Bringing Up IAC

Scott Blaha and Alexandr Pshenichkin

Abstract

Intelligent Adaptive Curiosity (IAC) is an attempt at motivation-based robot learning. IAC has very limited ability to predict and categorize time-dependent elements of its environment, however. In this paper, we attempt to integrate IAC's strong points into a more capable learning algorithm. We implement this by adding a simple recurrent network as a memory module for the previously memory-less IAC. We also explore modifications to the region formation system, in an attempt to create more sensible and dynamic region formation. Our modified system makes modest gains in understanding complex input, but remains unable to handle arbitrarily difficult time-dependencies.

Introduction

The three primary aspects of developmental robotics are abstraction, anticipation, and motivation. Abstraction is the ability of a robot to form coherent categories about its world. Anticipation is a robot's capacity for predicting its future environment. Motivation drives the robot to explore new areas of its world. A system which fully incorporates all three of these traits should be capable of operating and learning in a complex environment. Therefore, we decided to take an existing system and enhance its ability to exhibit all three developmental abilities.

Intelligent Adaptive Curiosity (Oudeyer and Kaplan 2004), or IAC for short, displays these traits in varying degrees. IAC was designed to answer the question of internal motivation; it does an excellent job of looking for novelty in its environment without direction on the part of the system's designer. IAC remains internally motivated to investigate and learn from its environment based on an imperative to minimize error combined with a random element of action selection. This makes IAC explore new parts of its environment while keeping learning progress high.

IAC groups sensory inputs into categories using a clustering algorithm. The intended result of this clustering is to abstract the continuous world into a discrete set of regions. While the general idea behind IAC's abstraction mechanism is sound, some of the issues of its implementation prohibit its use in a real world setting.

As our previous work with IAC (Blaha and Pshenichkin 2006) showed, IAC's main weakness is in its anticipation ability. Our experiments demonstrated that IAC cannot learn complex patterns in its environment, rendering its anticipatory abilities lacking. Therefore, most of our current work on IAC focuses on improving its anticipatory aspect.

Despite the fact that it is the ability of IAC to anticipate that is most directly failing, we found that this is the result of the robot not storing information about previous time steps (e.g. a memory). Because the robot has no memory, it lacks the ability to make totally informed predictions about its environment. Rather than modify IAC itself, we decided to encode memory as an input into the existing framework. This was accomplished by adding a simple recurrent neural network (Elman 1990) to IAC.

Another modification that was tested was an attempt to improve on IAC's abstraction mechanism. We tried replacing the region formation via k-means clustering currently in use by IAC with a Resource Allocating Vector Quantizer (Linaker and Niklasson 2000). RAVQ is a more intelligent category formation system, since it groups input vectors without maintaining a list of all inputs ever to the system, unlike IAC's naïve region formation. However, our attempts to add RAVQ to IAC were thwarted because of the nature of our experimental setup. The reasons for this are described in more detail below.

Experiments

Our experiments started with the same Python implementation of IAC used in our previous work, as well as the same Pyrobot environment. We made several changes to these, as described below. The main change made to our IAC implementation was the addition of a simple recurrent network. Additionally, the amount of random action performed by the IAC-driven robot was reduced. Finally, in some test runs, the region formation mechanism of IAC was changed to RAVQ.

For the overall structure of our memory addition to IAC, see Figure 1. The new simple recurrent network (SRN) receives the robot's current sensorimotor context as input and attempts to predict the robot's sensor values at the next time step. We use the SRN's context layer activations as an approximation of a state vector for the environment. The important point is that this state vector be completely generated by the robot. The hope is that the SRN's context layer begins to represent a form of memory, which IAC can exploit as additional "sensor" data. By concatenating this state vector onto our original sensorimotor context, we will have provided a form of implicit, developmental memory to the robot.

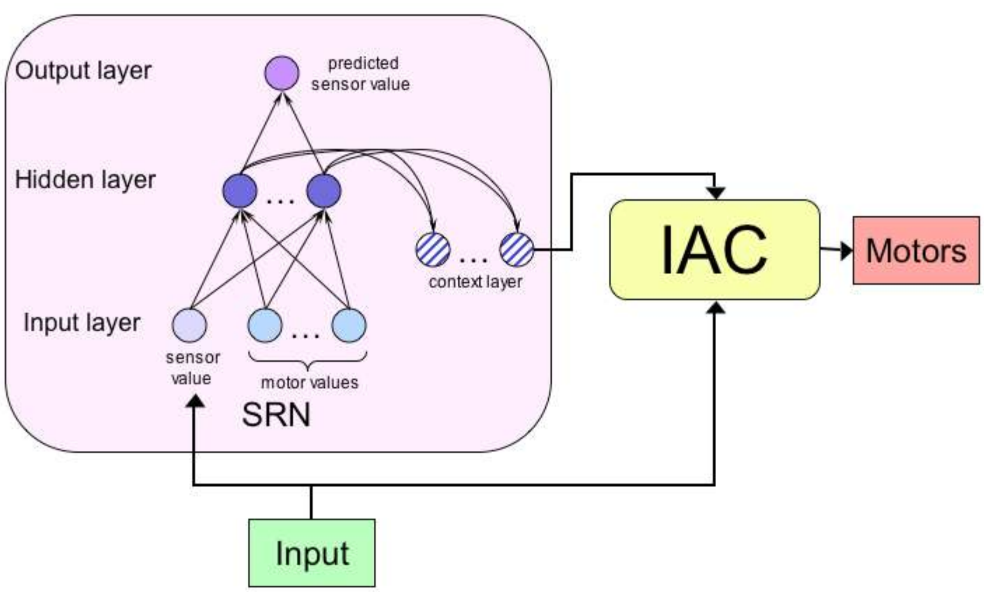

Figure 1: The structure of the memory module

In addition to the change above, we also modified the action selection method of IAC. Previously, IAC chose a random action 35% of the time, to encourage exploration of the environment. However, in our previous work, we found this behavior to be detrimental to the learning of time-dependent aspects of the environment. Therefore, we lowered this chance of random action to 5% while the robot was making positive learning progress (defined by Oudeyer and Kaplan as the approximation of the derivative of the robot's error function). See Figure 2 for an analysis of the effect of this change. We were forced to choose a non-zero chance of random action due to IAC's tendency to become fixated on sources of random noise without an element of chance in its actions.

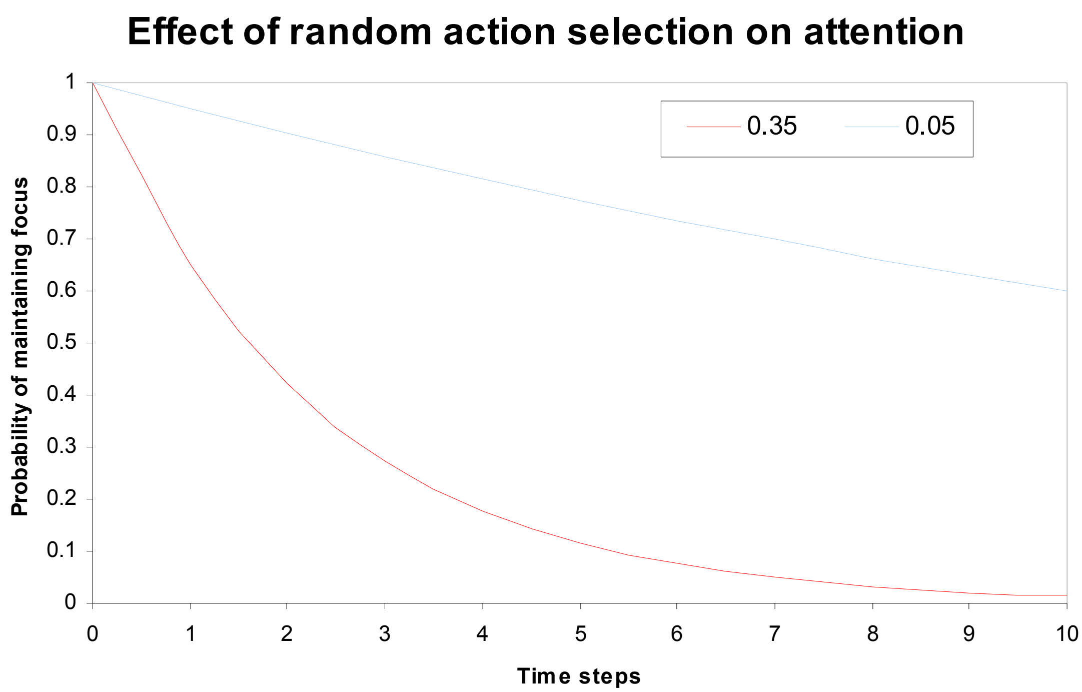

Figure 2: The effects of changing the probability of randomness

Our robot's environment was much the same as in our previous work. The robot sits stationary in the middle of a room, surrounded by light sources (see Figure 3). As before, the robot's only motor options are to turn to face each light source. The robot's only sensor is a light sensor, whose value is completely determined by the robot's heading and the simulation's current time step. For the function each light's intensity obeys, see Table 1. You will notice that these functions are more complex than those used in our previous work. This is because we hope that our modifications to IAC allow it to learn more difficult dependencies in time.

Light	1	2	3	4	5	6	7	8
Pattern	Constant	Random	Binary Random	1-1 Square Wave	2-2 Square Wave	4-4 Square Wave	4-1 Square Wave	2-1-2-3 Square Wave

Table 2: Light sources in the environment

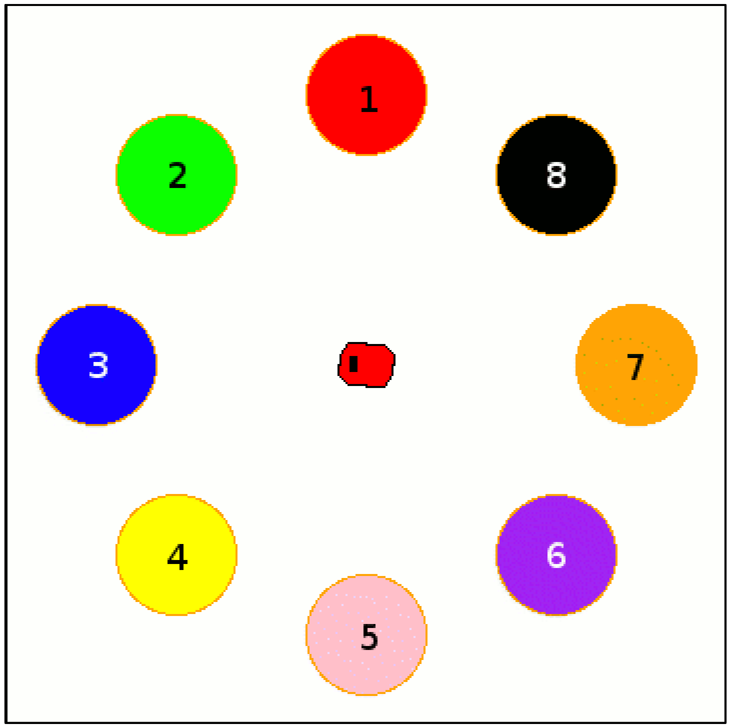

Figure 3: The robot's environment

As described above, we ran some experiments with IAC where the k-means clustering region formation was replaced by a RAVQ. However, RAVQ relies on the assumption that the robot's world is generally continuous to generate regions of sensorimotor space. Our simulated world does not follow this assumption, as sensor values fluctuate wildly over the course of the experiment. Therefore, as will be seen below, RAVQ fails to form meaningful abstractions about our world.

As before, we collected error and heading data on each time step during runs of the robot. Please refer to our previous paper for implementation and experimentation details. Individual experiments constrained the headings (and hence, the lights) available to the robot to better test our hypotheses. These individual experiments are described below.

Results

The first stage of our experiments was to find a reasonable value for the context layer size of the Simple Recurrent Network. An SRN with too small of a context layer will fail to learn anything; if the context layer is too big, in contrast, it will overwhelm the sensorimotor data going to IAC, and the SRN will begin to find nonexistent patterns in the data set. Having tested several values, we finally settled upon a context layer of equal size to our sensorimotor vector, as this seemed to provide a noticeable improvement over smaller sizes and a slight improvement over larger ones. In Figure 4 we see the results of different sizes of context layers on the prediction error on light 4.

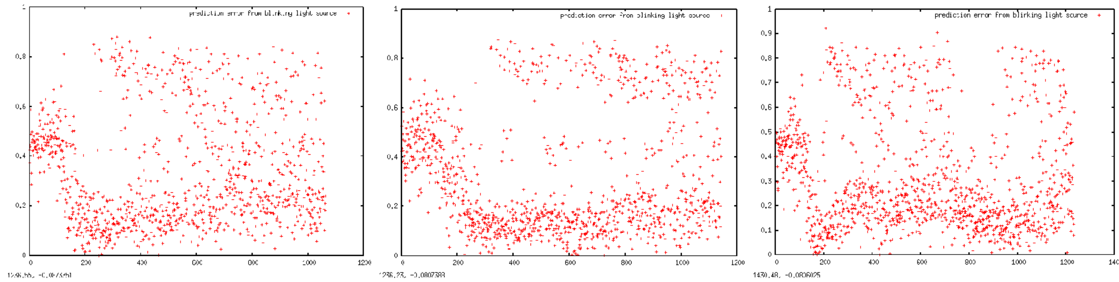

Figure 4: Prediction error of blinking light with small, medium, and large context layers respectively

RAVQ's inapplicability to our experimental setup became apparent during the early stages of testing. RAVQ expects real-valued input in a mostly continuous world and is designed to smooth out input over many time steps. Our experimental world features rapidly-changing discontinuous binary-like inputs. Because of this discrepancy, RAVQ would view changes in input as simply noise, and average the almost-binary inputs into flat lines. As Figure 5 shows, RAVQ fails to effectively learn the pattern of a blinking light, a simple task learned by both stock IAC and our modified version without RAVQ.

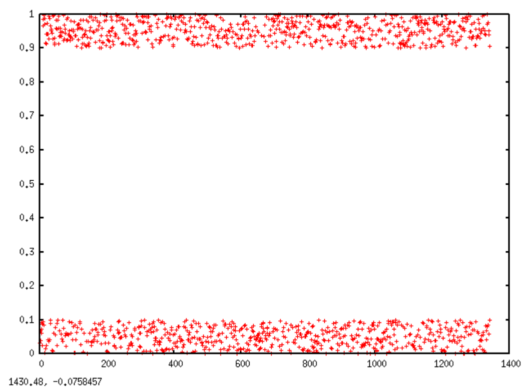

Figure 5: Prediction errors while looking at a blinking light with RAVQ

Since our primary goal was to create system that could learn more complex input patterns, our first set of experiments focused on reducing the robot's prediction error. To better determine whether learning was taking place, we narrowed the light choices so that the system could focus its attention easier on specific lights of our choosing. In each of these runs, the robot could choose between the binary random light (light 3) and the light we were testing to see if our modified IAC could learn to predict. We tested lights 5, 6, and 7 each individually against light 3. Our modified IAC was able to learn to predict the 2-2 square wave light's pattern, as shown by Figure 6. Unfortunately, it was unable to learn to predict any of the more difficult lights, as shown by Figures 7 and 8. In the case of the 4-1 square wave (light 7), we claim that it has not learned to predict the light's value (even though most of its predictions are very good later on!) because getting the right answer 80% of the time on this light is not good enough; the robot should be able to guess when the light will be off as well.

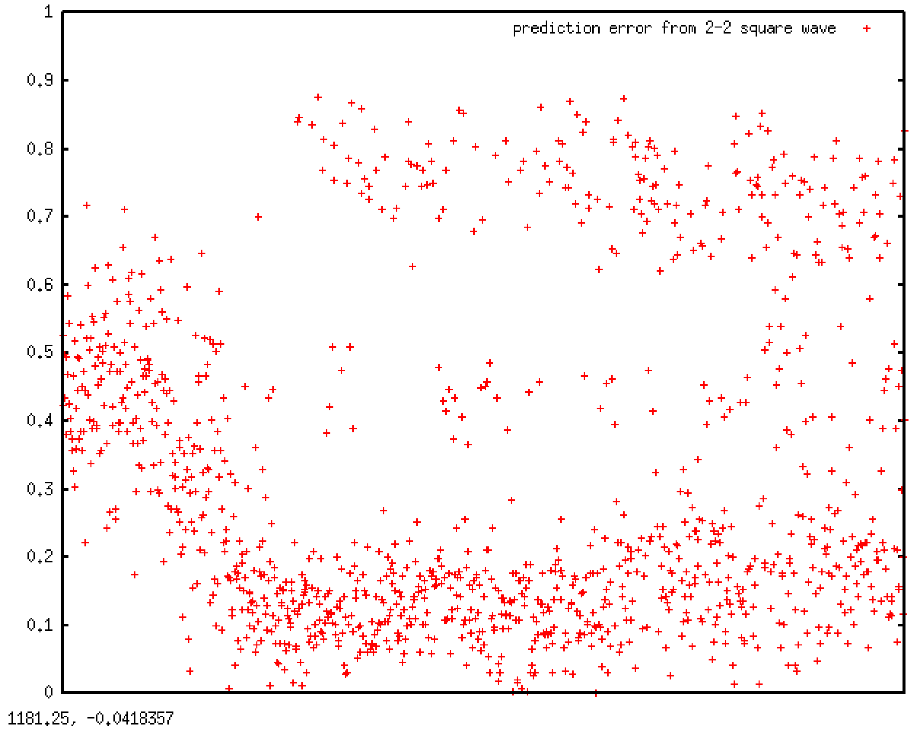

Figure 6: Prediction errors while looking at a 2-2 square wave

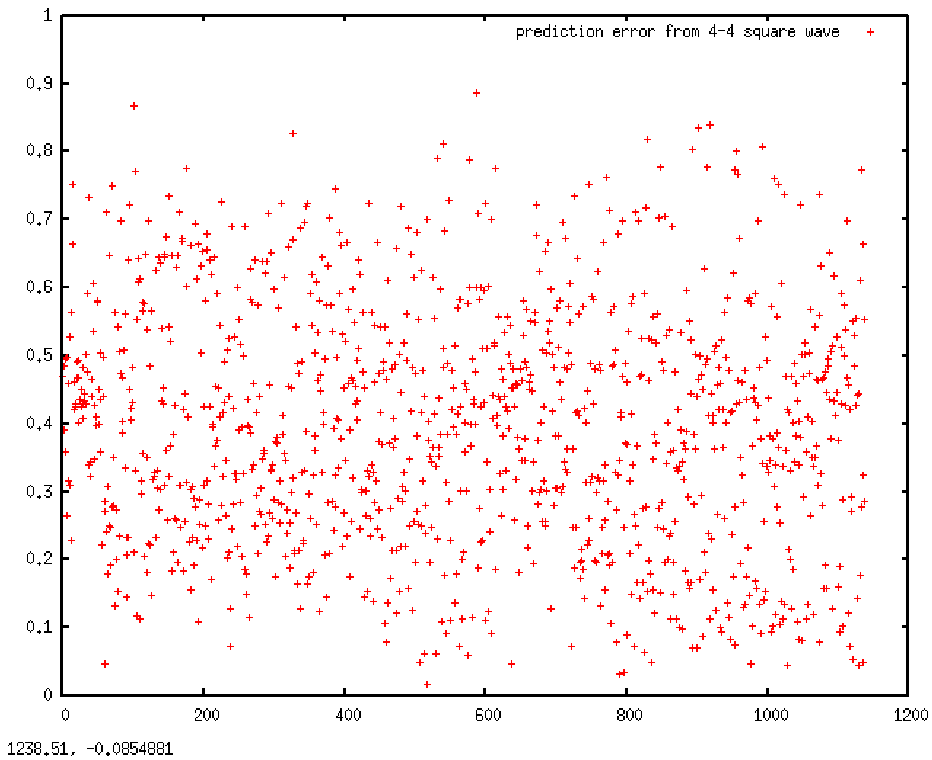

Figure 7: Prediction errors while looking at a 4-4 square wave

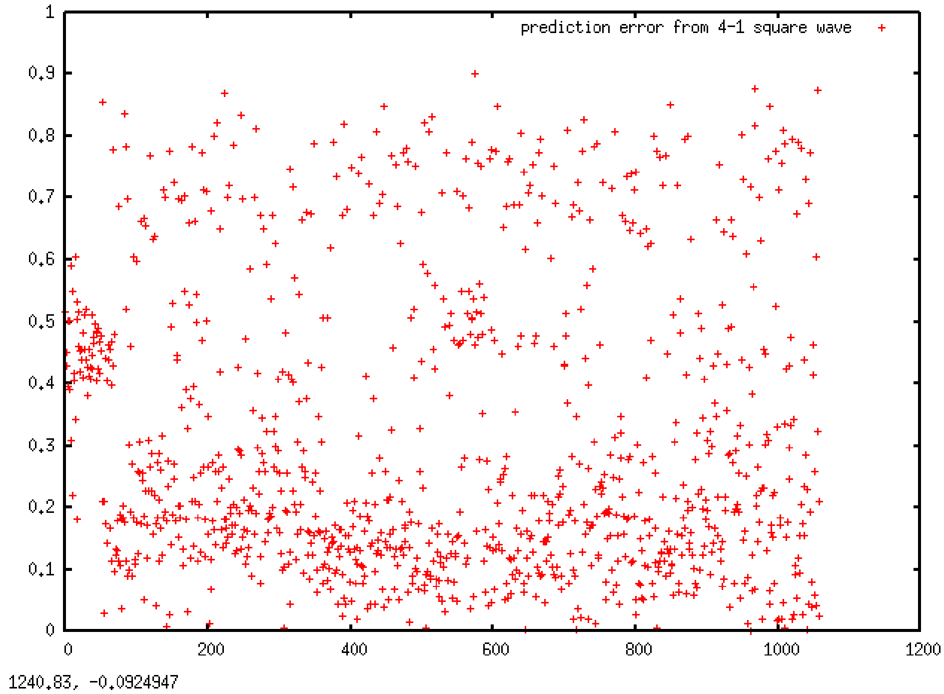

Figure 8: Prediction errors while looking at a 4-1 square wave

Having examined the question of whether or not the system learned and found the results a bit disappointing, we now turn to the question of whether or not the system was motivated to learn. We ran the full version of the experiment, permitting the robot to choose any of the eight input sources. Figure 9 shows the relative attention paid to the light sources at different points in the experiment. We have identified five distinct regions (i.e. periods) of the run, which are indicated on the graph by Roman numerals and discussed in more detail below. As we will see below, the robot tends to focus its attention on simpler tasks before harder tasks. This pattern has been very robust across all of our experiments, and we consider it to be one of the advantages of IAC.

In region I, we see that the system very quickly becomes acclimated to the constant input value (light 1). It then focuses significant attention on light 2, the non-binary random light source. A comparison of the attention devoted to light 2 and light 3, however, shows that the non-binary random does not attract extra attention due to its continuous nature. It is popular at the beginning, however, as IAC's initial guess (0.5) is likely to be closer to the true sensor value than it would be for binary-valued data.

In region II, the robot devotes most of its attention to the binary random light, but also looks at the blinking light. Surprisingly, the robot was unable to fully learn the blinking light in this run, which can probably be attributed to insufficient attention.

In region III, the robot now focuses on the 2-2 square wave (light 5). Its interest in the random light sources has declined sharply as the 2-2 square wave presents a better prospect for learning progress; the attention which IAC pays to the light indicates that this is a more difficult task to master than the blinking light.

Having learned the 2-2 light, the robot now observes light 6, which is the 4-4 square wave pattern. Attention to randomness is high once again, however, which, for such a complex pattern, supports the conclusion that that learning is not really achieved (see Figure 7).

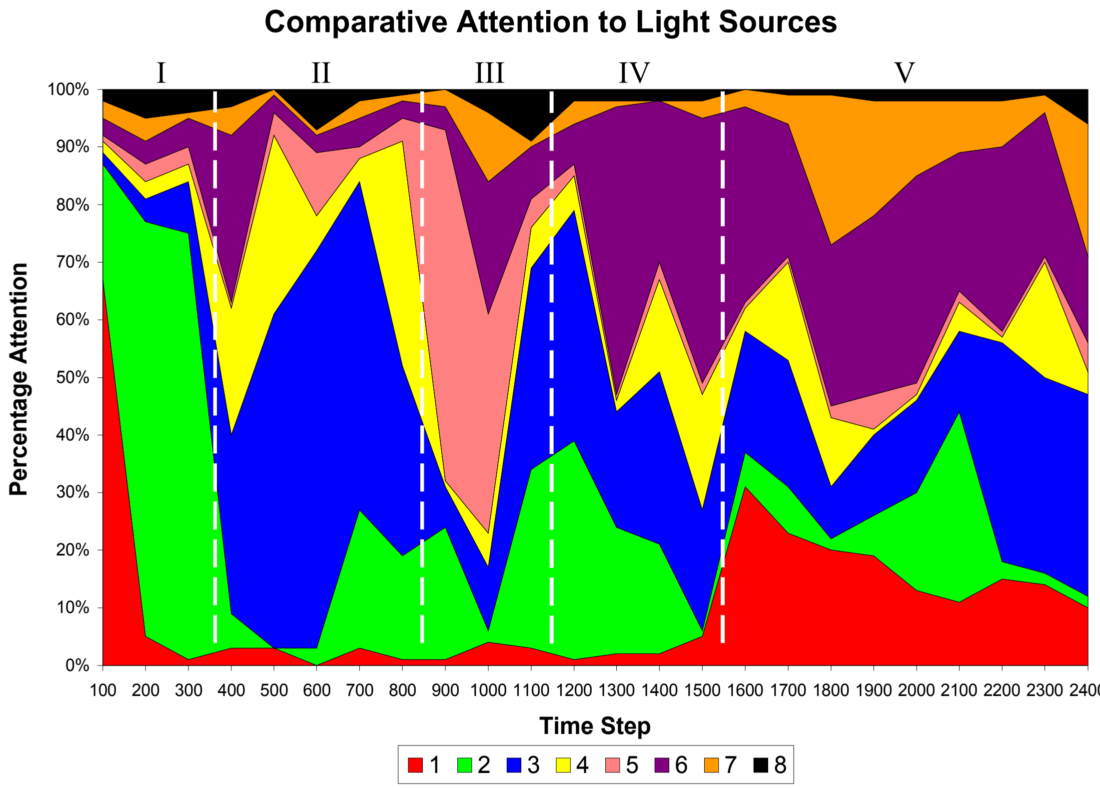

Figure 9: Attention paid to each light source over a full run

Finally, the system settles into a state of confused boredom, as indicated by the increased attention to previously-learned light sources. The system's continued attention to light 6 indicates that it is still failing to produce accurate guesses. Meanwhile, the constant input and the 4-1 square wave (lights 1 and 7, respectively) receive much more attention.

Discussion

The addition of an SRN memory module allowed IAC to learn to predict more complex time-dependent features of its environment. As Figure 10 shows when compared to Figure 6, our modifications to IAC have produced a tangible improvement in its learning ability for some time-dependent phenomena. This pattern, however, does not generalize to higher-order relationships.

It seems that the simplicity of our sensors is an obstacle to IAC learning more complex patterns. Because there is only one quasi-binary input, our range of possible patterns is extremely limited. In addition, there is only one sensor value as input, versus 8 motor values and 9 context layer values; this makes it difficult for the SRN to pick out this value as being more important than the others, particularly because, by adding the context layer as input, we've added many more real-valued inputs that are always on and always changing.

Since neural networks serve as both our memory and our experts, the issues we saw in our previous experiments are magnified. Since they use sigmoid activation functions, they have trouble producing extreme outputs, such as the quasi-binary values we are working with. As neural networks tend to smooth input, an uneven time dependency such as the 4-1 square wave (light 7) will tend to be simplified down to a constant value. In the case of our most complex pattern, the 2-1-2-3 square wave (light 8), there is simply no reason for a neural network to prefer it over random inputs: a real-valued random function will have lower prediction error, since the network will achieve more success by just guessing 0.5, whereas the output of a binary random function is not significantly different from the square wave's pattern.

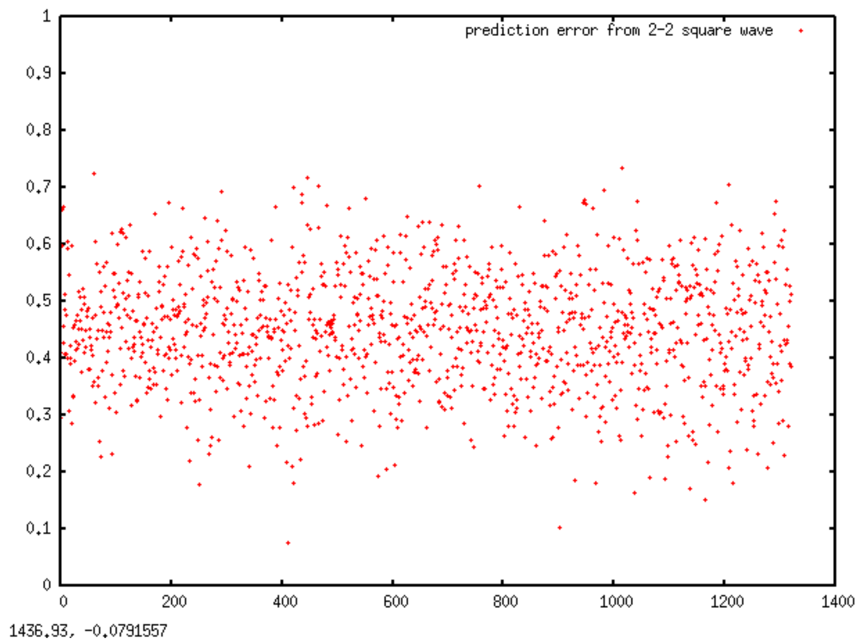

Figure 10: Original IAC's prediction errors while looking at 2-2 square wave

Any future learning task incorporating IAC and simple recurrent networks must necessarily have more complex input. Increasing the richness of input to the SRN should improve its ability to encode information about the robot's environment. Improving the abstraction mechanism, such as by replacing the region generation and experts with growing neural gas (Fritzke 1995) and a moving average, would allow the robot to process and categorize this more complex input.

References

- Blaha, S. and Pshenichkin, A. (2006). "The limitations of IAC". Unpublished work.
- Elman, J.L. (1990). "Finding structure in time". Cognitive Science, 14, 179-211.
- Fritzke, B. (1995). "A growing neural gas learns network topologies". In Advances in Neural Information Processing Systems, 7, 625-632.
- Linaker, F. and Nikalsson, L. (2000). "Sensory flow segmentation using a resource allocating vector quantizer". Advances in Pattern Recognition: Joint IAPR International Workshops, SSPR 2000 and SPR 2000, Alicante, Spain, August/September 2000. In Lecture Notes in Computer Science, 1876, 853.
- Oudeyer, P.-Y. and Kaplan, F. (2004). "Intelligent Adaptive Curiosity: a source of self-development". In Proceedings of the 4th International Workshop of Epigenetic Robotics, 117, 127-130.

Appendix

We have included, as a supplement, an alternative version of Figure 9 from another run; it is not part of the paper per se. We hope that it will be easier to separate typical and transient behaviors by examining both graphs. We have attempted to label the regions of this figure to coincide with those of the other one; region features are explained below.

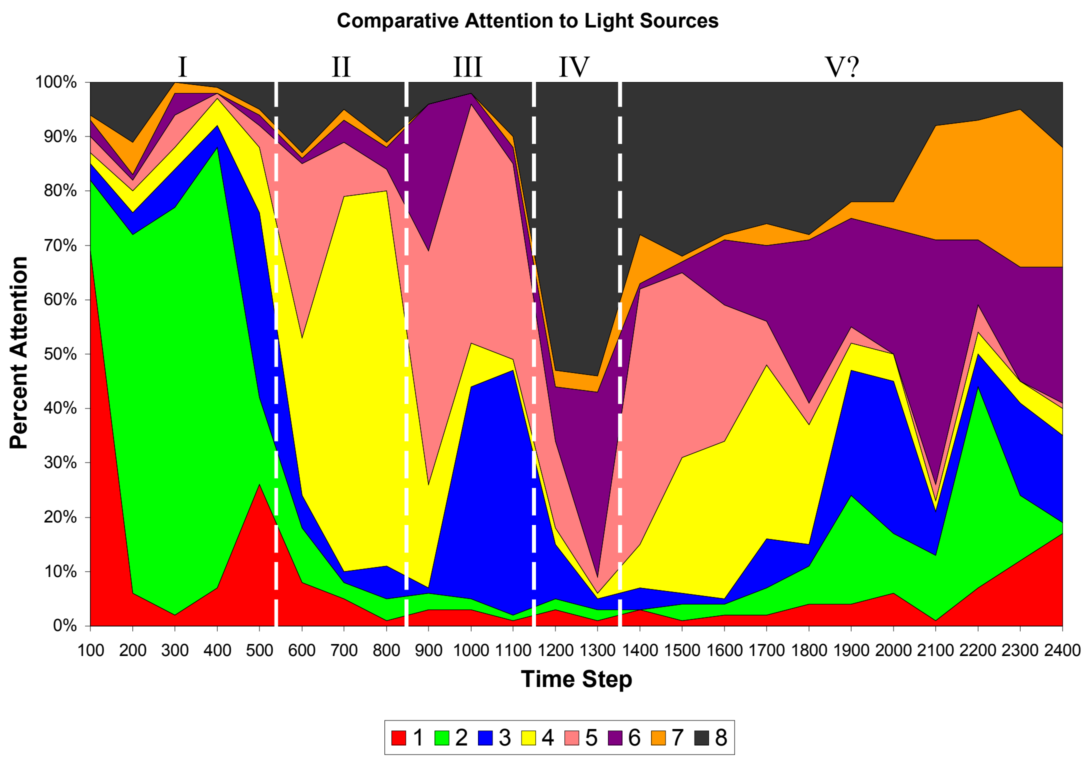

Figure 0: Attention paid to each light source over a full run

Region I is nearly identical to Figure 9's Region I. The robot seems to quickly learn the constant light and then focus a great deal of its attention on the smoothly random one.

Region II seems like an improvement over the previous case, as the robot is much more focused on the blinking light; examining its error, however, reveals strangely lackluster performance. There is a brief reduction in error, but it seems that the robot's learning is soon forgotten.

Both the emphasis on the 2-2 square wave and the resurgence of interest in the binary random light in Region III are similar to observations from the previous experiment.

Region IV is completely different from the last experiment's. Interestingly, the 2-1-2-3 square wave attracted more focus than the 4-4 square wave. With its attention equally split between the two, it seems the robot learned neither.

Finally, we've labeled the rest of the graph Region V. Based on prediction error, little if any learning progress happens here.