Adapting Behavior via Intrinsic Reward

by

Cameron Linke

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Learning about many things can provide numerous benefits to a reinforcement learning system. How to behave to best learn a collection of predictions in parallel in the reinforcement learning setting remains an open problem. The research presented in this work explores the behavior, and effect, that different intrinsic reward mechanisms drive in this multi-prediction, parallel learning, setting. The setup of this thesis is a staple thesis in which the core of the work is an accepted journal paper, with a new introduction and conclusion written by the author. The additional introduction and conclusion are original works. They serve to demonstrate the author's knowledge of the subject matter and journal paper in question, the author's individual perspective on topic area, and the future directions for research that the author would like to pursue. Adapting Behavior via Intrinsic Reward: A Survey and Empirical Study, has been accepted into the Journal of Artificial Intelligence Research (JAIR). The paper is co-authored by Cam Linke (the author of this thesis), Nadia M. Ady, Martha White, Thomas Degris and Adam White. The paper was a collaborative work that began as a proposed conference paper, was accepted as an extended abstract into the Reinforcement Learning and Decision Making (RLDM) conference, and finally was accepted into JAIR. The author's specific contributions were the open-sourced code and notebook for replicating the empirical work; writing sections of the intro, experiments and conclusion; and contributing to the large survey of previous literature performed in the paper.

Preface

The setup of this thesis is a staple thesis in which the core of the work is an accepted journal paper, with a new introduction and conclusion written by the author. The additional introduction and conclusion serve to demonstrate the author's knowledge of the subject matter and journal paper in question, the author's individual perspective on topic area, and the future directions for research that the author would like to pursue. Adapting Behavior via Intrinsic Reward: A Survey and Empirical Study, has been accepted into the Journal of Artificial Intelligence Research (JAIR)[1]. The paper is co-authored by Cam Linke, Nadia M. Ady, Martha White, Thomas Degris and Adam White. The paper was a collaborative work that began as a proposed conference paper, was accepted as an extended abstract into the Reinforcement Learning and Decision Making (RLDM) conference, and finally was accepted into JAIR. The author's specific contributions were the open-sourced code and notebook for the empirical work to be replicated; writing sections of the intro, experiments and conclusion; and contributing to the large survey of previous literature performed in the paper.

[1]Linke, C., Ady, N. M., White, M., Degris, T., & White, A. (2020). Adapting behavior via intrinsic reward: A survey and empirical study. *J. Artif. Intell. Res. 69*, 1287–1332

Acknowledgements

To start I would like to thank my co-supervisors Rich Sutton and Adam White. I could not imagine better people to be able to work with. Rich's insights, passion, and curiosity about the field are infinite and I am very thankful to have had the opportunity to work with him. His combination of ambition and generosity in research is something I hope to emulate. I would not have gotten to this point without Adam. He introduced me to the exciting topic of intrinsic motivation, patiently following different threads and ideas, and taught me how to be a grad student and researcher. Every time I felt confused, stuck, or demotivated Adam was there to encourage me, point me in an interesting direction, and help me to move forward. Adam's ability to think about big, challenging, ideas, while taking meaningful steps to understand them, has shaped how I think about research. Both Adam and Rich helped me fall in love with reinforcement learning, science, and research and I am forever grateful to both of them for that.

Secondly, I would like to thank all of my RLAI and Amii lab-mates. I have had an amazing time collaborating on projects, laughing and commiserating over drinks, and trying to get data to each other at three in the morning when everyone's wifi was down. Every part of grad school was made better going through it with this wonderful group of researchers and friends.

Finally, I would like to thank my family and friends who I am so grateful to have in my life. Specifically my wife and teammate, Ashlyn, for her love, support, feedback, and encouragement throughout the entire process. I could not be more blessed.

Table of Contents

Chapter 1

Introduction

Adaptation of behaviour based on learning can help autonomous agents, natural or artificial, operate effectively in a changing world. A kitten sneaking up on a leaf helps it learn to hunt, two mice wrestling allows them to learn to fend off future attackers, and a baby playing in its crib allows it to learn about the effects that it can have on objects around it (e.g. Gopnik, Meltzoff, and Kuhl, 1999). While later in life having learned these things can become useful, at the time they are interesting to the animal without knowing if it will be useful in the future. Instead, improving the ability to predict and control their environment is its own reward. Learning feels good to the agent, which encourages it to learn to broadly interact with its environment, without having to specifically know what the future holds (e.g. Barto, 2013). Like their natural counterparts, artificial agents also need to be able to learn in a general manner in order to operate independently in the world. A self-driving vehicle could experience far more diverse situations throughout its life than it could learn just from human training. Similarly, a garbage picking robot will likely won't know every environment, situation, or piece of trash it will need to deal with. If we want to have independent agents, we need them to be able to broadly learn many things on their own, without necessarily knowing what the future may require of them.

An agent's understanding of the world can be thought of as its ability to make predictions about what will happen in the future. These predictions can be thought of as an agent's knowledge of the world (as shown by A. White,

2015). For our garbage picking robot its knowledge of its environment can be thought of as the prediction of the streams of data coming in through its sensors. These sensorimotor streams give the robot a rich set of data from which to learn. Predicting whether or not it will keep moving after running into an object becomes the robot's "knowledge" of what a wall vs an object it can push is, without both specifically being labeled. The difference between a soda can, which the robot should throw out, and a reusable water bottle, which it shouldn't, becomes a prediction of its "not garbage" sensor turning on when it drops the item off. Building up on knowledge this way allows basic sensor streams to be combined into rich predictions that give the agent a strong overall understanding of its environment. Ring (in preparation) presented a thought experiment in which The ability to build layers of this predictive knowledge was shown—with a basic contact sensor is enough for a small robot to build up a rich understanding of the rooms it is in.

We would like our agents to constantly improve its predictions as soon as new data is acquired, to learn them online, and not have to wait to update them at some later point in time. This ability to incorporate new data both helps to keep the agent's predictions as up to date as possible, and helps the agent deal with non-stationarity—changes in the environment or sensors that could cause its predictions to become inaccurate. We don't want our agent's predictions to become fixed or to have to wait until some later point to incorporate the effects of these changes. In addition to learning online we want our agent to be able to learn about the effects that different ways of behaving, called policies, that the agent could be doing instead of just its current behavior. Learning predictions for the same way of behaving is called on-policy learning. An effective agent will learn both on-policy as well as learn to make predictions about what would have if it behaved in different ways—off-policy learning. These off-policy predictions can give our agent a more robust understanding of what will happen if it chooses any number of ways of behaving. For example, from the same stream of experience our robot can learn to predict what will happen to its streams of data if it behaves in a more aggressive manner or a more cautious manner. These both may be necessary

modes of moving about the lab at different times, and we would like our robot to learn them both simultaneously. Using off-policy learning has shown the ability for agents to learn to predict many signals at the same time, online, while the agent is in operates in the world (A. White, 2015).

Learning many signals has been shown to be helpful in a variety of settings. The signals can be seen as hints that help a neural network to learn (Suddarth and Kergosien, 1990). Similarly, executing temporally extended sequences of actions can help in sparse reward environments, where there is little or no reward signal. In these environments the agent may otherwise have little to learn from and will dither around looking for a signal (Riedmiller et al., 2018). Regularization of the network can be achieved by learning other predictions, beyond the main target of learning (such as the reward signal), in a deep reinforcement learning setting (Jaderberg et al., 2016). An agent can use the predictive signals as a knowledge of the world to predict what is "next", giving it a general awareness of its environment (Modayil, White, and Sutton, 2014). Parameterizing over possible goals allows an agent to generalize over goals as well as states (Schaul, Horgan, Gregor, and Silver, 2015).

Given an agent that can learn many things at once a question still remains: how should the agent behave so that it can best learn a set of predictions. This process of learning for the sake of learning can has been called many things in literature, including: curiosity, self-supervised learning, intrinsic motivation, or even the more general term of exploration. No matter the name, the heart of the activity is adapting the agent's behaviour to best improve all of the predictions it is trying to learn over time. Here we will use the terms intrinsic motivation to refer to the mechanism of adapting an agent's behaviour based on learning, and intrinsic reward to refer to the reward given to the agent to encourage this behaviour.

This these has three main contributions. The first is a simplified testbed that focuses on the core mechanism driving each intrinsic reward. The second is an empirical comparison evaluating many of the commonly used intrinsic rewards on this testbed. Third, we provide insight into the effect on the behavior of intrinsic rewards of introspective learners—learners that update

their learning rate based on learning progress.

In order to compare various intrinsic rewards we propose a new test bed to allow for a clear evaluation of the behavior of each proposed reward. Many intrinsic rewards have been proposed as to how learning focused behavior could be driven. These intrinsic rewards have been evaluated in different environments each with its own set of complications. Aspects such as function approximation can vary between rewards and have a large impact on the difference between the behavior and results of each of the intrinsic reward. Our testbed simplifies problem down to the basic decision that a curious agent has to make at each step—what action will maximize learning. The proposed testbed is a bandit like environment in which each action returns a scalar signal. These signals mimic the types of signals a real world agent may encounter: highly noisy signals, constant signals, and signals which changes over time. The agent seeks to learn to predict the expected value of the signal received from taking each action. In order for the agent to maximize learning it needs to chose the action that allows it to make the most learning progress. An agent that does well will adapt its behaviour to best learn over time and its predictions will track any changes in the environment.

We evaluated multiple existing methods in this testbed. The goal was not to exhaustively evaluate all methods, but to evaluate representative methods to show the effects of different classes of methods. We primarily focused on literature in the reinforcement learning setting, although in many cases the choice of intrinsic reward was originally inspired by a different domain such as child psychology. We focused on methods that would specifically apply to our problem domain where the agents behaviour would need to be adapted over time to best learn a number of sensorimotor predictions. We concentrated our efforts on model-free methods both to focus the problem and due to the single state nature of our problem domain.

Finally, our review of of different intrinsic reward mechanisms brought us insight into the effect introspective learners can have on an agent. Introspective learners are ones which are able to adapt their learning rate up or down based on the learning progress that the agent is making. We demonstrate that

introspective learners can be helpful with many intrinsic reward mechanisms. We show that a simple, but effective, intrinsic reward is the change in weights of the agent's predictions when those predictions are learned introspectively.

1.1 The Curiosity Test Bed

Many demonstrations of intrinsic motivation exist, and results have been shown in various experimental environments. However at times, the complicated nature of these environments can make it difficult to evaluate an intrinsic reward mechanism. Was it something in the method of function approximation that caused the behaviour? Was it intrinsic mechanism itself? For this reason we propose a simplified testbed that will allow us to focus specifically on the effects of intrinsic reward while holding out as many confounding factors as possible.

The testbed is similar to a multi-armed bandit. The agent is in one state and at each time step takes an action that returns it to that same state. Each action taken returns a signal to the agent. Each of the actions has a learner that is trying to predict the expected value of the signal received from that action. The signals received from each demon are of three types: 1) A constant signal always returning the same constant value, 2) a noisy signal that has a constant expected value, but returns a random number from a normal distribution around that value, 3) a slowly drifting signal whose value changes by a small random number at each time step.

This setting mimics the types of signals an intrinsically motivated agent could receive at each time step. Consider a robot that has sensor signals it is receiving at a given time. An intrinsically motivated robot will want to learn to predict those signal in order to effectively navigate its world. The signals themselves will have different properties as to their properties that effect their predictability. Some will be constant and will be easily learnable by the robot. Others, as popularized by the noisy tv problem (Schmidhuber, 1991a), will have a constant expected value, but at any given time step will have a large amount of noise in the signal. While some sensors are learnable and will need

to have their prediction constantly updated to keep the prediction up to date in a changing world. Each of the above signals is designed to distill the above scenarios to their essence. A robot may have a sensor that gives it a steady, constant stream of data. Here the signal is almost perfectly learnable and the robot should not spend much time on it after having learned its prediction. This is exemplified by the first, constant, signal in the testbed. The second type of signal a robot could encounter is one which is extremely noisy, but has a constant expected value. Static from a television, snow falling captured by a visual sensor, or a broken sensor returning noise, are all captured by the second type of signal in the testbed. These signals have learnable expected values but are highly noisy and will cause the agent's prediction error to be very high. In order to maximize learning the robot should quickly learn the expected value and ignore these signals, despite the high prediction error on any given step. Finally a robot should have many sensors capturing rich, learnable information about the world. A bump sensor will tell it when it has run into something, a distance sensor will tell it how close it is to different objects, and various sensors for things like battery levels will help it have an understanding of the environment it is operating in. All of these predictions will change over time as the environment, and sensors themselves, change. The third signal in the testbed represents these types of signals—ones which represent an expansive, constantly changing, environment. A robot that learns well will consistently take actions that update these predictions.

The testbed provides three different situations that simulate common patterns a learning agent will face. The Drifter-Distractor problem creates the most basic setting in which the agent has four signals it seeks to learn—one drifting signal, two noisy signals, and one constant signal. As mentioned previously this mimics a common set of signals a robot could receive as it operates. A well performing agent will learn to largely ignore the noisy and constant signals once their expectation has been learned—ultimately focusing on the learnable drifting signal.

The second problem, Switched Drifter-Distractor, switches the signals associated with each action after 50,000 timesteps. After this change there is

one noisy signal, one constant signal, and two drifting signals. This tests the agent's ability to adapt to large changes in the environment. A well performing agent should not become so rigid in what it has learned that it cannot adapt when there are changes to its environment. Additionally with two drifting signals a well performing agent will not prefer one drifting signal over the other, instead checking both with the same frequency to keep the predictions of both up to date.

Finally the Jumpy Eight-Action problem is designed to stress test a learning agent that requires signals to be sampled at different frequencies in order to perform well. The environment has two constant signals, three signals that drift at different rates, and three noisy signals. This simulates a scenario where the signals may change in a large, but somewhat regular, pattern. In order to perform well an agent will have to sample different signals at different frequencies—where the frequency is based on both the amount of drift and the amount of noise in the signal. An effective agent will need to learn ignore the noisy signals and focus on regular sampling of the drifting signals based on the amount they drift. A behavior that samples uniformly will perform quite poorly in this environment since the time away from focusing on the quickly drifting arms will cause the prediction error to increase rapidly.

1.2 Introspective agents

Introspective agents are ones which can adapt their rate of learning based on their learning progress. Some of examples of these are line search, the commonly used Adam optimizer (Kingma and Ba, 2014) used in deep neural networks, or AdaGain (Jacobsen et al., 2019) which adapts the rate of learning for stability. Introspective agents are able to use the rate of learning as a proxy for learning progress as an adaptive method will drive down the rate of learning on predictions that are not improving, and drive up the rate of learning where the agent's predictions are improving.

In our case we used a variant of the Autostep algorithm (Mahmood, Sutton, Degris, and Pilarski, 2012), Auto, to adapt the agent's step size parameter.

Autostep is a meta-descent method which seeks to adapt the step-size based on the partial derivative of the error. The basics of how it work are to keep a trace of the prediction errors and to lower the step-size if the sign of the error changes, indicating a lack of learning progress. If the sign stays the same learning is progressing and the step-size is increased. As an example, the noisy signal will cause the agent's prediction to have positive error at some points, and a negative error at others—the sign will flip back and forth. This will cause Autostep to drive the step-size of that learner down. Auto builds on top of IDBD (Sutton, 1992) and Autostep to extend it to the multi-state setting and provides some heuristics to allow for robust performance.

To see how being introspective could be helpful let us return to our three signals from the testbed. In the case of a constant signal the agent should be able to get to accurate predictions quickly. In this case the step-size would be driven up since the agent's predictions are good and the prediction error would remain low or zero. For the noisy signal at each time step the agent prediction error would be high and little to no learning progress would be made. Here the lack of learning progress will drive down the step-size. Finally in the case of the drifting arm the agent is able to continually make progress on learning the signal, keeping the step-size of the predictor high.

To see why this can help, consider a case where an agent was non-introspective and did not adapt its learning rate. If the learning rate was set high—say close to 0.5—the agent would be able to quickly learn the constant signal. The agent likely would be able to track the drifting signal as it would update its prediction to close to the expected value at each time step. However the noisy signal will pose a large problem as the prediction will constantly chase the noise of the signal, leading to a large prediction error. If the learning rate was set low—say 0.0001—the constant signal could be learned, albeit slowly. Learning the expected value of the noisy signal could be possible as it would average over many timesteps. However the prediction of the drifting target would not update quickly enough to keep its prediction up to date. Without adapting the step-size for each signal there is not an ideal step-size in between the large and small step-sizes. Either a step-size would need to be chosen ahead of time

based on prior knowledge of the signal's characteristics, or we need to use an introspective learner that adapts its learning rate.

We propose a simple intrinsic reward, weight change, that uses the change in the weights of a prediction. When combined with introspective learners, weight change is shown to be an effective intrinsic reward for driving learning progress. Weight change as an intrinsic reward can be thought of in two parts: first, the error signal seeking to focus the agent on the signal that the agent could potentially learn the most about. Second, since the adaptive learning rate adjusts up or down based on the learner's ability to make progress on predicting the signal, this tempers the amount weights are updated based on whether progress can be made. Combined, the change in the weights at each step are shown to be a good intrinsic reward. Reward increases when both error and learnability are high, and decreases when either error or predictability is low.

1.3 Summary

This work has three main contributions: proposing a new testbed for evaluating intrinsic reward mechanisms, an empirical evaluation of 14 intrinsic rewards representing common intrinsic reward categories, and proposing an intrinsic reward, weight change, that is shown to be simple and effective when combined with introspective learners.

The testbed itself represents a distilled version of the full problem of intrinsic reward. To look at what a full setting could be consider the Horde setting (Sutton et al., 2011) where an agent is learning thousands of predictions online as it operates in the world. Most of these predictions—called demons—are learned off-policy. The first step will be moving beyond the single state, bandit-like setting, to a multiple state setting. The behavior learner would use a temporal difference learning method, either a value based method such as SARSA (Sutton and Barto, 2018) or a policy gradient method such as Actor-Critic (Sutton, McAllester, Singh, and Mansour, 2000). The agent would have to learn what states and actions over time will lead to the most

learning progress across all of the demons. The learners are not independent of each other, as they are in this testbed, so different actions could lead to different amounts of learning for multiple predictors at the same time. In the full setting an agent will need to learn to take actions that can balance the learning of many predictions simultaneously.

The learners themselves would also need to move on from the single state, least means squares (LMS), learners used here. The predictors could leverage an online, off-policy, learning method such as gradient TD. The introspective learning mechanism will also need to be able to deal with the multi-state setting. Auto, used here, could be used in the multi-state setting. Other methods such as TIDBD (Kearney, Veeriah, Travnik, Sutton, and Pilarski, 2018), AdaGain (Jacobsen et al., 2019), and Adam (Kingma and Ba, 2014), have all been shown to be effective in different settings and are all candidates for being used. Research on the comparative effectiveness of different step-size adaptive methods remains an outstanding research question.

The world is large and complex, so the prediction learners will not be able to store every state that the agent visits. The learners will need some sort of function approximation mechanism to be able to generalize from one state to another. Tile coding has been shown to be an effective mechanism in this setting (Sutton et al., 2011) where the agents are able to learn online. More complex methods, such as artificial neural networks (ANNs), have been shown to be effective function approximators for dealing with large state spaces (e.g. Mnih et al., 2015). The combination of ANNs and reinforcement learning often requires a number of moving parts, such as experience replay and target networks (Mnih et al., 2015), in order to be effective. How that will interact with a system such as this is the subject of further research.

In this setting the challenge of exploration and exploitation returns. The agent will have to trade off exploiting the areas where it knows learning progress can be made, while exploring to find areas where it learning could be higher. At its most basic the system could use a simple epsilon-greedy method to mix exploration in. More complex methods such as pseudocounts (Bellemare et al., 2016) have been shown to perform better than epsilon greedy and

could be used here to help the agent explore more to find areas of maximum learning.

1.4 Thesis Structure

The structure of this thesis is not typical for a master's thesis. The setup is a staple thesis, also known as a sandwich thesis, in which the core of the work is an accepted journal paper. In addition to the paper, the author wrote a new introduction and conclusion. They serve to demonstrate the author's knowledge of the subject matter and journal paper in question, the author's individual perspective on the topic area, and the future directions for research that the author would like to pursue, and in some instances, is already pursuing.

Adapting Behavior via Intrinsic Reward: A Survey and Empirical Study, has been accepted into the Journal of Artificial Intelligence Research (JAIR). The paper is co-authored by Cam Linke, Nadia M. Ady, Martha White, Thomas Degris and Adam White. The paper was a collaborative work that began as a proposed conference paper, was accepted as an extended abstract into the Reinforcement Learning and Decision Making (RLDM) conference, and finally was accepted into JAIR. The author's specific contributions were the open-sourced code and notebook for replicating the empirical work; writing sections of the intro, experiments and conclusion; and contributing to the large survey of previous literature performed in the paper.

Journal of Artificial Intelligence Research 69 (2020) 1287-1332 Submitted 05/2020; published 12/2020

Adapting Behavior via Intrinsic Reward: A Survey and Empirical Study

Cam Linke CLINKE@UALBERTA.CA
Nadia M. Ady NMADY@UALBERTA.CA
Martha White WHITEM@UALBERTA.CA
Alberta Machine Intelligence Institute (Amii)
Department of Computing Science
University of Alberta

Thomas Degris DEGRIS@GOOGLE.COM
DeepMind London

Adam White ADAMWHITE@GOOGLE.COM
Alberta Machine Intelligence Institute (Amii)
Department of Computing Science
University of Alberta
DeepMind Alberta

Abstract

Learning about many things can provide numerous benefits to a reinforcement learning system. For example, learning many auxiliary value functions, in addition to optimizing the environmental reward, appears to improve both exploration and representation learning. The question we tackle in this paper is how to sculpt the stream of experience—how to adapt the learning system's behavior—to optimize the learning of a collection of value functions. A simple answer is to compute an intrinsic reward based on the statistics of each auxiliary learner, and use reinforcement learning to maximize that intrinsic reward. Unfortunately, implementing this simple idea has proven difficult, and thus has been the focus of decades of study. It remains unclear which of the many possible measures of learning would work well in a parallel learning setting where environmental reward is extremely sparse or absent. In this paper, we investigate and compare different intrinsic reward mechanisms in a new bandit-like parallel-learning testbed. We discuss the interaction between reward and prediction learners and highlight the importance of introspective prediction learners: those that increase their rate of learning when progress is possible, and decrease when it is not. We provide a comprehensive empirical comparison of 14 different rewards, including well-known ideas from reinforcement learning and active learning. Our results highlight a simple but seemingly powerful principle: intrinsic rewards based on the amount of learning can generate useful behavior, if each individual learner is introspective.

1. Balancing the Needs of Many Learners

Learning about many things can provide numerous benefits to a reinforcement learning system. Adding many auxiliary losses to a deep learning system can act as a regularizer on the representation, ultimately resulting in better final performance in reward maximization problems, as demonstrated with Unreal (Jaderberg et al., 2016). A collection of value

functions encoding goal-directed behaviors can be combined to generate new policies that generalize to goals unseen during training (Schaul et al., 2015). Learning in hierarchical robot-control problems can be improved with persistent exploration, given call-return execution of a collection of subgoal policies or skills (Riedmiller et al., 2018), even if those policies are imperfectly learned. In all these examples, a collection of general value functions (see Sutton et al., 2011) is updated from a single stream of experience. The question we tackle in this paper is how to sculpt that stream of experience—how to adapt the learning system's behavior—to optimize the learning of a collection of value functions.

One answer is to simply maximize the environmental (extrinsic) reward. This was the approach explored in Unreal and it resulted in significant performance improvements in challenging visual navigation problems. However, it is not hard to imagine situations where this approach would be limited. In general, the reward may be delayed and sparse: what should the agent do in the absence of external (environmental) motivations? Learning reusable knowledge such as skills (Sutton et al., 1999) or a model of the world might result in more long-term reward. Such auxiliary learning objectives could emerge automatically during learning (Silver et al., 2017). Most agent architectures, however, include explicit skill and model learning components. It seems natural that progress towards these auxiliary learning objectives could positively influence the agent's behavior, resulting in improved learning overall.

Learning many value functions off-policy from a shared stream of experience—with function approximation and an unknown environment—provides a natural setting to investigate no-reward intrinsically motivated learning. The basic idea is simple. The aim is to accurately estimate many value functions, each with an independent learner—where there is no external reward signal. Directly optimizing the data collection for all learners jointly is difficult because we cannot directly measure this total learning objective and because actions have an indirect impact on learning efficiency. There is a large related literature in active learning (Cohn et al., 1996; Balcan et al., 2009; Settles, 2009; Golovin and Krause, 2011; Konyushkova et al., 2017) and active perception (Bajcsy et al., 2018), from which to draw inspiration for a solution but which do not directly apply to this problem. In active learning the agent must sub-select from a larger set of items to choose which points to label. Active perception is a subfield of vision and robotics. Much of the work in active perception has focused on specific settings—namely visual attention (Bylinskii et al., 2015), localization in robotics (Patten et al., 2018) and sensor selection (Satsangi et al., 2018, 2020)—or assumes knowledge of the dynamics of the world (see Bajcsy et al., 2018).

An alternative strategy is to formulate our task as a reinforcement learning problem. We can use an intrinsic reward, internal to the learning system, that approximates the total learning across all learners. The behavior can be adapted to choose actions in each state that maximize the intrinsic reward, towards the goal of maximizing the total learning of the system. The choice of intrinsic rewards can have a significant impact on the sample efficiency of such intrinsically motivated learning systems. This paper provides the first formulation of parallel value function learning as a reinforcement learning task. Fortunately, there are many ideas from related areas that can inform our choice of intrinsic rewards.

Rewards computed from internal statistics about the learning process have been explored in many contexts over the years. Intrinsic rewards have been shown to induce behavior that resembles the development stages exhibited by young humans and animals (Barto, 2013;

Chentanez et al., 2005; Oudeyer et al., 2007; Lopes et al., 2012; Haber et al., 2018). Internal measures of learning have been used to improve skill or option learning (Chentanez et al., 2005; Schembri et al., 2007; Barto and Simsek, 2005; Santucci et al., 2013; Vigorito, 2016), and model learning (Schmidhuber, 1991b, 2008). Most recent work has investigated using intrinsic reward as a bonus to encourage additional exploration in single task learning (Itti and Baldi, 2006; Stadie et al., 2015; Bellemare et al., 2016; Pathak et al., 2017; Hester and Stone, 2017; Tang et al., 2017; Andrychowicz et al., 2017; Achiam and Sastry, 2017; Martin et al., 2017; Colas et al., 2018; Schossau et al., 2016; Pathak et al., 2019). Few have investigated the impact of making these internal measures the main objective of learning (Berseth et al., 2019), however previous studies have noted that intrinsic reward is useful even in single-task problems with a well-defined external goal (Bellemare et al., 2016).

It remains unclear, however, which of these measures of learning would work best in our no-reward setting. Most prior work has focused on providing demonstrations of the utility of particular intrinsic reward mechanisms. One study focused on a suite of large-scale control domains with a single scalar external reward (Burda et al., 2018), comparing different learning systems that use an intrinsic reward based on model-error as an exploration bonus. A large study has been conducted on learning progress measures for curriculum learning for neural networks (Graves et al., 2017), where the goal is to learn from which task to sample a dataset to update the parameters. Variants of their measures are related to the intrinsic rewards explored in this paper, but their setting differs substantially in that learning is offline from batch supervised learning datasets and the underlying problems are stationary. To the best of our knowledge, there has never been a broad empirical comparison of intrinsic rewards for the online multi-prediction setting with non-stationary targets.

A computational study of intrinsic rewards is certainly needed, but tackling this problem with function approximation and off-policy updating is not the place to start. Estimating multiple value functions in parallel requires off-policy algorithms because each value function is conditioned on a policy that is different than the exploratory behavior used to select actions. In problems of moderate complexity, these off-policy updates can introduce significant technical challenges. Popular off-policy algorithms like Q-learning and V-trace can diverge with function approximation (Sutton and Barto, 2018). Sound off-policy algorithms exist, but require tuning additional parameters and are relatively understudied in practice. Even in tabular problems, good performance requires tuning the parameters of each component of the learning system—a complication that escalates with the number of value functions. Finally, the agent must solve the primary exploration problem in order to make use of intrinsic rewards. Finding states with high intrinsic reward may not be easy, even if we assume the intrinsic reward is reliable and informative. To avoid these many confounding factors, the right place to start is in a simpler setting.

In this paper, we investigate and compare different intrinsic reward mechanisms in a new bandit-like parallel learning testbed. The testbed consists of a single state and multiple actions. Each action is associated with an independent scalar target to be estimated by an independent prediction learner. An ideal behavior policy will focus on actions that generate the most learning across the prediction learners. However, the overall task is partially observable, and learning is never done. The targets change without an explicit notification to the agent, and the task continually changes due to changes in action selection and learning of the individual prediction learners. Different configurations of the target

distributions can simulate unlearnable targets, non-stationary targets, and easy-to-predict targets. Our new testbed provides a simple instantiation of a problem where *introspective* learners should help achieve low overall error. An introspective prediction learner is one that can autonomously increase its rate of learning when progress is possible, and decrease learning when progress is not—or cannot—be made.

This paper summarizes a comprehensive empirical comparison of different intrinsic reward mechanisms, including several ideas from reinforcement learning and active learning. Our computational study of learning progress highlighted a simple principle: intrinsic rewards based on the *amount of learning* (e.g., Bayesian Surprise and simple change in weights) can generate useful behavior if each individual learner is introspective. Across a variety of problem settings we find that the combination of introspective learners and simple intrinsic rewards was most reliable, performant, and easy to tune. We conclude with a discussion about how these ideas could be extended beyond our one-state prediction problem to drive behavior in large-scale problems where off-policy learning and function approximation are required.

2. Problem Formulation

In this section we formalize a testbed for comparing intrinsic reward using a stateless prediction task and independent learners. This formalism is meant to simplify the study of balancing the needs of many learners to facilitate comprehensive comparisons.

We formalize our multiple-prediction learning setting as a collection of independent, online supervised learning tasks. On each discrete time step $t = 1, 2, 3, ...$, the *behavior agent* selects an action $A_t \in \{1, \ldots, N\}$ corresponding to task $i \in \{1, \ldots, N\}$[1], causing a target signal to be sampled from an (unknown) target distribution, $c_{t,i} \sim \theta(t, i)$, where $C_{t,i}$ denotes the random variable with distribution $\theta(t, i)$. This distribution $\theta(t, i)$ is indexed by time to reflect that it can change on each time step; this enables a wide range of different target distribution to be considered, to model this non-stationary, multi-prediction learning setting. We provide the definition we use in this work later in this section, in Equation (3).

Associated with each prediction task is a simple *prediction learner* that maintains a real-valued vector of weights $w_{t,i}$, to produce an estimate, $\hat{c}_{t,i} \in \mathbb{R}$, of the expected value of the target, $\hat{c}_{t,i} \approx \mathbb{E}[C_{t,i}]$. On a step where task i is selected, $w_{t,i}$ could be updated using any standard learning algorithm. In this work, we use a 1-dimensional weight vector, and so the update is a simple delta-rule (least-mean-squares (LMS) learners):

$$w_{t+1,i} \leftarrow w_{t,i} + \alpha_{t,i}\delta_{t,i} \tag{1}$$

where $\alpha_{t,i}$ is a scalar step-size parameter and $\delta_{t,i} \stackrel{\text{def}}{=} c_{t,i} - w_{t,i}$ is the prediction error of prediction learner i on step t. On a step where task i is not selected, $w_{t,i}$ is not updated, implicitly setting $w_{t+1,i}$ to $w_{t,i}$.

1. To clearly separate the action selected by the agent and the prediction task, we use A_t to denote the action selected at time t and i to denote prediction task. A_t is uppercase to indicate it is a random variable, with lowercase a corresponding to an actual action selected. In our setting, taking action a corresponds to observing data for task i, and so there is an equivalence between the actions and tasks. More generally, such as in the full reinforcement learning setting, this is not the case; for extensions on this work, it is useful to clearly delineate between actions and prediction tasks.

The primary goal is to minimize the Mean Squared Error up to time t for all of the N learners:

$$\texttt{MSE}(t) \stackrel{\text{def}}{=} \frac{1}{t}\sum_{k=1}^{t}\frac{1}{N}\sum_{i=1}^{N}(\hat{c}_{k,i} - \mathbb{E}[C_{k,i}])^2. \tag{2}$$

The behavior agent does not get to observe this error, both because it only observes one of the targets $c_{k,i}$ on each step, rather than all N, and because that target is a noisy sample of the true expected value $\mathbb{E}[C_{k,i}]$. The agent can nonetheless attempt to minimize this unobserved error.

In order to minimize Equation (2), we must devise a way to choose which prediction task to sample. This can be naturally formulated as a sequential decision-making problem, where on each time step t, the behavior agent chooses an task i (corresponding to action A_t), resulting in a new sample of $c_{t,i}$, and an update to $w_{t,i}$. In order to learn a preference over actions we associate an intrinsic reward $R_t \in \mathbb{R}$ with each action selection, and thus with each prediction task. We investigate different intrinsic rewards. Given a definition of the intrinsic reward, we can use a bandit algorithm suitable for non-stationary problems; we discuss two options below in Section 2.1.

The targets for each prediction learner are intended to replicate the dynamics of targets that a parallel auxiliary task learning system might experience, such as sensor values of a robot. To simulate a range of interesting dynamics, we construct each $\theta(t,i)$ as a Gaussian distribution with drifting mean:

$$\begin{aligned} &\theta(t,i) \stackrel{\text{def}}{=} \mathcal{N}(\mu_{t,i}, \sigma^2_{t,i}) \\ &\text{for} \quad \mu_{t+1,i} \leftarrow \Pi_{[-50,50]}\left(\mu_{t,i} + \mathcal{N}(0, \xi^2_{t,i})\right) \end{aligned} \tag{3}$$

where $\mu_{t,i} \in \mathbb{R}$, $\sigma^2_{t,i} \in \mathbb{R}^+$ controls the sampling noise, $\xi^2_{t,i} \in \mathbb{R}^+$ controls the rate of drift and $\Pi_{[-50,50]}$ projects the drifting $\mu_{t,i}$ back to the range $[-50, 50]$ to keep it bounded. The variance and drift are indexed by t because we explore settings where they change periodically. These changes are not communicated to the behavior agent, and the individual LMS learners are prevented from storing explicit histories of the targets. The purpose of this choice was to simulate partial observability common in many large-scale systems (e.g., Sutton et al., 2011; Modayil et al., 2014; Jaderberg et al., 2016; Silver et al., 2017). Given our setup, both prediction learners and the behavior learner would do well to treat their respective learning tasks as non-stationary and track rather than converge (Sutton et al., 2007), as long as $\xi^2_{t,i}$ is greater than zero. Each sample $c_{k,i} \sim \theta(t,i)$, and $\mu_{t,i}$ is bounded between $[-50, 50]$, and $\mu_{t,i}$ is updated on each step t regardless of which action is selected. Our formalism is summarized in Figure 1.

2.1 Non-stationary Bandit Algorithms for Prediction Learning

We do not focus on the bandit formalism itself nor bandit algorithms in this work. Rather, our goal is to investigate intrinsic rewards and their utility for learning multiple predictions, in the simplest setting in which we can obtain meaningful insights: a bandit-like setting. Our choice of bandit algorithm, therefore, is simply to facilitate this investigation, rather than investigate the properties of the bandit algorithms themselves. We use two different bandit

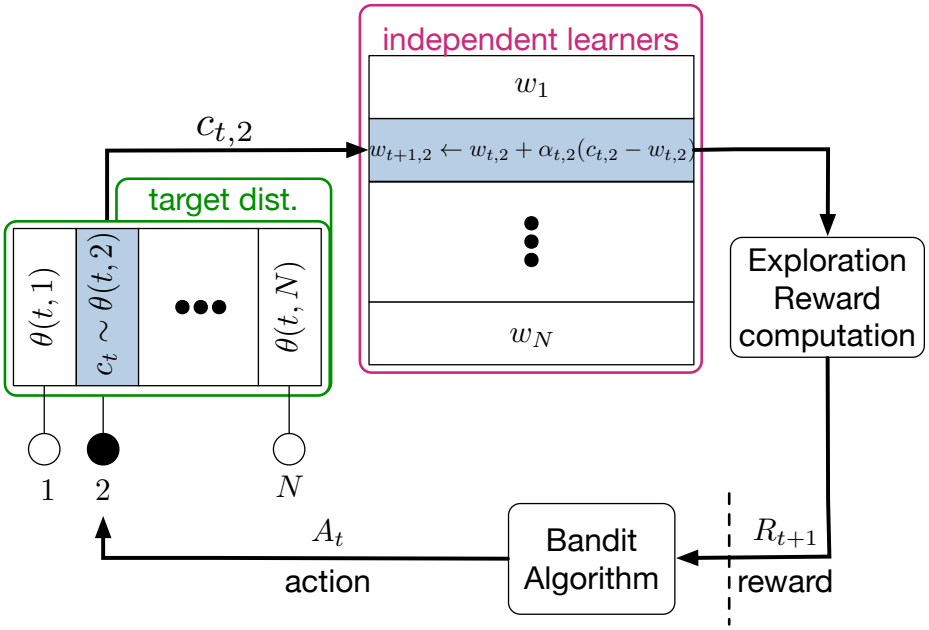

Figure 1: Our parallel multi-prediction learning formulation.

algorithms—a Gradient Bandit and an extension of Dynamic Thompson Sampling (DTS)—so as to ensure our conclusions are not due primarily to the choice of bandit algorithm. We describe these two algorithms below, as well as the reasons for choosing them.

We cannot simply pick any bandit algorithm, as our prediction learning setting differs from the usual multi-armed bandit setting in at least two ways. First, the distributions of the targets are non-stationary. Non-stationary learning problems have been studied under dynamic bandits, also called restless bandits. The general problem setting is known to be hard, but under some restrictions, some progress can be made. Some algorithms assume piecewise stationarity, such as Discounted UCB or Sliding-Window UCB (Garivier and Moulines, 2011), or those with a variation budget, which is used to decide how to restart stationary bandit algorithms (Besbes et al., 2014). More suitable for our setting is work assuming restrictions on drift, such as Brownian motion, including State-Oblivious UCB (Slivkins and Upfal, 2008) and Dynamic Thompson Sampling (DTS) (Gupta et al., 2011), or State-Oblivious UCB (Slivkins and Upfal, 2008) and the Gradient Bandit (Sutton and Barto, 2018).

Second, our objective is to minimize error across all learners, but we only see an intrinsic reward corresponding to the target we selected for that step. The need to minimize error across all learners—our second issue—is related to partial monitoring (see Lattimore and Szepesvári, 2019 for an overview). In partial monitoring, the learning system only receives limited feedback about the true loss incurred. For our prediction setting, the true loss is the MSE over all the predictions. The feedback is only about the prediction for the action selected, and, depending on the intrinsic reward, it is a noisy and indirect measure of the MSE for that prediction. Partial monitoring encompasses a wide range of problems, but it is difficult to develop algorithms and regret bounds for the fully general setting. To the best

of our knowledge, current algorithms rely on finite outcomes and involve estimating distributions over outcomes. Given the difficulty of even that restricted setting, the additional complication of non-stationarity does not as yet seem to have been tackled.

Fortunately, for our prediction setting, the structure of our problem (described in Section 2) admits a simple approach that performs well in practice: to err on the side of taking an action periodically. Selecting any action is not detrimental, as it provides information about one of the targets. Particularly in a non-stationary setting, each action should be taken periodically, to check if expected reward estimates remain accurate. One reasonable strategy is to obtain a distribution over the actions—not find the single best action—and sample proportionally to that distribution, as is done by the Gradient Bandit. For DTS, we prevent the variance for the Bayesian estimate for each action from dropping below a minimum level, both to account for non-stationarity and to increase the probability that an action will be selected. We find these simple choices to be sufficient for reasonable behavior in our multi-prediction problem setting.

We now describe these two bandit algorithms. The Gradient Bandit, specified in Sutton and Barto (2018, Section 2.8), attempts to maximize the expected average reward by modifying a vector of action preferences $h \in \mathbb{R}^N$—indexed by action—based on the difference between the reward and average reward baseline:

$$h_{t+1}(a) \leftarrow \begin{cases} h_t(a) + \alpha(R_{t+1} - \bar{r})(1 - \pi_t(a)) & \text{if } A_t = a; \\ h_t(a) - \alpha(R_{t+1} - \bar{r})\pi_t(a) & \text{otherwise.} \end{cases}$$

where $\bar{r} \in \mathbb{R}$ is the average of all the rewards up to time t, maintained using an unbiased exponential average (introduced by Sutton and Barto (2018) [Eq. 2.9]), and $\bar{r}$ and $h_0(a)$ are both initialized to zero. Actions are selected probabilistically according to a softmax distribution which converts the preferences to probabilities:

$$Pr\{A_t = a\} = \pi_t(a) \stackrel{\text{def}}{=} \frac{e^{h_t(a)}}{\sum_{b=1}^{N} e^{h_t(b)}}$$

The Gradient Bandit will sample all the actions infinitely often, though if an action preference is very low then that action will be rarely taken. Notice that the Gradient Bandit algorithm is similar to policy gradient methods in reinforcement learning.

The second non-stationary bandit algorithm we use is Dynamic Thompson Sampling (DTS) (Gupta et al., 2011). The algorithm maintains a posterior distribution over the expectation and variance of the reward for each action, using a Bayesian update. The posterior variance is increased after each update, to account for non-stationarity in the rewards. This ensures that, before the posterior is treated like a prior for the next update, it reflects the uncertainty in that prior information, due to the fact that the environment is non-stationary. Otherwise, the posterior would concentrate over time. The distribution over expected rewards is then used in the standard way in Thompson sampling: an estimate is sampled for each action, and the action with maximal sampled value is executed.

The algorithm we use is an extension of DTS, which was only specified for Bernoulli rewards. We extended the approach to Gaussian rewards. The behavior agent assumes the rewards for each action (indexed by a) come from a Gaussian distribution, $\mathcal{N}(\mu_a, \sigma_a^2)$ with unknown mean μ_a and unknown variance σ_a^2. The behavior agent maintains Bayesian

estimates, meaning it maintains a normal-inverse-gamma (NIG) distribution over (μ_a, σ_a^2), which is the conjugate prior for a Gaussian with unknown mean and unknown variance. For each action, we maintain three parameters for a NIG: (m_a, v_a, n_a), where m_a is an estimate of the mean, n_a maintains a count and v_a is an estimate of n_a times the variance.[2] To sample a mean μ_a for each action, to use for action selection, you first sample σ_a^2 from an inverse-gamma with parameters $(n_a/2, v_a)$ and then sample μ_a from $\mathcal{N}(m_a, \sigma_a^2/n_a)$. After picking action $A_t = \tilde{a}$ with the largest μ_a, the behavior agent receives a reward r for taking that action and updates the estimate for $\tilde{a}$. These parameters are updated with the standard Bayesian update:

$$
\begin{aligned}
v_{\tilde{a}} &\leftarrow v_{\tilde{a}} + \frac{n_{\tilde{a}}}{n_{\tilde{a}}+1}\frac{(r-m_{\tilde{a}})^2}{2} \\
m_{\tilde{a}} &\leftarrow \frac{n_{\tilde{a}} m_a + r}{n_{\tilde{a}}+1} \\
n_{\tilde{a}} &\leftarrow n_{\tilde{a}} + 1
\end{aligned}
$$

in the order specified above. If the problem was stationary, then this is the complete update.

But, the problem is non-stationary, due to the fact that the rewards can change over time. Notice that the posterior variance for μ_a, which is $\frac{v_a}{n_a^2/2 - n_a}$ for this NIG, would gradually shrink to zero as the count n_a increases. To account for non-stationarity, the simple idea behind DTS is to increase this posterior variance after the update in such a way as to minimally impact the mean. For an NIG, this means that we would modify the count and variance parameter for the selected action using

$$
\begin{aligned}
v_{\tilde{a}} &\leftarrow \max((1-\alpha)v_{\tilde{a}}, 10^{-2}) \\
n_{\tilde{a}} &\leftarrow (1-\alpha)n_{\tilde{a}}
\end{aligned}
$$

for decay $\alpha \in (0,1)$, that behaves like a step-size parameter. The count is decayed by $1-\alpha$, providing an exponential decay on older samples and providing an upper bound on n_a of $\sum_{i=0}^{\infty}(1-\alpha)^i = 1/\alpha$. The $v_{\tilde{a}}$ is similarly decayed. The max with 10^{-2} is to ensure v_a never goes to zero, and so that the variance remains at a minimal level. The new variance of μ_a, according to the NIG after decaying v_a^{old} and n_a^{old}, is a strict increase

$$
\begin{aligned}
\frac{v_a}{n_a^2/2 - n_a} = \frac{(1-\alpha)v_a^{\text{old}}}{(1-\alpha)((1-\alpha)(n_a^{\text{old}})^2/2 - n_a^{\text{old}})} &= \frac{v_a^{\text{old}}}{(1-\alpha)(n_a^{\text{old}})^2/2 - n_a^{\text{old}}} \\
&> \frac{v_a^{\text{old}}}{(n_a^{\text{old}})^2/2 - n_a^{\text{old}}}
\end{aligned}
$$

where we assume $n_a^{\text{old}} > 2$. The mean value m_a, though, remains unchanged when we increase the posterior variance. During the standard update to m_a above, however, notice that is resembles an exponential moving average because older values are multiplied by n_a, using $n_a m_a$. The algorithm requires an initial mean estimate $m_a = m_0$—a good choice being a large positive value for m_0 to encourage exploration—with the initial estimate $v_a = m_0^2$.

2. A NIG typically has four parameters. For us, the parameter typically called α, which is used to normalize v_a, exactly equals $n_a/2$, so we do not maintain it explicitly. It is only used to sample the action, using inverse-gamma parameters $(\alpha, v_a) = (n_a/2, v_a)$.

3. Simulating Parallel Prediction Problems

We consider several prediction problems corresponding to different settings of $\xi^2_{t,i}$ and $\sigma^2_{t,i}$ to define task distribution $\theta(t, i)$ in Equation (3). We introduce three problems, with target data simulated from those problems show in Figure 2.

The *Drifter-Distractor* problem has four targets, one for each action: (1) two (stationary) high-variance targets as *distractors* (2) a slowly *drifting* target and (3) a *constant* target, with $\xi^2_{t,i}$ and $\sigma^2_{t,i}$ for each of these types in Table 2. A distractor target is simply a noisy, stationary target: the variance is high enough such that an agent might oversample the target even after the mean estimate is accurate. This is inspired by the noisy TV problem (Schmidhuber, 2008).

The *Switched Drifter-Distractor* problem is similar to Drifter-Distractor except, after 50,000 time-steps the associations between the actions and the target distributions are permuted as detailed in Table 2. To do well in this problem, the learning system must be able to respond to changes. In addition, in phase two of this problem, two targets exhibit the same drift characteristics; the behavior agent should prefer both actions equally.

The *Jumpy Eight-Action* problem is designed to require sampling different prediction tasks with different frequencies. In this problem, there are two drifters and three distractors, with different drifting rates and different amounts of sampling variance, as summarized in Table 3. The best approach is to select several actions probabilistically depending on their drift and sampling variance. We add an additional target type, that drifts more dramatically over time, with periodic shifts in the mean:

$$\begin{aligned} \mu_{t+1,6} &\leftarrow \Pi_{[-50,50]}\left(\tilde{\mu}_{t+1,6}\right) \\ &\text{where } \tilde{\mu}_{t+1,6} \leftarrow \mu_{t,6} + \zeta_t \text{Bernoulli}(0.005)\mathcal{N}(10, 1.0) \end{aligned} \tag{4}$$

and where indicator $\zeta_0 = 1$ and $\zeta_t \in \{-1, 1\}$ switches sign if $|\tilde{\mu}_{t+1,6}| > 50$. The sample from a Bernoulli ensures the jumps are rare, but the large mean of the Gaussian makes it likely for this jump to be large when it occurs, as shown in Figure 2. This problem simulates a prediction problem where the target changes by a large magnitude in a semi-regular pattern, but then remains constant. This could occur due to changes in the world outside the prediction learner's control and representational abilities.

target type	μ_0	σ^2	ξ^2
constant	uniform(-50,50)	0	0
distractor	0	1	0
drifter	0	0	0.1

Table 1: These parameters define each target distribution used in the *Drifter-Distractor* and the *Switched Drifter-Distractor* problems. The parameter μ_0 specifies the initial mean of each target, σ^2 is the sampling variance, and ξ^2 is the drift variance.

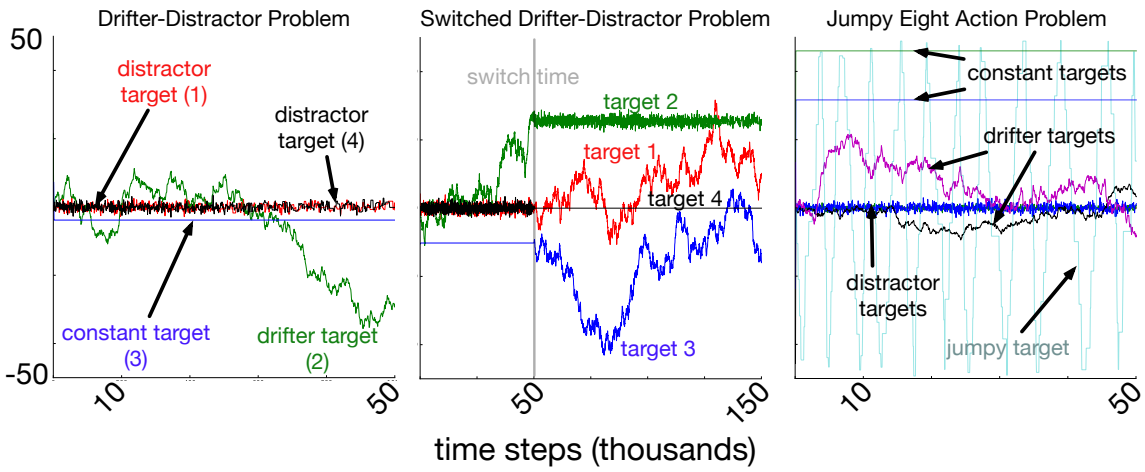

Figure 2: Each subplot shows the target data generated by one run of the problem, with Drifter-Distractor (left), Switched Drifter-Distractor (middle), and Jumpy Eight-Action Problem (right).

target	*phase 1*	→	*phase 2*
target 1	distractor	→	drifter
target 2	drifter	→	distractor
target 3	constant	→	drifter
target 4	distractor	→	constant

Table 2: The target distributions in the *Switched Drifter-Distractor* change part way through the task. Phase one lasts for 50,000 time steps, then targets are permuted and remain fixed for the remainder of the experiment (another 100,000 steps). The initial parameters for each target type—constant, distractor and drifter—are the same as in the Drifter-Distractor Problem described in Table 1

Task	1	2	3	4	5	7 & 8
σ^2	0.1	0.5	1.0	0.01	0.01	0.0
ξ^2	0.0	0.0	0.0	0.01	0.05	0.0

Table 3: Parameters defining $\theta(t, i)$ for each prediction task in the *Jumpy Eight-Action* problem, where σ^2 is the sampling variance and ξ^2 is the drift variance for Equation (3). The first three prediction tasks have distractor targets; four and five have drifter targets; and seven and eight have constant targets. The distractors range from low variance to high variance, and the drifters with low drift to higher drift. **Prediction Task 6** is special, defined in Equation (4).

4. Introspective Prediction Learners

The behavior of a learning system that maximizes intrinsic rewards relies on the underlying prediction learning algorithms as well as the definition of the intrinsic reward. In this section we introduce a distinction between two categories of learners, for which behavior can be substantially different: introspective and non-introspective learners. We consider a learner to be **introspective** if the algorithm can modulate its own learning without help from an external process. More concretely, an introspective learner stops updating if it cannot make progress. For example, in the case of prediction learning, an introspective learner would regulate its updates to mitigate noise in its prediction targets. A **non-introspective learner**, on the other hand, will continually update regardless of learning progress.

In this paper we consider two basic settings representing non-introspective and introspective learners, used as prediction learners in our multi-prediction problem. We use basic LMS learners with a constant step-size parameter as our non-introspective learner. With a *constant step-size* parameter, the LMS algorithm will always try to adapt its estimates toward the sample targets on each time step. It does not matter if the target exhibits high variance—say centered mean zero—or if the target is actually constant; the LMS algorithm will continue to adapt its estimates attempting to track each target in the online setting. Consider how a constant global step-size parameter would work on our Drifter-Distractor Problem discussed above. If the step-size parameter value is too large for the distractor target, then the prediction learner will continually make large updates due to the sampling variance, never converging to low error. If the step-size parameter is too small for the tracking target, then the prediction learner's estimate will often lag, causing high-error. A constant global step-size parameter cannot balance the need to track the drifter targets, and the need to learn slowly on the distractor targets.

To create a simple introspective learner for our setting, we simply combine our LMS predictors with a step-size adaption method called Autostep. Autostep is a simple meta-learning algorithm that adapts the step-size parameter of each LMS learner over-time (Mahmood et al., 2012). The basic idea behind Autostep is to increase the step-size parameter when learning is progressing, and lower the step-size parameter value when learning is not progressing. It does so by keeping a trace, $h \in \mathbb{R}$, of the previous prediction errors. Roughly speaking, if the error changes sign often then the predictions are not improving and the step-size parameter value should be lowered. If the error is mostly of the same sign, then the step-size parameter value should not be reduced. Autostep has one key hyper-parameter, the meta learning-rate: this controls how quickly the algorithm changes the step-size parameter (α). The full pseudocode, specialized to our stateless tracking tasks, is given below.[3] Note that Autostep changes the step-size parameter with a multiplicative exponential, which allows geometric or rapid changes to the LMS learners step-size parameter.

To give some intuition about how Autostep changes the step-size parameter, consider what happens when we apply it to the Drifter-Distractor Problem in Figure 3. Here we simply plot α over time for four LMS learners—one for each target—with each step-size

3. Our implementation of Autostep clips the step-size in step 3, given by Degris and White (2020), and so differs slightly from the form given by Mahmood et al. (2012).

Algorithm 1 : *The Autostep algorithm specialized to stateless prediction*
κ is the meta learning-rate parameter
n and h are scalar memory variables initialized to 1 and 0
δ is the prediction error and α (initialized to 1.0) the step-size parameter of predictor i

1: **procedure** Autostep(δ)
2: $\quad n \leftarrow \max(|\delta h|, n + \frac{1}{10000}\alpha(|\delta h| - n))$
3: $\quad \alpha \leftarrow \min(\alpha \exp(\kappa \frac{\delta h}{n}),\ 0.5)$
4: $\quad h \leftarrow h(1-\alpha) + \alpha\delta$

parameter adapted by Autostep. We used the Gradient Bandit and Weight Change reward [4] to generate the behavior. The initial α of each LMS learner were set to one. The lines for the constant target (blue) and drifter target (green) are overlapping, and the lines for the distractor targets (red and black) are overlapping. Autostep progressively decreases α for the distractor targets, as the updates oscillate around zero. The update magnitude (or error) for the constant target goes to zero, and so Autostep stops changing α. This makes sense: why change the α if the prediction is perfect. Autostep keeps the α high for the learner estimating the drifter target, because continual progress is possible. On each time step the LMS learner moves its estimate towards the recent sample and most of these updates are in the same direction, at least over a recent window of time. In terms of prediction performance, Autostep significantly improves tracking, enabling different update rates for different prediction learners and reducing α on unlearnable targets or noisy targets once learning is complete—as you will see in our main experiments below.

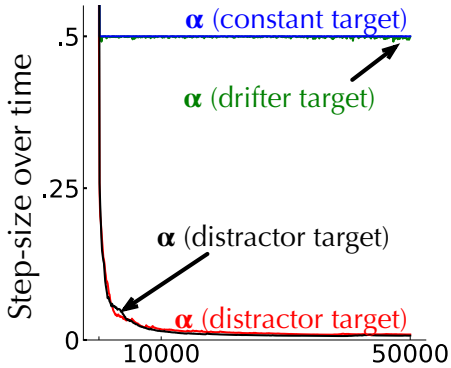

Figure 3: Sample run showing how Autostep changes the step-size parameters (α) over time with *Weight Change* reward. The lines for the constant target (blue) and drifter target (green) are overlapping, and the lines for the distractor targets (red and black) are overlapping.

We experimented with other step-size adaption methods, including AdaDelta and RMSProp, but the results were qualitatively similar. In this study we chose Autostep because (a) it was specifically designed for non-stationary, incremental, online tracking tasks like

4. The details of the intrinisic reward function used to generate the data do not matter for the purpose of the this experiment. Nevertheless, the Weight Change reward will be defined in the next section.

ours, (b) it uses a simple and easy to interpret update rule, and (c) there is a long literature demonstrating the practical utility of this method dating back to its origins in the IDBD method (Sutton, 1992).

The choice of using meta-learning to obtain introspective learners not only works well in our multi-prediction tasks, but also should scale to larger tasks with function approximation in future work. Step-size adaption methods like Adam and RMSProp can speed up training in neural networks and make learning more robust to non-stationarity. In online reinforcement learning, extensions of Autostep have shown to improve prediction and control performance with function approximation (Kearney et al., 2018; Günther et al., 2020). We discuss these extensions, and how our results go beyond stateless tracking at the end of the paper. It is worth emphasizing that introspective learners are not optimal learners and that they are in fact the most common type of agent used in deep reinforcement learning. The main criterion is that an introspective agent should regulate its own updates based on an internal measure of learning progress.

5. Intrinsic Rewards for Multi-prediction Learning

Many learning systems draw inspiration from the exploratory behavior of humans and animals, uncertainty reduction in active learning, and information theory—and the resulting techniques could all be packed into the suitcase of curiosity and intrinsic motivation. In an attempt to distill the key ideas and perform a meaningful yet inclusive empirical study, we consider only methods applicable to our problem formulation of multi-prediction learning. Although few approaches have been suggested for off-policy multi-task reinforcement learning—approaches by Chentanez et al. (2005); White et al. (2014) as notable exceptions—many existing approaches can be used to generate intrinsic rewards for multiple, independent prediction learners (see the excellent summary by Barto, 2013). We first summarize methods we evaluate in our empirical study. The specific form of each intrinsic reward discussed below is given in Table 4, with italicized names below corresponding to the entries in the table. We conclude by mentioning several rewards we did not evaluate, and why.

Several intrinsic rewards are based on **violated expectations**, or surprise. This notion can be formalized using the prediction error itself to compute the instantaneous *Absolute Error* or *Squared Error*. We can obtain a less noisy measure of violated expectations with a windowed average of the error, which we call *Expected Error*. Regardless of the specific form, if the error increases, then the intrinsic reward increases encouraging further sampling for that target. Such errors can be normalized, such as was done for *Unexpected Demon Error* (White et al., 2014), to mitigate the impact of noise in and magnitude of the targets.

Another category of methods focus on **learning progress**, and assume that the learning system is capable of continually improving its policy or predictions. This is trivially true for approaches designed for tabular stationary problems (Chentanez et al., 2005; Still and Precup, 2012; Little and Sommer, 2013; Meuleau and Bourgine, 1999; Barto and Simsek, 2005; Szita and Lorincz, 2008; Lopes et al., 2012; Schossau et al., 2016). The most well-known approaches for integrating intrinsic motivation make use of rewards based on improvements in (model) error: including *Error Reduction* (Schmidhuber, 1991b, 2008), and model *Error Derivative* approach (Oudeyer et al., 2007). Improvement in the value function can also be used to construct rewards, and can be computed from the *Positive Error Part*

(Schembri et al., 2007), or by tracking improvement in the value function over all states (Barto and Simsek, 2005). As our experiments reveal, however, intrinsic rewards requiring improvement can lead to less desireable behavior in non-stationary tracking problems.

An alternative to learning progress is to reward **amount of learning**. This does not penalize errors becoming worse, and instead only measures that estimates are changing: the prediction learner is still adjusting its estimates and so is still learning. *Bayesian Surprise* (Itti and Baldi, 2006) formalizes the idea of amount of learning. For a Bayesian learner, which maintains a distribution over the weights, Bayesian Surprise corresponds to the KL-divergence between this distribution over parameters before and after the update. This KL-divergence measures how much the distribution over parameters has changed. Bayesian Surprise can be seen as a stochastic sample of Mutual Information, which is the expected KL-divergence between prior and posterior across possible observed targets. We discuss this more in Section 6. Other measures based on Information Gain have been explored (Still and Precup, 2012; Little and Sommer, 2013; Achiam and Sastry, 2017; de Abril and Kanai, 2018; Berseth et al., 2019). In the tabular case different variations of information-gain reward perform similarly to Bayesian Surprise empirically (Little and Sommer, 2013).

Though derived assuming stationarity and Bayesian learners, we provide an approach to approximate Bayesian Surprise for our non-stationary setting with non-Bayesian learners. The prediction learner's main objective is to estimate an unknown mean. A Bayesian learner maintains a distribution over this unknown mean, based on the chosen distribution for the targets. A simple choice is to use a Gaussian distribution for the targets, with an unknown mean but a known variance, giving a Gaussian conjugate prior. The variance is not actually known; we maintain an estimate $v_t^{(y)}$ of the variance of the target $\text{Var}[C_{t,i}]$.[5] The posterior uses the learner's mean estimate and the posterior variance for a Bayesian update, which is proportional to $v_t^{(y)}/t$. Effectively, the approximate Bayesian surprise is maintaining a posterior, but uses the learners mean estimate instead of its own. To additionally account for non-stationarity, we use the same idea behind DTS: increasing the posterior variance after each update (see Table 4 for the formula).[6] We make no claims that this is the ideal strategy to approximate to Bayesian Surprise for non-Bayesian learners in non-stationary problems; it is rather a reasonable simple strategy in an effort to include it as best as possible in our experiments.

We can additionally consider non-Bayesian strategies for measuring amount of learning, including those based on change in error (*Error Derivative*), *Variance of Prediction*, *Uncertainty Change*—how much the variance in the prediction changes—and the *Weight Change*, which we discuss in more depth in the next section. Note that several learning progress measures can be modified to reflect amount of learning by taking the absolute value, and so removing the focus on increase rather than change (this must be done with care as we likely do not want to reward model predictions becoming worse, for example).

5. Typically, a Bayesian learner would simply maintain a distribution over both the mean and variance, if they are both unknown. Our goal here, though, is to approximate Bayesian surprise for a non-Bayesian learner. Since the learner only estimates the mean, we assume that the corresponding Bayesian learning can only maintain a distribution over the mean.
6. The count n is decayed by $1-\beta$, and the variance is prevented from decreasing below 10^{-3}. To keep the update simpler, we assume the initial variance for the prior is very large—which is in fact reasonable as it implies no prior knowledge about the unknown mean. Therefore the initial variance can be omitted in the posterior variance, as it has a negligible affect.

Reward Name	$R_{t,i}$
Error Derivative (Oudeyer et al., 2007)	$\left\lvert \frac{1}{\eta+1}\sum_{j=0}^{\eta}\delta^2_{t-j-\tau,i} - \frac{1}{\eta+1}\sum_{j=0}^{\eta}\delta^2_{t-j,i} \right\rvert$
$\tau \leq \eta < t$, where η specifies the length of the window and τ the amount of overlap	
Expected Error	$\left\lvert \overline{\delta_{t,i}}^{\beta} \right\rvert$
$\overline{\delta_{t,i}}^{\beta} = (1-\beta)\overline{\delta_{t-1,i}}^{\beta} + \beta\delta_{t,i}$ is the exponentially weighted average, with $0 < \beta < 1$	
Step-size Change	$\lvert \alpha_{t-1,i} - \alpha_{t,i} \rvert$
Error Reduction (Schmidhuber, 1991a)	$\lvert \delta_{t-1,i} \rvert - \lvert \delta_{t,i} \rvert$
Squared Error (Gordon and Ahissar, 2011)	$\delta^2_{t,i}$
Bayesian Surprise (Itti and Baldi, 2006)	$\frac{1}{2}\log_2\left(\frac{v_{t,i}}{v_{t-1,i}}\right) + \frac{v_{t-1,i} + (\hat{c}_{t-1,i} - \hat{c}_{t,i})^2}{2v_{t,i}} - \frac{1}{2}$
$v_{t,i} = \max(v_t^{(y)}/n_t, 10^{-3})$ where $n_t = (1-\beta)n_{t-1}+1$. Note $v_{t,i}$ is the posterior variance estimate and $v_t^{(y)}$ an estimate of the target variance $\mathrm{Var}[C_{t,i}]$ obtained using an exponential average variant of Welford's algorithm: $v_t^{(y)} = (1-\beta)v_{t-1}^{(y)} + \beta(c_{t,i} - \hat{c}_{t-1,i})(c_{t,i} - \hat{c}_{t,i})$.	
Unexpected Demon Error (White et al., 2014; White, 2015)	$\left\lvert \frac{\overline{\delta_{t,i}}^{\beta}}{\sqrt{\mathrm{Var}[\delta_i]} + \epsilon} \right\rvert$
ϵ is a small constant, set to 10^{-6} in our experiments. $\mathrm{Var}[\delta_i]$ is a standard sample variance.	
Uncertainty Change	$\lvert \mathrm{Var}[\hat{c}_{t-1,i}] - \mathrm{Var}[\hat{c}_{t,i}] \rvert$
Variance of Prediction	$\mathrm{Var}[\hat{c}_{t,i}]$
where variances $\mathrm{Var}[\hat{c}_{t,i}]$ are estimated using an exponential average variant of Welford's algorithm: $v_t = (1-\beta)v_{t-1} + \beta(\hat{c}_{t,i} - \bar{c}_{t-1,i})(\hat{c}_{t,i} - \bar{c}_{t,i})$ for $0 < \beta < 1$, with $\bar{c}_{t,i} = (1-\beta)\bar{c}_{t-1,i} + \beta\hat{c}_{t,i}$ the exponential average of the predictions.	
Weight Change	$\lVert w_{t,i} - w_{t-1,i} \rVert_1 = \alpha_t \lVert \hat{c}_{t,i} - \hat{c}_{t-1,i} \rVert_1$
Absolute Error* (Schmidhuber, 1991b)	$\lvert \delta_{t,i} \rvert$
Positive Error Part* (Mirolli and Baldassarre, 2013)	$\max(\delta_{t,i}, 0)$
Variance of Error*	$\mathrm{Var}[\delta_{t,i}]$
Uncertainty Reduction*	$\mathrm{Var}[\hat{c}_{t-1,i}] - \mathrm{Var}[\hat{c}_{t,i}]$

Figure 4: Intrinsic rewards investigated in this work. Separate statistics are maintained for each task i, and only updated when task i is selected by the behavior agent. Starred rewards performed poorly and where excluded from the results. Sample averages are computed using an unbiased exponentially-weighted average introduced by Sutton and Barto (2018)[Eq 2.9].

There are several strategies which we omit, because they would (1) result in uniform exploration in our pure exploration problem, (2) require particular predictions about state to drive exploration, (3) are designed for the offline batch setting, or (4) are based on statistics of the targets rather than the statistics generated by the prediction learners. Count-based approaches (Brafman and Tennenholtz, 2002; Bellemare et al., 2016; Sutton and Barto, 2018) are completely unsupervised, rewarding visits to under-sampled states or actions—resulting in uniform exploration in our problem. Though count-based approaches are sometimes used in learning systems, they reflect novelty rather than learning progress or surprise (Barto et al., 2013).

The second set of strategies we omit are methods that use a model to encourage exploration (Schmidhuber, 2008; Chentanez et al., 2005; Stadie et al., 2015; Pathak et al., 2017, 2019) such as by using Bayesian Surprise for next-state prediction (Houthooft et al., 2016). Subgoal discovery systems (Kulkarni et al., 2016; Andrychowicz et al., 2017; Péré et al., 2018) define rewards to reach particular states. Empowerment and state control systems are explicitly designed to respect and use the fact that some tasks or regions of the state-space cannot be well learned. Often such systems use only unsupervised signals relating to statistics of the exploration policy, ignoring the statistics generated by the learning process itself (Karl et al., 2017). Like count-based approaches, unsupervised measures like this would induce uniform exploration in our stateless task.

Curriculum learning—learning what task to sample next—is closely related to our multi-prediction problem. Graves *et al.* (2017) introduce several measures for batch curriculum learning that are related to the ideas underlying the intrinsic rewards discussed above. Most related, Prediction Gain corresponds to *Error Reduction*, albeit assuming a batch of data rather than an online instance. An approximation, called Gradient Prediction Gain, corresponds to the norm of the gradient; for our setting, this is the same as the *Absolute Error*. Several of Graves' measures require the ability to sample new batches of data, such as Supervised Prediction Gain and Target Prediction Gain. Finally, Graves *et al.* investigated several Complexity Gain measures for the neural networks, measuring KL divergence between the posterior and a learned prior. The prior is updated towards the previous posterior, and so the resulting KL is related to Bayesian surprise. The KL itself, though, is not used: rather, the gain in complexity is measured by looking at the difference in two KLs, before and after an update. These approaches require Bayesian learners with a separate prior distribution to be learned just to measure the complexity. The most simple and computationally feasible of these is L2 Gain, which is simply the difference in ℓ_2 norm of the weights before and after and update: $\|w_{t,i}\|_2^2 - \|w_{t-1,i}\|_2^2$. This rewards the learning system for making the weights smaller, and performed worse than random for curriculum learning (Graves et al., 2017).

Finally, we do not test intrinsic rewards based only on targets, such as variance of the target. To see why, consider a behavior that estimates the variance for a constant target, and quickly determines it only needs to select that action a few times. The prediction learner, however, could have a poor estimate of this target, and may need many more samples to converge to the true value. Separately estimating *possible* amount of learning from *actual* amount of learning has clear limitations. Note that in the stationary bandit setting, with a simple sample average learner, the variance of the prediction target provides a measure of uncertainty for the learned prediction (Audibert et al., 2007; Garivier and Moulines,

2011; Antos et al., 2008), and has been successfully applied in education applications (Liu et al., 2014; Clement et al., 2015). When generalizing to other learners and problem settings, however, variance of the target will no longer obviously reflect uncertainty in the predictions. We therefore instead directly test intrinsic rewards that measure uncertainty in predictions, including Uncertainty Change and Variance of Prediction.

6. Optimal Behavior for Multi-prediction Learning, and Approximations

One natural question given this variety of intrinsic rewards, is if there is an optimal approach. In some settings, there is in fact a clear answer. In a stationary, stateless problem where the goal is to estimate means of multiple targets, it has been shown that the behavior agent should take actions proportional to the variance of each target to obtain minimal regret (Antos et al., 2008). For a stationary setting, with state, an optimal approach would be to take actions to maximize Information Gain—the reduction in entropy after an update—across learners (Orseau et al., 2013). We therefore use Information Gain as the criterion to measure optimal action selection. In this section, we describe how to maximize Information Gain in an ideal case, and how to approximate it otherwise. The goal of this section is to provide intuition and motivation, as we do not yet have theoretical claims about the approximation strategies.

We first show that Information Gain is maximized when maximizing expected Bayesian surprise, assuming Bayesian learners. A Bayesian learner updates weights w for a parameterized distribution p_w on the parameters θ needed to make the prediction $\hat{c}$. The parameters can be seen as a random variable, Θ, with distribution p_w. The goal is to narrow this distribution around the true parameters θ^* that generate c—that is, c is sampled from $p(c|\theta^*)$. After seeing each new sample, the posterior distribution over parameters is computed using the previous distribution $p_{w_t}(\theta)$ and the new sample, c_t, using the update

$$p_{w_{t+1}}(\theta) \stackrel{\text{def}}{=} p_{w_t}(\theta|c_t) = \frac{p(c_t|\theta)p_{w_t}(\theta)}{p_{w_t}(c_t)}.$$

The term in the denominator is dependent on w_t because $p_{w_t}(c_t) = \int p(c_t|\theta)p_{w_t}(\theta)d\theta$. A *Bayesian learner* is one that uses exact updates to obtain the posterior. We assume the prior is appropriately specified so that $p_{w_t}(c) \neq 0$, and so $p(\theta|c_1, \ldots, c_n)$ has non-zero support as $n \to \infty$ almost surely for any stochastic sequence $c_1, \ldots, c_n$.

Bayesian surprise is defined as the KL divergence between the distribution over parameters before and after an update (Itti and Baldi, 2006)

$$\text{KL}(p_{w_{t+1}}||p_{w_t}) = \int p_{w_{t+1}}(\theta) \log \frac{p_{w_{t+1}}(\theta)}{p_{w_t}(\theta)} d\theta. \tag{5}$$

The Bayesian surprise is high when taking an action that produces a stochastic outcome c_t that results in a large change in the prior and posterior distributions over parameters. The expectation of the KL-divergence over stochastic outcomes, with a Bayesian learner, corresponds to the Information Gain. This result is well-known, but we explicitly show it in the following theorem for completeness. Notice that Information Gain defined in Equation (6) is relative to the model class of our learner, rather than some objective notion of information content.

Theorem 1 *Assume targets $C \in \mathcal{C}$ are distributed according to true parameters θ^*, with density $p_{\theta^*} : \mathcal{C} \to \infty$ and event space . For a Bayesian learner, that maintains distribution p_{w_t} over parameters Θ, the mutual information (also called the Information Gain) $I(C, \Theta)$ equals the expected KL-divergence between the posterior and prior*

$$I(C, \Theta) = \mathbb{E}[KL(p_{w_{t+1}} || p_{w_t})] \tag{6}$$

where the expectation is over stochastic outcomes C that produce w_{t+1} from w_t.

Proof

$$\begin{aligned}
I(C, \Theta) &= \int \int p_{\theta^*, w_t}(c, \theta) \log \frac{p_{\theta^*, w_t}(c, \theta)}{p_{\theta^*}(c) p_{w_t}(\theta)} dc d\theta && \triangleright\ p_{\theta^*, w_t}(c, \theta) = p_{\theta^*}(c) p_{w_t}(\theta|c) \\
&= \int \int p_{\theta^*}(c) p_{w_t}(\theta|c) \log \frac{p_{w_t}(\theta|c)}{p_{w_t}(\theta)} dc d\theta && \triangleright\ p_{\theta^*}(c) \text{ cancels in the fraction} \\
&= \int p_{\theta^*}(c) \left[\int p_{w_t}(\theta|c) \log \frac{p_{w_t}(\theta|c)}{p_{w_t}(\theta)} d\theta \right] dc && \triangleright\ p_{\theta^*}(c) \text{ does not involve } \theta \\
&= \int p_{\theta^*}(c) \left[\int p_{w_{t+1}}(\theta) \log \frac{p_{w_{t+1}}(\theta)}{p_{w_t}(\theta)} d\theta \right] dc && \triangleright\ p_{w_{t+1}}(\theta) = p_{w_t}(\theta|c) \\
&= \int p_{\theta^*}(c) \mathrm{KL}(p_{w_{t+1}} || p_{w_t}) dc && \triangleright\ \text{Equation (5)} \\
&= \mathbb{E}[\mathrm{KL}(p_{w_{t+1}} || p_{w_t})]
\end{aligned}$$

The weights w_{t+1} are dependent on the observed c. By definition, this integral gives an expected KL, across possible observed c. ■

To make this more concrete, consider Bayesian surprise for a Bayesian learner with a simple Gaussian distribution over parameters. For our simplified problem setting, the weights for the Bayesian learner are $w_t = (\mu_t, \sigma_t^2)$ for the Gaussian distribution over the parameters θ, which in this case is the current estimate of the mean of the target, $\hat{c}_t$. The Bayesian surprise is

$$\mathrm{KL}(p_{w_{t+1}} || p_{w_t}) = \log \frac{\sigma_{t+1}^2}{\sigma_t^2} + \frac{\sigma_t^2 + (\mu_t - \mu_{t+1})^2}{2\sigma_{t+1}^2} - \frac{1}{2}.$$

We can make this even simpler if we consider the variance σ^2 to be fixed, rather than learned. The Bayesian surprise then simplifies to

$$\begin{aligned}
\mathrm{KL}(p_{w_{t+1}} || p_{w_t}) &= \log \frac{\sigma^2}{\sigma^2} + \frac{\sigma^2 + (w_t - w_{t+1})^2}{2\sigma^2} - \frac{1}{2} \\
&= 0 + \frac{\sigma^2}{2\sigma^2} + \frac{(w_t - w_{t+1})^2}{2\sigma^2} - \frac{1}{2} \\
&= \frac{(w_t - w_{t+1})^2}{2\sigma^2}.
\end{aligned} \tag{7}$$

This value is maximized when the squared change in weights $(w_t - w_{t+1})^2$ is maximal. Therefore, though Bayesian surprise in general may be expensive to compute, for some settings it is as straightforward as measuring the change in weights.

Additionally, we can also consider approximations to Bayesian surprise for non-Bayesian learners. A non-Bayesian learner typically estimates the parameters θ_t directly, such as by maximizing likelihood or taking the maximum a posteriori (MAP) estimate

$$\theta_{t+1} \stackrel{\text{def}}{=} \arg\max_{\theta} p(\theta|c_1, \ldots, c_t).$$

Now instead of maintaining the full posterior $p(\theta|c_1, \ldots, c_t)$ as $p_{w_{t+1}}$, the prediction learner need only learn θ_{t+1} directly. Because θ_{t+1} is the mode of the posterior, for many distributions θ_{t+1} will actually equal a component of w_{t+1}. For the Gaussian example above with a learned variance, θ_{t+1} equals the first component of w_{t+1}, the mean μ_{t+1}. For a fixed variance, θ_{t+1} exactly equals w_{t+1}. Therefore, the non-Bayesian learner would have the exact same Information Gain, measured by the Bayesian surprise in (7). Note that MAP makes the same probabilistic assumptions for the posterior as the Bayesian learner, but is considered non-Bayesian here because it does not maintain the parameters of this posterior.

This direct connection, for Bayesian and non-Bayesian learners, only exists for a limited set of distributions. One such class is the natural exponential family distribution over the parameters. Examples include the Gaussian with fixed variance and mean w_t and the Gamma distribution with a fixed shape parameter and scale parameter w_t. Each natural exponential family has the property that the KL-divergence between two distributions with parameters w_t and w_{t+1} corresponds to a (Bregman) divergence directly on the parameters (Banerjee et al., 2005). For a Gaussian, this divergence is the squared error normalized by the variance, as above in Equation (7). Another distribution that has this connection is a Laplace distribution with mean w_t and fixed variance $2b^2$. Then the KL-divergence is $KL(p_{w_{t+1}}||p_{w_t}) = |w_t - w_{t+1}|/b$.

This connection is limited to certain posterior distributions, but is true for general problem settings, even the general reinforcement learning setting. The distributions before and after an update, p_{w_t} and $p_{w_{t+1}}$ respectively, are over the parameters of the prediction learner. These parameters are more complex in settings with state—such as parameters to a neural network—but we can nonetheless consider exponential family distributions on those parameters.

This discussion motivates a simple proposal to approximate Bayesian surprise and Bayesian learners for a general setting with non-Bayesian learners: using weight change with *introspective* learners. An introspective learner is not a precise definition, but rather a scale. A perfectly introspective learner would be a Bayesian learner, or in some cases the corresponding MAP learner. A perfectly non-introspective learner could be a random update. The more closely the learner approximates the weights to the perfectly introspective learner, the better its solution and the better the Bayesian surprise reflects the Information Gain. Further, because the underlying distribution may not be known, we use the change in weights as an approximation.

For concreteness, consider the following learning system. Each prediction learner is augmented with a procedure to automatically adapt the step-size parameter $\alpha_{k,i}$, based on the errors produced over time ($\delta_{i,0:k}$). In this paper we use the Autostep algorithm (Mahmood et al., 2012) described in Section 4. Recall, that the Autostep algorithm automatically reduces $\alpha_{k,i}$ towards zero if the target is unlearnable, increase $\alpha_{k,i}$ when successive errors have the same sign, and does not change $\alpha_{k,i}$ if the error is zero. We call a learner with

a fixed step-size parameter, on the other hand, non-introspective, because the learner will forever chase the noise. The weight change for such a learner would not be reflective of Information Gain, reflecting instead only the inadequacy of the learner. A learner equipped with Autostep, on the other hand, like a Bayesian or MAP learner, will stop learning once new samples provide no new information.

This proposal reflects a simply reductionist philosophy: there should be an explicit separation in the role of the behavior agent and the role of the prediction learners. The behavior agent should balance data generation amongst parallel prediction learners. The prediction learner's primary responsibility is to estimate their target accurately. If the behavior agent trusts that the prediction learners are using the data appropriately, then the learning system can make use of intrinsic rewards based solely on the prediction learner's parameters, such as the change in the weights. The alternative is to assume that the intrinsic rewards must be computed to overcome poor learning. This approach would require the learning system to recognize when a prediction learner is non-introspective, and decrease the reward associated with that learner. If the learning system can measure this, though, then presumably so too can the prediction learner—they are after all part of the same system. The learner should then be able to use the same measure to adjust its own learning.

In this work, we define the change in weights using the ℓ_1 norm,

$$\text{Weight Change}(w_t, w_{t+1}) \doteq \|w_t - w_{t+1}\|_1. \tag{8}$$

In our setting, the Weight Change is simply Absolute Error scaled by the step-size parameter, emphasizing the role that learner capability plays in ensuring an effective reward.

$$\|w_t - w_{t+1}\|_1 = \alpha_{t,i}\|\hat{c}_{t,i} - \hat{c}_{t-1,i}\|_1 = \alpha_{t,i}|\delta_{t,i}| \tag{9}$$

Remark: The above discussion applies to the non-stationary setting, by treating the non-stationarity as partial observability. We can assume that the world is stationary, driven by some hidden state, but that it appears non-stationary to the learning system because it only observes partial information. If a Bayesian prediction learner had the correct model class, it could still maximize Information Gain. For example, the prediction learner could know there is a hidden parameter ξ defining the rate of drift for the mean of the distribution over C. It could then maintain a posterior over both ξ and the mean and covariance of C, based on observed data. As above, it would be unlikely for the prediction learner to have this true model class. It remains an important open theoretical question how such approximations influence the behavior agent's ability to maximize Information Gain.

7. Experimental Setup

We conducted five experiments, across the three problems described in Section 3. The goal of these experiments is to (a) assess the utility of different intrinsic rewards in our testbed with many different target distributions, and (b) to understand how the ability of the underlying prediction learners—introspective or not—impact the results.

Each component of the learning system is modulated by several hyper-parameters that interact in different ways. The behavior agent (gradient bandit) makes use of a step-size parameter α and the step-size parameter of the average reward estimate α_r. For non-introspective learners, each prediction learner makes use of a (shared) step-size parameter

Hyper-parameters	
Behavior agent (Gradient Bandit)	**Step-size parameter** $\alpha \in \{2^{-8}, 2^{-7}, \ldots, 2^{-2}\}$ **Average Reward rate** $\alpha_r \in \{10^{-5}, 10^{-4}, \ldots, 10^{-1}\}$
Behavior agent (Dynamic Thompson Sampling)	**Step-size parameter** $\alpha \in \{2^{-8}, 2^{-7}, \ldots, 2^{-2}\}$ **Initial mean estimate** $m_a = 100$
Non-introspective prediction learner (LMS with a constant step-size parameter)	**Step-size parameter** $\alpha_p \in \{2^{-7}, 2^{-6}, \ldots, 2^{-2}\}$, with $\alpha_i = \alpha_p$
Introspective prediction learner (LMS with Autostep)	**Meta learning-rate** $\kappa \in \{0.01, 0.05, 0.1\}$ **Initial step-size** $\alpha_{0,i} = 1.0$
Smoothing parameter (Variance of Prediction, Uncertainty Change, Bayesian Surprise, UDE, Expected Error)	$\beta \in \{10^{-7}, 10^{-6}, \ldots, 10^{-1}\}$
Error Derivative Window (all combinations s.t. $\eta > \tau$)	$\eta \in \{1, 5, 10, 25, 100, 1000\}$ $\tau \in \{1, 5, 10, 25, 100\}$

Table 5: The hyper-parameter configurations investigated across all three experiments. There was a total of 50,000 combinations of intrinsic reward function and hyper-parameter setting, with each of these evaluated using 200 independent runs.

α_p, with $\alpha_i = \alpha_p$ for all i. For introspective learners, the step-size adaption method Autostep uses a meta learning-rate parameter κ. Finally, many of the intrinsic rewards have their own tunable parameters. For example, *UDE* uses an exponential average of recent errors which requires a smoothing parameter β. Oudeyer's *Error Derivative* reward makes use of two windows of recent errors determined by scalar parameters η and τ. In most cases the key parameters of the prediction learner, behavior agent, and intrinsic reward correspond to different timescales—slower or faster—and so required noticeably different values. Because these choices have such a big impact on behavior, as we show, we needed extensive sweeps and analysis to gain insight into the methods. This warranted investigating each result deeply, to communicate a nuanced picture.

We extensively sweep all the key performance parameters of every learner and reward function, to ensure an accurate characterization of performance. Table 5 lists all the parameter settings we tested. In some cases we report results for several parameters to gain more specific insights into the behavior induced by an intrinsic reward. When providing overall results, we report the best performance of the learning system for each intrinsic reward, using the best performing parameters across all parameters tested. The best performing parameters were those that achieved the lowest total RMSE (defined in Equation 2) over the duration of the experiment, averaged over 200 independent runs. All told we tested over 50,000 parameter configurations, 200 times each across our three experiments.

When reporting results under the best parameters, we jointly tune hyper-parameters for the intrinsic reward and the prediction learners. These hyper-parameters are all part of the agent; the best hyper-parameters reflect the best the agent could do for that intrinsic reward and prediction learner. Even under ideal circumstances, many intrinsic rewards can fail to induce the desired behavior, highlighting issues with the intrinsic rewards or with the use of non-introspective learners.

Nonetheless, reporting the best parameters does not provide the full picture, and though we attempt to highlight certain key results for other hyper-parameters, we cannot and do not attempt to show the full picture. Ideally, we could slice down further, to provide this nuance. At the extreme, this could consist of showing all possible intrinsic rewards—*Error Derivative* with the smallest step-size parameter, *Error Derivative* with a the largest step-size parameter, and so on—for each of the many combinations of prediction learner and behavior agent (with different hyper-parameter settings). This is infeasible[7], and part of the role of the empirical analysis is to summarize key outcomes. We have provided what we believe are the key slices: different intrinsic rewards (under their ideal circumstances) with two types of prediction learners (non-introspective and introspective). When intrinsic rewards fail under idealized scenarios, this reflects how they might perform across hyper-parameter settings. When intrinsic rewards result in near-ideal behavior, we then dig deeper to understand if this was an accident of idealized hyper-parameter tuning, or more generally a characteristic of the intrinsic reward.

We follow the same basic template in the presentation of the results. First we report the behavior of the best configuration for each reward function using non-introspective learners (*i.e.,* without Autostep). For a given reward, the behavior is depicted by the probability of selecting each action over time according to the behavior agent's policy. This gives us insight into how each reward drives action selection over time. We then investigate the RMSE over time, plotting both the error of each predictor and the average. Finally, in each experiment we investigate the performance sensitivity of several intrinsic rewards with respect to the tunable parameters. This provides more detailed understanding of how the parameters interact and helps explain when some intrinsic rewards produce unexpected behaviors.

As a final note, the behavior plots (showing the action selection probability) do not include error bars. The error bars over 200 runs are negligibly small, except in some cases where action probabilities across runs varied significantly for poorly performing intrinsic rewards (e.g., the selection of the distractor targets with Squared Error reward in Figure 9). The variance across runs, however, can both make the plots difficult to read and hides how the action probabilities can vary over time within one run. For these plots, therefore, we instead show a sampling of individual runs. All learning curves (plotting RMSE) include standard error bars. The error bars in all learning curves for Experiments One and Two are not visible because they are smaller than the width of the mean line. In Experiment Three the error bars are visible and the results are significant.

7. To enable the reader to do this on their own, we have provided a python notebook to explore the full set of data, at http://jair.adaptingbehavior.com .

8. Experiment One: Drifter-Distractor

We start with our simplest task: the Drifter-Distractor problem. This problem has 1 constant target, 2 distractor targets and 1 drifter target (see Figure 2 in Section 3). This four-action problem highlights some key features we want out of our learned behaviors. The behavior should not be continually distracted by noisy or unlearnable things (the two distractor targets). It should be able to quickly learn about simple targets (the constant target), and ultimately focus action selection on targets that result in continual learning progress (the drifter target). We test if such a behavior is learned, with non-introspective and introspective learners, under different intrinsic rewards.

Let us first be more precise about how the behavior should look in this problem. Consider the ideal setting, where we have a Bayesian prediction learner. The behavior should try out all the actions in the beginning. The prediction learner associated with the constant target should quickly narrow the posterior—quickly reduce its error—and the behavior should stop selecting the corresponding action. The prediction learners associated with the high-variance distractor targets will take longer to learn due to the target variance, but eventually the posterior for these narrows as well and the learner converges to the correct prediction of zero. Once that happens the behavior should stop choosing the actions corresponding to the distractor targets. Finally, the prediction learner corresponding to the drifter target cannot ever reduce its error to zero: unending learning progress is possible. A Bayesian learner for the drifter target is effectively performing filtering, and needs to see samples constantly to track the changing mean. Therefore the behavior should eventually settle on selecting the action corresponding to the drifter target the majority of the time. This behavior is the ideal behavior, in that it most efficiently gathers the data needed for each prediction tasks.

There are a few common degenerate behaviors that are possible in this problem. The first is over-selecting the actions corresponding to the distractor targets. Every time the behavior takes one of these actions, the corresponding non-introspective prediction learner updates toward a random target and so its predictions can oscillate around the optimum. Over short windows of time, the variance of the drifter target is smaller than the distractor targets; within that window, the errors generated by the distractor targets will appear larger. This results in the behavior frequently selecting the distractor targets, occasionally selecting the drifter target and cycling between the three. Any methods that rely on prediction learners to not chase noise should exhibit this degenerate behavior, such as *Weight Change*. With non-introspective learners, this can only be prevented if the intrinsic reward can somehow distinguish between distractor targets and drifter targets.

The other common degenerate behavior is selecting all actions nearly equally. This strategy does not result in the lowest possible RMSE, but it does result in lower RMSE than other behaviors such as mostly selecting the actions corresponding to the distractor targets. The uniform strategy emerges because there is no setting of the parameters of the intrinsic reward to force the behavior to follow the ideal strategy described above.

8.1 Results with Non-introspective Learners

Figure 5 summarizes the behavior of the Gradient Bandit with several intrinsic reward functions, with non-introspective learners. The bold dash lines reflect the probabilities averaged over 200 runs, while the light stroke solid lines depict probabilities of individual runs. Sev-

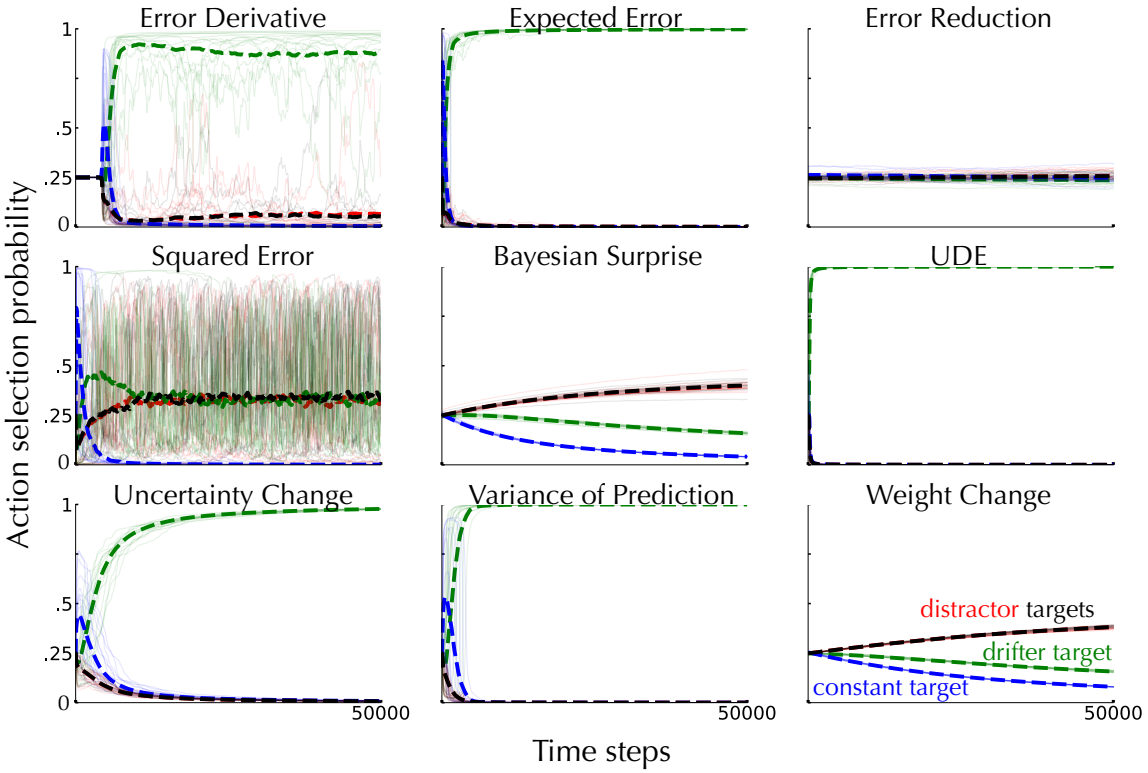

Figure 5: **Behavior** in the **Drifter-Distractor** problem, with **Non-Introspective Learners**. Each subplot corresponds to the behavior of the Gradient Bandit with a different intrinsic reward. Each line depicts the action selection probabilities learned by the behavior agent, over 50000 steps. The bold dashed lines show the mean probability of each action, averaged over 200 repetitions of the experiment. The light stroke solid lines show the probabilities computed by the Gradient Bandit for each action on individual runs—we only show a small random subset of 15 runs for readability. The **green line** corresponds to the drifter target, the **blue line** corresponds to the constant target, and the **red** and **black lines** correspond to the distractor targets. Intrinsic rewards based on variance estimates and averaging errors over time induce near-ideal action selection.

eral rewards induced the ideal behavior described above to varying degrees. Rewards based on simple moving averages of each learner's prediction error, including *Expected Error* and *UDE*, quickly latch on to the action corresponding to the drifter target. The parameter sweep chose a short averaging window, because the δ are more consistently the same sign for the drifter target, making the Expected Error higher for the drifter target. Using the variance of each predictors estimate, as in *Variance of Prediction* and *Uncertainty Reduction*, the behavior also converges to mostly selecting the action corresponding to the drifter target, after exploring the actions corresponding to the constant and distractor targets initially a bit longer. A parameter corresponding to a long window is used, because the predictions for the drifter target change much more over time than those for the distractor targets. Perhaps unsurprisingly the *Squared Error* and *Error Reduction* produce inappro-

priate behavior. *Bayesian Surprise* and *Weight Change* cause the Gradient Bandit to be distracted by the distractor targets resulting in sub-optimal behavior. The *Error Derivative* reward induces behavior that looks near-ideal in expectation, albeit there is more variance across runs than exhibited by other intrinsic reward functions.

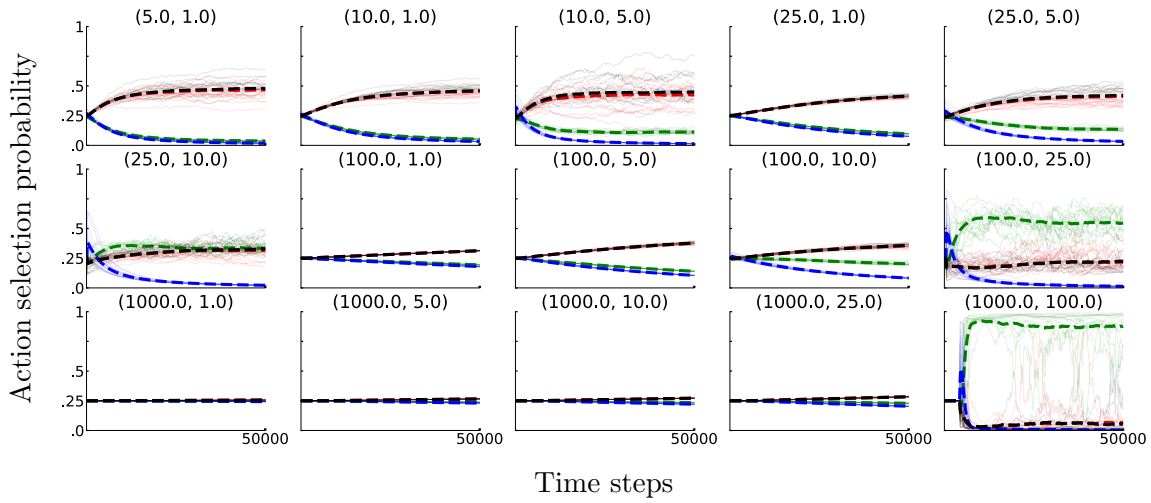

Figure 6: The impact of varying the window length parameters η and τ of **Error Derivative** reward, in the **Drifter-Distractor** problem, with **Non-introspective Learners**. Each subplot depicts the behavior of the Gradient Bandit algorithm with *Error Derivative* reward for many combinations of η, τ as indicated by the labels. As in Figure 5, each subplot shows both the average action selection probability for each action over time, and a small subset of individual runs. A large diversity of behaviors can be induced by changes to the window length parameters. Only one setting induced correct behavior: $\eta = 1000, \tau = 100$. This explains why the initial action selection was uniform in Figure 5: the reward is zero until the windows fill, which takes 1000 steps for $\eta = 1000$.

Performance in the Drifter-Distractor problem with non-introspective learners is largely dependent on setting the hyper-parameters of the each reward correctly. To illustrate this sensitivity, consider the *Error Derivative* reward, which is parameterized by two scalars η and τ. The η parameter controls the size of the window used to average recent errors, and τ controls how much each of the two windows overlap. Figure 6 shows the behavior of the Gradient Bandit, in terms of action selection probability over time, for every combination of η and τ. For each pair of (η, τ) we selected all the other hyper-parameters in the learning system to minimize the total RMSE; each subplot of the figure represents the best performance possible for a given (η, τ) pair according to RMSE. Across these combinations, we see the full gamut of behaviors. Only one setting out of twelve exhibited the described good behavior; most were uniform or focused on the distractor targets.

The hyper-parameters of the other components of the learning system also interact with the reward function. Figure 7 shows the best behavior—in terms of RMSE—of the Gradient Bandit for different values of the LMS predictor step-size parameter α_p. As the predictors learn faster, the *Error Derivative* reward induces nearly uniform action selection. If we slow

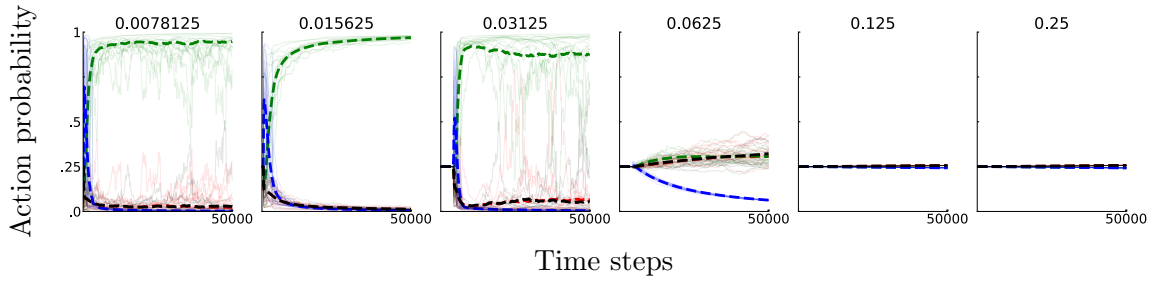

Figure 7: The impact of varying the LMS step-size parameter α_p with the **Error Derivative** reward, in the **Drifter-Distractor** problem, with **Non-introspective Learners**. Each subplot depicts the behavior of the Gradient Bandit algorithm with *Error Derivative* reward for for different values of α_p as indicated by the labels. Large α_p—faster target tracking—induces a uniform behavior, and smaller α_p produce action selection more similar to the ideal behavior but RMSE is higher because predictions are learned slowly. The third subplot, corresponding to $\alpha_p = 0.03125$, achieved the lowest total RMSE, because it allowed for somewhat faster learning for the predictions, but was still slow enough for the behavior to estimate learning.

the prediction learners updates with a smaller step-size parameter value, then the behavior strongly favors the action for the drifter target. This makes sense because with a small α_p, the intrinsic reward for the distractor targets becomes smaller and much bigger for the drifter target because the step-size parameter value is not large enough to track quickly. Though the action selection by the behavior is correct, this is not what we want from the learning system: we want the prediction learners to learn quickly, rather than artificially slowly so that the behavior can more easily track what they know. In fact, with small step-size parameter values, the RMSE is much worse than we can get with the introspective learners, where it is much easier to estimate learning progress and prediction learners can learn more aggressively.

Finally, let us investigate the error over time for each intrinsic reward. Figure 8 shows the exponential average of the RMSE over time for each reward function. We choose an exponential average to smooth the results (with a decay constant of 0.999). We plot both the error of each target, and the error average across targets. All rewards except *UDE* result in perfect prediction of the constant target; even *UDE* has near-zero error, indicating only minor under-selection of the action for the constant target. Rewards that induce nearly uniform action selection generate larger prediction error in aggregate (*Error Reduction* and *Surprise*). Reward functions that do not induce a strong preference for the drifter target exhibit high or growing error (*Weight Change*). Rewards that induce strong preference for the distractor targets do achieve better error on those predictions at the cost of accuracy in predicting the drifter targets (*Squared Error*). Achieving the lowest overall error requires first selecting the actions for the constant and distractor targets at first, and then focusing on the drifter target (i.e., *UDE*, *Uncertainty Change*, and *Variance of Prediction*).

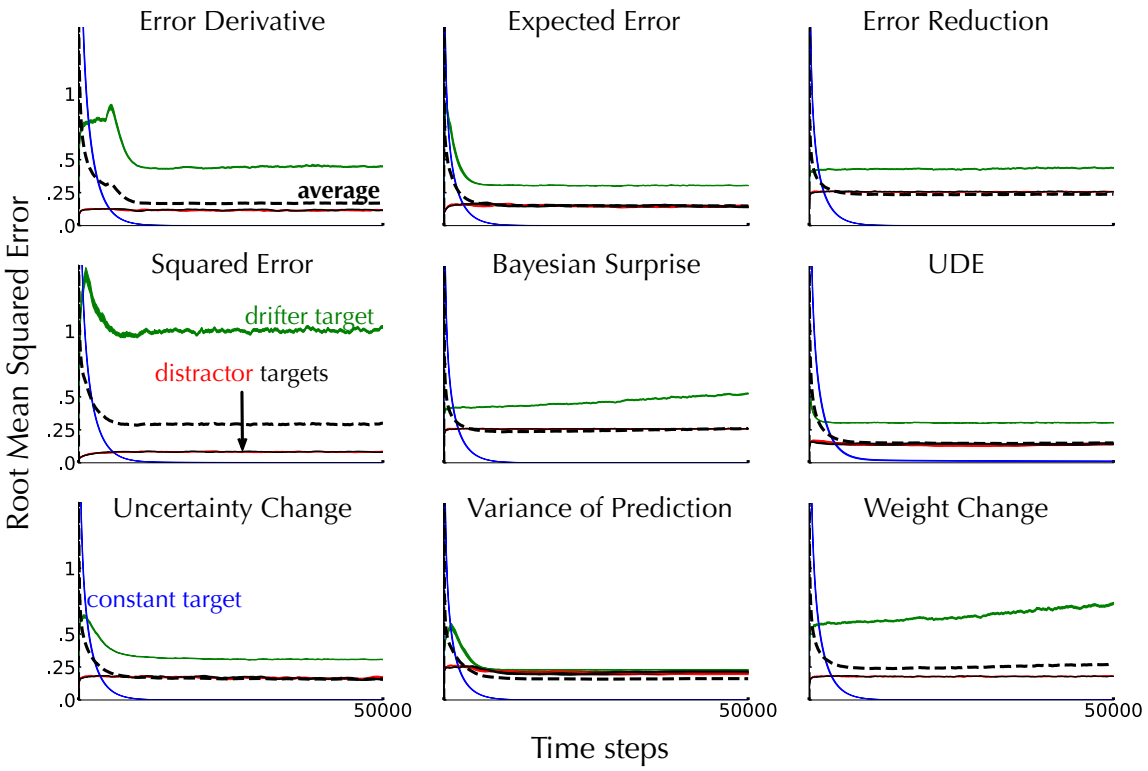

Figure 8: **RMSE** over time corresponding to each intrinsic reward function in the **Drifter-Distractor** problem with **Non-introspective Learners**. Each subplot corresponds to a different reward as labelled. The line colors correspond exactly as in the previous plots: **green drifting**, **black** and **red high-variance**, and **blue constant**. Each line is the exponentially weighted moving average of the LMS predictor's RMSE. The RMSE is computed with an exponential average, with a decay 0.999. The final results are averaged over 200 independent runs (standard error bars are plotted but not visible). The heavy stroke black dashed line reports average of the other four. Although many rewards induce similar action selection strategies, they can produce different RMSE curves.

8.2 Results with Introspective Learners

In this section we analyze the impact of different intrinsic rewards with introspective learners. We use LMS learners with Autostep, a step-size adaption method, to obtain introspective prediction learners. First let us recall how the step-size parameter for each LMS learner might change over time (see Figure 3 in Section 4 for reference), based on the errors generated by each of our three target types. The distractor targets are noisy—even if the mean is stable—so the LMS learner will experience positive and negative errors. The Autostep algorithm will reduce the step-size parameter corresponding to these targets, allowing each LMS learner to mitigate the variance and converge to the correct prediction of zero. The constant target on the other hand is easy to predict. Autostep will keep the step-size parameter large because the errors will be of the same sign. However, the error on the constant error can easily be reduced to zero with repeated sampling. Once the prediction error is

zero Autostep will modify the step-size parameter no further. The drifter target has noise, like the distractor targets, but the mean is not centered at zero, and it exhibits temporal structure. Consequently, the Autostep algorithm will keep the step-size parameter value high for the duration of the experiment. It is not hard to see that introspective learners should efficiently reduce error across all the targets, at least compared with a global, constant step-size parameter value. More subtly, an intrinsic reward that takes into account the dynamic values of the step-size parameter could exploit this additional information to adapt behavior to reduce error even faster.

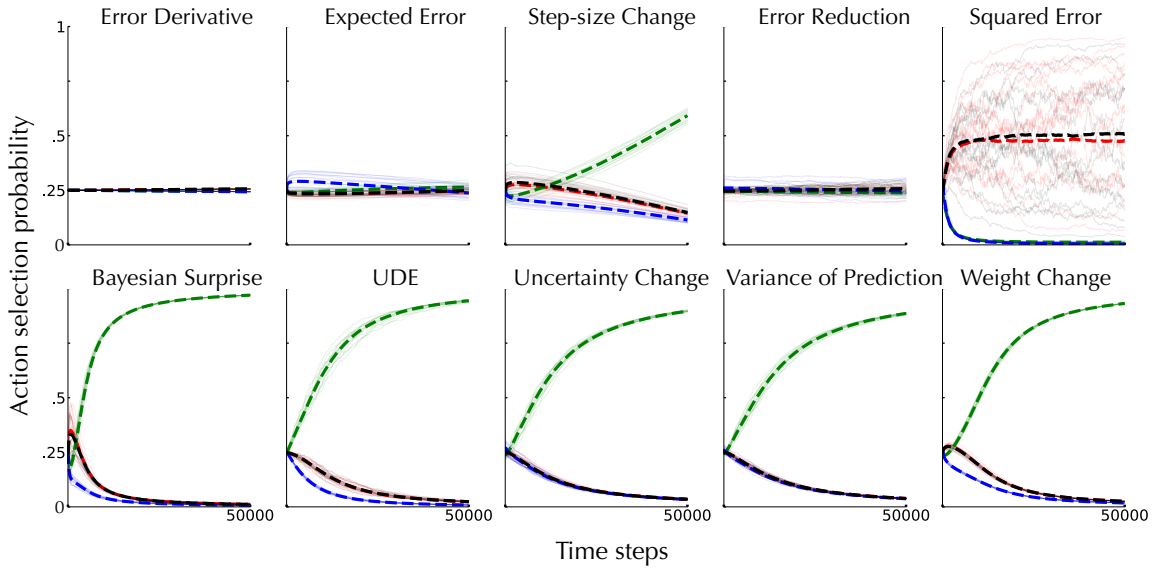

Figure 9: **Behavior** in the **Drifter-Distractor** problem with **Introspective Learners**. Each subplot corresponds to the behavior of the Gradient Bandit with a different intrinsic reward. Each LMS learner uses the Autostep algorithm to adapt the step-size parameter over time. The line coloring, labelling, and semantics mirror Figure 5. With Autostep, *Weight Change* induces near-ideal action selection. *Error Derivative* and *Expected Error* rewards, on the other hand, induce inappropriate action selection.

The setup of our second experiment was identical to the first except that each LMS learner maintained its own step-size parameter $\alpha_{t,i}$ updated via Autostep. We also include an intrinsic reward based on the change in the step-size parameter to assess the utility of rewarding action choices that caused changes in the step-size parameter values. This reward only makes sense if the step-size parameter can change over time, and thus was not included in the previous experiment.

The results of our second experiment are summarized in Figure 9. As before we plot the action selection probabilities to summarize the behavior. Weight change reward now induces near-ideal action selection. The step-size parameters for the distractor targets decay to a relatively small values causing the weight change to reduce—those actions become less and less rewarding. Autostep keeps the step-size parameter value relatively high for the drifter target, on the other hand, and the change in weights remains relatively high. Finally, even though the step-size parameter does not decay to zero for the constant target, the prediction

error for the constant target does go to zero. Consequently, the magnitude of the update also goes to zero, meaning the weight change goes to zero and preference for the constant target diminishes over time. *Bayesian Surprise* induces similar behavior as *Weight Change* as suggested by our analysis in Section 6. The variance-based rewards and *UDE* induce the same overall action preferences as without Autostep.

Across the board there is an improvement in RMSE reduction as shown in Figure 10. The RMSE is about half of that for the non-introspective learners. The differences in RMSE between the intrinsic rewards appear more minor, but the differences are meaningful. The total RMSE is well correlated with our definition of ideal behavior in this domain—reward functions that result in lower error exhibit the expected action preferences over time. To see larger differences, though, we need more actions. This first experiment was primarily designed to investigate qualitative behavior; the final experiment uses more actions and provides a better insight into quantitative differences.

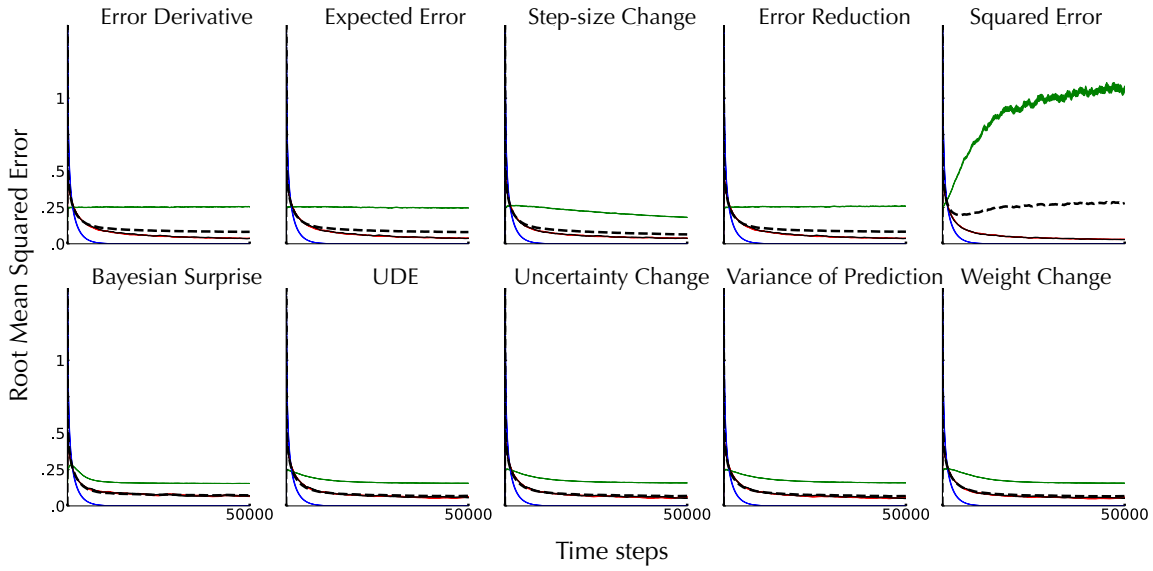

Figure 10: **RMSE** over time corresponding to each intrinsic reward function in the **Drifter-Distractor** problem with **Introspective Learners**. Results averaged over 200 runs, and standard error bars included. In this experiment, reward functions that induce similar action preferences produce similar RMSE reduction over profiles. Using *Weight Change* reward produces the lowest RMSE (0.108), however both *UDE* (0.109) and *Uncertainty Change* (0.110) result in similar performance. *Squared Error* results in the worst performance overall (0.292), and rewards that induce uniform action selection like *Error Derivative* result in larger error (0.124) compared with *Weight Change*.

For non-introspective learners, we observed that careful tuning of hyper-parameters allowed for the correct behavior for certain intrinsic rewards, by slowing prediction learning. This was the case for the *Error Derivative*, where in Figure 7 we observed that if the predictors learned too quickly, the drifter target did not produce the highest *Error Derivative*. For introspective learners, prediction learning cannot be slowed: they increase learning when learning is possible. We might expect *Error Derivative* to therefore perform poorly, and

be unable to find an appropriate hyper-parameter setting. We find this is the case: *Error Derivative* with Autostep does not induce the action selection preferences we expect—it causes nearly uniform action behavior—and no setting of the window parameters resulted in appropriate action preferences (Figure 11).

Overall, the preference for the drifter target is less pronounced with introspective learners, as seen in Figure 9. Instead, the behavior selects the actions for the distractor targets for longer. This is because step-size adaption is a meta (or second order) learning process, and so a non-trivial amount of data is required to recognize that learning is oscillating.

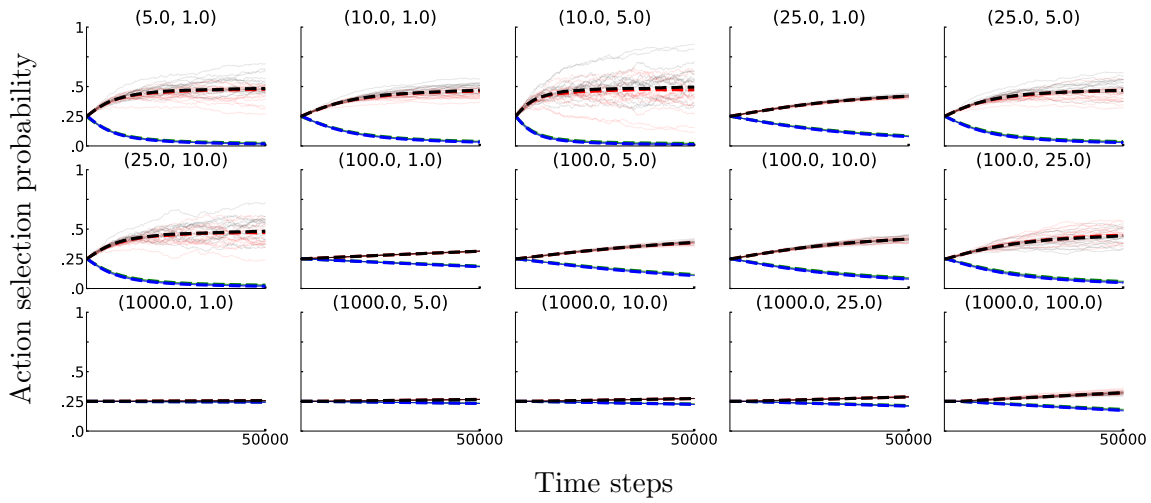

Figure 11: The impact of varying the window length parameters η and τ of **Error Derivative** reward in the **Drifter-Distractor** problem with **Introspective Learners**.

One might therefore wonder if rewards like the *Weight Change* reward simply hide the hyper-parameter tuning issue inside the step-size adaption algorithm. This seems not to be the case: the parameters of Autostep are straightforward to tune, and the behavior is largely insensitive to these choices as shown in Figure 12. Small meta learning-rate parameter values slow learning but do not prevent preference for the drifter target. The results of our first experiment highlight the utility of both simple intrinsic rewards—one's without hyper-parameters—and introspective learners in multi-prediction learning systems.

One final point of note is the surprising difference between *UDE* and *Expected Error*. In the previous experiment, with non-introspective learners, they performed similarly. In this experiment, with introspective learners, *Expected Error* results in uniform action selection whereas *UDE* provides the correct behavior. This is surprising, as *UDE* corresponds to *Expected Error* divided by the long-run sample standard deviation of the target. If we look more closely at the behavior induced by *Expected Error* with different smoothing parameters β, in Figure 13, it becomes more clear why this is the case. A small β in this problem results in early errors dominating the moving average; consequently, the constant target is preferred, as it generates high error at first. A larger β is needed to avoid this issue, but this unfortunately causes poor estimates of the true expected error for the distractor

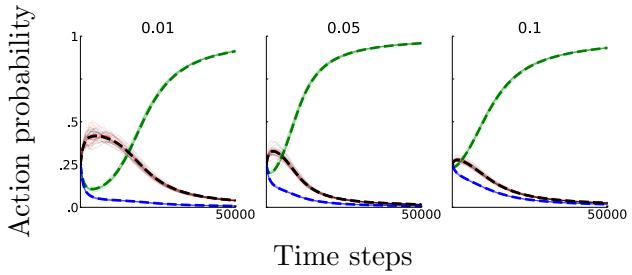

Figure 12: Action selection probabilities for the Gradient Bandit with **Weight-Change** reward under different meta learning-rate parameter values κ, in the **Drifter-Distractor** problem with **Introspective Learners**.

targets (which should be zero). In fact, it makes the errors for those target appear higher. Consequently, for the four smaller β, the constant target is preferred and for the two large, the distractor targets are preferred; there is no β amongst our set that lets the behavior focus on the drifter target.

UDE, on the other hand, has a way to overcome this: the long-run variance estimate makes the drifter target appear better. The variance of the drifter target appears small in the beginning of learning, and it takes many steps to start to recognize that it is actually high variance. In contrast, the variance estimate for the distractor targets are learned quickly, and the variance for the constant target looks higher initially due to consistent decrease in the error. This behavior is perhaps a bit accidental, and again highlights the complex interactions between all these hyper-parameters. This only further motivates the utility of intrinsic rewards with no hyper-parameters, that rely on introspective prediction learners.

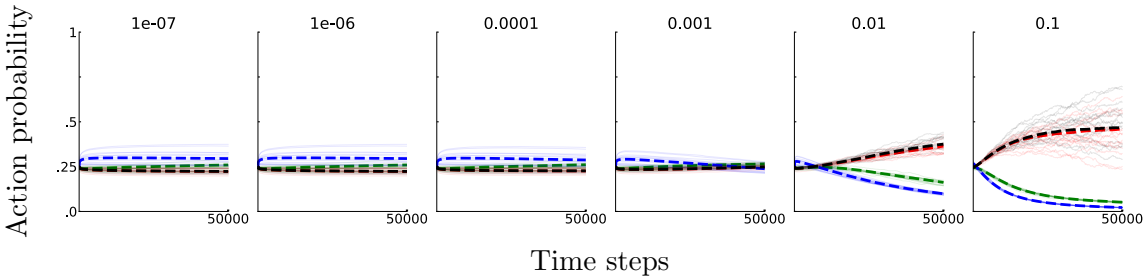

Figure 13: The impact of varying the smoothing parameter β of **Expected Error** reward in the **Drifter-Distractor** problem with **Introspective Learners**.

8.3 Results with Another Bandit Algorithm

A natural question is if the above results are specific to the Gradient Bandit behavior learner. To verify that our conclusions were not somehow overfit to the Gradient Bandit algorithm, we also repeated Experiment One with a Dynamic Thompson Sampling (DTS) algorithm, described in Section 2.1. This bandit algorithm is representative of a different

class of algorithms used in online learning: DTS estimates action-values instead of action-preferences and uses optimism (Thompson sampling) to increase exploration.

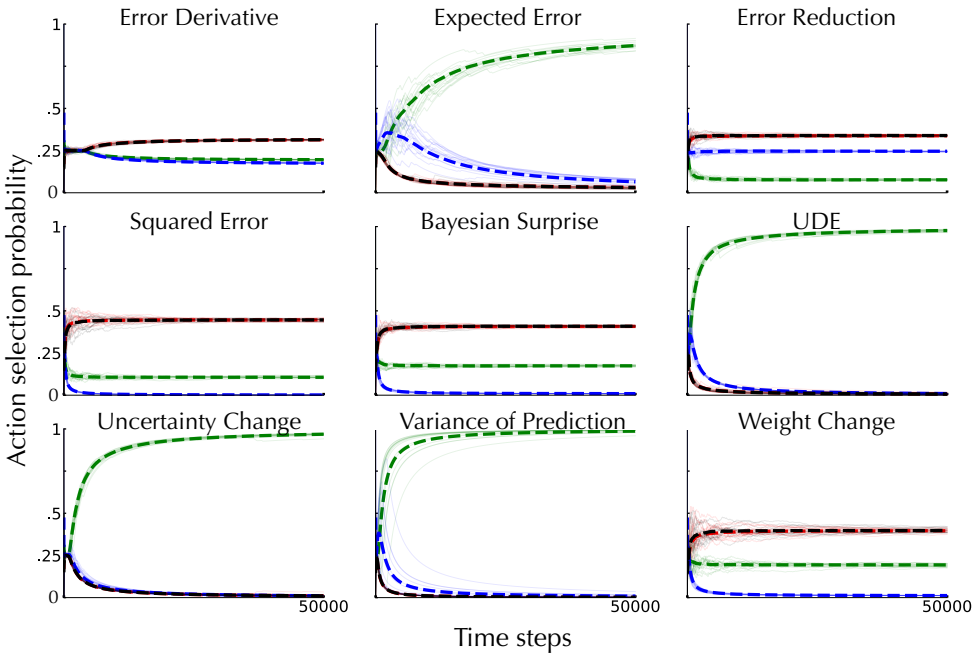

Figure 14: **Behavior** in the **Drifter-Distractor** problem, with **Non-Introspective Learners** where the behavior is learned using Dynamic Thompson Sampling.

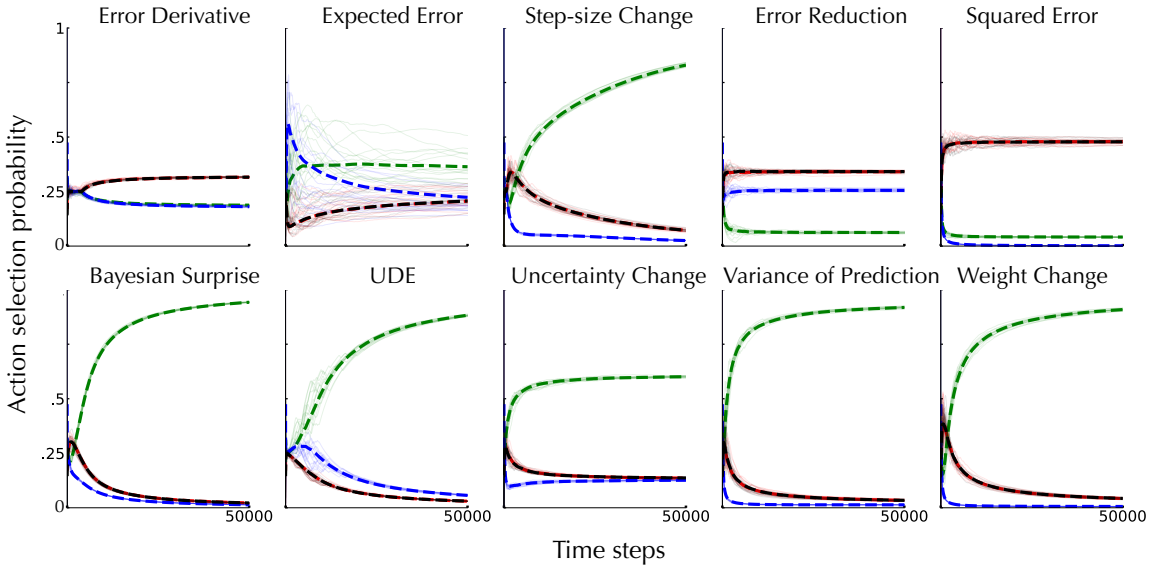

Figure 15: **Behavior** in the **Drifter-Distractor** problem, with **Introspective Learners** where the behavior is learned using Dynamic Thompson Sampling.

The qualitative conclusions are similar, shown in Figures 14 and 15. The primary difference is that the DTS algorithm is less likely to settle on uniform random behavior; rather it is more likely to exhibit a preference. This could be because the algorithm is inherently designed to identify the best action, under drift, whereas the Gradient Bandit samples according to preferences. The Gradient Bandit algorithm is less greedy: if two actions have similar action preferences, even if one is clearly higher than the other, then the Gradient Bandit will spend time selecting both actions. Deterministic methods select the action with the highest value. Despite this difference, we observed the same basic behaviors and the same qualitative differences amongst different intrinsic rewards. The only noteworthy difference is the behavior with *Error Derivative* and non-introspective learners: with DTS this reward no longer induces the expected action selection. As we saw, *Error Derivative* only worked for a narrow range of its hyper-parameters in Experiment One, and so it is not surprising that it was not a stable result.

9. Experiment Two: Switched Drifter-Distractor Problem

Our second experiment is similar to the first except we introduce an unexpected change in the target distributions to tax the reward function's ability to help keep track of the relevant actions. Our first experiment reveals that several intrinsic rewards could help the Gradient Bandit algorithm ignore unhelpful actions and focus on the one corresponding to a drifting prediction target. The intrinsic rewards that were most helpful are based on moving estimates of the error or the variance of the prediction itself.

To further evaluate these rewards we introduce a simple unpredictable change in the targets. For the first 50,000 steps the task is the same as Experiment One, then the targets suddenly switch according to Table 2. The ideal behavior before the *switch* should be the same as Experiment One: choose the actions corresponding the constant and distractor targets until their error is reduced, then focus on the drifter. After the switch, two of the targets drift. So the ideal behavior should focus on those two actions equally after some initial transient period due to the change.

This task is partially observable by design. The idea is to simulate a situation where the learning system encounters an unexpected change. The question is how does the sudden change interact with each reward function's internal estimates? Can we find a setting of the smoothing parameter and window lengths the help the intrinsic reward identify the appropriate actions?

9.1 Results with Non-introspective Learners

Figure 16 summarizes the behavior of the Gradient Bandit with several intrinsic reward functions. In the first phase of the experiment, the behavior is fairly similar to Experiment One. *Weight Change*, *Squared Error*, *Expected Error*, *Bayesian Surprise* and the variance-based rewards perform almost the same, though there is reduced selection of the action for the drifter target. Notably both *Error Derivative* and *UDE* exhibit substantially different behavior. *Error Derivative* induces uniform action preferences for the entire duration of the experiment. *UDE* has trouble inducing a strong preference between the drift and constant targets in phase one. In the second phase *UDE* incorrectly focuses action selection on only one of the two drifter targets.

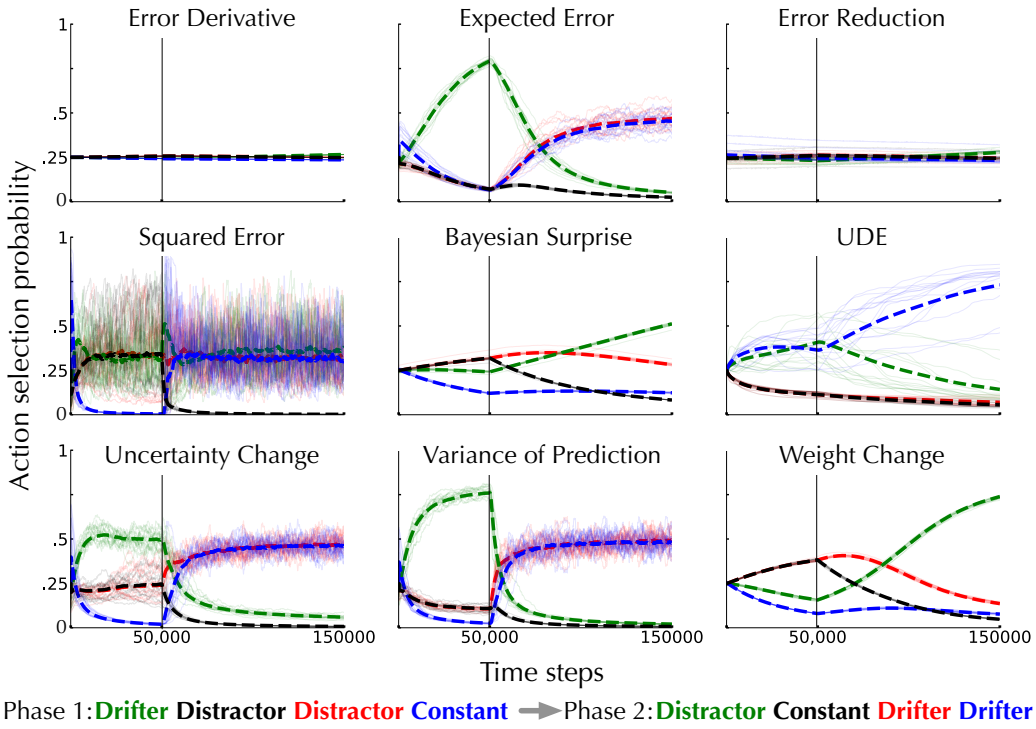

Figure 16: **Behavior** in the **Switched Drifter-Distractor** problem with **Non-introspective Learners**. Each subplot corresponds to the behavior of the Gradient Bandit with a different intrinsic reward. The black vertical line in each subplot marks the time of the switch.

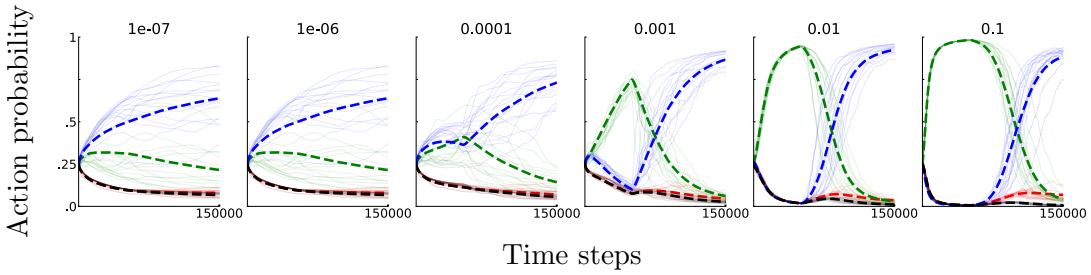

Figure 17: The impact of varying the smoothing parameter β with the **UDE** reward in the **Switched Drifter-Distractor** problem with **Non-introspective Learners**. Large β—more emphasis on recent errors—induces the correct behavior in phase one, but not in phase two.

Let us take a closer look first at *UDE* to get a better sense of why its behavior is so different in Experiment Two. Figure 17 illustrates how the behavior changes as of function of the smoothing parameter of *UDE*. As in Experiment One, a large value of β induces strong preference for the drifter target in phase one. In phase two, however, mainly one of the drifter targets is selected, resulting in higher RMSE. The best RMSE is achieved

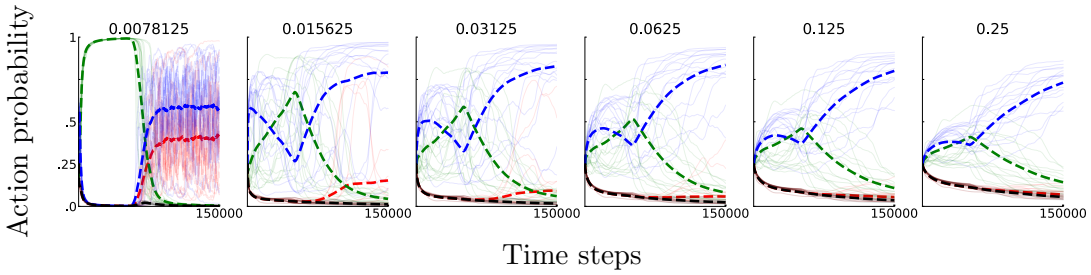

Figure 18: The impact of varying the step-size parameter α_p of LMS with the **UDE** reward in the **Switched Drifter-Distractor** problem with **Non-introspective Learners**. Interestingly very small α_p—slow prediction learning—induces the action preferences closer to what we expect. Unfortunately when $\alpha_p = 0.0078125$ the RMSE is 1.376, compared to a RMSE of 0.313 when $\alpha_p = 0.25$.

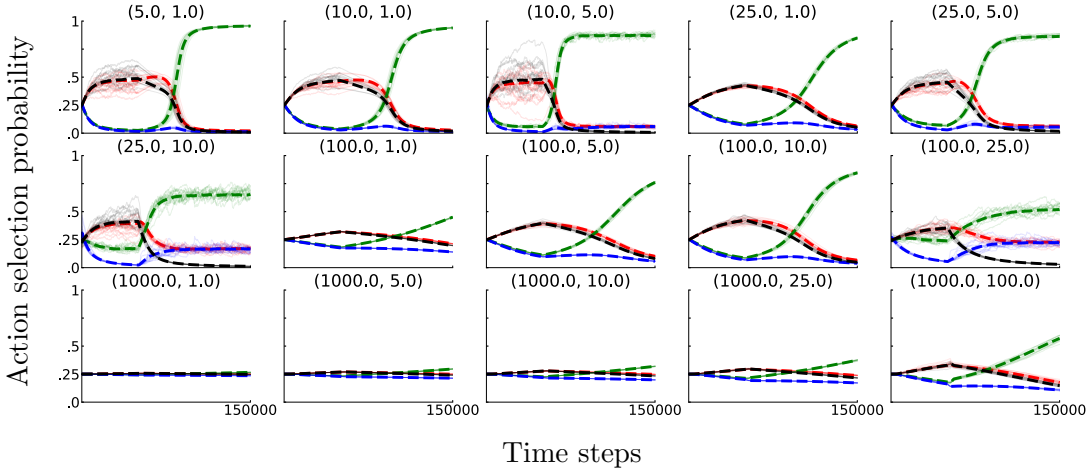

Figure 19: The impact of varying the window length parameters η and τ of **Error Derivative** reward in the **Switched Drifter-Distractor** problem with **Non-introspective Learners**.

with smaller β that (a) over-selects the action for the constant target in phase one, (b) over-selects the distractor target in phase two, and (c) under-selects both actions for the drifter targets in phase two. Figure 18 illustrates how the behavior changes as of function of the step-size parameter α_p. We see the same phenomenon here with *UDE* that we saw with *Error Derivative* in Experiment One. If α_p is small, then the prediction of the drifter target is less accurate and thus the behavior induced by *UDE* favors the drifter target, but results in much higher overall error.

The *Error Derivative* has similar problems. The behavior of the Gradient Bandit with *Error Derivative* does not exhibit the expected shape under any configuration of the window length parameters we tested (see Figure 19). If we inspect the behavior changes as a function of the step-size parameter α_p (Figure 20), then again we observe that slower learning is

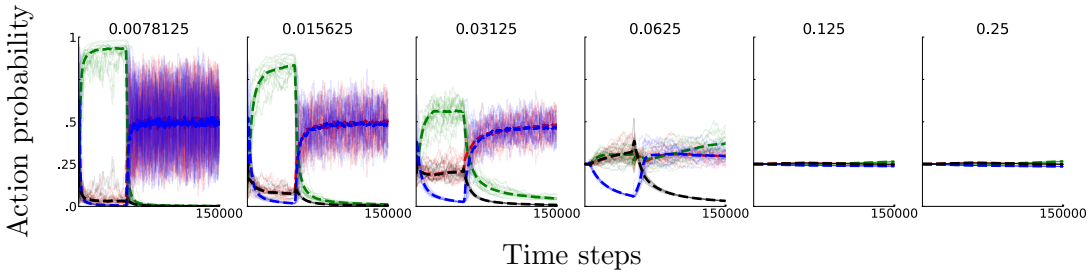

Figure 20: The impact of the step-size α_p with the **Error Derivative** reward in the **Switched Drifter-Distractor** problem with **Non-introspective Learners**. When $\alpha_p = 0.0078125$ the behavior looks somewhat similar to the ideal behavior but the RMSE is 0.650. With $\alpha_p = 0.25$ the RMSE is 0.300, the behavior induced is uniform.

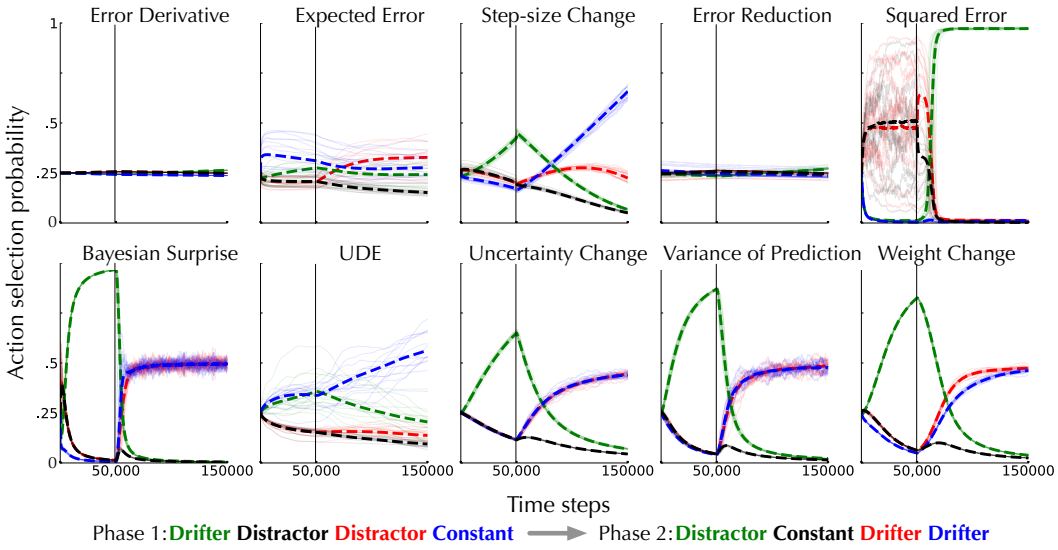

Figure 21: **Behavior** in the **Switched Drifter-Distractor** problem with **Introspective Learners**, with the Gradient Bandit.

required for the *Error Derivative* reward to be highest for the drifter target, at the cost of high RMSE.

9.2 Results with Introspective Learners

The results of our second experiment, this time with Autostep, are summarized in Figure 21. As before we plot the action selection probabilities to summarize the behavior. As before, *Weight Change* and *Surprise* rewards now induce near-ideal action selection. As in Experiment One, *Expected Error* and *Error Derivative* do not induce the expected behavior because Autostep increases the effective learning rate of the LMS predictors. *UDE* induces similar behavior on this problem with or without Autostep.

10. Experiment Three: Jumpy Eight-Action Problem

Our final experiment, in the Jumpy Eight-Action problem, quantitatively compares the best performing intrinsic rewards in a setting where the behavior agent should prefer several different actions. To achieve good performance, the agent must continually sample three actions, with different probabilities, and eventually ignore three noisy targets and two constant targets. The jumpy target follows a pattern increasing towards 50 and then decreasing toward -50 repeatedly. The idea was to design a problem where the intrinsic rewards based on variance would induce the wrong preferences over arms. The jumpy target exhibits high variance, as do the other two drifter targets. Intrinsic rewards based exclusively on the variance of the predictions might over-reward the jumpy-target action.

This final experiment provides the quantitative comparison for performance, so we include a baseline uniform random behavior. There are eight targets and three of them continually drift. The drift is fast enough that wasting action selection will result in high error. By the design of the experiment, therefore, uniform behavior should not be optimal, and we should expect some of the intrinsic rewards to significantly outperform uniform random behavior. We exclude rewards that did not succeed in inducing useful behavior in Experiments One and Two, and so have little hope to provide improvements here. This includes *Error Reduction* and *Squared Error* rewards.

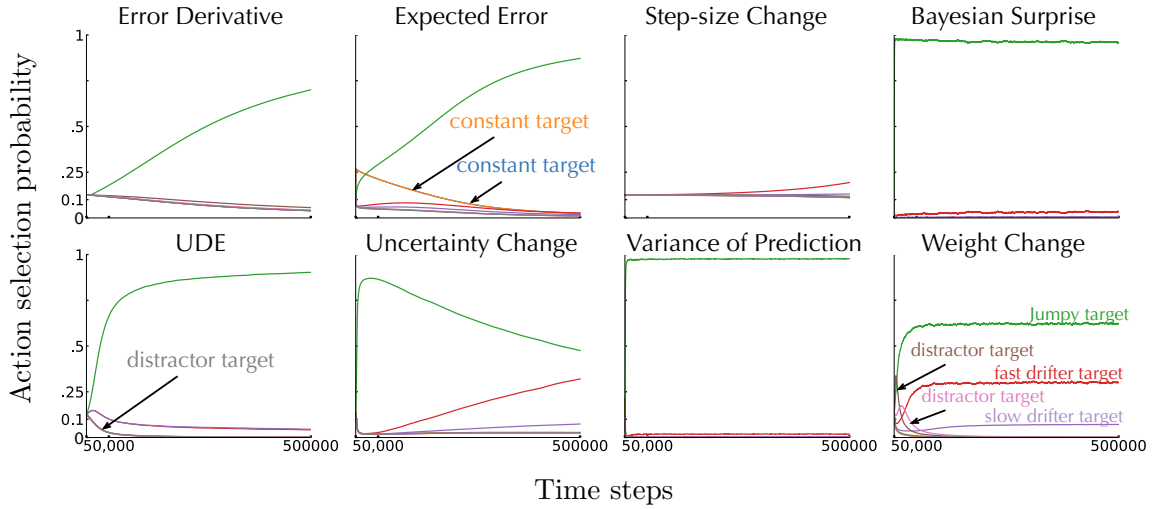

Figure 22: **Behavior** in the **Jumpy Eight-Action** problem with **Introspective Learners**. Plotted is the probability of selecting each action versus time (averaged over 400 runs), with Autostep prediction learners.

Figure 22 shows the behavior of the Gradient Bandit with eight different intrinsic rewards. We report the results using the hyper-parameter settings that resulted in the lowest total RMSE. We expect the behavior should initially select all the actions, eventually learning to ignore the constant and distractor targets once they have been learned. Let us inspect the behavior with *Weight Change*. Initially, the actions corresponding to the high-error constant targets are preferred. Next the action corresponding to the distractor target (with variance 1.0) is strongly preferred for a while, as is the other other distractor target (with

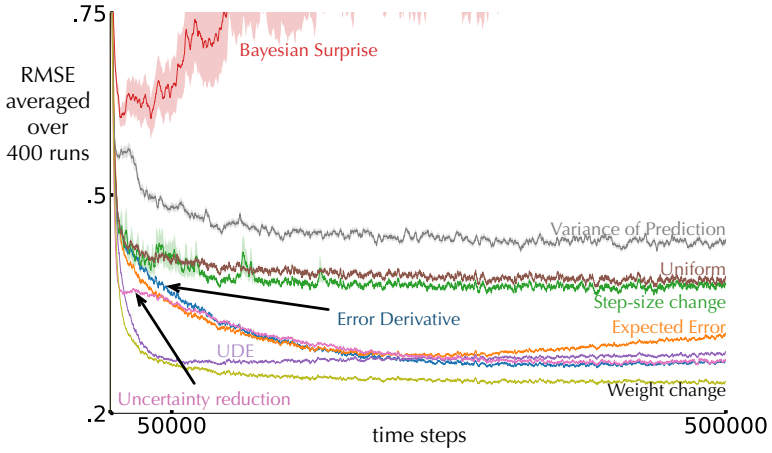

Figure 23: **RMSE** for the **Jumpy Eight-Action** problem with **Introspective Learners**. Standard error of 400 runs is shown as shading around each line. We include the performance of the uniform behavior to provide a baseline of comparison. The behavior agent based on *Weight Change* learned the fastest and achieved the lowest RMSE at the end of learning. Most of the differences in the final RMSE are significantly different, in that the 95% confidence intervals do not overlap (2 times the standard error), except for the difference between *Uniform* and *Step-size Change* and between *Uncertainty Reduction* and *Error Derivative*.

variance 0.5). The remaining noisy target (with variance 0.1), is easy to quickly learn and is not preferred after the beginning. Eventually the steady-state behavior selects the action for the jumpy target most often, followed by the actions for the two drifter targets. As you can see in Figure 23, the behavior induced by *Weight Change* significantly outperforms uniform action selection, and all other intrinsic rewards we tested.

Several rewards cause the behavior to over select the action for the jumpy target, at the cost of under selecting the actions for the two drifter targets. *Bayesian Surprise* and *Variance of Prediction* eventually select the jump action nearly 100% of the time. In Figure 23 we can see that both these rewards results in high RMSE compared to uniform action selection. This makes sense for intrinsic rewards based on variance because the variance of the jumpy target is significantly higher than the others. *Bayesian Surprise* likely did not work because our distributional assumptions are wrong—recall we assumed all targets were Gaussian with a moving estimate of the variance of the target, to provide an approximate Bayesian surprise for non-Bayesian learners. This is simply a problem with using Bayesian surprise outside its intended use-case—it should be used with Bayesian learners. But, as motivated above, we wanted to verify if a more explicit form of Bayesian surprise, rather than the much simpler *Weight Change*, could provide benefit, even for non-Bayesian learners.

Many of the other rewards also over selected the action for the jumpy target, though less excessively. *Expected Error* takes a long time to induce a preference for the jumpy target, initially favoring the high-variance ones. *UDE* induces a preference for the correct actions, but over-rewards the action for the jumpy target—especially toward the end of the experiment. This resulted in increasing RMSE over time, as we see in the upward

trend of the error of *UDE* in Figure 23. *Uncertainty Change* eventually induces near-ideal action selection, though it starts by over selecting the action for the jumpy target. *Error Derivative*, after 500,000 steps, also appears to be trending towards over selection of the action for the jumpy target, while *Step-size Change* seems to be getting it all wrong. *Weight Change* induces a preference for all three non-stationary targets after initially selecting all the actions. We cannot say if this is the optimal behavior, but it does cause the prediction learners to learn faster and achieve lower error compared with all other intrinsic rewards we tested.

11. Adapting the Behavior of a Horde of Demons

The ideas and algorithms promoted in this paper may be more impactful when combined with policy-contingent, temporally-extended prediction learning. Imagine learning hundreds or thousands of off-policy predictions from a single stream of experience, as in the Unreal (Jaderberg et al., 2016) and Horde (Sutton et al., 2011) architectures. In these settings, the behavior must balance overall speed of learning with prediction accuracy. That is, balancing action choices that generate off-policy updates across many predictions, with the need to occasionally choose actions in almost total agreement with one particular policy. In general we cannot assume that each prediction target is independent as we have done in this paper; selecting a particular sequence of actions might generate useful off-policy updates to several predictions in parallel (White et al., 2012). There have been several promising investigations of how intrinsic rewards might benefit single (albeit complex) task learning (see Pathak et al., 2017; Hester and Stone, 2017; Tang et al., 2017; Colas et al., 2018; Pathak et al., 2019). However, to the best of our knowledge, no existing work has studied adapting the behavior based on intrinsic rewards of a model-based or otherwise parallel off-policy learning system.

Simple intrinsic reward schemes and the concept of an introspective learning system should scale nicely to these more ambitious problem settings. We could swap our stateless LMS learners for Q-learning with experience replay, or gradient temporal difference learning (Maei et al., 2010). The *Weight Change* reward could be computed for each predictor with computation linear in the number of weights. It would be natural to learn the behavior policy with an average-reward actor-critic architecture, instead of the gradient bandit algorithm used here. Finally, the notion of an introspective learner still simply requires that each prediction learner can adapt its rate of learning. This can be achieved with quasi second order methods like Adam (Kingma and Ba, 2015), or extensions of the Autostep algorithm to the case of temporal difference learning and function approximation (Kearney et al., 2018, 2019; Günther et al., 2020; Jacobsen et al., 2019). It is not possible to know if the ideas advocated in this paper will work well in a large-scale off-policy prediction learning architecture like Horde, however they will certainly scale up.

In this paper we focused on (non-Bayesian) mean prediction learners. This is a natural first step, as most prediction algorithms in reinforcement learning form point estimates (e.g., value functions). It is possible, however, to go beyond mean predictions, both for our testbed and for the Horde setting. For example, it is straightforward to use distributional prediction learners—those that estimate the distribution over the targets. For mean prediction learners we used the normed difference between weights before and after an update; for distributional

learners, we could use a KL divergence between the distribution over targets before and after an update. Note that this differs from Bayesian surprise, which is the KL divergence between the distributions over the *weights* before and after an update. We expect similar conclusions with distributional prediction learners, as conceptually the definition of *Weight Change* simply uses a modified difference measure—a norm versus a divergence. The extension to Bayesian prediction learners—for either the mean or distribution over targets—is more complex, though improvements in Bayesian neural networks make it a more feasible line of investigation.

Maximizing intrinsic reward as presented in this paper is not a form of exploration, its the objective of the learning system—the agent must explore in order to maximize the intrinsic reward. The intrinsic rewards do not provide a bonus to help improve exploration. In our stateless prediction task, sufficient exploration was provided by the stochastic behavior policy. This will not always be the case, and additional exploration will likely be needed. Efficient exploration is an open problem in reinforcement learning. Combining the ideas advocated in this paper with exploration bonuses or planning could work well, but this topic is left to future work.

12. Conclusion

The goal of this work was to systematically investigate intrinsic rewards for a multi-prediction setting. This paper has three main contributions. The first is a new benchmark suite for comparing intrinsic rewards. Our bandit-like task requires the learning system to demonstrate several important capabilities: avoiding dawdling on noisy outcomes, tracking non-stationary outcomes, and seeking actions for which consistent learning progress is possible. Second, we provide a survey of intrinsically motivated learning systems, and empirically investigated 10 different analogs of well-known intrinsic reward schemes. Finally, we found that simple intrinsic rewards based on amount of learning, can induce effective behavior—avoiding classic degenerative behavior—if the base prediction learners are introspective. Introspective prediction learners can decide for themselves when learning is done. Previous work focused on designing more complex intrinsic rewards to mitigate bad behavior. Our results suggest the opposite: we should focus on designing better learners and use simple (ideally parameter-free) intrinsic rewards. We found that intrinsic rewards based on amount of learning—like *Weight Change*—can perform well in problems specifically designed to distract the learning system. This work provides several new insights into the strengths and weakness of different intrinsic reward mechanisms, and may provide guidance for constructing larger more complex intrinsically motivated reinforcement learning systems where an extensive and systematic study like ours is not feasible.

Acknowledgments

We would like to thank our generous funders for supporting this work, specifically the NSERC Discovery grant program and CIFAR for funding the Alberta Machine Intelligence Institute and the CIFAR Canada AI Chairs program. We would like to thank Rich Sutton for his ideas and insights that shaped this project early on. We would also like to thank

Andrew Jacobsen, Andrew Patterson and Tor Lattimore for helpful comments on the text. Finally, we would like to thank our colleagues at the Reinforcement Learning and Artificial Intelligence Lab at the University of Alberta and DeepMind Alberta for providing an exciting and stimulating environment for research.

References

Achiam, J. and Sastry, S. (2017). Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv:1703.01732*.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. (2017). Hindsight experience replay. In *Advances in Neural Information Processing Systems*.

Antos, A., Grover, V., and Szepesvári, C. (2008). Active learning in multi-armed bandits. In *International Conference on Algorithmic Learning Theory*.

Audibert, J.-Y., Munos, R., and Szepesvári, C. (2007). Variance estimates and exploration function in multi-armed bandit. In *CERTIS Research Report 07–31*.

Bajcsy, R., Aloimonos, Y., and Tsotsos, J. K. (2018). Revisiting active perception. *Autonomous Robots*.

Balcan, M.-F., Beygelzimer, A., and Langford, J. (2009). Agnostic active learning. *Journal of Computer and System Sciences*.

Banerjee, A., Merugu, S., Dhillon, I. S., and Ghosh, J. (2005). Clustering with Bregman divergences. *Journal of Machine Learning Research*.

Barto, A. G. (2013). Intrinsic motivation and reinforcement learning. In *Intrinsically Motivated Learning in Natural and Artificial Systems*.

Barto, A. G., Mirolli, M., and Baldassarre, G. (2013). Novelty or surprise? *Frontiers in Psychology*, 4:907.

Barto, A. G. and Simsek, O. (2005). Intrinsic motivation for reinforcement learning systems. In *Yale Workshop on Adaptive and Learning Systems*.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*.

Berseth, G., Geng, D., Devin, C., Finn, C., Jayaraman, D., and Levine, S. (2019). SMiRL: Surprise minimizing RL in dynamic environments. *arXiv:1912.05510*.

Besbes, O., Gur, Y., and Zeevi, A. (2014). Stochastic multi-armed-bandit problem with non-stationary rewards. In *Advances in Neural Information Processing Systems*, pages 199–207.

Brafman, R. I. and Tennenholtz, M. (2002). R-max – A general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*.

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., and Efros, A. A. (2018). Large-scale study of curiosity-driven learning. In *International Conference on Learning Representations*.

Bylinskii, Z., DeGennaro, E., Rajalinghamd, R., Ruda, H., Zhang, J., and Tsotsos, J. (2015). Towards the quantitative evaluation of visual attention models. *Vision Research*.

Chentanez, N., Barto, A. G., and Singh, S. P. (2005). Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems*.

Clement, B., Roy, D., Oudeyer, P.-Y., and Lopes, M. (2015). Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining*.

Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*.

Colas, C., Sigaud, O., and Oudeyer, P.-Y. (2018). GEP-PG: Decoupling exploration and exploitation in deep reinforcement learning algorithms. In *International Conference on Machine Learning*.

de Abril, I. M. and Kanai, R. (2018). Curiosity-driven reinforcement learning with homeostatic regulation. In *International Joint Conference on Neural Networks*.

Degris, T. and White, A. (2020). Automatic step-size adaptation for reinforcement. *Manuscript in Preparation*.

Garivier, A. and Moulines, E. (2011). On upper-confidence bound policies for switching bandit problems. In *International Conference on Algorithmic Learning Theory*.

Golovin, D. and Krause, A. (2011). Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*.

Gordon, G. and Ahissar, E. (2011). Reinforcement active learning hierarchical loops. In *International Joint Conference on Neural Networks*.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. In *International Conference on Machine Learning*.

Günther, J., Ady, N. M., Kearney, A., Dawson, M. R., and Pilarski, P. M. (2020). Examining the use of Temporal-Difference Incremental Delta-Bar-Delta for real-world predictive knowledge architectures. *Frontiers in Robotics and AI*.

Gupta, N., Granmo, O.-C., and Agrawala, A. (2011). Thompson sampling for dynamic multi-armed bandits. In *International Conference on Machine Learning and Applications and Workshops*.

Haber, N., Mrowca, D., Fei-Fei, L., and Yamins, D. L. (2018). Learning to play with intrinsically-motivated self-aware agents. In *Advances in Neural Information Processing Systems*.

Hester, T. and Stone, P. (2017). Intrinsically motivated model learning for developing curious robots. *Artificial Intelligence*.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. (2016). Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*.

Itti, L. and Baldi, P. F. (2006). Bayesian surprise attracts human attention. In *Advances in Neural Information Processing Systems*.

Jacobsen, A., Schlegel, M., Linke, C., Degris, T., White, A., and White, M. (2019). Meta-descent for online, continual prediction. In *AAAI Conference on Artificial Intelligence*.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. (2016). Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations*.

Karl, M., Soelch, M., Becker-Ehmck, P., Benbouzid, D., van der Smagt, P., and Bayer, J. (2017). Unsupervised real-time control through variational empowerment. *arXiv:1710.05101*.

Kearney, A., Veeriah, V., Travnik, J., Pilarski, P. M., and Sutton, R. S. (2019). Learning feature relevance through step size adaptation in temporal-difference learning. *arXiv:1903.03252*.

Kearney, A., Veeriah, V., Travnik, J. B., Sutton, R. S., and Pilarski, P. M. (2018). TIDBD: Adapting temporal-difference step-sizes through stochastic meta-descent. *arXiv:1804.03334*.

Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *International Conference on Machine Learning*.

Konyushkova, K., Sznitman, R., and Fua, P. (2017). Learning active learning from data. In *Advances in Neural Information Processing Systems*.

Kulkarni, T. D., Narasimhan, K., Saeedi, A., and Tenenbaum, J. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in Neural Information Processing Systems*.

Lattimore, T. and Szepesvári, C. (2019). Cleaning up the neighborhood: A full classification for adversarial partial monitoring. In *International Conference on Algorithmic Learning Theory*.

Little, D. Y.-J. and Sommer, F. T. (2013). Learning and exploration in action-perception loops. *Frontiers in Neural Circuits*.

Liu, Y.-E., Mandel, T., Brunskill, E., and Popovic, Z. (2014). Trading off scientific knowledge and user learning with multi-armed bandits. In *International Conference on Educational Data Mining*.

Lopes, M., Lang, T., Toussaint, M., and Oudeyer, P.-Y. (2012). Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Advances in Neural Information Processing Systems*.

Maei, H. R., Szepesvári, C., Bhatnagar, S., and Sutton, R. S. (2010). Toward off-policy learning control with function approximation. In *International Conference on Machine Learning*.

Mahmood, A. R., Sutton, R. S., Degris, T., and Pilarski, P. M. (2012). Tuning-free step-size adaptation. In *International Conference on Acoustics, Speech and Signal Processing*.

Martin, J., Sasikumar, S. N., Everitt, T., and Hutter, M. (2017). Count-based exploration in feature space for reinforcement learning. In *International Joint Conference on Artificial Intelligence*.

Meuleau, N. and Bourgine, P. (1999). Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*.

Mirolli, M. and Baldassarre, G. (2013). Functions and mechanisms of intrinsic motivations. In *Intrinsically Motivated Learning in Natural and Artificial Systems*. Springer.

Modayil, J., White, A., and Sutton, R. S. (2014). Multi-timescale nexting in a reinforcement learning robot. *Adaptive Behavior*.

Orseau, L., Lattimore, T., and Hutter, M. (2013). Universal knowledge-seeking agents for stochastic environments. In *International Conference on Algorithmic Learning Theory*, pages 158–172. Springer.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. (2007). Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*.

Pathak, D., Gandhi, D., and Gupta, A. (2019). Self-supervised exploration via disagreement. In *International Conference on Machine Learning*.

Patten, T., Martens, W., and Fitch, R. (2018). Monte Carlo planning for active object classification. *Autonomous Robots*.

Péré, A., Forestier, S., Sigaud, O., and Oudeyer, P.-Y. (2018). Unsupervised learning of goal spaces for intrinsically motivated goal exploration. In *International Conference on Learning Representations*.

Riedmiller, M., Hafner, R., Lampe, T., Neunert, M., Degrave, J., Van de Wiele, T., Mnih, V., Heess, N., and Springenberg, J. T. (2018). Learning by playing-Solving sparse reward tasks from scratch. In *International Conference on Machine Learning*.

Santucci, V. G., Baldassarre, G., and Mirolli, M. (2013). Which is the best intrinsic motivation signal for learning multiple skills? *Frontiers in Neurorobotics*.

Satsangi, Y., Lim, S., Whiteson, S., Oliehoek, F., and White, M. (2020). Maximizing information gain in partially observable environments via prediction rewards. In *International Conference on Autonomous Agents and Multiagent Systems*.

Satsangi, Y., Whiteson, S., Oliehoek, F. A., and Spaan, M. T. (2018). Exploiting submodular value functions for scaling up active perception. *Autonomous Robots*.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. (2015). Universal value function approximators. In *International Conference on Machine Learning*.

Schembri, M., Mirolli, M., and Baldassarre, G. (2007). Evolving childhood's length and learning parameters in an intrinsically motivated reinforcement learning robot. In *International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*.

Schmidhuber, J. (1991a). Curious model-building control systems. In *International Joint Conference on Neural Networks*.

Schmidhuber, J. (1991b). A possibility for implementing curiosity and boredom in model-building neural controllers. In *International Conference on Simulation of Adaptive Behavior: From Animals to Animats*.

Schmidhuber, J. (2008). Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. In *Workshop on Anticipatory Behavior in Adaptive Learning Systems*.

Schossau, J., Adami, C., and Hintze, A. (2016). Information-theoretic neuro-correlates boost evolution of cognitive systems. *Entropy*, 18(1).

Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Silver, D., van Hasselt, H., Hessel, M., Schaul, T., Guez, A., Harley, T., Dulac-Arnold, G., Reichert, D., Rabinowitz, N., Barreto, A., and Degris, T. (2017). The Predictron: End-to-end learning and planning. In *International Conference on Machine Learning*.

Slivkins, A. and Upfal, E. (2008). Adapting to a changing environment: The brownian restless bandits. In *International Conference on Learning Theory*.

Stadie, B. C., Levine, S., and Abbeel, P. (2015). Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv:1507.00814*.

Still, S. and Precup, D. (2012). An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, 131(3):139–148.

Sutton, R. S. (1992). Adapting bias by gradient descent: An incremental version of delta-bar-delta. In *AAAI Conference on Artificial Intelligence*, pages 171–176.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction, 2nd Edition*. MIT press.

Sutton, R. S., Koop, A., and Silver, D. (2007). On the role of tracking in stationary environments. In *International Conference on Machine Learning*.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *International Conference on Autonomous Agents and Multiagent Systems*.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*.

Szita, I. and Lorincz, A. (2008). The many faces of optimism: A unifying approach. In *International Conference on Machine learning*.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, X., Duan, Y., Schulman, J., DeTurck, F., and Abbeel, P. (2017). #Exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*.

Vigorito, C. M. (2016). *Intrinsically Motivated Exploration in Hierarchical Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst.

White, A., Modayil, J., and Sutton, R. S. (2012). Scaling life-long off-policy learning. In *International Conference on Development and Learning and Epigenetic Robotics*.

White, A., Modayil, J., and Sutton, R. S. (2014). Surprise and curiosity for big data robotics. In *AAAI Workshop on Sequential Decision-Making with Big Data*.

White, A. M. (2015). *Developing a Predictive Approach to Knowledge*. PhD thesis, University of Alberta.

Chapter 3

Conclusion and Future Work

In this work we propose a testbed that simplifies the problem of intrinsic motivation for the learning of many predictions down to its most simple essence—what action should the agent take to maximize learning. The testbed is setup to evaluate common issues that intrinsically motivated agents face. The constant signal represents continuous signals that can quickly be learned. The noisy signal represents signals which have a learnable expected value, but are extremely noisy. This is commonly seen as the noisy tv problem that intrinsic rewards need to be able to handle. The drifting signal represents the signals that an agent could receive that are learnable but require the agent to regularly track and update its predictions in order to keep them accurate. Together these signals comprise a testbed that allows us to investigate the core of each intrinsic reward.

The testbed itself has three main environments: Drifter Distractor, Switched Drifter Distractor, and the Jumpy Eight Action problem. Each of these utilize the three signal types to test the different reward mechanisms. In the proposed testbed we used these three environments to evaluate 14 representative intrinsic rewards. This work empirically looked at how each method behaved when trying to decide what action to take to best learn to predict the signals the agent was receiving. The chosen rewards covered different classes of representative methods commonly proposed in intrinsic motivation literature.

We propose that a simple method, weight change, can be a simple but effective method for producing an intrinsic reward to drive behaviour. This

is shown throughout the three testbeds to be effective at learning the right behavior and tracking the signals to achieve strong performance on predicting each of the signals. While extensive tuning is often needed in order for intrinsic rewards to perform well, weight change has few parameters and performs robustly across many parameter combinations.

3.1 Future Work

This work represents a first step in evaluating model-free methods of intrinsic motivation and using introspective learners to improve the agent's ability to adapt behaviour based on learning. Building on this there are a number of paths forward to continue in this line of research.

The first is to expand this beyond a one state setting to a full reinforcement learning environment, with multiple states and prediction demons that will all be learned over time. The effect of off-policy learning methods, and introspective learners in that context remains to be fully evaluated - even in the tabular setting. Scaling this research up to a Horde (Sutton et al., 2011) of off-policy prediction learners, each learning a variety of signals over multiple timescales, could first be done in the tabular setting to evaluate if the results of the different learning methods hold. Similar to this testbed the demons should have representative signals the demon is learning to predict—some noisy, some constant, and some drifting over time. A good intrinsically motivated agent will need to combine the learning of all of these demons in producing its intrinsic reward.

Moving beyond the tabular setting, the effect of function approximation methods needs to be evaluated. Evaluating how these methods scale in the linear and non-linear settings can be important. Methods such as tile-coding (Sutton and Barto, 2018) has been shown to scale efficiently while learning on a robot powered by a standard laptop (see A. White, 2015). Using an artificial neural network as a function approximator, with multiple predictions (Jaderberg et al., 2016) has been shown to be an effective combination in complex vision problems. How function approximation affects both the demons and the

behaviour learning itself needs to be evaluated. Whether or not weight change is still effective when moving from the linear to non-linear setting could be of particular interest.

The effect of different methods for the introspective learners remains to be explored. Different meta-learning methods adapt their learning rate in different ways. Different step-size adaptation methods, such as RMSProp (Tieleman and Hinton, 2012), were tried and they had similar results to Auto. However a full exploration into the effect of different classes of introspective learners needs to be evaluated to understand if there are any major differences between methods.

The tradeoff between exploration and exploitation returns when we scale up the setting. The agent must explore to find the areas it can exploit to learn the most in. Exploration is an ongoing area of research in reinforcement learning and the effect of different exploration methods on the agents learning remains to be explored. The relative importance of different policies may also come into play. Demons will likely have different policies and the agent may care more about learning some policies over others.

The focus of this paper was on model-free methods of reinforcement learning. In order to be fully effective at adapting its behaviour for learning, an agent will likely need to move into the model based reinforcement learning setting (Sutton and Barto, 2018). Using a model could allow the agent to "think" about parts of the state space it hasn't visited, or perhaps hasn't visited in a while. These areas could be more interesting to the agent and could be visited to improve the agent's predictions. The effect of mechanisms such as experience replay (Lin, 1992) or Dyna (Sutton, 1991) on an intrinsically motivated agent is a further area of research.

When moving out of simulation into the real world the ability to run large parameter sweeps become infeasible. The ability for intrinsic reward mechanisms to rely on as few parameter sweeps as possible becomes very beneficial in a real world setting. We show in the paper that some methods were able to perform well without introspective learners. However, this required a large number of parameters to be swept to find the right combination that worked

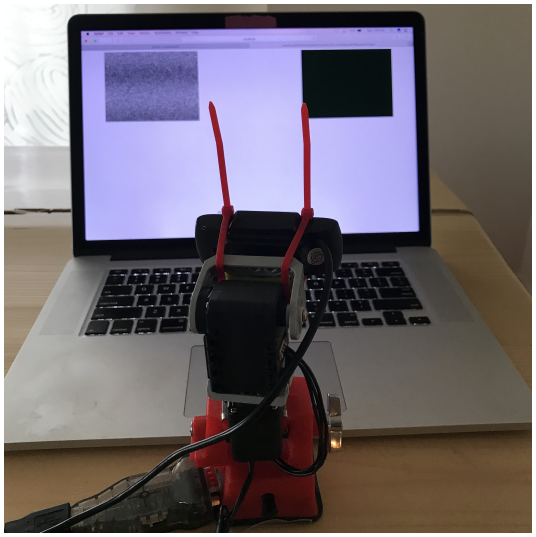

Figure 3.1: Drifter Distractor problem with Dynamixel robot. The robot chooses where to look on the screen and tries to predict the pixel values the camera captures.

for that intrinsic reward. This has been another line of research that I have begun to follow. An initial step was the implementation of the Drifter Distractor, and Switched Drifter Distractor, on a robot (Figure 3.1). The robot consisted of two dynamixel servos with a webcam attached. At each step the robot chose where on a laptop screen it wanted to focus its camera. On the screen was a noisy tv section, a constant section, and a section whose color drifted over time. The robot seeks to predict the pixel values coming in through the camera. The implementation details remained largely the same to the experiments in the paper—gradient bandit for behavior and Autostep for introspective learners. Here running even 5000 steps taking 100ms/step took over eight minutes per run. Weight change, using the proposed standard Autostep parameters (see Mahmood et al., 2012), proved effective in this setting with little to no parameter sweeps. Most of the other methods proved too difficult to parameter tune to reach the same level of performance as those with introspective learners. In order for intrinsic reward to be effective in the real world, we would like to understand the methods that can allow for as little parameter tuning as possible.

Finally, showing the effects of the combination of all of these components

on the ultimate behaviour and performance of an agent remains to be seen. The drive to continue to grow competence in predicting and controlling its world should lead an agent to display ever more complex and masterful skills (as shown by R. White, 1959). Both showing the qualitative behaviour improvement, and that quantitative reward improvement of this type of agent remains an outstanding challenge.

References

Achiam, J. & Sastry, S. (2017). Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv:1703.01732*.

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., ... Zaremba, W. (2017). Hindsight experience replay. In *Advances in neural information processing systems*.

Antos, A., Grover, V., & Szepesvári, C. (2008). Active learning in multi-armed bandits. In *International conference on algorithmic learning theory*.

Audibert, J.-Y., Munos, R., & Szepesvári, C. (2007). Variance estimates and exploration function in multi-armed bandit. In *Certis research report 07–31*.

Auer, P., Lim, S. H., & Watkins, C. (2011). Models for autonomously motivated exploration in reinforcement learning. In *International conference on algorithmic learning theory*.

Bajcsy, R., Aloimonos, Y., & Tsotsos, J. K. (2018). Revisiting active perception. *Autonomous Robots*.

Balcan, M.-F., Beygelzimer, A., & Langford, J. (2009). Agnostic active learning. *Journal of Computer and System Sciences*.

Baldassarre, G. & Mirolli, M. (2013). Intrinsically motivated learning systems: An overview. In *Intrinsically motivated learning in natural and artificial systems*.

Banerjee, A., Merugu, S., Dhillon, I. S., & Ghosh, J. (2005). Clustering with Bregman divergences. *Journal of Machine Learning Research*.

Barto, A. G. (2013). Intrinsic motivation and reinforcement learning. In *Intrinsically motivated learning in natural and artificial systems*.

Barto, A. G., Mirolli, M., & Baldassarre, G. (2013). Novelty or surprise? *Frontiers in Psychology*, *4*, 907.

Barto, A. G. & Simsek, O. (2005). Intrinsic motivation for reinforcement learning systems. In *Yale workshop on adaptive and learning systems*.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., & Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. In *Advances in neural information processing systems*.

Berseth, G., Geng, D., Devin, C., Finn, C., Jayaraman, D., & Levine, S. (2019). SMiRL: Surprise minimizing RL in dynamic environments. *arXiv:1912.05510*.

Besbes, O., Gur, Y., & Zeevi, A. (2014). Stochastic multi-armed-bandit problem with non-stationary rewards. In *Advances in neural information processing systems* (pp. 199–207).

Brafman, R. I. & Tennenholtz, M. (2002). R-max – A general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research.*

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., & Efros, A. A. (2018). Large-scale study of curiosity-driven learning. In *International conference on learning representations.*

Bylinskii, Z., DeGennaro, E., Rajalinghamd, R., Ruda, H., Zhang, J., & Tsotsos, J. (2015). Towards the quantitative evaluation of visual attention models. *Vision Research.*

Chentanez, N., Barto, A. G., & Singh, S. P. (2005). Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems.*

Choshen, L., Fox, L., & Loewenstein, Y. (2018). Dora the explorer: Directed outreaching reinforcement action-selection. In *International conference on learning representations.*

Clement, B., Roy, D., Oudeyer, P.-Y., & Lopes, M. (2015). Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining.*

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research.*

Colas, C., Sigaud, O., & Oudeyer, P.-Y. (2018). GEP-PG: Decoupling exploration and exploitation in deep reinforcement learning algorithms. In *International conference on machine learning.*

de Abril, I. M. & Kanai, R. (2018). Curiosity-driven reinforcement learning with homeostatic regulation. In *International joint conference on neural networks.*

Degris, T. & White, A. (2020). Automatic step-size adaptation for reinforcement. *Manuscript in Preparation.*

Garivier, A. & Moulines, E. (2011). On upper-confidence bound policies for switching bandit problems. In *International conference on algorithmic learning theory.*

Gehring, C. & Precup, D. (2013). Smart exploration in reinforcement learning using absolute temporal difference errors. In *International conference on autonomous agents and multiagent systems.*

Golovin, D. & Krause, A. (2011). Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research.*

Gopnik, A., Meltzoff, A. N., & Kuhl, P. K. (1999). *The scientist in the crib: Minds, brains, and how children learn.* William Morrow & Co.

Gordon, G. & Ahissar, E. (2011). Reinforcement active learning hierarchical loops. In *International joint conference on neural networks.*

Graves, A., Bellemare, M. G., Menick, J., Munos, R., & Kavukcuoglu, K. (2017). Automated curriculum learning for neural networks. In *International conference on machine learning*.

Günther, J., Ady, N. M., Kearney, A., Dawson, M. R., & Pilarski, P. M. (2020). Examining the use of Temporal-Difference Incremental Delta-Bar-Delta for real-world predictive knowledge architectures. *Frontiers in Robotics and AI*.

Günther, J., Kearney, A., Ady, N. M., Dawson, M. R., & Pilarski, P. M. (2019). Meta-learning for predictive knowledge architectures: A case study using TIDBD on a sensor-rich robotic arm. In *International conference on autonomous agents and multiagent systems*.

Gupta, N., Granmo, O.-C., & Agrawala, A. (2011). Thompson sampling for dynamic multi-armed bandits. In *International conference on machine learning and applications and workshops*.

Haber, N., Mrowca, D., Fei-Fei, L., & Yamins, D. L. (2018a). Emergence of structured behaviors from curiosity-based intrinsic motivation. *arXiv:1802.07461*.

Haber, N., Mrowca, D., Fei-Fei, L., & Yamins, D. L. (2018b). Learning to play with intrinsically-motivated self-aware agents. In *Advances in neural information processing systems*.

Hester, T. & Stone, P. (2017). Intrinsically motivated model learning for developing curious robots. *Artificial Intelligence*.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., & Abbeel, P. (2016). Variational information maximizing exploration. In *Advances in neural information processing systems*.

Itti, L. & Baldi, P. F. (2006). Bayesian surprise attracts human attention. In *Advances in neural information processing systems*.

Jacobsen, A., Schlegel, M., Linke, C., Degris, T., White, A., & White, M. (2019). Meta-descent for online, continual prediction. In *AAAI conference on artificial intelligence*.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., & Kavukcuoglu, K. (2016). Reinforcement learning with unsupervised auxiliary tasks. In *International conference on learning representations*.

Karl, M., Soelch, M., Becker-Ehmck, P., Benbouzid, D., van der Smagt, P., & Bayer, J. (2017). Unsupervised real-time control through variational empowerment. *arXiv:1710.05101*.

Kearney, A., Veeriah, V., Travnik, J. B., Sutton, R. S., & Pilarski, P. M. (2018). TIDBD: Adapting temporal-difference step-sizes through stochastic meta-descent. *arXiv:1804.03334*.

Kearney, A., Veeriah, V., Travnik, J., Pilarski, P. M., & Sutton, R. S. (2019). Learning feature relevance through step size adaptation in temporal-difference learning. *arXiv:1903.03252*.

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Kingma, D. P. & Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *International conference on machine learning*.

Konyushkova, K., Sznitman, R., & Fua, P. (2017). Learning active learning from data. In *Advances in neural information processing systems*.

Kulkarni, T. D., Narasimhan, K., Saeedi, A., & Tenenbaum, J. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*.

Lattimore, T. & Szepesvári, C. (2019). Cleaning up the neighborhood: A full classification for adversarial partial monitoring. In *International conference on algorithmic learning theory*.

Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, *8*(3-4), 293–321.

Linke, C., Ady, N. M., White, M., Degris, T., & White, A. (2020). Adapting behavior via intrinsic reward: A survey and empirical study. *J. Artif. Intell. Res. 69*, 1287–1332.

Little, D. Y.-J. & Sommer, F. T. (2013). Learning and exploration in action-perception loops. *Frontiers in Neural Circuits*.

Liu, Y.-E., Mandel, T., Brunskill, E., & Popovic, Z. (2014). Trading off scientific knowledge and user learning with multi-armed bandits. In *International conference on educational data mining*.

Lopes, M., Lang, T., Toussaint, M., & Oudeyer, P.-Y. (2012). Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Advances in neural information processing systems*.

Maei, H. R., Szepesvári, C., Bhatnagar, S., & Sutton, R. S. (2010). Toward off-policy learning control with function approximation. In *International conference on machine learning*.

Mahmood, A. R., Sutton, R. S., Degris, T., & Pilarski, P. M. (2012). Tuning-free step-size adaptation. In *International conference on acoustics, speech and signal processing*.

Martin, J., Sasikumar, S. N., Everitt, T., & Hutter, M. (2017). Count-based exploration in feature space for reinforcement learning. In *International joint conference on artificial intelligence*.

Matiisen, T., Oliver, A., Cohen, T., & Schulman, J. (2017). Teacher-student curriculum learning. *arXiv:1707.00183*.

Mellor, J. & Shapiro, J. (2013). Thompson sampling in switching environments with bayesian online change detection. In *Artificial intelligence and statistics*.

Meuleau, N. & Bourgine, P. (1999). Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*.

Mirolli, M. & Baldassarre, G. (2013). Functions and mechanisms of intrinsic motivations. In *Intrinsically motivated learning in natural and artificial systems*. Springer.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529–533.

Modayil, J., White, A., & Sutton, R. S. (2014). Multi-timescale nexting in a reinforcement learning robot. *Adaptive Behavior*.

O'Donoghue, B., Osband, I., Munos, R., & Mnih, V. (2018). The uncertainty bellman equation and exploration. In *International conference on machine learning*.

Orseau, L., Lattimore, T., & Hutter, M. (2013). Universal knowledge-seeking agents for stochastic environments. In *International conference on algorithmic learning theory* (pp. 158–172). Springer.

Oudeyer, P.-Y., Kaplan, F., & Hafner, V. (2007). Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*.

Pathak, D., Agrawal, P., Efros, A. A., & Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*.

Pathak, D., Gandhi, D., & Gupta, A. (2019). Self-supervised exploration via disagreement. In *International conference on machine learning*.

Patten, T., Martens, W., & Fitch, R. (2018). Monte Carlo planning for active object classification. *Autonomous Robots*.

Péré, A., Forestier, S., Sigaud, O., & Oudeyer, P.-Y. (2018). Unsupervised learning of goal spaces for intrinsically motivated goal exploration. In *International conference on learning representations*.

Riedmiller, M., Hafner, R., Lampe, T., Neunert, M., Degrave, J., Van de Wiele, T., ... Springenberg, J. T. (2018). Learning by playing-Solving sparse reward tasks from scratch. In *International conference on machine learning*.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv:1609.04747*.

Santucci, V. G., Baldassarre, G., & Mirolli, M. (2013a). Which is the best intrinsic motivation signal for learning multiple skills? *Frontiers in Neurorobotics*.

Santucci, V. G., Baldassarre, G., & Mirolli, M. (2013b). Which is the best intrinsic motivation signal for learning multiple skills? *Frontiers in Neurorobotics*.

Satsangi, Y., Lim, S., Whiteson, S., Oliehoek, F., & White, M. (2020). Maximizing information gain in partially observable environments via prediction rewards. In *International conference on autonomous agents and multiagent systems*.

Satsangi, Y., Whiteson, S., Oliehoek, F. A., & Spaan, M. T. (2018). Exploiting submodular value functions for scaling up active perception. *Autonomous Robots*.

Schaul, T., Horgan, D., Gregor, K., & Silver, D. (2015). Universal value function approximators. In *International conference on machine learning*.

Schembri, M., Mirolli, M., & Baldassarre, G. (2007). Evolving childhood's length and learning parameters in an intrinsically motivated reinforce-

ment learning robot. In *International conference on epigenetic robotics: Modeling cognitive development in robotic systems*.

Schmidhuber, J. (1991a). A possibility for implementing curiosity and boredom in model-building neural controllers. In *International conference on simulation of adaptive behavior: From animals to animats*.

Schmidhuber, J. (1991b). Curious model-building control systems. In *International joint conference on neural networks*.

Schmidhuber, J. (2008). Driven by compression progress: A simple principle explains essential aspects of subjective beauty, novelty, surprise, interestingness, attention, curiosity, creativity, art, science, music, jokes. In *Workshop on anticipatory behavior in adaptive learning systems*.

Schmidhuber, J. (2010). Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*.

Schossau, J., Adami, C., & Hintze, A. (2016). Information-theoretic neurocorrelates boost evolution of cognitive systems. *Entropy*, *18*(1).

Settles, B. (2009). *Active learning literature survey*. University of Wisconsin-Madison Department of Computer Sciences.

Silver, D., van Hasselt, H., Hessel, M., Schaul, T., Guez, A., Harley, T., ... Degris, T. (2017). The Predictron: End-to-end learning and planning. In *International conference on machine learning*.

Slivkins, A. & Upfal, E. (2008). Adapting to a changing environment: The brownian restless bandits. In *International conference on learning theory*.

Stadie, B. C., Levine, S., & Abbeel, P. (2015). Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv:1507.00814*.

Still, S. & Precup, D. (2012). An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, *131*(3), 139–148.

Suddarth, S. C. & Kergosien, Y. (1990). Rule-injection hints as a means of improving network performance and learning time. In *European association for signal processing workshop* (pp. 120–129). Springer.

Sutton, R. S. (1991). Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, *2*(4), 160–163.

Sutton, R. S. (1992). Adapting bias by gradient descent: An incremental version of delta-bar-delta. In *AAAI conference on artificial intelligence* (pp. 171–176).

Sutton, R. S. & Barto, A. G. (2018). *Reinforcement learning: An introduction, 2nd edition*. MIT press.

Sutton, R. S., Koop, A., & Silver, D. (2007). On the role of tracking in stationary environments. In *International conference on machine learning*.

Sutton, R. S., McAllester, D. A., Singh, S. P., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems* (pp. 1057–1063).

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., & Precup, D. (2011). Horde: A scalable real-time architecture for learning

knowledge from unsupervised sensorimotor interaction. In *International conference on autonomous agents and multiagent systems*.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*.

Szita, I. & Lorincz, A. (2008). The many faces of optimism: A unifying approach. In *International conference on machine learning*.

Tamar, A., Di Castro, D., & Mannor, S. (2016). Learning the variance of the reward-to-go. *Journal of Machine Learning Research*.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, X., Duan, Y., ... Abbeel, P. (2017). #EXPLORATION: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*.

Tieleman, T. & Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning.

Vigorito, C. M. (2016). *Intrinsically motivated exploration in hierarchical reinforcement learning* (Doctoral dissertation, University of Massachusetts Amherst).

White, A. (2015). *Developing a predictive approach to knowledge* (Doctoral dissertation, University of Alberta).

White, A. M. (2015). *Developing a predictive approach to knowledge* (Doctoral dissertation, University of Alberta).

White, A., Modayil, J., & Sutton, R. S. (2012). Scaling life-long off-policy learning. In *International conference on development and learning and epigenetic robotics*.

White, A., Modayil, J., & Sutton, R. S. (2014). Surprise and curiosity for big data robotics. In *AAAI workshop on sequential decision-making with big data*.

White, R. (1959). Motivation reconsidered: The concept of competence. *Psychological review*, *66*(5), 297.