CURIOUS MODEL-BUILDING CONTROL SYSTEMS

In Proc. International Joint Conference on Neural Networks, Singapore, volume 2, pages 1458-1463. IEEE, 1991.

Jürgen Schmidhuber*
Department of Computer Science
University of Colorado
Campus Box 430, Boulder, CO 80309, USA

Abstract

A controller is a device which receives inputs from a (dynamic) environment and produces outputs that manipulate the environmental state. A model-building control system is a controller with an additional module (the 'world model') which is trained to predict future inputs from previous input/action pairs. The novel *curious* model-building control system described in this paper is a model-building control system which actively tries to provoke situations for which it *learned to expect to learn* something about the environment. Such a system has been implemented as a 4-network system based on Watkins' Q-learning algorithm which can be used to maximize *the expectation of the temporal derivative of the adaptive assumed reliability of future predictions.* An experiment with an artificial non-deterministic environment demonstrates that the system can be superior to previous model-building control systems (the latter do not address the problem of modelling the reliability of the world model's predictions in uncertain environments and use ad-hoc methods (like random search) to train the world model).

1 INTRODUCTION

Much of the recent research on adaptive neuro-control and reinforcement learning focusses on systems with sub-modules that learn to predict inputs from the environment. These sub-modules often are called 'adaptive world models'; they are useful for a whole variety of control tasks. For instance, Werbos' and Jordan's architectures for neuro-control [15][2] contain an adaptive world model in form of a back-propagation module (the model network) which is trained to predict the next input, given the current input and the current output of an adaptive control network. The model network allows to compute error gradients for the controller outputs. This is essential, since with typical adaptive neuro-control tasks there is no teacher who provides desired controller outputs. There is only a desired environmental input. Extensions of this approach [9] rely on the same basic principles. 'DYNA-systems' [11] use adaptive world models for limiting the number of 'real-world experiences' necessary to solve certain reinforcement learning tasks.

There are at least two important problems with all of these approaches that have not been addressed so far:

1. Previous model-building control systems are not well-suited for uncertain non-deterministic environments. In particular, they do not model the reliability of the predictions of the adaptive world models. Therefore, if credit assignment for the controller is based on the assumption of a correct world model, unexpected results may be obtained.

2. Previous model-building control systems employ some ad-hoc method for establishing the world model. For instance, [2], [3], [10], and others use random search to train the world model. [11] uses a local input/output representation and makes the probability of making a certain training experiment dependent on the time that went by since the system made the last experiment of the same type. These methods

*This work has been done at Technische Universität München, Germany work fine for certain problems, but they do not address the challenges of real world tasks in uncertain environments. There are at least two (related) sources of efficiency which are neglected by these approaches:

2A. Not much additional training time should be wasted on exploring those parts of the world which are already well-modelled. 2B. Not much additional training time should be wasted on exploring those parts of the world where the expectation of future improvement of the world model is low.

The first contribution of this paper (section 2) is to show how one can adaptively model the reliability of a predictor's predictions.

The second (and most important) contribution of this paper (section 3) is to show how reinforcement learning can be used for teaching a model-building control system to actively generate training examples for increasing the reliability of the predictions of its world model. This is relevant for the problem of 'on-line state space exploration'. The approach is based on learning to estimate the effects of further learning.

2 ADAPTIVE CONFIDENCE

Consider an adaptive discrete time 'predictor' M (not necessarily a neural network) whose input at time t is the real vector $i_M(t)$ and whose output at time t is the real vector $o_M(t) = f_M(i_M(t), h_M(t))$, where the real vector $h_M(t)$ represents the internal state of M. Meaningful internal states are required if the prediction task requires to memorize past events. At time t there is a target output $d_M(t)$. The predictor's goal is to make $o_M(t) = d_M(t)$ for all t.

After having provided a number of training examples for M, M usually will still make some errors, particularily if the training environment is noisy. How can we model the reliability of M's predictions?

We introduce an additional 'confidence module' C (not necessarily a neural network) whose input at time t is the real vector $i_C(t) = i_M(t)$ and whose output at time t is the real vector $o_C(t) = f_C(i_C(t), h_C(t))$, where the real vector $h_C(t)$ is the internal state of C. At time t there is a target output $d_C(t)$ for the confidence module. $d_C(t)$ should provide information about how reliable M's prediction $o_M(t)$ can be expected to be [8] [5] [7].

In what follows, v^j is the jth component of a vector v, E denotes the expectation operator, $dim(x)$ denotes the dimensionality of vector x, $\mid c \mid$ denotes the absolute value of scalar c, $P(A \mid B)$ denotes the conditional probability of A given B, and $E(A \mid B)$ denotes the conditional expectation of A given B. For simplicity, we will concentrate on the case of $h_C(t) = h_M(t) = 0$ for all t. This means that M's and C's current outputs are based only on the current input. There is a variety of simple ways of representing reliability in $d_C(t)$:

1. Modelling probabilities of global prediction failures. Let $d_C(t)$ be one-dimensional. Let $d_C(t) = P(o_M(t) \neq d_M(t) \mid i_M(t))$. $d_C(t)$ can be estimated by $\frac{n_1}{n_2}$, where n_2 is the number of those times $k \leq t$ with $i_M(k) = i_M(t)$ and where n_1 is the number of those times k with $i_M(k) = i_M(t), o_M(k) \neq d_M(k)$.

2. Modelling probabilities of local prediction failures. Let $d_C(t)$ be $dim(d_M(t))$-dimensional. Let $d_C^j(t) = P(o_M^j(t) \neq d_M^j(t) \mid i_M(t))$ for all appropriate j. $d_C^j(t)$ can be estimated by $\frac{n_1}{n_2}$, where n_2 is the number of those times $k \leq t$ with $i_M(k) = i_M(t)$ and where n_1 is the number of those times k with $i_M(k) = i_M(t), o_M^j(k) \neq d_M^j(k)$.

Variations of method 1 and method 2 would not measure the probabilities of exact matches between predictions and reality but the probability of 'near-matches' within a certain (e.g. euclidian) tolerance.

3. Modelling global expected error. Let $d_C(t)$ be one-dimensional. Let

$$d_C(t) = E\left\{\frac{1}{2}(d_M(t) - o_M(t))^T(d_M(t) - o_M(t)) \mid i_M(t)\right\}.$$

If C is a back-propagation net (e.g. [14]), an approximation of $d_C(t)$ can be obtained by using gradient descent (with a small learning rate) for training C at time t to emit M's error $\frac{1}{2}(d_M(t) - o_M(t))^T(d_M(t) - o_M(t))$. This is a special case of the method described in [8] (there a fully recurrent net was employed). Of course, other error functions are possible. For instance, with the experiments described below the confidence network predicted the the absolute value of the difference between M's (one-dimensional) output and the current target value.

4. Modelling local expected error. Let $d_C(t)$ be $dim(d_M(t))$-dimensional. Let

$$d_C^j(t) = E\{(d_M^j(t) - o_M^j(t))^2 \mid i_M(t)\}$$

for all appropriate j. If C is a back-propagation net, an approximation of $d_C(t)$ can be obtained by using gradient descent (with a small learning rate) for training C at time t to emit M's local prediction errors

$$\left((d_M^1(t) - o_M^1(t))^2, \ldots, (d_M^m(t) - o_M^m(t))^2\right)^T,$$

where $m = dim(o_M(t))$.

3 ADAPTIVE CURIOSITY

If M is used as a 'world model', then with many applications $i_M(t) = o_A(t) \circ x(t)$ and $d_M(t) = x(t+1)$, where $o_A(t)$ is the output vector of a controller A at time t, '$\circ$' is the concatenation operator, and $x(t)$ is the environmental input at time t. In general, $o_A(t)$ influences the state of the environment. Therefore it may have an influence on $x(t+1)$.

In [7] confidence modules have been successfully applied to the problem of meaningful hierarchical sequence chunking. This section (which provides the major contribution of this paper) describes how they can help to make the *construction* of a world model more efficient.

We define curiosity as the desire to improve a predictor of the reactions of an environment (a 'world model'). In [8] and [4] the following basic idea for 'on-line state space exploration by implementing dynamic curiosity and boredom' has been formulated: *Spend reinforcement for a model-building control system whenever there is a mismatch between the expectations of the adaptive world model and reality.* Any sensible reinforcement learning algorithm can be used to encourage the controller to generate action sequences that provoke situations where the world model tends to make bad predictions. Since the model is adaptive, its predictions often will improve. This in turn will lead to less reinforcement for the control system. Therefore the corresponding action sequences will become discouraged. The controller will get *'bored'* with the corresponding situations and will start to focus on yet unpredictable parts of the environment.

The particular implementation described in [8] employed a recurrent confidence network with a one-dimensional output for modelling the expected error of the model network (this error was called the 'curiosity reinforcement'). The confidence network was not called so: It was part of the model network (which predicted the next state of the environment plus a reinforcement vector including all kinds of reinforcement, not just 'curiosity reinforcement'). The target activation of the single output unit of the confidence net was a function of the current error of the model network. In the simplest case this function was linear. The controller's goal was to activate the error-predicting unit by creating action sequences for provoking mismatches between expectations and reality. The gradient computed for the error predictor also served to change the internal representations of the whole network (whose error function simply contained an additional term). Recently [12] described related ideas (they use the term 'competence network' instead of the term 'confidence network' as used in [7] and [5]).

One problem with the idea above is that in non-deterministic environments the controller will focus on parts of the environmental dynamics which are inherently unpredictable. This is because the adaptive model usually will produce incorrect predictions for the uncertain parts of the environment. Therefore the control system will receive reinforcement *although it cannot be expected that the world model will improve.*

A related problem is that often certain parts of the environment can be represented only by a complex mapping which is difficult to learn while other parts are 'easy to learn'. If we want a system which first tries to solve the easy tasks before focussing on the harder tasks then the system will need an (adaptive) internal representation of something like the *expectation of how difficult certain learning tasks will be.*

Both problems are related in the sense that both require to *learn something about the effects of further learning.* In what follows an approach for coping with these problems will be described. Instead of simply learning to predict errors as the approach described in [8] the new approach learns to predict cumulative error *changes.*

3.1 THE BASIC PRINCIPLE

This subsection discusses a rather general principle of adaptive curiosity. Here we do not have to care whether the adaptive world model is implemented as a back-propagation network, as a lookup table, or as something else. There are certain natural implementations of the ideas; they are discussed in the following subsections.

The basic principle can be formulated as follows: *Learn a mapping from actions (or action sequences) to the expectation of future performance improvement of the world model. Encourage action sequences where this expectation is high.*

One way to do this is the following (section 4 will describe alternatives): *Model the reliability of the predictions of the adaptive predictor as described in section 2. At time t, spend reinforcement for the model-building control system in proportion to the current change of reliability of the adaptive predictor. The 'curiosity goal' of the control system (it might have additional 'pre-wired' goals) is to maximize the expectation of the cumulative sum of future positive or negative changes in prediction reliability.*

More formally: The control system's curiosity goal at time t_0 is to maximize

$$E\{\sum_{t \geq t_0} -\gamma^{t-t_0} \triangle o_C(t+1)\}.$$

Here $0 \leq \gamma < 1$ is a discount factor for avoiding infinite sums, and $\triangle o_C(t)$ is the (positive or negative) change of assumed reliability caused by the observation of $i_M(t)$, $o_M(t)$, and $x(t+1)$.

For instance, if method 1 or method 3 from section 2 is employed, then $\triangle o_C(t) = o_C(t) - \bar{o_C}(t)$, where $\bar{o_C}(t)$ is C's response to $i_M(t)$ after having adjusted C at time t.

So far the discussion did not have to refer to a particular reinforcement learning algorithm. Every sensible reinforcement learning algorithm ought to be useful (e.g [1][16][13][9]). For instance, [6] describes how adaptive critics [1][15] can be used to build a 'curious' model-building control system based on the principle described above. The following subsection focusses on Watkins' recent 'Q-learning' method.

3.2 A CURIOUS SYSTEM BASED ON Q-LEARNING

Here we describe how a reinforcement learning method called Q-learning can be used to build a 'curious' model builder. The notation is the same as above. Following [13] we introduce an adaptive function Q for evaluating pairs of inputs $x(t)$ and actions $a(t)$ as well as an utility function U for evaluating inputs $x(t)$.

After random initialization of C, M, A, U, and Q, at each time step t the following algorithm is performed:

1. Randomly select $p \in [0, \ldots, 1]$. If $p \leq \mu \in [0, \ldots, 1]$ then $a(t) = o_A(t)$ else $a(t)$ is chosen randomly.
2. Compute $o_M(t)$, execute $a(t)$, obtain $x(t+1)$, and adjust M to improve its prediction in similar situations. Adjust C according to one of the methods described in section 2. Obtain $r(t) = r_{ext}(t) + \triangle o_C(t)$, where $r_{ext}(t)$ is the current externally defined reinforcement (if there is any) and where $\triangle o_C(t)$ is the current change of confidence in M's current predictions.
3. Set $Q(x(t), a(t)) \leftarrow (1-\alpha)Q(x(t), a(t)) + \alpha(r(t) + \gamma U(x(t+1)) - Q(x(t), a(t)))$, where α is a learning rate and $0 \leq \gamma \leq 1$.
4. Adjust A to emit a in response to $x(t)$ such that $Q(x(t), a) = max_b Q(x(t), b)$.
5. $U(x(t)) \leftarrow Q(x(t), a)$.

Note that the algorithm does not specify the implementation of C, M, and A. All three can be implemented as lookup tables or (in hope for useful 'generalizations') as back-propagation networks, Boltzmann-machines, etc. Q and U may be replaced by back-propagation networks, too (see the experiments described in section 5).

4 PREDICTING ERROR CHANGES DIRECTLY

The reinforcement generating mechanism for the reinforcement learning systems described above can be modified in various ways. For instance, define $\bar{o_M}(t)$ as M's response to $i_M(t)$ *after* having adjusted M at time t. We can replace the confidence network by a network H which at every time step receives the current input $i_M(t)$ and whose target output is the current *change* of M's output $\triangle o_M(t) = o_M(t) - \bar{o_M}(t)$ caused by M's learning algorithm (H should have a small learning rate). H will learn approximations of the expectations

$$E\{\triangle o_M(t) \mid i_M(t)\}$$

of the changes of M's responses to given inputs. The *absolute value* $\mid o_H(t) \mid$ of H's output $o_H(t)$ (an approximation of $\mid E\{\triangle o_M(t) \mid i_M(t)\} \mid$) should be taken as the reinforcement for the adaptive critic or the Q-learning algorithm (the reinforcement learning algorithm does not have to be specified here): The control system's curiosity goal at time t_0 is to maximize

$$E\{\sum_{t \geq t_0} -\gamma^{t-t_0} \mid o_H(t) \mid\},$$

where $0 \leq \gamma < 1$ is a discount rate. An alternative would be to make predictions about the (discounted) sum of future changes of M's *weight vector* and use these predictions in an analoguous manner.

5 EXPERIMENTS

A 'curious' adaptive agent based on Watkins' Q-learning method was tested in artificial non-deterministic discrete-state environments. M (the world model), C, a controller A, and a module for evaluating pairs of environmental states and actions Q were implemented as general back-propagation networks.

The agent was able to move around in a two-dimensional world with 100 different states. The environment was reactive. M's task was to predict the reactions of the environment which were partly random and partly deterministic.

The 'curious' system was tested against the conventional random search method. With both methods, at time t the sum $E(t)$ of the squared differences between the values of the possible *deterministic* reactions and the corresponding predictions of M was used as a criterion for judging the quality of M.

With guidance by the principle of adaptive curiosity $E(t)$ decreased up to 10 times faster than with random search (see [6] for details). The reason for this superior performance was that the 'curious' system soon found out that there were certain states of the environment where further performance improvement of M could be expected. *It started to focus on these particular states.* The random search method was not selective at all, therefore it wasted a lot of time on senseless exploration of states of the environment that did not allow performance improvement.

The more complex the environment the more benefits should be expected from the principle of adaptive curiosity. Ongoing experiments focus on increasingly complex worlds, non-local input/output representations and on the expected 'generalization capabilities' of non-trivial networks with hidden units.

6 CONCLUSION

The central idea of this paper is to construct an adaptive system which *learns to predict the effects of further learning.* This is done by training an adaptive sub-module to predict (the expectation of the sum of) future error *changes* caused by a particular learning algorithm. Here one adaptive module learns to make estimates about the effects of the learning procedure of another adaptive module. In other words, there is a module which learns to make a statement about learning itself. This is related to the concept of *'meta-learning'*: In a very limited sense the system learns how to learn.

The method represents a general strategy for learning to select training examples such that the expected performance improvement is maximized. Therefore the usefulness of the approach is not limited to model-building control systems. The principles above are general enough to be of interest whenever the task is to select appropriate training examples for any kind of learning system.

7 ACKNOWLEDGEMENTS

I wish to thank Reiner Wahnsiedler who conducted the experiments.

References

[1] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13:834–846, 1983.

[2] M. I. Jordan and D. E. Rumelhart. Supervised learning with a distal teacher. Technical Report Occasional Paper #40, Center for Cog. Sci., Massachusetts Institute of Technology, 1990.

[3] Nguyen and B. Widrow. The truck backer-upper: An example of self learning in neural networks. In *Proceedings of the International Joint Conference on Neural Networks*, pages 357–363. IEEE Press, 1989.

[4] J. Schmidhuber. Dynamische neuronale Netze und das fundamentale raumzeitliche Lernproblem. Dissertation, Institut für Informatik, Technische Universität München, 1990.

[5] J. Schmidhuber. Talk at the NIPS'90 workshop on dynamic networks led by R. Rohwer, 1990.

[6] J. Schmidhuber. Adaptive curiosity and adaptive confidence. Technical Report FKI-149-91, Institut für Informatik, Technische Universität München, April 1991.

[7] J. Schmidhuber. Adaptive decomposition of time. In T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, editors, *Artificial Neural Networks*, pages 909–914. Elsevier Science Publishers B.V., North-Holland, 1991.

[8] J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In J. A. Meyer and S. W. Wilson, editors, *Proc. of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pages 222–227. MIT Press/Bradford Books, 1991.

[9] J. Schmidhuber. Reinforcement learning in Markovian and non-Markovian environments. In D. S. Lippman, J. E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems 3*, pages 500–506. San Mateo, CA: Morgan Kaufmann, 1991.

[10] J. Schmidhuber and R. Huber. Learning to generate artificial fovea trajectories for target detection. *International Journal of Neural Systems*, 2(1 & 2):135–141, 1991.

[11] R. S. Sutton. Integrated architectures for learning, planning and reacting based on dynamic programming. In *Machine Learning: Proceedings of the Seventh International Workshop*, 1990.

[12] S. Thrun and K. Möller. On planning and exploration in non-discrete environments. Technical report, Gesellschaft für Mathematik und Datenverarbeitung, D-5205 St. Augustin, Germany, March 1991.

[13] C.J.C.H Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, Oxford, 1989.

[14] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* PhD thesis, Harvard University, 1974.

[15] P. J. Werbos. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man, and Cybernetics*, 17, 1987.

[16] R. J. Williams. Toward a theory of reinforcement-learning connectionist systems. Technical Report NU-CCS-88-3, College of Comp. Sci., Northeastern University, Boston, MA, 1988.