Recurrent Transition Hierarchies for Continual Learning: A General Overview

Mark Ring
IDSIA / University of Lugano / SUPSI
Galleria 2
6928 Manno-Lugano, Switzerland
Email: mark@idsia.ch

Abstract

Continual learning is the unending process of learning new things on top of what has already been learned (Ring 1994). Temporal Transition Hierarchies (TTHs) were developed to allow prediction of Markov-k sequences in a way that was consistent with the needs of a continual-learning agent (Ring 1993). However, the algorithm could not learn arbitrary temporal contingencies. This paper describes Recurrent Transition Hierarchies (RTH), a learning method that combines several properties desirable for agents that must learn as they go. In particular, it learns online and incrementally, autonomously discovering new features as learning progresses. It requires no reset or episodes. It has a simple learning rule with update complexity linear in the number of parameters.

Overview

Continual learning is the unending process of learning new things on top of what has already been learned (Ring 1994). Temporal Transition Hiererchies (TTHs) were introduced as a continual-learning method for non-Markov reinforcement domains (Ring 1993). They allow a learning agent to build a hierarchy of skills incrementally (when mapping observations to actions), but are equally suited to modeling an environment's dynamics (mapping arbitrary inputs to outputs).

The TTH algorithm was the first that allowed learning of long time lags by incrementally expanding the length of the history relevant to prediction and decision making. Though fast at learning k-order Markov sequences, the algorithm cannot model arbitrary temporal dependencies. The power of the networks increases enormously when the network is augmented with recurrent connections, which allow information to be retained over indefinite time intervals.

The resulting algorithm, Recurrent Transition Hierarchies (RTH) has many appealing properties: it is fully incremental, can reuse internal structure hierarchically, is easily extensible, can produce non-linear decision boundaries, does online gradient descent, keeps only a minimal history (n-steps for hierarchies of height n), and self constructs with no prior knowledge of state information, including knowledge of the number of states (in fact, the algorithm has no prespecified notion of state, describing all its interaction purely in terms of sensory-motor experience).

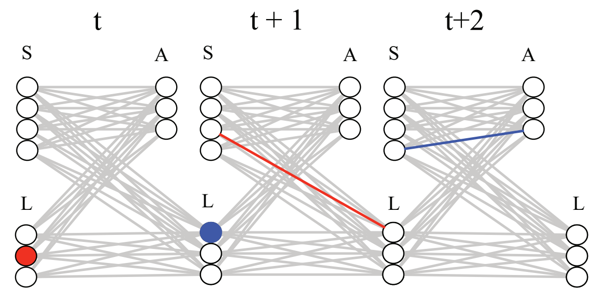

Figure 1: RTH network with four observation/input units (S), three output/action units (A), and three high-level units (L) over three time steps. At each step, values are propagated forward. Each high-level unit also modifies one specific connection at the following time step; e.g., the red L unit modifies the red connection; the blue L unit modifies the blue connection.

Formal Description

Temporal Transition Hierarchies are linear function approximators, augmented with higher-order units that detect and respond to temporal events. The network is composed of three kinds of units: observation/input units ($o \in O$), action/output units ($a \in A$), and high-level units ($l \in L$). The A and L units are collectively referred to as "non-input" units, ($n \in N = \{A \cup L\}$). The values of the high-level units are computed by the network at every time step and serve to gate the flow of information from observation units to non-input units depending on temporal contingencies. In particular the value of each non-input unit is calculated as:

$$n^i(t+1) = \sum_j \hat{w}_{ij}(t+1) o^j(t). \tag{1}$$

where n^i is the i^{th} non-input unit, o^j is the j^{th} observation unit, and the weight from j to i might be modified dynamically by a dedicated l unit:

$$\hat{w}_{ij}(t+1) = \begin{cases} w_{ij} + l_{ij}(t) & \text{if } \exists\, l_{ij} \\ w_{ij} & \text{otherwise.} \end{cases} \tag{2}$$

At most one l unit is assigned to a connection weight. The w_{ij}'s are the parameters of the learning system and are modified by online gradient descent to minimize the error, E.

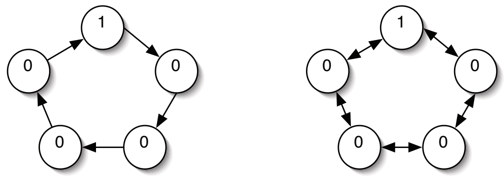

Figure 2: The *cycleworld* from Tanner and Sutton (2005) (left), and the *ringworld* from the same paper. The agent's observation is shown inside the state (circle); the agent receives the observation upon landing in the state.

Task	RTHs	TD(λ) Network
cycleworld 6	80	5,000
ringworld 5	1,900	5,000
ringworld 8	65,000	125,000

Figure 3: Steps to reach RMSE values of 0.05 or less on the given tasks. RTH values are averages over 50 runs.

Weights are changed on every step,

$$w_{ij}(t+1) = w_{ij}(t) - \alpha \Delta w_{ij}(t)$$

$$\Delta w_{ij}(t) = \sum_{\tau=0}^{t} \frac{\partial E(t)}{\partial w_{ij}(\tau)} \ ,$$

where α is the learning rate. The computation is straightforward and is linear in the number of connections.

Each l unit is trained to predict the next *error* of the weight it modifies; i.e., if a high-level unit l_{ij} is assigned to weight w_{ij}, then its target at t is $\Delta w_{ij}(t+1)$. Because the prediction occurs one step in advance of the error, it can be used to modify the weight at the following time step, as in Equation 2, to dynamically reduce the connection's contribution to the global error. As the prediction improves, the error due to that connection is reduced. Because each l unit also faces a prediction task, the full functionality of the network is brought to bear on it. This recursive definition allows a simple process for feature discovery and node creation that builds new units for weights with large persistent error. An important aspect of TTHs and RTHs is that new units can be introduced without disturbing the functioning of the network already present, reducing catastrophic forgetting.

Illustration

"Cycleworld" (Figure 2, left) is a deterministic environment with k states, one of which generates observation 1 (all others generate 0). The agent moves one state clockwise at every step. "Ringworld" (Figure 2, right) has the same structure but is stochastic: with equal probability the agent moves clockwise or counter-clockwise.

"Cycleworld" can be learned with standard TTHs, because it is a Markov-k environment and can be predicted with information from a fixed number of previous steps. Results for 6 states are given in Table 3.

"Ringworld," shown in two different sizes, is far more challenging than it seems at a casual glance, because the agent can take an arbitrary number of steps between sightings of a "1". The results show that the representation constructed by the RTH is sufficient for prediction in these cases. It is somewhat surprising that the algorithm actually learns faster than the TD(λ) Network, whose structure was hand-designed specifically for this task. In fact, much of the RTH learning process is spent building the network structure itself. An average of 81 units were created for the Ringworld of size 5, with a standard deviation of 5.3; the Ringworld of size 8 was capped at 190 new units, which was always reached during training.

In other experiments (not shown), the network has demonstrated the ability to map arbitrary, factored input-output vectors and can learn to separate and model multiple, simultaneous dynamic processes.

Discussion

Each RTH high-level unit is constructed to fill a very specific role: it must predict the error of the weight that it modifies. These predicted errors are then useful as inputs to the rest of the network and are sufficient for allowing the system to retain information over an indeterminate time period.

RTH networks have been proven capable of modeling any POMDP (proof still unpublished). They construct units without causing catastrophic forgetting; build new structure during learning; learn incrementally and online, and can grow in size for as long as the agent survives. In fact, they have all the same properties as TTHs and are therefore exactly adapted to continual learning, yet with the additional ability to learn arbitrary temporal dependencies.

Furthermore, RTH is fully incremental, can build and reuse internal structure hierarchically, is easily extensible, can produce non-linear decision boundaries, does online gradient descent, keeps only a minimal history (n-steps for hierarchies of height n), and self constructs with no prior knowledge of state information (including knowledge of the number of states), and can without modification predict multiple simultaneous processes.

Acknowledgments This work was funded in part by the following grants to J. Schmidhuber: EU project FP7-ICT-IP-231722 (IM-CLeVeR) and SNF Project 200020-122124.

References

Ring, M. B. 1993. Learning sequential tasks by incrementally adding higher orders. In Giles, C. L.; Hanson, S. J.; and Cowan, J. D., eds., *Advances in Neural Information Processing Systems 5*, 115–122. San Mateo, California: Morgan Kaufmann Publishers.

Ring, M. B. 1994. *Continual Learning in Reinforcement Environments*. Ph.D. Dissertation, University of Texas at Austin, Austin, Texas 78712.

Tanner, B., and Sutton, R. S. 2005. Td(λ) networks: temporal-difference networks with eligibility traces. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, 888–895. New York, NY, USA: ACM Press.