Reasoning and Learning in Real-World Systems for Long-Term Autonomy (LTA)
Papers from the AAAI 2018 Fall Symposium

Predictions, Surprise, and Predictions of Surprise in General Value Function Architectures

Johannes Günther [1], Alex Kearney[1], Michael R. Dawson[1], Craig Sherstan[1] and Patrick M. Pilarski[1, 2]
[1]Departments of Computing Science and Medicine, University of Alberta, Edmonton, Alberta, Canada; [2]DeepMind
{gunther, pilarski}@ualberta.ca

Abstract

Effective life-long deployment of an autonomous agent in a complex environment demands that the agent has some model of itself and its environment. Such models are inherently predictive, allowing an agent to predict the consequences of its actions. In this paper, we demonstrate the use of General Value Functions (GVFs) for learning and representing such a predictive model on a robotic arm. Our model is composed of three types of signals: (1) predictions of sensorimotor signals, (2) measures of surprise using Unexpected Demon Error (UDE) and (3) predictions of surprise. In a proof-of-principle experiment, where the robot arm is manually perturbed in a recurring pattern, we show that each perturbation is detected as a jump in the surprise signal. We demonstrate that the recurrence of these perturbations not only can be learned, but can be anticipated. We propose that introspective signals like surprise and predictions of surprise might serve as a rich substrate for more abstract predictive models, improving an agent's ability to continually and independently learn about itself and its environment to fulfill its goals.

Introduction

Autonomous agents facing long-term deployment may encounter many challenges when interacting with the real world. The conditions of the environment and the agent itself may change over time. Further, it is impossible for engineers to fully anticipate all that such an agent must know ahead of time. The only way to overcome these shortcomings autonomously is for an agent to independently and continuously learn about itself and the environment in terms of its ongoing sensorimotor experience. One potential way to learn and represent information is to use predictions and predictive knowledge (Clark 2013). To this end, predictive models, such as General Value Functions (GVFs) (Sutton et al. 2011), present a method by which an agent might construct and represent information from its own experience. Such models should enable the agent to predict upcoming events and the outcomes of its actions, key information for successfully acting on its own. The usefulness of machine-made predictions has recently proven to be beneficial for various complex problems, even in challenging and changing environments. Examples include, but are not limited to, industrial laser welding (Günther et al. 2016), artificial limbs (Pilarski et al. 2013; Sherstan, Modayil, and Pilarski 2015) and robot navigation (Kahn et al. 2017). However, most research has focused on the use and prediction of signals generated by the environment (i.e., signals originating outside the agent, from the world or its physical body) and not *internal signals* (here defined as signals relating to the computational workings of the learning machine itself).

While knowledge about the environment is valuable for an autonomous agent to successfully interact with the environment on its own, further insight might be required to evaluate the consequences of the agent's actions. As stated by Schultz and Dickinson (2000, p. 476), "In general terms, learning can be viewed as the acquisition of predictions of outcomes (reward, punishment, behavioral reactions, *external stimuli, internal states*)" [emphasis added]. It is therefore necessary to not only learn about external sources of information but also about internal ones. Many authors have looked at using various internally generated metrics to drive exploration (White and White 2010; Gehring and Precup 2013), adapt algorithm parameters (White and White 2016; 2010; Sakaguchi and Takano 2004), adapt to changes in the reward function (White and White 2010), and minimize risk (Tamar, Castro, and Mannor 2016). Further, Sherstan et al. (2016) argued that internally generated signals, such as learning errors and statistical measures, should be made available to the agent as state information, enabling an agent to learn to make better decisions on its own. Learning external *and* internal signals by employing GVFs will result in a large number of predictions. Recent work has demonstrated the ability to learn a large number of online predictions for the sensor values of a mobile robot (Modayil, White, and Sutton 2014); in Pilarski and Sherstan (2016), a precursor to the present work, $\sim$18k GVFs were deployed in real time on the data stream of a robotic prosthesis.

In this paper we build on this prior work to provide an example of how GVFs can be used to make thousands of predictions about both external and internal signals at different time scales on a real-world problem domain. Using a proof-of-principle experiment, we learn thousands of predictions about incoming sensor readings provided by the sensors of a robotic artificial limb. Furthermore, we investigate measures that are related to these predictions to gain knowledge about the internal state of the prosthesis. One particular measure that we investigate in detail is the Unexpected Demon

Error (UDE) (White 2015). The UDE provides information about the comparison of the current prediction error to an average of previous errors. It can be seen as a measure of surprise, as it takes previous experiences into account and will only increase when current experience significantly differs from previous experience. Such a differing experience might be due to changing conditions, either in the environment or in the agent itself, providing important knowledge about the agent's functioning within said environment. We furthermore learn predictions about the UDE to provide the agent with a sense of how much surprise it might experience.

As a main contribution of the present work, we propose that predictions of raw perceptual data from an agent's data stream, along with sensations and predictions of surprise with respect to this data stream, can be used as a platform on which to build more powerful and more abstract predictive models of an agent's operation and interactions with its world. In the remainder of this paper, we demonstrate that such introspective information can be learned in a tractable, scalable way for use during long-term operation.

General Value Functions

As suggested, General Value Functions (GVFs) are a means to learn predictive knowledge (Sutton et al. 2011). A GVF v is defined in terms of the return, G_t. The return at time t is defined as $G_t = \sum_{k=0}^{\infty} \gamma^k C_{t+k+1}$, where C is the cumulant and γ is the discount rate. The cumulant is the signal of interest. The discount rate describes how future cumulants are weighted in the return. In the simplest case, $\gamma = 0$, the return is equal to the next cumulant. This setting is called myopic. As γ increases and approaches 1, future cumulants contribute more to the return. For $\gamma = 1$, the return is undiscounted and all future cumulants contribute equally.

A GVF v is defined as $v(s; \pi, \gamma, C) = \mathbb{E}_\pi[G_t | s_t = s]$. It maps from a state s to the expected return, given the agent follows the policy π and starts in the state s. The policy π specifies the behavior by providing the probability of taking an action a for a given state s. Together, the three parameters π, γ and C define what a GVF is about and are called *question parameters* (White 2015).

A way to learn General Value Functions is temporal-difference (TD) learning (Sutton 1988). TD learning allows for online and incremental computation of the value function by using estimates to make updates. This property makes it ideal to compute a sufficiently big number of GVFs to represent all information of interest. In this work, the value function is approximated by the inner product of a binary feature vector $x(s)$ that represents the sensor readings and a learned weight vector w. The value for a state is therefore computed as $v(s) = w^\top x(s)$. To update the value function, the TD error δ is computed after each time step as stated in line 3 in Algorithm 1. The TD error is then used to update the weights by taking a step towards the new estimate, based on the step size $0 < \alpha$. To potentially speed up learning by assigning credit to previously visited states, eligibility traces z are used. These traces decay according to the decay rate $\lambda \in [0, 1]$. The whole algorithm can be found in Algorithm 1 and an extensive introduction to TD learning can be found in Sutton and Barto (2018). The parameters α and λ are called *answer parameters*, as they define how the GVFs are learned. A collection of GVFs is called a *Horde* (Sutton et al. 2011).

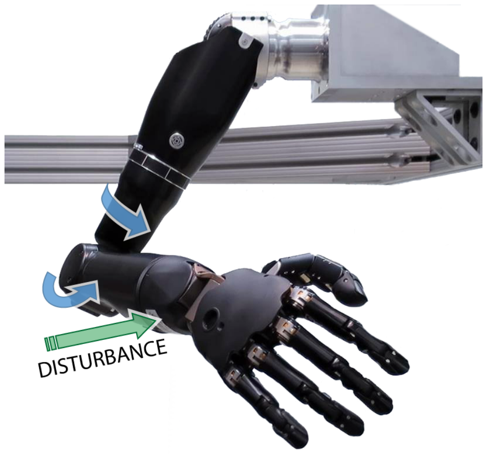

Figure 1: The Modular Prosthetic Limb (MPL) used for the experiments. The arrows indicate the nature of the repeated disturbance imposed during the experiment. The green arrow indicates the direction of the provided perturbation, while the blue arrows indicate the resulting joint movement.

Algorithm 1 TD(λ)

```
1: Initialize vectors $z \in 0^n$ and $w \in 0^n$; initialize a small
   scalar $\alpha$; observe state $s$
2: Repeat for each observation $s'$ and cumulant $C$:
3:     $\delta \leftarrow C + \gamma w^\top x(s') - w^\top x(s)$
4:     For $i = 1, 2, \cdots, n$:
5:         $z_i \leftarrow z_i \gamma \lambda + x_i(s)$
6:         $w_i \leftarrow w_i + \alpha_i \delta z_i$
7:         $s \leftarrow s'$
```

Unexpected Demon Error

One of the error measures we are most interested in for this paper is the Unexpected Demon Error (UDE) (White 2015). It provides a measure for unexpected changes in a signal due to changes in the environment. Mathematically, the UDE is calculated as

$$\text{UDE} = \left| \frac{\bar{\delta}^\beta}{\sqrt{\text{var}(\delta)} + \epsilon} \right|, \qquad (1)$$

where $\bar{\cdot}^\beta$ is a moving average over the TD error δ and ϵ is a small constant to prevent division by zero. During learning, small changes in the TD error are to be expected, as the learner updates the value function and acquires knowledge about the world. The way the UDE is defined, it will neither react to the regular occurring learning nor to random noise, as both are considered in the mean and the variance of the TD error. The UDE will only significantly increase if the TD error behaves significantly differently due to changes in the

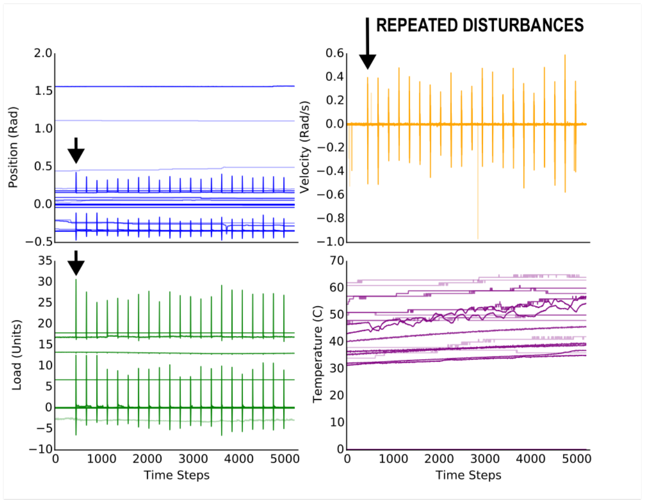

Figure 2: Decoded percept data from the robot over the 20min duration of the experiment. The 21 disturbances are clearly identifiable in data from the position, velocity and the load sensors. The temperature sensors show an increasing temperature over the experiment, with additional increases for some sensors due to the perturbations.

environment, triggering an unexpected amount of error, as the name implies. UDE can therefore very well be seen as a measure of genuine surprise and might provide important insight into the agent's learning.

Implementation and Experiments

Experimental Setup

The robotic arm used in the experiment is the Modular Prosthetic Limb (MPL v3) (Bridges, Para, and Mashner 2011), which can be seen in Figure 1. The MPL includes 26 articulated joints in the shoulder, elbow, wrist, and hand. Complex coordinated movements are possible via custom motors that provide up to 17 active degrees of freedom. Each of these motors is outfitted with sensors for load, position, temperature, and current. Additionally, each fingertip of the hand is instrumented with a 3-axis accelerometer and 14 pad pressure sensor arrays to provide a detailed sensor stream that is transmitted over a Controller Area Network bus and then sent to the control computer as User Datagram Protocol (UDP) broadcast packets. As one of the most advanced research prostheses currently available, the MPL provides a unique sensory-rich platform for testing GVF architectures.

When using sensor data for machine learning, the first decision to make is the choice of how infromation is represented to the learning machine. As in related work in the Atari Learning Environment (ALE) (Bellemare et al. 2013), we follow an approach of presenting the learning machine with raw binary data, prior to this data being decoded into real-valued numbers, states, or observations. Similar to the random access memory data used for learning on the ALE by Bellemare et al. (2013) and subsequent follow-up studies, the UDP packets that convey the MPL's data stream provide

Bit/Line	Data / sensor readings
0-39	Header
40-903	Position
904-1767	Velocity
1768-2631	Load
2632-3495	Temperature
3496-3520	Footer & Checksum

Table 1: UDP packet structure: the position of data in the MPL binary percepts (each bit corresponding to the respective line in the figures that follow).

a straight-forward, interpretable testbed with different dynamics for a system to learn about, including bits that do and do not vary with respect to perturbations, pseudo-random bits, and non-stationary bits that drift or slowly change activity over time (summarized in Table 1). As such, these bits—and not the real-valued signals that they help transmit—are the primary focus of our explorations below.

The experiment was conducted for approximately 20 minutes, resulting in 5217 time steps in total. For each time step, predictions, UDE and predictions of the UDE were computed and calculated, which took 0.23s per step on average. For the duration of the experiment, the arm was controlled to be compliant but motionless, streaming data regarding its current sensor readings: the position, velocity, load and temperature of all actuators. The experiment began with the arm remaining motionless in its resting position. This enabled the predictors to reach an acceptable level of accuracy. After two minutes, the arm was manually perturbed by an experimenter as indicated by the arrows in Figure 1, resulting in a change in the sensor readings for position, velocity and load. The arm then was allowed to move automatically back to its original resting position according to its compliance settings. The arm was perturbed in a similar way every minute for the remainder of the experiment, resulting in a recurring yet noisy pattern of sensory information. In total, 21 disturbances were recorded. The stream of 108 decoded sensor readings from this pattern are shown in Figure 2.

Predictive Architecture and Algorithmic Implementation

To create all signals of interest (predictions, surprise and predictions of surprise), at least two different hordes are necessary—one for the predictions and the surprise (Horde 1) and one for the predictions of the surprise (Horde 2). A third Horde (Horde 3) predicts the future values of the 108 decoded sensor values, and was added simply as an example of how predictions of raw data can be used as a substrate for more complex predictive questions. These Hordes are structured into two layers, as shown in Figure 3. Each Horde has two inputs, the cumulant, denoted as C, and the state vector, denoted as X. The first predictive layer receives the binary sensor signals from the MPL as state information and cumulant. As the signal consists of 3520 signals, there are 3520 GVFs in this Horde, one for each possible cumulant. The outputs of the first Horde are myopic ($\gamma = 0$) predictions about the binary inputs, and the predictions' UDE.

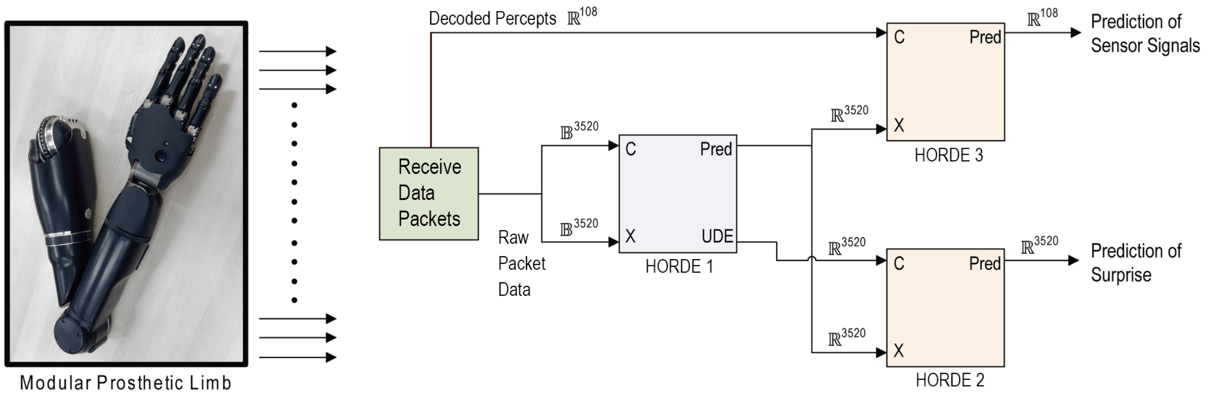

Figure 3: The prediction architecture used in the experiments. The sensor stream from the MPL on the left side is received over the network as a 3520 bit UDP packet, subsequently decoded into 108 floating point signals. These 3520 bits are delivered directly as both cumulant (C) and state (X) for Horde 1. The output of the first Horde is then fed into Horde 2 and 3 as the state X used in predicting the 108 decoded sensor signals and also to make predictions about the UDE (surprise) of Horde 1.

The outputs of the first predictive layer (Horde 1) are then used as inputs for the second predictive layer. There are two independent Horde architectures present in this layer (Horde 2 and 3). Horde 2 in the second layer receives the predictions from the first layer as state inputs $x(s)$ and the UDE as cumulants. As this layer predicts 3520 cumulants, it again consists of 3520 GVFs. Its outputs are predictions about the UDE of the first layer, with a discount rate of $\gamma = 0.999$, which corresponds to a prediction of 1000 steps into the future. The discount rate was chosen such that the predictions can reliably learn about the imposed perturbations, which are about 260 time steps apart.

Horde 3 receives predictions from the first layer as its state representation input $x(s_t)$, and its cumulants are the floating point decodings of sensor readings from the MPL. There are 108 GVFs in this Horde. Its outputs are predictions about the sensor signals, based on a discount or termination signal $\gamma = 0.9$, which corresponds to 10 time steps into the future.

For all predictive layers in this architecture, the same eligibility trace decay rate $\lambda = 0.99$ was chosen as a standard intermediate value of λ (Sutton and Barto 2018).

Experimental Results

To provide further intuition, we created synthetic data to demonstrate the expected behavior of the internal signals for a variety of potential external signal types, shown in Figure 4. Subplot (a) shows a potential data stream, including signals that do not change, recurring patterns, and random noise. Subplot (b) shows the predictions and should therefore match subplot (a), if the predictions are accurate. Subplot (c), which shows the UDE, should only spike for surprising changes in the original data and not for consistent noise. While the TD error for noise will constantly change, the UDE should only increase for the first occurrence of noise, as it keeps track of previous TD errors and will therefore expect TD errors of the same magnitude. It should furthermore not react to signals that are constant.

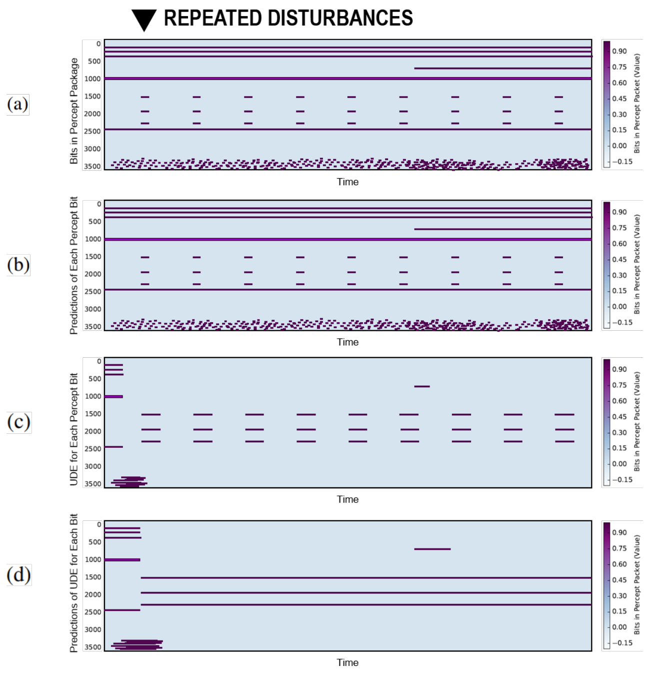

Figure 4: Simplified plots for the ideal relationship between (a) binary data, (b) predictions, (c) UDE and (d) predictions of UDE with $\gamma = 0.999$ for synthetic data.

The UDE should, however, react to recurring patterns, as the short moving average will forget about these signals over time. Subplot (d), which shows the predictions for the UDE, should show a longer activation where the UDE is active. The predictions of the UDE are only consistently active for the recurring pattern, as the predictions are consistently reinforced. The actual recorded data for all sensors over the whole duration of the experiment is shown in Figure 5.

Bits and Bit Predictions

To provide insight into the experimental results, Figure 5(a) shows the binary features of the data stream for all sensors. These features are created from the sensor readings by plotting the full contents of the UDP sensor packet received from the robot arm. Table 1 shows the line numbers of each sensor value in Figures 5, 6 and 7.

Purple bits are highly active, while light blue bits are not active, as indicated by the legend. Some bits do not change their value over time—this corresponds to constant sensor readings. For example, most sensor readings from the hand will be constant, as it is not moving during the experiment. Figure 6(a) shows a zoom in on the position and velocity bits for 200 time steps. As expected, most of the sensor stream is constant. However, around time step 4090 some of the values significantly change, as a result of the perturbation to the prosthetic arm. Other bits will be constantly changing. This may be due to sensor noise or due to inherently shifting signals, e.g. increasing temperature, or, in the case of the load sensors (Figure 7(a)), because the actuators need to keep the arm in place, resulting in the load sensors frequently being active and their values varying by small amounts.

As the predictions for the bits are myopic, they should ideally be the same as the actual bits. Figure 5(b) clearly shows that the predictions for the bits that show a constant behavior are identical. Even when zoomed in as shown in Figure 6(b) and 7(b), the bits that are constant are matching the predictions. The changing bits are not as trivial to predict. Bits that change randomly should in fact not be predictable and the predicted value should be distributed around the expectation, i.e. 0.5. Such random behavior can be seen in Figure 7(b) for some of the bits related to load between lines 2000 and 2050. For the position and velocity, shown in Figure 6(b) however, the predictions clearly map the disturbance, as can be seen by the changing predicted value, as the perturbation occurs around time step 4090.

UDE and UDE Predictions

To provide a meaningful measure of surprise, the UDE should show its highest activation both at the beginning of the experiment, when the sensor readings are new, and upon the disturbances, as the readings will significantly change when the arm is perturbed. Figure 5(c) clearly reveals the experimental design. The subplot shows the repetitive pattern of the arm displacement around lines 100, 950 and 1800. These binaries correspond to the position, velocity and load, respectively. Every time the arm is perturbed, the UDE significantly spikes as the sensor readings change. Noise still shows up in the UDE plot, but the intensity is lower, due to the UDE taking previous errors into account. The dampening of the noise is clearly displayed in Figure 6(d). Between lines 900 and 1000, some of the velocity binary features are highly volatile, as seen in subplot (a). The UDE, as shown in subplot (d), in comparison, only spikes twice, around time steps 4000 and 4090, where the actual displacements occur. Figure 7 elucidates a further aspect of the functionality of UDE: After the perturbation around time step 4310, the TD error in subplot (c) stays quite volatile until around time step 4460. The UDE decreases over this period and only spikes again when the TD error suddenly drops and stays low.

When looking at the predictions in Figure 5(d), it can be seen that the predictions about the UDE are not significantly active until the first disturbance occurs. After that, they are consistently high for the binaries that are affected by the perturbations. As the termination signal $\gamma = 0.999$ allows the predictions to consider 1000 time steps in expectation, the UDE predictions learn about the reoccurring movements and correctly predict the spikes in UDE. Figures 6(e) and 7(e) show in detail that the predictions anticipate that there will be changes in UDE due to perturbations, and at the same time filter the impact of UDE spikes that are not directly related, e.g. in lines 2000 to 2050 in Figure 7(e).

Discussion

This work presented the use of a predictive architecture to capture important information about the sensor stream of a prosthetic limb. The raw sensor stream of the MPL was received as binary values and served as an input to the first predictive layer that learned to predict these inputs in a myopic way and produced the Unexpected Demon Error (UDE) as a measure of the surprise with regard to the inputs.

In the original binary sensor data, the temporal structure of the perturbations is hidden by a significant amount of noise and general changes in the sensor values, for example due to changes in temperature. The (myopic) predictions of the sensor values match the original sensor values quite well for a large amount of the readings. However, some binary features behave randomly or almost randomly, resulting in predictions that are not accurate.

The UDE, however, is able to capture the perturbations and their effects on the position, velocity and load sensors. Each time the arm is manually moved, the surprise for each sensor peaks and falls afterwards. The UDE can therefore be seen as a valuable measure to inform the system about changes in its own functioning. Furthermore, the predictions about the UDE are consistently high after the first displacement, effectively capturing knowledge about the recurring pattern. At the same time, UDE and the predictions about the UDE are capable of filtering the noisy sensor readings to some degree, providing a better distinction between the perturbations and the normal, unperturbed running. For example, the UDE is consistently low for the checksum and the temperature, but spikes for signals that are impacted by perturbations of the arm. Intuitively, the system has learned about the potential changes in its functioning and to some degree can predict and expect these perturbations.

The internal signals that are generated by the suggested architecture can not only be thought of as direct inputs for a potential controller but can be looped in as additional context to improve the accuracy of these and other internal signals. For example, one could imagine using the predictions about surprise as additional context, incorporating unexpected motions into the agent's knowledge to improve its predictions of the sensor values. Including internal signals may improve the representation of the system, enabling the agent to learn more complex dependencies about itself and improve its performance autonomously.

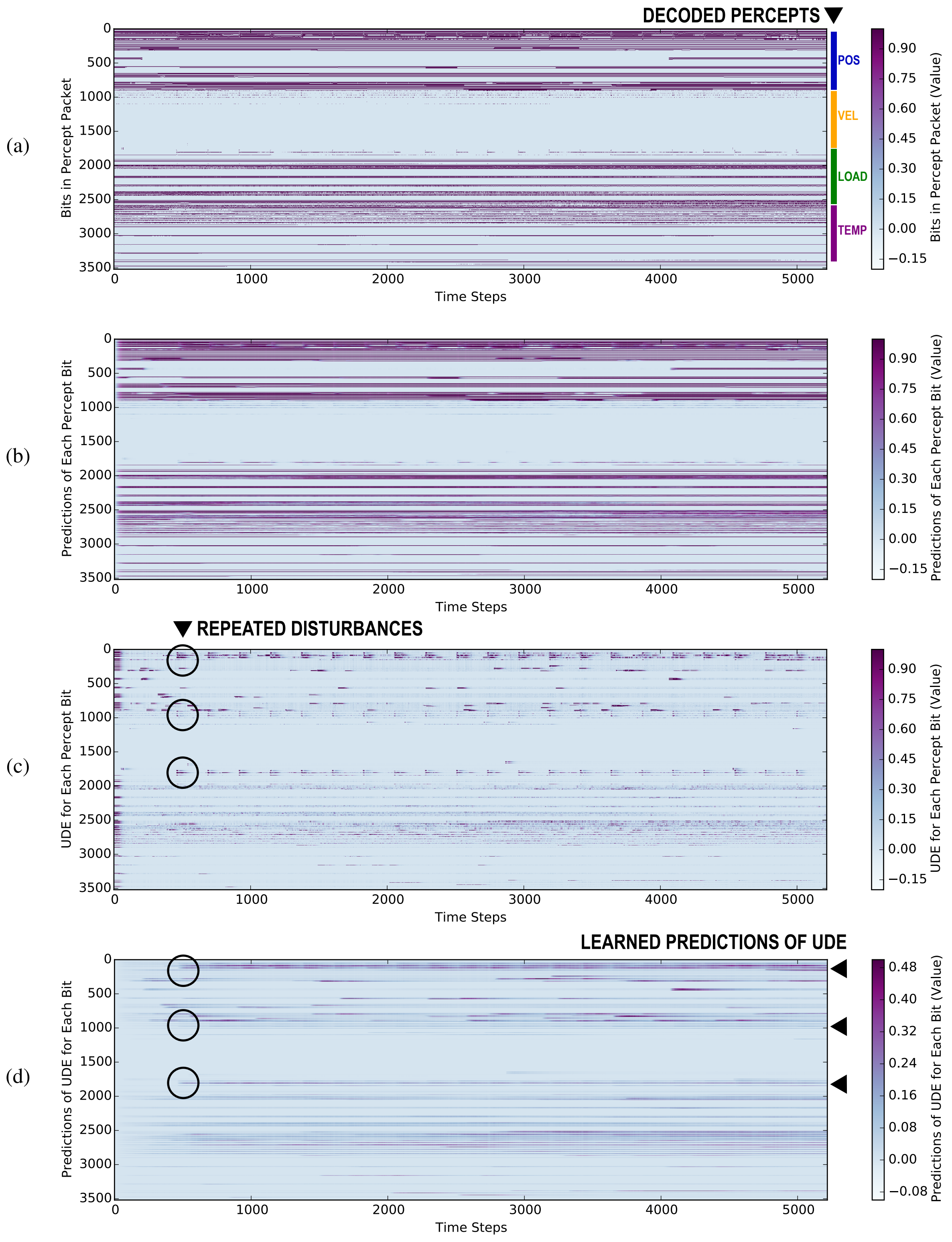

Figure 5: All recorded data for the experiment. The first subplot (a) shows the sensor stream from the MPL as decoded binaries. The second subplot (b) contains the myopic predictions for the binaries, provided by the first predictive layer. In the third subplot (c), the UDE is shown, followed by (d) the predictions about the UDE for a termination signal $\gamma = 0.999$.

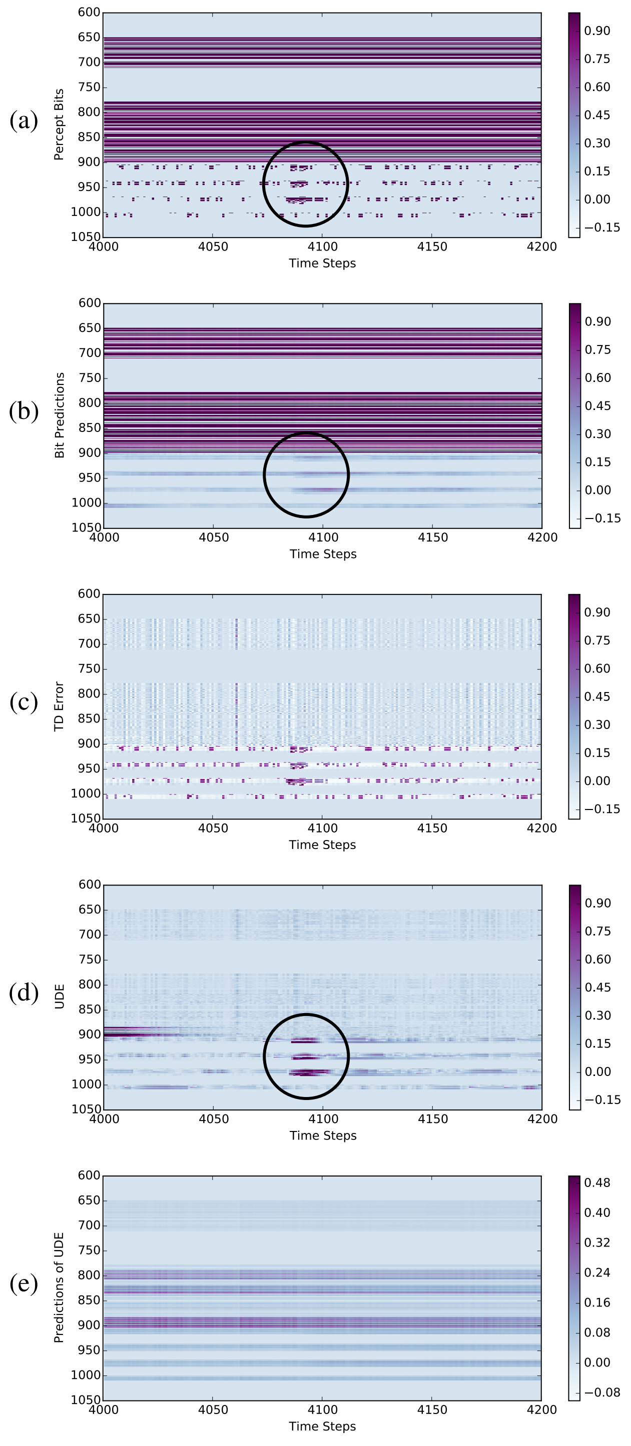

Figure 6: (a) Sensor data, (b) predictions ($\gamma = 0$), (c) prediction error, (d) UDE and (e) predictions of UDE ($\gamma = 0.999$) for position and velocity sensors.

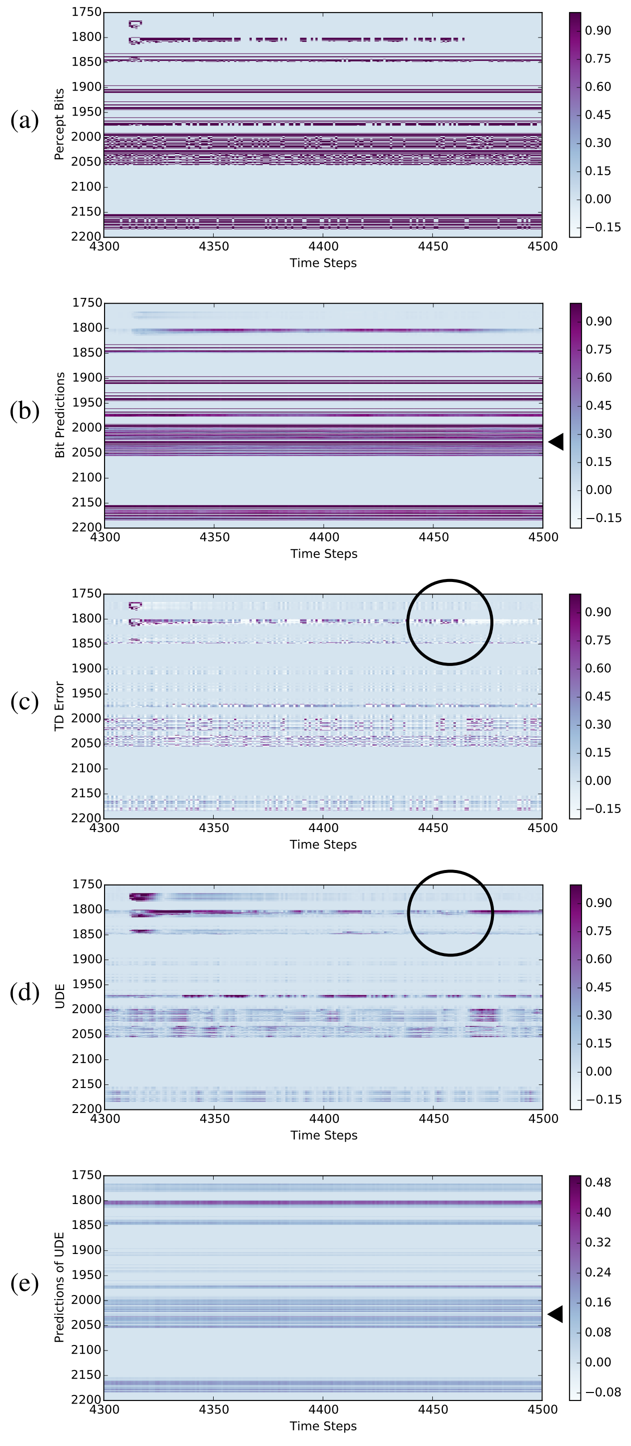

Figure 7: (a) Sensor data, (b) predictions ($\gamma = 0$), (c) prediction error, (d) UDE and (e) predictions of UDE ($\gamma = 0.999$) for load sensors.

Conclusion

The experiments in this paper were conducted to demonstrate how a predictive architecture can learn predictions, measure surprise, and learn predictions of surprise for a recurring pattern of sensor data from a prosthetic limb. The results show that important information about the underlying domain can be revealed by generating signals of interest from the ongoing operation of a Horde of General Value Function learners. The architecture in this paper learns surprise and predictions of surprise but does not make use of them. We suggest that the use of these signals in control learning is a natural extension that promises benefits: introspective signals can potentially help a learning agent to extend its knowledge not only about the environment but also about its own state within this environment. The present work can therefore be viewed as the process of learning a grounded, rudimentary model of actions and their consequences, which may create a foundation for learning more complicated concepts and relationships.

In the case of a learning artificial limb, predictions of surprise should provide knowledge of a change in the dynamics of the prosthesis before the change happens. If successfully learned, such predictions might serve as indicators not only of external variability like a new domain, a handshake, or unpredictable contact with objects, but also of changes in the function of the limb; the latter is a first step towards detecting the need for maintenance before the system breaks down. We suggest that introspective knowledge as presented in this work can be a valuable extension to systems that continually, autonomously learn and adapt in real-world settings.

Acknowledgements

This research was undertaken, in part, thanks to funding from the Canada Research Chairs program, the Canada Foundation for Innovation, the Alberta Machine Intelligence Institute, Alberta Innovates, and the Natural Sciences and Engineering Research Council. The authors also thank Kory Mathewson, Adam White, Richard Sutton, and Joseph Modayil for suggestions and helpful discussions, and Nadia Ady for her in-depth editorial feedback.

References

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research* 47:253–279.

Bridges, M. M.; Para, M. P.; and Mashner, M. J. 2011. Control System Architecture for the Modular Prosthetic Limb. *Johns Hopkins APL Technical Digest* 30(3):217–222.

Clark, A. 2013. Whatever Next? Predictive Brains, Situated Agents, and the Future of Cognitive Science. *Behavioral and Brain Sciences* 36(3):181–204.

Gehring, C., and Precup, D. 2013. Smart Exploration in Reinforcement Learning Using Absolute Temporal Difference Errors. In *Autonomous Agents and Multiagent Systems (AAMAS)*, 1037–1044.

Günther, J.; Pilarski, P. M.; Helfrich, G.; Shen, H.; and Diepold, K. 2016. Intelligent Laser Welding Through Representation, Prediction, and Control Learning: An Architecture with Deep Neural Networks and Reinforcement Learning. *Mechatronics* 34:1–11.

Kahn, G.; Villaflor, A.; Ding, B.; Abbeel, P.; and Levine, S. 2017. Self-supervised Deep Reinforcement Learning with Generalized Computation Graphs for Robot Navigation. *arXiv preprint arXiv:1709.10489*.

Modayil, J.; White, A.; and Sutton, R. S. 2014. Multi-timescale Nexting in a Reinforcement Learning Robot. *Adaptive Behavior* 22(2):146–160.

Pilarski, P. M., and Sherstan, C. 2016. Steps Toward Knowledgeable Neuroprostheses. In *Proceedings of the International Conference on Biomedical Robotics and Biomechatronics*, 220–220. IEEE.

Pilarski, P. M.; Dawson, M. R.; Degris, T.; Carey, J. P.; Chan, K. M.; Hebert, J. S.; and Sutton, R. S. 2013. Adaptive Artificial Limbs: A Real-Time Approach to Prediction and Anticipation. *IEEE Robotics & Automation Mag.* 20(1):53–64.

Sakaguchi, Y., and Takano, M. 2004. Reliability of Internal Prediction/Estimation and Its Application. I. Adaptive Action Selection Reflecting Reliability of Value Function. *Neural Networks* 17(7):935–952.

Schultz, W., and Dickinson, A. 2000. Neuronal Coding of Prediction Errors. *Annual Rev. Neurosci.* 23(1):473–500.

Sherstan, C.; Machado, M. C.; White, A.; and Pilarski, P. M. 2016. Introspective Agents: Confidence Measures for General Value Functions. In *International Conference on Artificial General Intelligence*, 258–261.

Sherstan, C.; Modayil, J.; and Pilarski, P. M. 2015. A Collaborative Approach to the Simultaneous Multi-joint Control of a Prosthetic Arm. In *Proceedings of the International Conference on Rehabilitation Robotics*, 13–18. IEEE.

Sutton, R. S., and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 2nd edition.

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 *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, 761–768. AAMAS.

Sutton, R. S. 1988. Learning to Predict by the Methods of Temporal Differences. *Machine Learning* 3(1):9–44.

Tamar, A.; Castro, D. D.; and Mannor, S. 2016. Learning the Variance of the Reward-To-Go. *Journal of Machine Learning Research* 17(13):1–36.

White, M., and White, A. 2010. Interval Estimation for Reinforcement-Learning Algorithms in Continuous-State Domains. In *Advances in Neural Information Processing Systems*, 2433–2441.

White, M., and White, A. 2016. A Greedy Approach to Adapting the Trace Parameter for Temporal Difference Learning. In *International Conference on Autonomous Agents and Multiagent Systems*, 557–565.

White, A. 2015. *Developing a Predictive Approach to Knowledge*. Ph.D. Dissertation, Dept. of Computer Science, University of Alberta.