with very high bias values such as 80.0 regardless of the direction of bias.

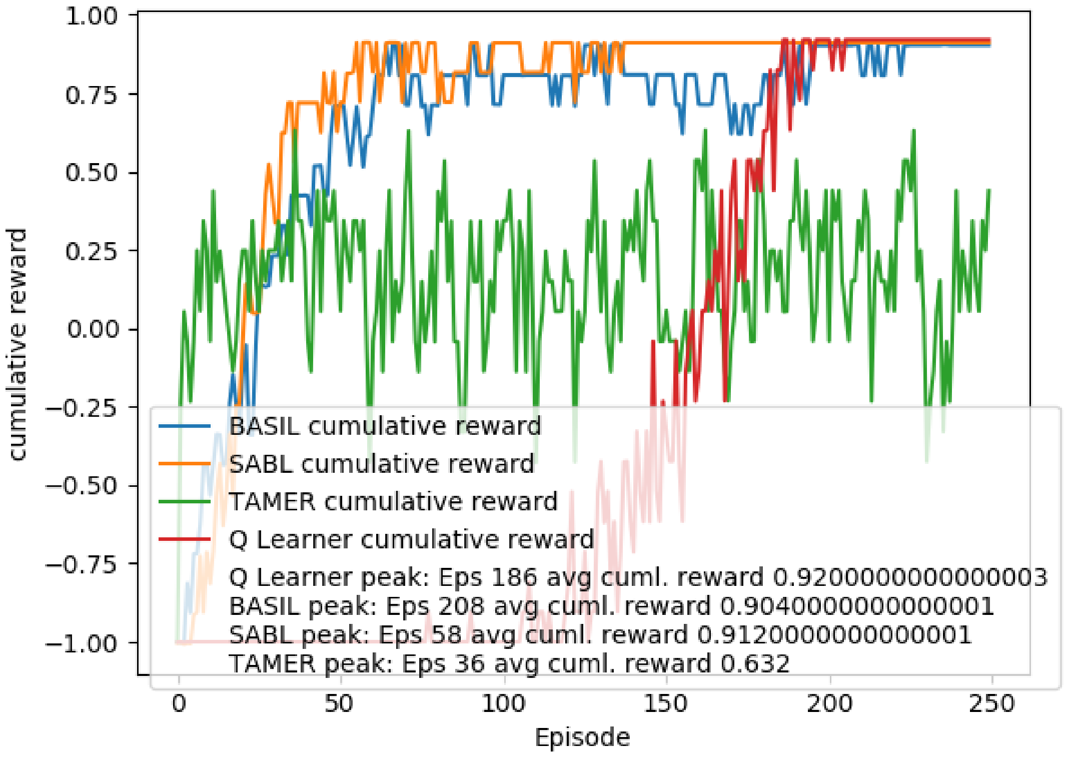


Figure 5.8: Cumulative Reward for each algorithm when feedback contains bias with $\mu = 0.0$ but high variance $\sigma = 2.0$, and feedback is always given.

BASILTAMER: High Variance in Feedback

As can be seen in Table 5.10, having higher than expected variance in the feedback signal is a problem for both forms of TAMER we tested. This is to be expected as this variance amounts to noise in the feedback signal and the only counter to this is to use a smaller alpha value and collect more feedback instances. Supplemental experiments were performed with the $\mu = 0.0$ and $\sigma = 3.0$ values using smaller values for alpha and this was confirmed to restore eventual performance to the levels of the $\mu = 0.0$ and $\sigma = 0.0$ case.

5.4 BASILTAMER Extensions: Adaptive Variance BASILTAMER Extension Results

In figures 5.12, 5.13, and 5.14 you can see the results of our tests of the Adaptive Variance BASILTAMER Extension, when the variance and standard deviation are equal to 1.0, with no bias and a bias of 5.0, respectively. As can be seen, from these results, the Adaptive Variance Extension of BASILTAMER does manage to preserve the performance of BASILTAMER in the Tetris environment to a high degree. Figure

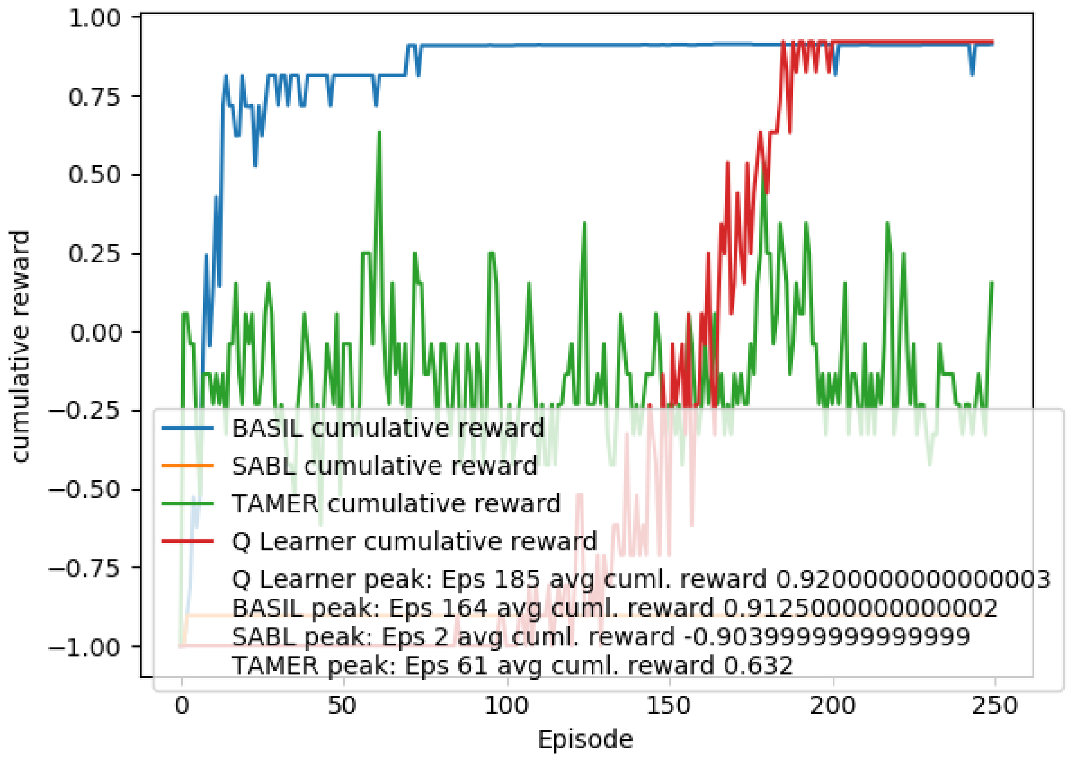


Figure 5.9: Cumulative Reward for each algorithm when feedback contains moderate positive bias $\mu = 0.5$ with moderate variance $\sigma = 0.3$, and negative feedback is always given, but positive feedback is only given with probability 0.1.

5.14 shows the results of the test with bias, but no variance, and demonstrates that the Adaptive Variance Extension still functions comparably to unmodified BASILTAMER when bias is present. The Adaptive Variance Extension does learn slightly slower than the unmodified BASILTAMER which we attribute to the Adaptive Variance Extension lowering its learning rate in response to the variance that inherently emerges from the stochastic environment of Tetris. The results of testing with neither variance nor bias are generally the same as the results depicted in figure 5.14 except that in that case TAMER performs comparatively to BASILTAMER and the Adaptive Variance BASILTAMER Extension. It should be noted that turning down the learning rate will result in a slower improvement in the learned policy, therefore the Adaptive Variance version of BASILTAMER does pay some cost to maintain it's performance and the cumulative reward for the extension when faced with variance does not increase as quickly as the cumulative reward for the regular BASILTAMER algorithm when not subjected to variance, as seen in our earlier experiments. However, we believe that to be unavoidable as there are few cures for noisy data other than increasing the sample size, as even if the noisy data points can be identified, individually they must be discounted meaning that more data will be needed. With

μ	σ	Classic TAMER	BASIL TAMER
0.0	0.0	75.2	74.3
5.0	0.0	0.0	73.5
20.0	0.0	0.0	73.8
80.0	0.0	0.0	71.7
-5.0	0.0	49.0	73.8
-20.0	0.0	3.5	76.2
-80.0	0.0	1.5	75.6
0.0	3.0	15.2	14.1
5.0	3.0	0.0	16.5
20.0	3.0	0.0	14.7
80.0	3.0	0.0	14.0
-5.0	3.0	36.8	17.0
-20.0	3.0	4.4	17.6
-80.0	3.0	1.3	18.4
0.0	30.0	1.3	0.4
5.0	30.0	1.1	0.7
20.0	30.0	0.1	0.7
80.0	30.0	0.0	1.1
-5.0	30.0	1.4	1.5
-20.0	30.0	1.3	1.0
-80.0	30.0	2.1	0.4

Table 5.3: Averages over the last ten episodes of training across all samples for each TAMER algorithm for each permutation of bias parameters.

that in mind, we consider these results to be a success for the Adaptive Variance Extension.

5.5 BASILTAMER Extensions: Silence Experiments Results

In figures 5.15, 5.16, 5.18, 5.19, and 5.20, you can see the results of our tests of the Zero Silence and Average Silence BASILTAMER Extensions. Before discussing the performance of the extensions, it is worth considering the performance of unmodified BASILTAMER in the different conditions. Notably, in this domain, that BASILTAMER maintains performance better when negative feedback is withheld rather than positive feedback. As we previously observed, BASILTAMER, inherently, has some resistance to feedback withholding, as an emergent property of its bias countering properties. The performance of the Average Silence Extension is disappointing, as it, overall, performs worse than unmodified BASILTAMER in the negative feedback withholding cases, although, it does outperform BASILTAMER in the positive feedback withholding cases. The standout performer in these results is the Zero Silence Extension. The Zero Silence Extension maintains a reasonable performance in all experiments and it is often a top performer across the different experimental

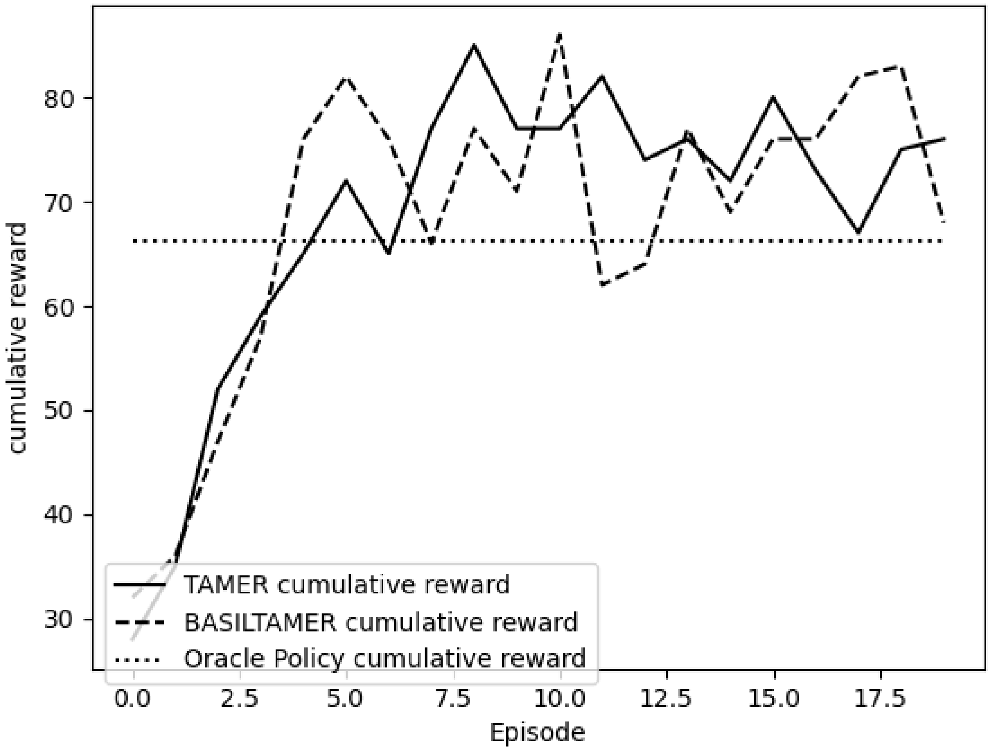


Figure 5.10: Cumulative Reward for each algorithm when feedback contains no bias.

conditions. It is important to remember that these results might not translate across different domains, particularly as they regard the effects of withholding positive feedback versus withholding negative feedback. The results to demonstrate that while the Average Silence Extension did not perform effectively, in general, the Zero Silence Extension did. The results also reiterate that the BASIL technique, inherently, serves as a partial counter to feedback withholding, as feedback withholding does result in a bias in the feedback given.

5.6 Deep BASILTAMER: Results

In Figure 5.21, we have the average cumulative reward as a function of time for each algorithm with each representation. Each plot is the average of policies learned from three separate training sessions with a human instructor. Each training session produced a series of checkpoints of the model as it learned from the human trainer, specifically, only the online trained portion of the model is check-pointed as the feature extraction portion of the model does not change. These checkpoints are taken at irregularly timestamps based upon the training session with timestamps being taken when the human instructor provides feedback, but no more than once per second, with the last version of the model in a given second being the version check-pointed.

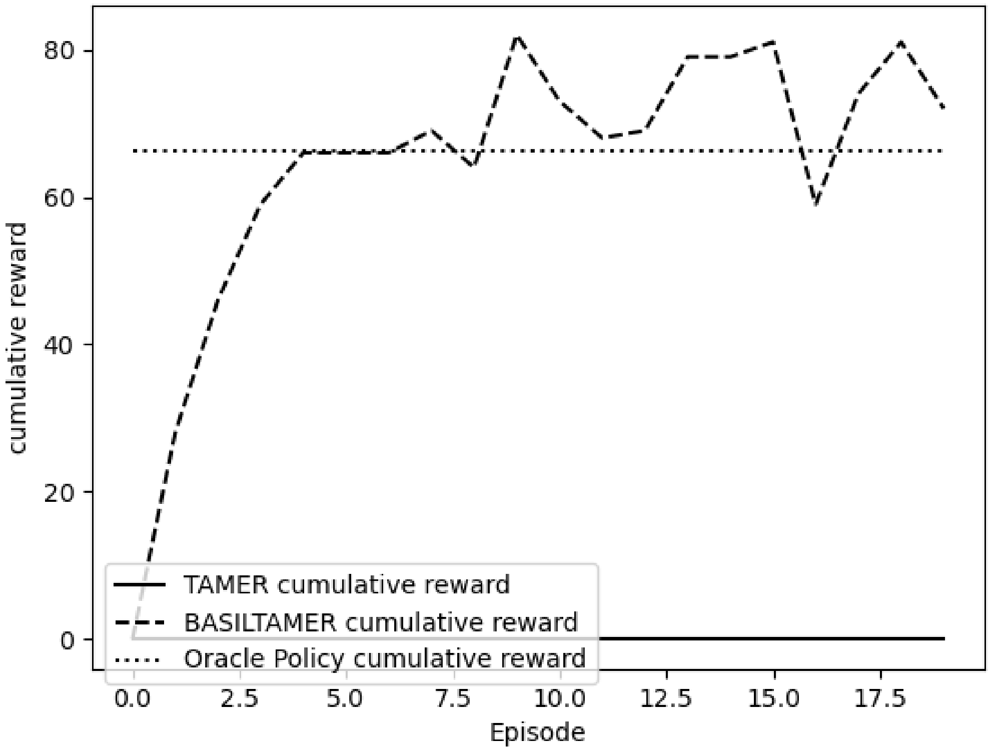


Figure 5.11: Cumulative Reward for each algorithm when feedback contains moderately high positive bias $\mu = 5.0$ with no variance $\sigma = 0.0$.

Each checkpoint is evaluated on five games of Atari bowling to produce an average score for the checkpoint. This creates a curve of data points for a training session, forming a cumulative reward curve for that training session. The reward curves for a particular set of training conditions, meaning a pairing of algorithm and agent presentation, are interpolated and averaged to produce the cumulative reward curve for that training condition. We apply the same technique to produce the later figures using groupings of testing conditions.

As can be observed in Figure 5.21, those training conditions which utilized the BASIL algorithm and either the PKD or Tron-X agent presentations fall behind other PKD and Tron-X testing condition curves. This suggests that the addition of the BASIL technique had a negative effect on the performance of Deep TAMER. This can be very clearly observed in Figure 5.22.

However, the AIBO representation curves are remarkably different. As can be seen in Figures 5.21 and 5.23, the baseline Deep Tamer algorithms had significantly worse performances with the AIBO representation. We believe this is because human trainers had a tendency to avoid punishment with the AIBO representation which is consistent with Bartneck, Reichenback, and Carpenter's[2] as well as Loftin et. al's[25] findings, and our expectations in designing the experiment. Notably, as Fig-

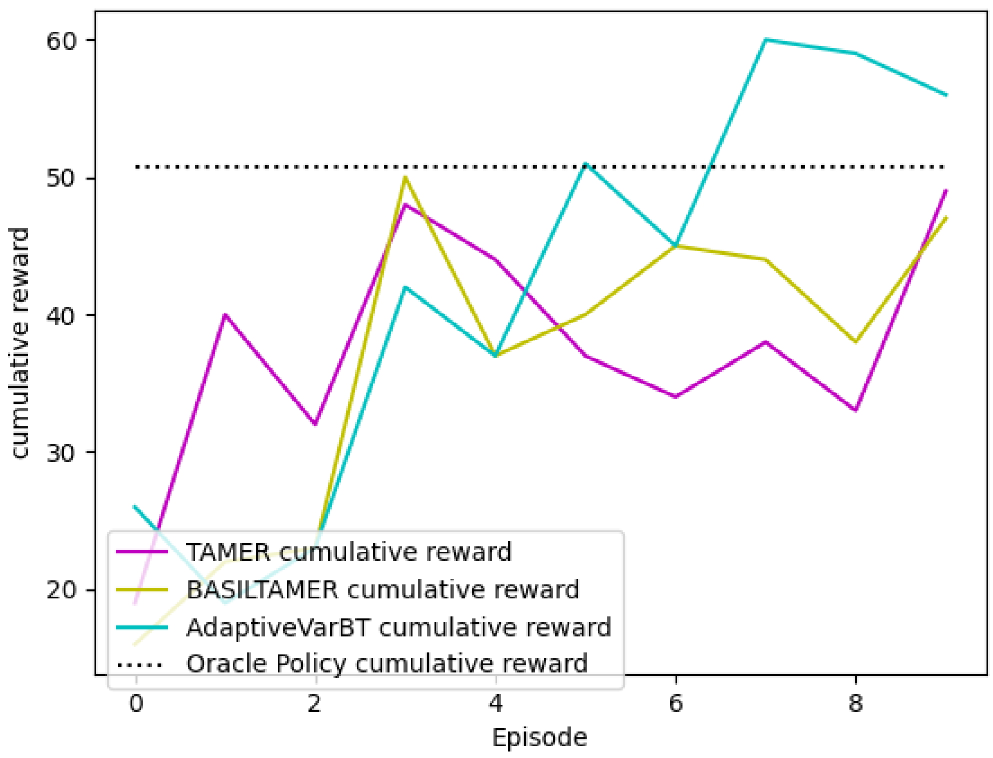


Figure 5.12: Cumulative Reward for each algorithm in Adaptive Variance experiments when feedback contains no bias $\mu = 0.0$ but does contain variance $\sigma = 1.0$

ure 5.22 demonstrates, the BASIL technique was able to account for the bias against punishment for the human trainers under the AIBO conditions and re-contextualized the feedback given in a way that brought the performance of the BASIL DeepTAMER agent in line with the performance of other agents under different conditions. From this we can conclude that a cost is paid when adding the BASIL technique to Deep TAMER if there is not a significant enough bias to account for. However, under circumstances in which there is significant bias in human feedback which would denigrate the learning agent's performance, the BASIL technique is effective in altering the given feedback to counteract the bias and to achieve a better performance.

5.7 Results for Multi-Circumstance Aware Deep BASILTAMER

In Figure 5.24, our results for the Multi-Circumstance Aware Deep BASILTAMER experiments can be seen The results show that the Multi-Circumstance Aware Deep BASILTAMER was able to learn from the mixed presentation feedback much more quickly than any of the algorithms tested in this and other earlier experiments. It is, of course, expected that an algorithm with approximately twice the human feedback at any given time-step would learn faster than an algorithm receiving the normal

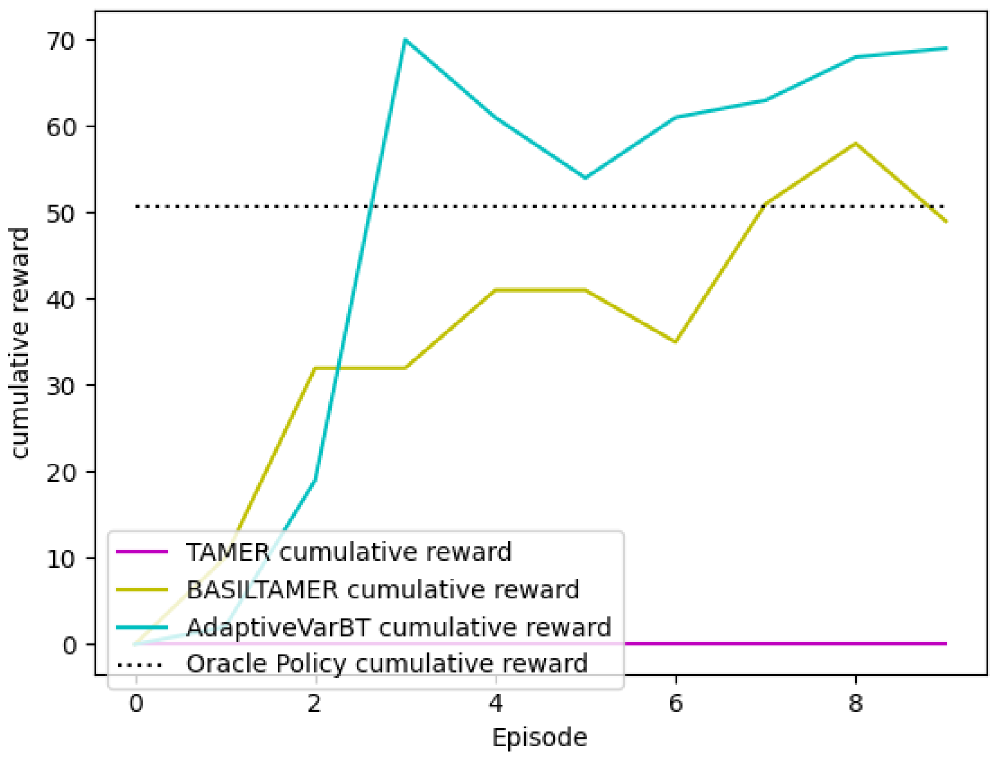


Figure 5.13: Cumulative Reward for each algorithm in Adaptive Variance experiments when feedback contains both bias $\mu = 5.0$ and variance $\sigma = 1.0$

amount of feedback, if all of that feedback can be meaningfully incorporated. In this experiment, we also observed that without explicit modification to support multiple circumstances the BASIL technique does not have an advantage over non-BASIL techniques when both are presented with feedback drawn from two very different biasing conditions. This is unsurprising as a BASIL implementation for normally distributed bias that assumes only one set of bias parameters will inevitably determine a feedback adjustment that is not correct for either of the independent bias distributions. It is, of course, possible that a BASIL based algorithm utilizing only one set of bias parameters, but assuming a more complicated bias distribution, such as a bimodal distribution, in place of the normal distribution, may perform better than the normally distributed bias BASIL algorithm used in our experiment. However, testing this alternative approach was beyond the scope of our experiment and would require a more computationally expensive EM step.

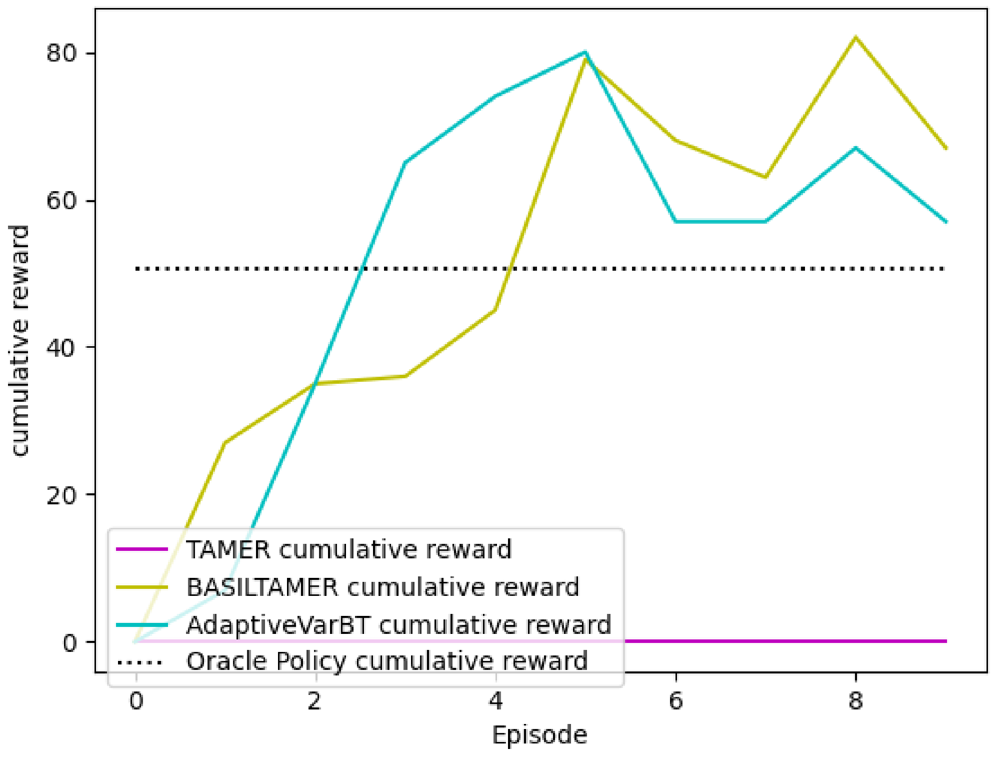


Figure 5.14: Cumulative Reward for each algorithm in Adaptive Variance experiments when feedback contains bias $\mu = 5.0$ but no variance $\sigma = 0.0$

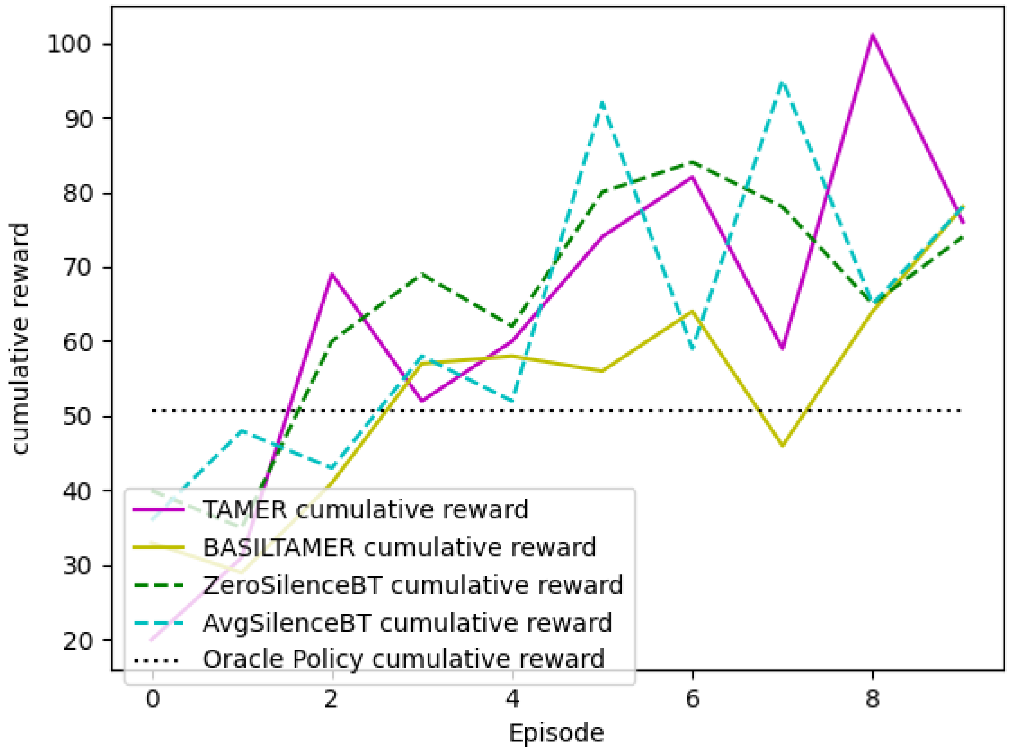


Figure 5.15: Cumulative Reward for each algorithm in the silence experiments when feedback contains no bias $\mu = 0.0$ and is never withheld.

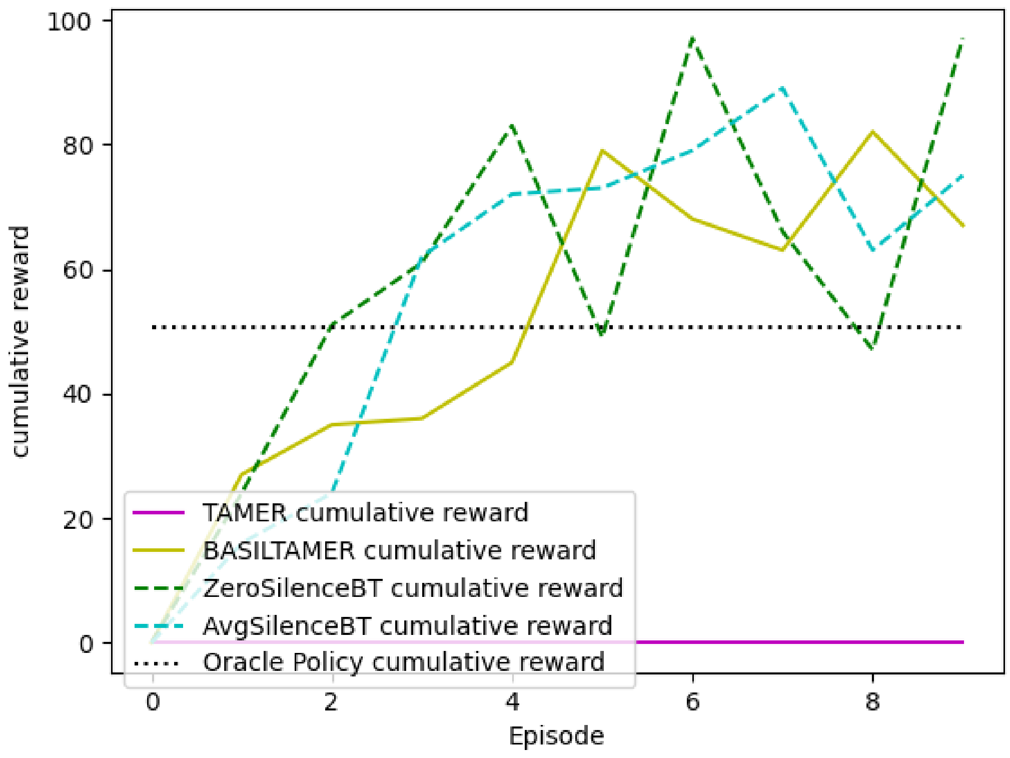


Figure 5.16: Cumulative Reward for each algorithm in the silence experiments when feedback contains bias $\mu = 5.0$ and is never withheld.

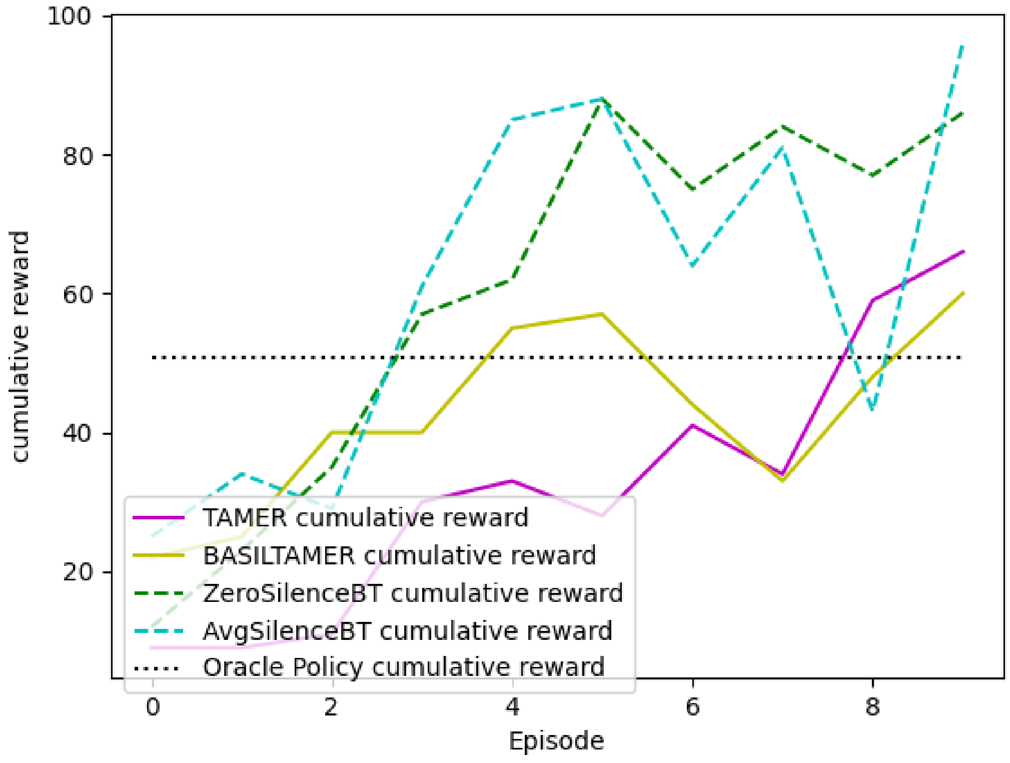


Figure 5.17: Cumulative Reward for each algorithm in the silence experiments when feedback contains no bias $\mu = 0.0$ and positive feedback is withheld at a 50% rate.

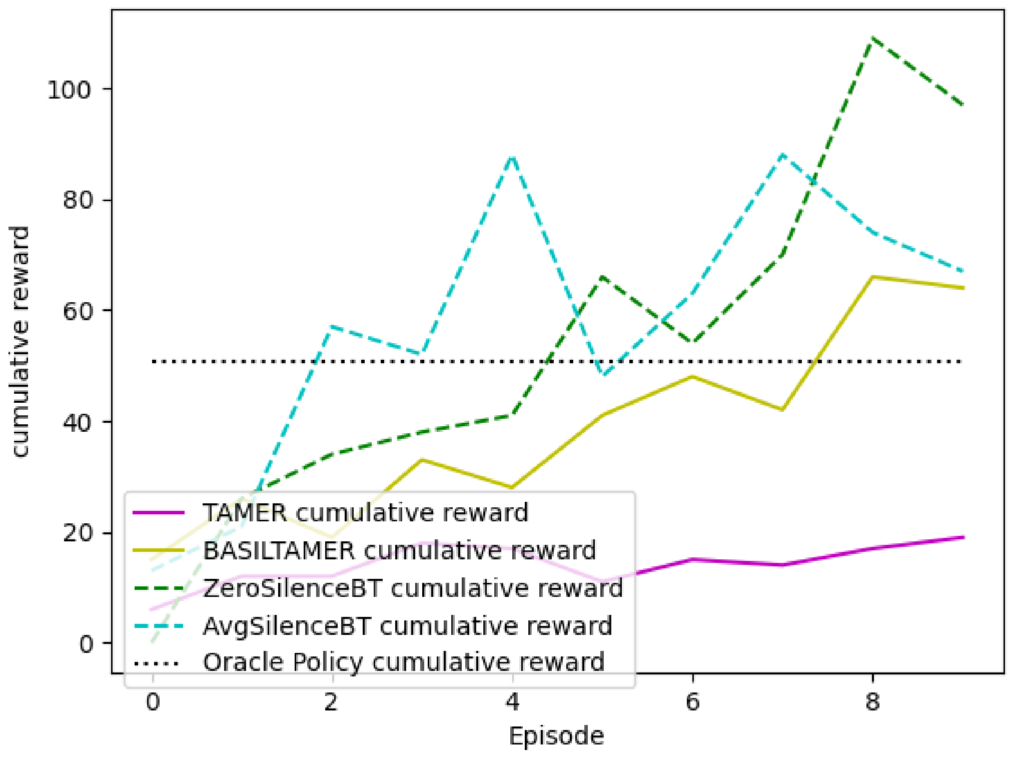


Figure 5.18: Cumulative Reward for each algorithm in the silence experiments when feedback contains no bias $\mu = 0.0$ and positive feedback is always withheld.

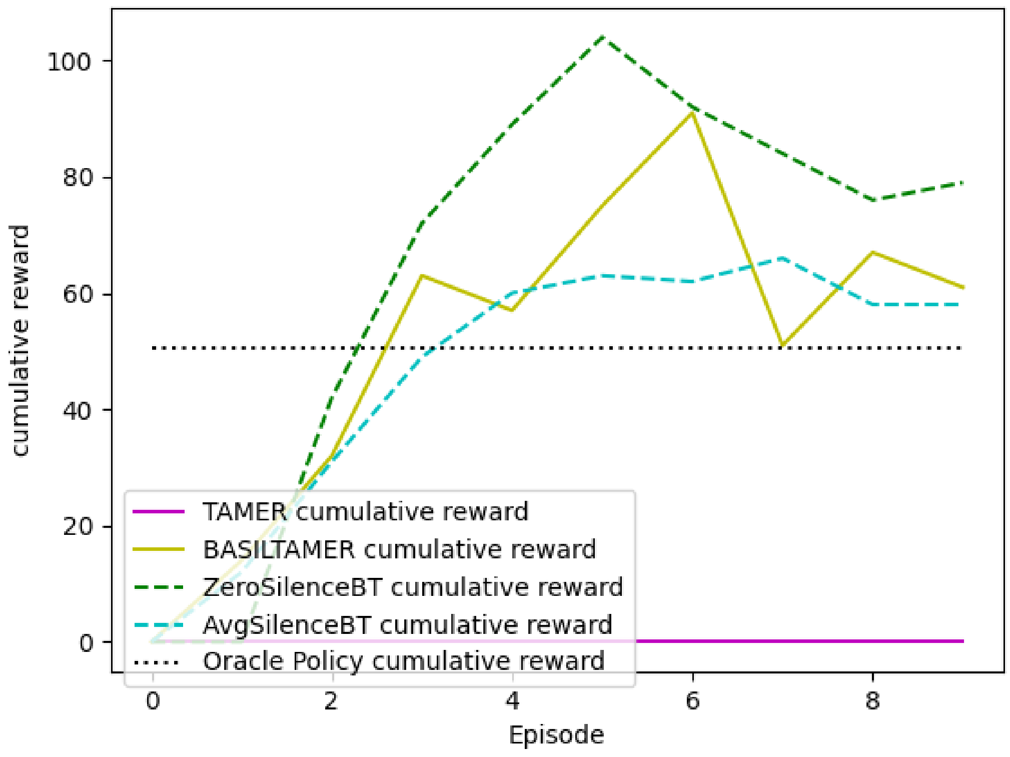


Figure 5.19: Cumulative Reward for each algorithm in the silence experiments when feedback contains no bias $\mu = 5.0$ and negative feedback is withheld at a 50% rate.

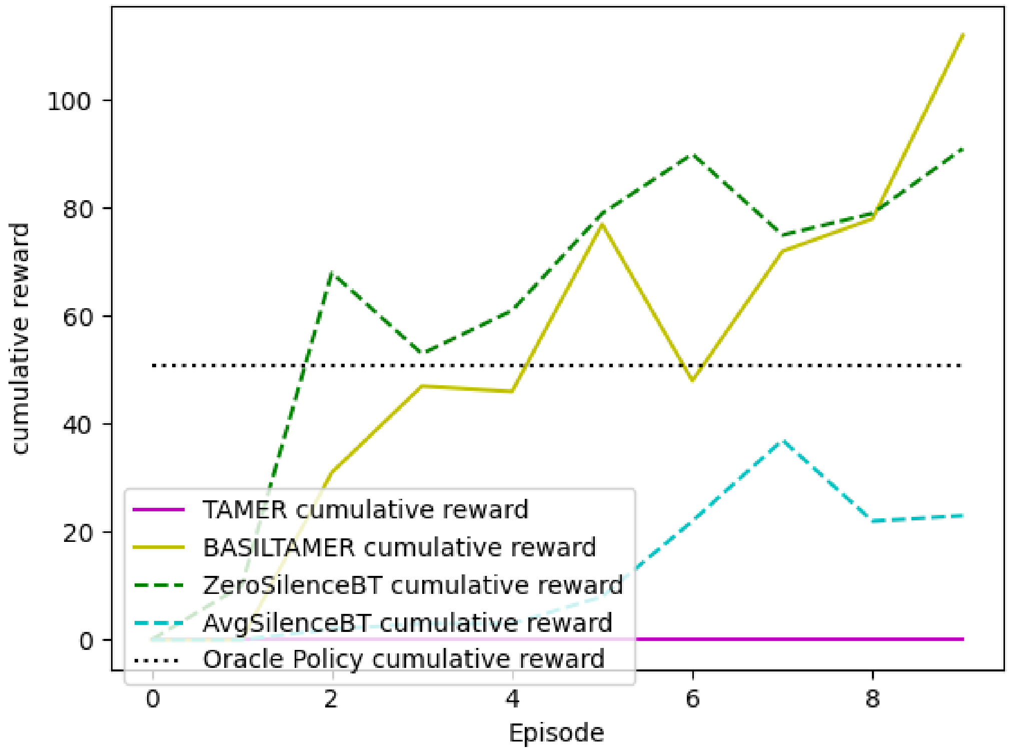


Figure 5.20: Cumulative Reward for each algorithm in the silence experiments when feedback contains no bias $\mu = 5.0$ and negative feedback is always withheld.

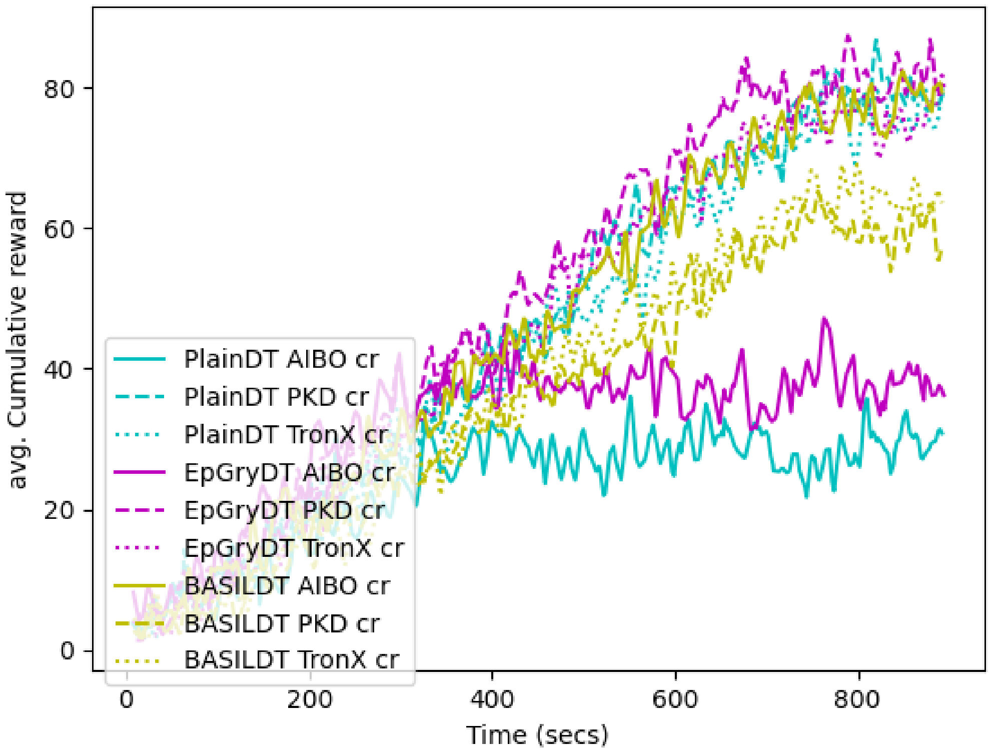


Figure 5.21: Cumulative reward of learned policies as a function of training time for each testing condition combination. Averaged from all applicable data. Observe that the BASIL plots for PKD and TronX fall below the baselines, but the BASIL-AIBO plot significantly exceeds other AIBO plots.

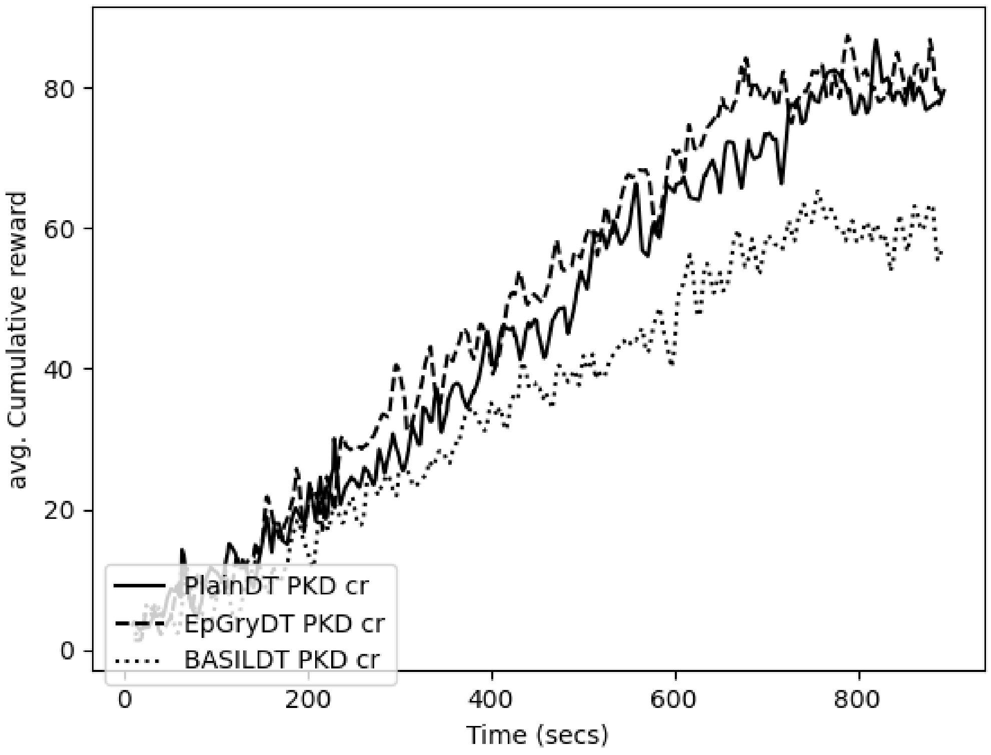


Figure 5.22: Cumulative reward of learned policies as a function of training time for each algorithm with the PKD presentation average from all applicable data. Observe that in the PKD presentation the BASIL technique hinders rather than helps the performance of Deep TAMER.

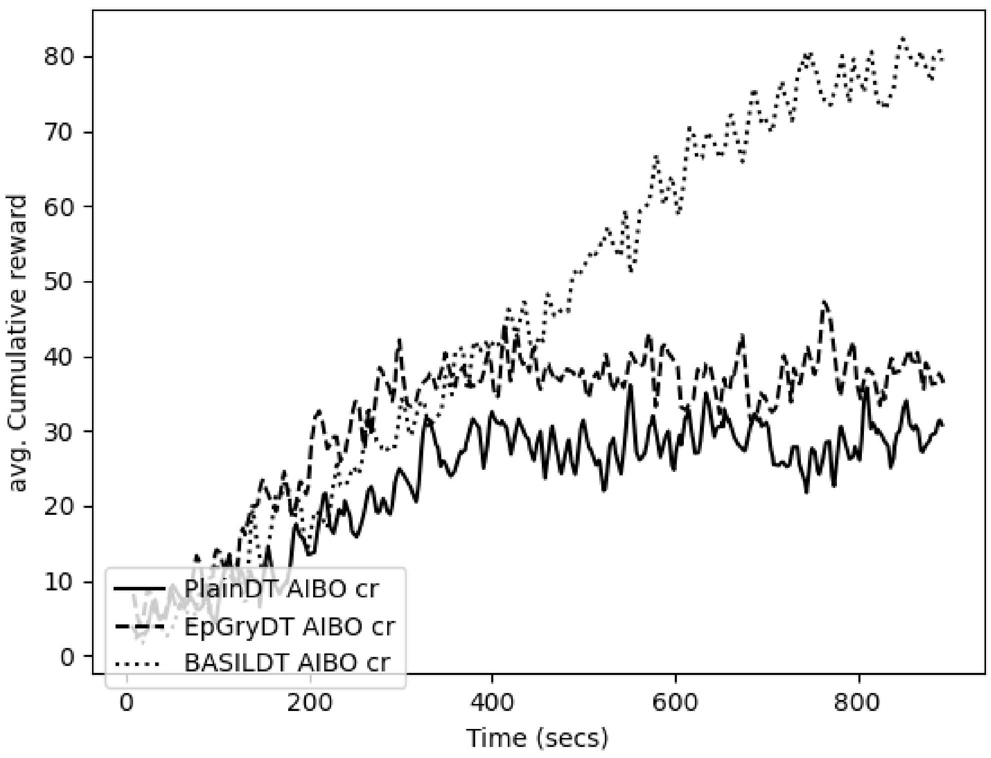


Figure 5.23: Cumulative reward of learned policies as a function of training time for each algorithm with the AIBO presentation average from all applicable data. Observe that in the AIBO presentation the BASIL technique significantly improves the performance of Deep TAMER relative to the baselines.

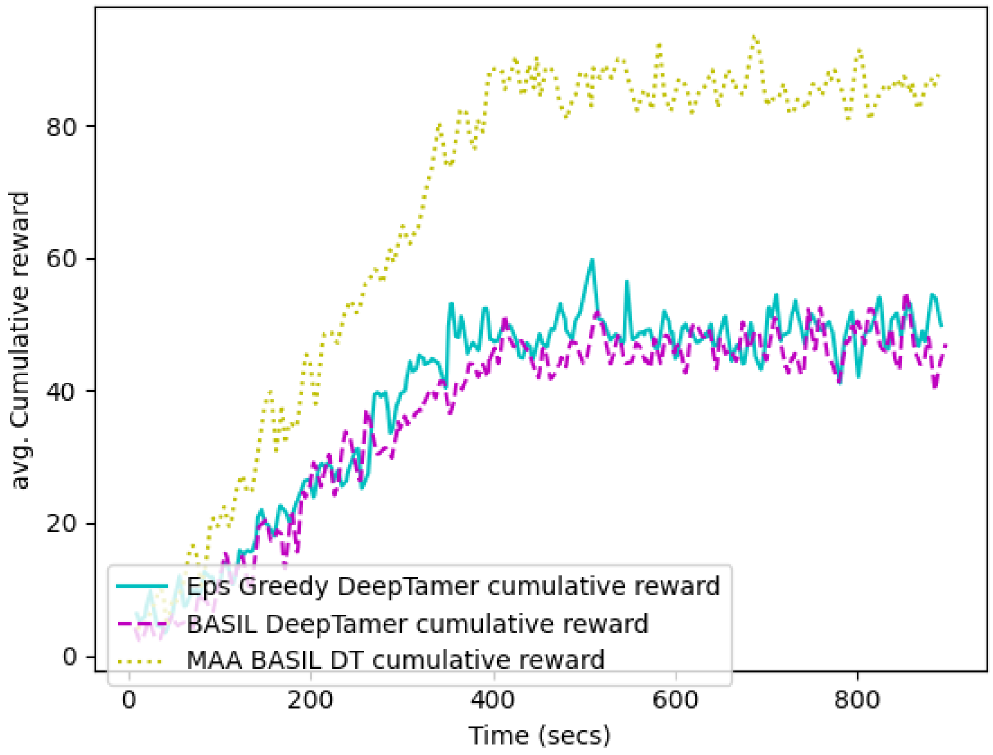


Figure 5.24: Cumulative Reward for epsilon greedy enhanced Deep TAMER (Eps Greedy Deep TAMER cumulative reward), Deep BASILTAMER (BASIL Deep TAMER cumulative reward), and Multi-Circumstance Aware Deep BASILTAMER (MAA BASIL DT cumulative reward) trained offline on AIBO and PKD training sessions recorded in earlier experiments. At each time-step, an algorithm has received all of the feedback given during one PKD training session and one AIBO training session up to that time-step.

Chapter 6 Discussion

In the following sections, we discuss our experimental findings and tests of the BASIL technique.

6.1 Understanding Bias in Human Trainers: Discussion

Our results for our experiment studying the effects of an agent's gender representation are statistically significant, but the size of the effect is relatively mild within the context of our experiment. It is possible the effect could be much more significant in more complex environments that might more strongly invoke human factors. For example, violent or competitive activities like the classic first person shooter game mode of *Death match* may invoke cultural or cross-cultural gender roles regarding violence in competition. It is also possible that stronger effects could be produced with more detailed agent representations such as a higher resolution graphical representation, an audible voice, or more realistic movement.

Regardless of the size of the effect, we have shown that a bias definitely exists based on the gendered presentation of a virtual agent's representation. The presence of this bias could have unforeseen consequences on an agent's training if not accounted for. Whether the bias results in overall better feedback for male or female representations is task and algorithm dependent. For example, classic reinforcement learning algorithms perform best with strong, consistent feedback signals. Our results indicate that this could be better achieved by using the William representation since the feedback given to this agent was less variable than the feedback given to Alice. If one wants to encourage primarily positive feedback during training, as is often the case when people professionally train living beings such as dogs or children, then our results indicate that the Alice representation would be more suitable. In some applications, neither bias may be universally preferable.

The implications of our research for social psychology are complex. A bias is confirmed to exist, with the agents receiving different amounts of reward and punishment despite behaving absolutely identically. The magnitude of this effect is small and this could be a reflection of several different possibilities. Previous research that found larger disparities in punishment for example, could be attributed to a mixture of bias and behavioral differences. Our effect could be muted by the knowledge that our agent is virtual and the limits of our agent presentation, or, a stronger effect might be observed with a different task than grid navigation. The strong similarity between the bias produced by the agent's presented gender and the effect of the instructor's gender is intriguing. If we assume that the human subjects would provide feedback with the same biases to a human student as they did to the virtual agent, then we might conclude that people teach as they have been taught. In other words, trainers may be more likely to favor harsher punishments if they were more likely to get them and more likely to favor greater rewards if they were more likely to receive them, etc. However we have only shown correlation here, and much more study is needed.

The ethical implications are similarly complex. The source of the observed bias could be either learned or neurologically inherent, and may be a combination of those factors. Given that bias was observed, the question of the ethicality of sculpting the agent's presentation to attain the best pattern of feedback is raised. On the positive side, this could result in more effective agents who better perform a desired task and therefore benefit from human interest, while, on the negative side, such use could reinforce these biases in culture, which could apply to interactions between humans. We suspect that, in practical terms, the desire to make agent training as effective as possible will be yielded to, however, this is worthy of greater discussion and research in the future.

An additional theoretical issue is agents that can influence their own presentation. Given sufficient learning capabilities and data to learn from, such agents could sculpt their presentations to maximize feedback given to them by human instructors. As machine agents have an increasing presence in online interactions, video games, etc., this has the potential to create circumstances where agents manipulate humans to give them the best possible feedback, while not performing their intended function. This is similar to the 'Big Red Button' problem, where an embodied agent, one with a physical real world presence, such as a robot, learns that a large red button or other input device terminates its learning session, thus denying it the chance for future positive feedback and either takes steps to disable the button or to disable the human who would push the button. While such issues may be someways out from a practical stand point, they remain issues that must be resolved eventually.

The perceived age of the agent representations is a potential compounding factor that we attempted to address, but cannot entirely mitigate within the limits of pixel art representation. Effort was made in the design of the pixel art representations to make them appear as 'adult' as possible regarding hairstyle, the presence of facial hair, and clothing, while, simultaneously, attempting to convey a clear gender identity. Despite these efforts, we have received some additional feedback that suggests some subjects still perceived the pixel-art as representing children. For this study, we failed to anticipate that there would still be confusion over the representations' presented age, and so, did not explicitly survey subjects for what age they perceived the representations to be. Therefore, the compounding effects of perceived age is a matter for future work.

The grid world task is limited in that 'bad behavior' is restricted to time wasting and self-endangerment, and does not include anything more directly disruptive, aggressive, or violent. These limitations have applications for interactive machine learning applications which will often be using tasks that differ greatly from the grid world game. Others may consequently observe different biases in the feedback given. However, even in this incredibly simple environment with a very limited representation, a statistical significant bias exists and this indicates that the effects of agent representation are something users of interactive machine learning should consider.

This research has completed our first research goal for this work by increasing our understanding of the ways in which factors, such as agent presentation, can influence feedback given to interactive reinforcement learning algorithms. However, our research has revealed that there remains much work to be done in this field, particularly as to how an agent's gender representation affects the feedback given by a human trainer and to what extent it affects the feedback given; as well as in what way biased feedback is provided for a variety of tasks performed by gendered agents, to provide a few examples of the avenues which this research can take.

6.2 The BASIL Technique in a Gridworld: Discussion

As demonstrated in the results, the BASIL technique allows for very rapid learning under even very heavy biases that warp all human feedback to the negative or positive reward values. However, this is not without cost. When bias is not present, BASIL has a slightly more erratic performance than traditional interactive reinforcement learning algorithms that do not account for bias.

It is also necessary to choose a correct distribution type for the bias modeling. Depending upon the distribution chosen and the way bias is actually distorted, you may get better results accounting for the wrong type of bias than for not accounting for any bias at all; however, this is not guaranteed. The BASIL technique is only applicable for distribution types where the maximum likelihood function can be calculated. For some problems performing the EM calculation may be costly. This may be offset by not preforming it at every new instance of feedback. It is up to an agent's engineers to find a balance that works for their problem.

In our experiments, we tested only one algorithm using the BASIL technique. There are many ways in which the technique could be used in conjunction with other algorithms. For example, there is no reason an engineer could not develop an algorithm that uses feedback provided by a human trainer filtered with the BASIL technique in addition to an environmental reward algorithm. It is also possible to integrate the BASIL technique into existing interactive learning algorithms such that bias is filtered before feedback is provided to the main algorithm.

The BASIL technique satisfies our secondary research goal of developing a technique and algorithms for interactive reinforcement learning agents that account for bias in feedback given by human trainers.

6.3 BASILTAMER: Discussion

As demonstrated in the results of both the Gridworld and Tetris experiments, the BASIL technique allows for very rapid learning under even very heavy biases that warp all human feedback to the negative or positive reward values. This is not without cost. It is necessary to choose a correct distribution type for the bias modeling. Depending upon the distribution chosen and the way bias is actually distorted, you may get better results accounting for the wrong type of bias than for not accounting for bias at all; however, this is not guaranteed. The BASIL technique is only applicable for distribution types where the maximum likelihood function can be calculated. For some problems, performing the EM calculation may be costly. This may be offset by not preforming it at every new instance of feedback. It is up to an agent's engineers to find a balance that works for their problem. In our experiments, we tested only a few algorithms using the BASIL technique. There are many ways in which the technique could be used in other algorithms. For example, there is no reason an engineer could not develop an algorithm that uses feedback provided by a human trainer filtered with the BASIL technique in addition to environmental reward. It is also possible to integrate the BASIL technique into other existing interactive learning algorithms besides TAMER such that bias is filtered before feedback is provided to the main algorithm.

6.4 BASILTAMER Extensions: Discussion

Readers should recall that we performed a collection of experiments testing three extensions to BASILTAMER. One of these extensions, which we refer to as Adaptive Variance, was intended to take advantage of BASIL's EM process to compute the variance in the feedback given and, in turn, utilized that to dynamically adjust the algorithm's learning rate to counter noisy feedback. The other extensions, were both designed to address feedback withholding, similar to SABL and ISABL. These two extensions were tested against each other alongside the baselines of TAMER and BASILTAMER. An astute reader might question why these algorithms were not compared to SABL or ISABL. This is because neither SABL nor ISABL effectively scale to domains, such as Tetris, without heavy modification to their forms, as presented by Loftin et. all[25]. While it may be possible to make such adaptions to the SABL and ISABL algorithms, we felt that doing so would require changes that were significant enough that the resulting algorithms would be new derived works and not clearly the same as the original algorithms. Furthermore, as the BASIL technique does owe a significant heritage to ISABL, we felt that implementing another baseline that was intended to be a translation of SABL or ISABL into the more complicated domain of Tetris would be redundant.

Regarding the experiments testing these extensions, the results were mixed. The Adaptive Variance Extension was clearly effective at its intended purpose, although it introduces an additional hyper-parameter that must be set and it will have a different optimal value in different domains and circumstances. In our silence experiments, we found that the Average Silence Extension was not generally effective for the purpose for which it was intended, although it did outperform unmodified BASILTAMER in some circumstances. The Zero Silence Extension, on the other hand, was fully effective at countering silence in our experiments. This result surprised us as the Zero Silence approach is far simpler and, some might argue, so simple as to be trivial. However, our results demonstrate that, sometimes, simpler solutions are more effective and serve as a warning against over-complicated solutions.

These Extensions complete our tertiary goal of combining our techniques for bias with techniques to address variance and inconsistency in feedback given by human trainers.

6.5 Deep BASILTAMER: Discussion

Our Deep BASILTAMER experiments demonstrated that the Deep BASILTAMER algorithm did not preform as well as the Deep TAMER algorithms unless the bias of the feedback was particularly high as was demonstrated by the performance of the algorithms in regards to the AIBO representation. This was most likely due to human trainers avoiding punishing the AIBO representation. Again, this is consistent with effect observed by Bartneck, Reichenback, and Carpenter's research[2] as well as with Loftin et. all's[25] findings, and our expectations in designing the experiment. We were able to show that the BASIL technique, when applied to Deep TAMER, was able to account for the bias against punishment possessed by the human trainers under the AIBO conditions and re-contextualized the feedback given in a way that brought the performance of the Deep BASILTAMER agent in line with the performance of other agents under different conditions. However, we can concluded from these experiments that the cost paid to augment Deep Tamer with the the BASIL technique is too high when there is not a significant enough bias that needs to be accounted for. However, under circumstances in which there is significant bias in human feedback which would denigrate the learning agent's performance, the BASIL technique is effective in altering the given feedback to counteract the bias and to achieve a better performance.

It is worth discussing the overall performance of our Deep TAMER implementation in comparison to the results of Warnell et. all[34]. Notably, our implementation received peak scores in the 80s while Warnell et. all report Deep TAMER as being capable of achieving a score of 160 for all of their trainers. It is our belief that this discrepancy arises from areas where we could not uncover the details of Warnell et. all's implementation and had to determine our own design for the implementation.

The area of concern for our implementation involves the Deep auto-encoder that serves for Deep TAMER as a feature extraction. While Warnell et. all provided sufficient details on the structure of the network, as far as we are aware they have not published information on the specifics of the training, such as hyper-parameters, number of training steps, etc.; nor have they published the weights of their auto-encoder, to the best of our knowledge. Our own auto-encoder was only somewhat effective, however, as it did not reliably reconstruct the position of the ball within the image when utilized in conjunction with the decoder and it is our belief that our auto-encoder has significant room for improvement. Because of the limitations of our auto-encoder, during the experimental design phase, we also pursued utilizing hand coded feature extractors in place of an auto-encoder. We found our results with our hand coded feature extractor comparable to our auto-encoder implementation during our initial testing. Therefore, we utilized the hand coded feature extractor in our experiments because it was less computationally demanding than the auto-encoder and could be deployed on a wider variety of machines. It could be argued that replacing the auto-encoder with a feature extractor that is not a neural network fundamentally changes the algorithm so that it is no longer precisely Deep TAMER, however, we hold that because of the fundamental separation between the visual auto-encoder and the online trained portion of the network that takes features and produces an evaluation of actions that it is appropriate to treat the auto-encoder as something that can be replaced with other techniques, as even two auto-encoders with the exact same structure may produce very different sets of features for the agent to evaluate. We suspect that our feature extractor's limitations is the primary cause of the discrepancy in the performance of our Deep TAMER implementation and Warnell et. all's.

There is one other area that we also suspect may have contributed to the relatively low performance of our implementation compared to Warnell et. all's, which is the interface for human trainers to provided feedback to the agent. While Warnell et. all did provide a wealth of detail on the way in which feedback instances were matched with game states, timestamps, and actions; we have been unable to find any details on the user interface that their human subjects provided feedback through. It is very possible that differences between Warnell et. all's user interface and our own played a role in the performance discrepancy.

It is worth noting that while the performance we observed was well behind Warnell et. all'ss, it is generally comparable to or exceeds the performance of non-interactive reinforcement learning techniques, such as those that Warnell et. all presented as a baseline in their experiments. While our results could be interpreted as a failure to reproduce Warnell et. all's work, we do believe that failure stems solely from insufficient details provided by Warnell et. all in some areas of their available research for us to duplicate their experiment and not from any problem or issue with the Deep TAMER algorithm.

6.6 Multi-Circumstance Aware Deep BASILTAMER: Discussion

Our experiments show that in order to gain benefits from the BASIL technique when feedback is biased with biases drawn from two highly distinct bias distributions, such as the feedback given to our AIBO and PKD representations, that it is beneficial to use a modification of the BASIL technique that explicitly accounts for the multiple distributions, such as the one we presented. We showed that without such adaptions the BASIL technique may not provide benefits, in this circumstance. We also observed that our Multi-Circumstance Aware BASIL algorithm was able to reach peak performance in approximately 400 seconds of training session time by utilizing feedback from multiple human trainers. In comparison to the algorithms from our earlier Deep BASILTAMER Experiments which obtained peaks of a similar score in approximately 700 to 800 seconds, the Multi-Circumstance Aware BASIL algorithm reached the same peaks in almost half the time. While Multi-Circumstance Aware Deep BASILTAMER algorithm received approximately the same number of training feedback instances to reach that peak, the ability to source that feedback from multiple human trainers may enable this technique to be used on larger problems where it is unfeasible for individual human trainers to contribute all of the feedback needed to achieve a desired level of performance.

Copyright© Jonathan Indigo Watson, 2023.

Chapter 7 Conclusion

In conclusion, Interactive Reinforcement Learning is a variant of reinforcement learning that includes humans as an additional source of information to learn behaviors from. On many tasks, it can allow agents to learn effective behavior much more quickly than using classical reinforcement learning. However, introducing humans directly into the process introduces complicated human factors that can skew the feedback given and may vary depending upon the humans in question, the presentation of the agent, the problem being presented, and myriad other factors.

In this body of work, we had three goals:

- Increasing our understanding of the ways in which factors, such as agent presentation, can influence feedback given to interactive reinforcement learning algorithms.
- Developing a technique and algorithms for interactive reinforcement learning agents that account for bias in feedback given by human trainers.
- Combining our techniques for bias with techniques to address variance and inconsistency in feedback given by human trainers.

To accomplish our first goal, we explored the effects of an agent's presented gender on human feedback provided in a *Wizard of Oz* experiment where we compared feedback given to a virtual agent who was presented as either male or female. Human trainers were tasked with helping a virtual agent learn to navigate a grid environment with obstacles and hazards present. We also gathered demographic information from the human instructors and analyzed feedback variation between demographic groups where possible. We found a statistically significant effect of the agent's presented gender on feedback given by a human subject. The size of the effect was relatively mild within the context of our experiment; however, the effect was very statistically significant and could be much greater in more complex environments. We also discovered a very similar effect of the human instructor's gender identity on the feedback given. We discussed the implication of these effects and their similarity as it regards interactive machine learning and social psychology. This experiment also provides a template for further research in how human bias effects feedback given.

To complete our second goal, we presented a general technique called Bias Adaptive Statistical Inference Learning (BASIL) to filter out bias from feedback. We presented a simple algorithm using the BASIL technique for normally distributed bias and tested it against well established interactive reinforcement learning and classical reinforcement learning algorithms. We found that the BASIL technique was able to filter both positive and negative bias out of feedback provided by simulated human trainers. This allowed our algorithm to perform well on highly distorted feedback that rendered other interactive reinforcement algorithms incapable of learning any path to the goal state. Our BASIL based algorithm also performed relatively well on other types of feedback distortions such as those that SABL [25] was designed to handle. However, it did not perform as well under those distortions as algorithms that specifically accounted for them.

In order to expand the scope of problems to which our BASIL technique could be applied, we presented an algorithm using the BASIL technique applied to the existing TAMER algorithm for normally distributed bias and tested it against classic TAMER in the domain of Tetris. We found that our algorithm, BASILTAMER, was able to filter both positive and negative bias out of feedback provided by simulated human trainers. This allowed our algorithm to perform well on highly distorted feedback that rendered classic TAMER incapable of learning how to clear any lines. We discussed ways to integrate the BASIL technique into other interactive reinforcement learning work and the options it provides for studying the bias in human feedback in the future.

We also presented three extensions of the BASILTAMER algorithm designed to handle variance and silence, completing our third goal of combining our techniques for bias with techniques to address variance and inconsistency in feedback given by human trainers. While one of our silence techniques was not as effective as we had hoped, the very simple technique of treating no feedback given as an explicitly neutral feedback, and then allowing the BASIL technique to address the bias in that value as 'normal' was quite successful at countering feedback withholding in the domain of Tetris. Our Zero Silence BASILTAMER Extension is a lesson in simple solutions and a demonstration of the inherent power of the BASIL technique. Our Adaptive Variance Extension was also successful in serving its intended function, which was to take advantage of the BASIL technique's need to calculate feedback distribution parameters to modify the learning rate and to increase resilience to noisy feedback.

In order to fully realize our second goal, we sought to adapt the BASIL technique to Deep Learning. In order to do this, we have presented an algorithm using the BASIL technique applied to the Deep TAMER algorithm and have determined that the costs of implementing Deep BASILTAMER resulted in the technique being useful for tasks where a high bias is expected. We also observed similar results to the work of Bartneck, Reichenback, and Carpenter [2] in which the human trainers provided avatars of varying anthropomorphism different feedback. By applying the BASIL technique to Deep TAMER, we completed our secondary goal.

We have also produced The Multi-Circumstance Aware Deep BASILTAMER algorithm and have demonstrated that it was able to learn from mixed presentation feedback much more quickly than any of the algorithms tested. This extension to Deep BASILTAMER adapts the technique to not only the scale of Deep Learning, but, additionally, to separate human trainers with differing biases. With the development of the Multi-Circumstance Aware Deep BASILTAMER algorithm we hold that our second goal is fully realized and all of our goals have been accomplished. We have developed a broad and general technique for handling some of the complicated factors that introducing humans into the loop of reinforcement learning brings.

Our work has a number of implications for machine learning and society as a whole. Our first study provides a framework for studying inter-human biases as applied to machine learning agents. The main body of our work produced several algorithms of increasing complexity and scope for training interactive machine learning agents to accomplish any task that can be modeled as a Markov Decision Process in a manner that accounts for human biases in the feedback that they provide to interactive reinforcement learning agents. The technique is also provided in a general form that can be adapted to additional algorithms as needed. Consequentially, it is now possible to apply IRL to problems where human biases would typically pollute the feedback to a degree that would harm the agent's potential to learn. As machine learning becomes an increasing larger part of people's lives, it is necessary to provide a means to non-programmers of training an artificial intelligence to perform a variety of tasks relevant to them and their individual needs.

In general, IRL provides a framework for fulfilling this need. However, it may be stymied by biases in the trainer's feedback and non-experts cannot reasonably be expected to have a full understanding of what constitutes effective training. Therefore, the BASIL technique provides a base for producing a large variety of tools to enable non-expert users to train machine learning agents to accomplish tasks that would otherwise not be feasible for machine learning algorithms or to reach levels of performance that would, otherwise, require advanced domain expertise. Furthermore, the BASIL technique enables providers of large scale products and services to integrate user feedback from a large user base and to address the inherent bias in the collective feedback for IRL applications. The BASIL technique also enables a high degree of automatic customization to fit individual users desires and preferences without the user explicitly needing to determine or set the values for those preferences. This sort of automatic customization will be increasingly beneficial as machine learning technology is increasingly used to provide digital personal assistants, potentially allowing each user of such systems to have a personalized assistant uniquely tailored to their preferences, and tasks that they would like to accomplish.

Copyright© Jonathan Indigo Watson, 2023.

Bibliography

[1] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning, 2004.

[2] Christoph Bartneck, Juliane Reichenbach, and Julie Carpenter. The carrot and the stick: The role of praise and punishment in human–robot interaction. *Interaction Studies*, 9(2):179–203, 2008.

[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[4] Maya Cakmak, Crystal Chao, and Andrea L Thomaz. Designing interactions for robot active learners. *IEEE Transactions on Autonomous Mental Development*, 2(2):108–118, 2010.

[5] Maya Cakmak and Manuel Lopes. Algorithmic and human teaching of sequential decision tasks. In *AAAI*, 2012.

[6] Clara Cannon and Abrar Anwar. Calibrated feedback for reinforcement learning. *Preprint*, 2022.

[7] Thomas Cederborg, Ishaan Grover, Charles L Isbell, and Andrea Lockerd Thomaz. Policy shaping with human teachers. In *IJCAI*, pages 3366–3372, 2015.

[8] Sonia Chernova and Andrea L Thomaz. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(3):1–121, 2014.

[9] Francois Chollet et al. Keras, 2015.

[10] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. 1977.

[11] A Chris Downs and Darryl C Gowan. Sex differences in reinforcement and punishment on prime-time television. *Sex Roles*, 6(5):683–694, 1980.

[12] Andrew J Elliot. Color and psychological functioning: a review of theoretical and empirical work. *Frontiers in Psychology*, 6:368, 2015.

[13] Lee Ellis and Christopher Ficek. Color preferences according to gender and sexual orientation. *Personality and Individual Differences*, 31(8):1375–1379, 2001.

[14] Paolo Frassanito and Benedetta Pettorini. Pink and blue: the color of gender. *Child's Nervous System*, 24(8):881–882, 2008.

[15] James F Gregory. The crime of punishment: Racial and gender disparities in the use of corporal punishment in us public schools. *Journal of Negro Education*, pages 454–462, 1995.

[16] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles Lee Isbell, and Andrea Lockerd Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In *NIPS*, 2013.

[17] Charles R. Harris, K. Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020.

[18] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[19] Charles Lee Isbell, Christian R. Shelton, Michael Kearns, Satinder P. Singh, and Peter Stone. A social reinforcement learning agent. In *Agents*, 2001.

[20] Taylor A. Kessler Faulkner, Elaine Schaertl Short, and Andrea L. Thomaz. Interactive reinforcement learning with inaccurate feedback. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7498–7504, 2020.

[21] Faisal Khan, Xiaojin Zhu, and Bilge Mutlu. How do humans teach: On curriculum learning and teaching dimension. In *NIPS*, 2011.

[22] W. Bradley Knox, Brian D. Glass, Bradley C. Love, W. Todd Maddox, and Peter Stone. How humans teach agents - a new experimental perspective. *International Journal of Social Robotics*, 4:409–421, 2012.

[23] W. Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: the tamer framework. In *K-CAP*, 2009.

[24] Mika Lehdonvirta, Yosuke Nagashima, Vili Lehdonvirta, and Akira Baba. The stoic male: How avatar gender affects help-seeking behavior in an online game. *Games and culture*, 7(1):29–47, 2012.

[25] Robert Tyler Loftin, James MacGlashan, Bei Peng, Matthew E Taylor, Michael L Littman, Jeff Huang, and David L Roberts. A strategy-aware technique for learning behaviors from discrete human feedback. In *AAAI*, pages 937–943, 2014.

[26] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[27] Steven R Shaw and Jeffery P Braden. Race and gender bias in the administration of corporal punishment. *School Psychology Review*, 19(3):378–383, 1990.

[28] Sonja B Starr. Estimating gender disparities in federal criminal cases. *American Law and Economics Review*, 17(1):127–159, 2014.

[29] Murray A Straus and Julie H Stewart. Corporal punishment by american parents: National data on prevalence, chronicity, severity, and duration, in relation to child and family characteristics. *Clinical child and family psychology review*, 2(2):55–70, 1999.

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

[31] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.

[32] Andrea L Thomaz and Cynthia Breazeal. Transparency and socially guided machine learning. In *5th Intl. Conf. on Development and Learning (ICDL)*, 2006.

[33] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[34] Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, and Peter Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. 02 2018.

[35] Chris Watkins. Learning from delayed rewards. 1989.

[36] Adam Waytz, Joy Heafner, and Nicholas Epley. The mind in the machine: Anthropomorphism increases trust in an autonomous vehicle. *Journal of Experimental Social Psychology*, 52:113–117, 2014.

[37] Mike Younger and Molly Warrington. Differential achievement of girls and boys at gcse: Some observations from the perspective of one school. *British Journal of Sociology of Education*, 17(3):299–313, 1996.

Vita

Jonathan Indigo Watson

Academic Degrees and Educational Achievements:

James Madison University- Bachelor of Science: 5/10/2014

James Madison University- Distinguished Graduate: 5/5/2016

James Madison University and the CNSS- Certified Information Systems Security Professional: 5/5/2016

James Madison University- Master of Science: 5/7/2016

Publications:

Co-author: We Don't Really Need Quaternions in Geometric Modeling, Computer Graphics and Animation: Here Is Why. Computer-Aided Design and Applications 20(6), 2023, 1061-1073.

Primary author: Bias Adaptive Statistical Inference Learning Agents for Learning from Human Feedback.FLAIRS-34 proceedings 2021-05-11.