Minimizing Environmental Swings with a Recurrent Neural Network Control System

Sam Skrivan, Dr. Jianna Zhang, Dr. Debra Jusak

Western Washington University, Department of Computer Science
samsk@microsoft.com
Jianna.Zhang@wwu.edu
Deb.Jusak@wwu.edu

Abstract

Maintaining environmental stability in a dynamic system is a difficult challenge. In your living room, when you set your thermostat to 68 degrees the actual temperature cycles above and below 68 degrees. We attempt to use a Recurrent Neural Network (RNN) in an Aquarium Control System that reduces such environmental swings (see Figure 1).

Introduction

Artificial Neural Networks (ANNs) have been heavily used in control systems. Early work with ANNs to fine tune a controller in highly non linear environments is surveyed by Panos Antsaklis [4]. More recent work uses Recurrent Neural Networks (RNN) in areas from network restoration when faced with physical changes [6] to scheduling jobs in a machine flow shop [7]. Neural Networks provide the control system with the ability to adapt to unseen circumstances. More efficient learning algorithms have been developed to provide the performance needed for real-time systems.

The Aquarium Control System (ACS) is a project that uses middleware to control and monitor environmental aspects of an Aquarium. This project will adapt the behavior of this control system using a RNN. The RNN will learn and reduce these environmental swings in the aquarium.

The Problem Addressed

The Aquarium Control System uses a Programmable Logic Controller (PLC) to control the instruments and actuators attached to the system. The environmental conditions it controls are the temperature, the salinity and the ph level of the aquarium. The PLC is programmed with the thresholds for each of these readings. When the reading crosses a programmed threshold the corresponding actuator is turned on. There is a lag between the time the reading crosses a threshold and the amount of time it takes for the actuators to change to be picked up in the readings.

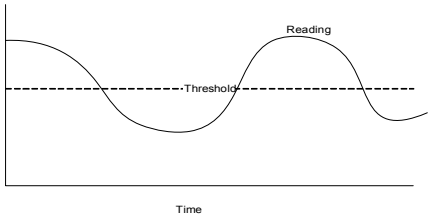

Figure 1. Environmental Swings

By the time the sensor picks up the changes the actuators have been on for too long and the environmental condition continues to rise or fall. This project will use a RNN to learn these curves and minimize the swings.

Project Design

In setting up the problem, one of the major challenges was that the desired output of the system is not known. What is known is that the system should turn off the heater before the temperature reaches the threshold. Looking at the graph, one can observe that minimizing the swings in the graph is minimizing the rate of change of the measured input. Intuitively it makes sense that if the rate of change was increasing, the system needed to take action to slow it down. To come up with the desired output and the network error calculation, we calculate the error proportional to the slope of the input lines.

We break up the graph into four quadrants: (1) the reading is above the threshold with a positive slope, (2) the reading is above the threshold with a negative slope, (3) the reading is below the threshold with a negative slope and (4) the reading is below the threshold with a negative slope. In the 1st and 3rd quadrant, we know what the desired output is, either the temperature is hot and getting hotter and the heater should be shut off, or cold and getting colder and the heater should be turned on. In the 2nd and 4th quadrants however, we know we want to take the opposite action before we get to the threshold, but we don't know when. In these cases we train the network to take the opposite action proportional to the slope. For example, in

the 2nd quadrant, where the temperature is falling, we calculate the instance error, with the following formula:

$e_k = O_k(O_k - 1)(0 - O_k)\mathrm{M}$, where M is the slope of the input line, and the expected output is 0(heater off). As the rate of change increases, the correction on the network to drive the output to 0 increases.

The feedback in the RNN allows the network to encode some knowledge of previous states in the system. It has been shown that the memory encoded in the weights degrades over time, and "forgets" what has happened in the distant past. There has been much work to look at these issues. [2,3,5]. Back propagation Through Time(BPTT) has shown promise in learning time series data.[3] BPTT turns a feedback system into a feed forward system by calculating the output and error over time.

Our network has three inputs: temperature, salinity and ph readings. There are three output nodes for each of the actuators: heater, fresh water pump, and CO2 injector. There is one hidden layer, and a feedback loop from each of the hidden nodes back to itself. The network uses the BPTT algorithm in the online phase of the project. Figure 2 shows the architecture of the RNN.

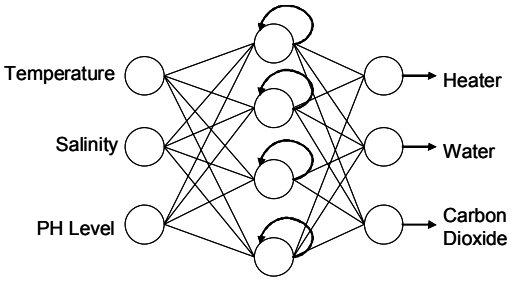

Figure 2. RNN Design

Approach

Initialization Phase. When the system starts, the network will be trained on a training set, which uses the thresholds programmed into the PLC. By the end of the initialization the temperature output will output close to 1 when the temperature input is above the threshold and close to 0 when it is below the threshold. The same will be true for the salinity, and nitrogen output. During this phase the hidden layer won't have feedback and the network will be trained using Back propagation. The initialization phase of the project has been implemented.

Online Phase. Once the error of the system on the training set reaches an acceptable level, it will shift to the online phase. During this phase the outputs of the network will be sent to the PLC to control the actuators. The network will continue to learn, however the error calculation will change. The error of the system will be calculated based on the difference of the n reading with the n-1 reading, for all of the sensors. The correction of the system will be proportional to the difference in the two readings. The system will be attempting to learn to minimize the variation in the readings. During this phase the feedback is added to the hidden layer and the BPTT algorithm will be used in a continuous learning mode.

Testing

We expect to find that the system using the neural networks performs better than the conventional approach. We will measure this success as the area above and below the threshold for the temperature, salinity and ph level for a 24 hour period.

We will run the system with the PLC operating in normal mode, based on the programmed thresholds, for a 24 hour period. We will compute the areas for the three readings. We will then run the system with the RNN added to the control system and the areas above and below the threshold will be computed again.

Conclusions

Recurrent Neural Networks provide a promising technique for adapting control systems. The Aquarium Control System, which has an open distributed architecture, is an excellent test bed for putting these assertions to the test. We expect to find that using the RNN will be successful in adapting the systems behavior to reduce the amplitude of the swings in temperature, salinity and ph levels. In future projects we would like to try other RNN algorithms and topologies. We would also like to work on different error calculations for the online phase of the project.

References

[1] Mitchell, Tom, "Machine Learning", 1997, McGraw Hill.

[2] Mandic, Danilo, & Chambers, Jonathan, "Recurrent Neural Networks For Prediction", 2001, John Wiley & Sons LTD.

[3] Ludik, Prins, Meert, Catfolis, "A Comparative Study of Fully and Partially Recurrent Networks", Neural Networks, Vol. 1, pp.292-297, 1997.

[4] Antsaklis, Panos,"Neural Networks for Control Systems", IEEE Transactions on Neural Networks, Vol. 1, Number 1, pp. 149-153, 1990.

[5] Jodouin, J., "Putting the Simple Recurrent Network to the Test", Neural Networks, 1993, IEEE International Conference on, Vol. 2, pp 1141-1146, 1993.

[6] Kumar, G. Prem & Venkataram, P.,"Network Restoration Using Recurrent Neural Networks", Int. J. Network Mgmt., 8, 264–273 1998.

[7] Lee, I. , Gupta, J. , Amar, D., "A Multi-Neural-Network Learning for Lot Sizing and Sequencing on a Flow-Shop", Proceedings of the 2001 ACM symposium on Applied computing, Las Vegas, Nevada, pp. 36 – 40, 2001.