Temporal Difference Learning for Recurrent Neural Networks

Risheek Garrepalli (garrepar@oregonstate.edu)
School of EECS,Oregon State University

Abstract

Truncated back-propagation through time(TBPTT) is one of the most common methods for training artificial recurrent neural networks(RNNs) for temporal credit assignment(TCA). There have been various proposed theories on how neural circuits in the brain might approximate back-propagation algorithm in solving credit assignment problem for feed forward neural networks, but it remains unclear how the equivalent of TBPTT could be implemented in brains. Temporal difference learning with eligibility traces is a key universal approach used in reinforcement learning for multi-step value prediction problems(TCA). In this work, we apply TD learning with eligibility trace to train RNNs which is biologically plausible as it encodes errors locally,does not need a separate backward computation and hence does not have an issue of weight symmetry.

Keywords: Recurrent neural networks; BPTT; temporal difference learning; dopamine; predictive coding

Background

Predictive Coding (Rao & Ballard, 1999),(Friston & Kiebel, 2009) or brain as prediction machine is one of the dominant models of cognitive neuroscience, where all the neuronal dynamics and connectivity is optimized to minimize prediction error. Within this framework brain constructs internal models of the world and it performs constant inference to predict what happens next, this is one of core signal or global objective w.r.t which synaptic weight modifications are performed in brain. These internal models are essentially dynamical systems allows brains to perform multi-step predictions instead of just single step predictions, RNNs are one of the simplest models of dynamical systems in connectionist models with feedback loops and are potentially more close to biological neuronal systems. Hence in this work, we use RNNs to learn internal world models which would also enable deliberate planning and model-based reinforcement learning(RL).

Credit assignment in brain: The back-propagation of error algorithm (BP) is a fundamental credit assignment mechanism in artificial neural networks but it is impossible to implement in a real brain, but there are many BP inspired proposed theories (Lee, Zhang, Fischer, & Bengio, 2015),re-circulation, and related methods such as contrastive divergence, etc. for understanding how brain might approximate BP including a scalability evaluation of such approaches (Bartunov et al., 2018).

(Whittington & Bogacz, 2019) summarizes many of the recently proposed theories on how neural circuits in the brain might approximate BP, and they focus mainly on temporal error(difference) models such as equilibrium propagation (Scellier & Bengio, 2017) which encode error (locally) in the difference between activity phases and these account for spike-time-dependent plasticity (STDP). Another class of models are explicit error models which include predictive coding and dendritic error (Richards & Lillicrap, 2019) models which are inspired by properties of pyramidal neurons. But most of these approaches focus on energy-based models which compute gradients numerically unlike back propagation which computes analytical gradients.

Except equilibrium propagation there has not been much work which focused on biologically plausible TBPTT and it is unclear how brain might implement it.(Lillicrap & Santoro, 2019) points out that compared to BP there is even less conviction about whether and how BPTT can be implemented in brain, but they focus on broader TCA problem where they suggest use of attention and memory-based architectures for TCA.

Though we believe memory is a necessary component of the overall system (as suggested by (Kumaran, Hassabis, & McClelland, 2016)) we believe that much of prefrontal cortex(PFC) is a hierarchical recurrent neural network with attention effectively making it a complex dynamical system and hence credit assignment, TCA of this network is needed.

Dopamine for sensory prediction errors: Dopamine neurons have been thought to report reward prediction error(RPE) and this hypothesis has several lines of evidence, but there seems to be evidence that dopamine neurons respond to novel stimuli too.(Gardner, Schoenbaum, & Gershman, 2018) proposes to extend dopaminergic modulation for sensory prediction errors(SPE), and it has been pointed out that it is unclear what exactly dopamine is contributing to model-based learning because prediction errors do not require TD errors. They apply TD learning to successor representation in RL, which has more information than model-free RL but cannot support deliberate planning which is possible in the model-based framework.
In this work, we learn internal predictive models of the world in recurrent neural networks and apply TD learning with eligibility traces for TCA, and hence it supports a hypothesis that dopamine neurons may contribute to SPEs and model-based RL,and it is also consistent with three factor learning rule.

Temporal difference learning:

(Sutton & Barto, 2018),(Sutton, 1988)Temporal error models encodes errors in differences in neural activity across time and that is exactly how temporal difference(TD) methods work too, the conventional (explicit-error) prediction learning methods assign credit by means of difference between predicted and actual observations instead TD methods assign credit by means of difference between temporally successive predictions and hence allow incremental learning and also local

learning. In multi-step prediction problems, partial information relevant to correctness of prediction is revealed at each step and TD uses this information. *The key idea of TD(0) is to represent multi-step prediction error as the sum of changes in predictions* as illustrated in fig(1).

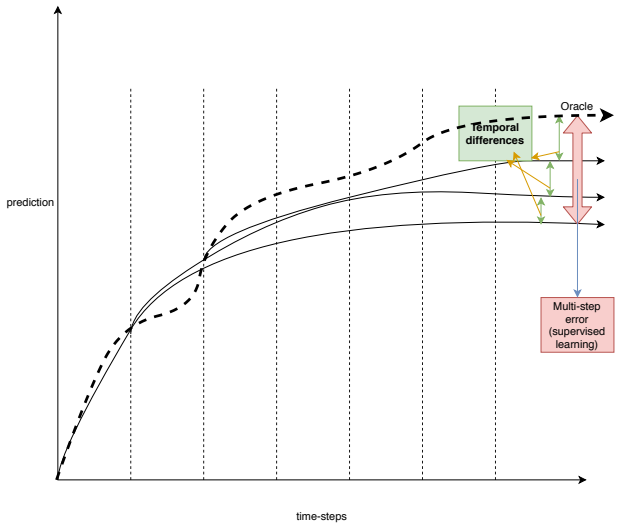

Figure 1: Multi-step erros as sum of TD errors

Multi-step prediction setup

Let us consider a multi-step prediction problem in which experience comes in observation sequences of form $S_{t=1}^{(j=0)}, S_{t=2}^{(j=0)}, S_{t=3}^{(j=0)}, S_{t=4}^{(j=0)}, S_{t=5}^{(j=0)}$ where each observation at time 't' in the sequence is $S_t^{(j=0)}$.
For any fixed 'k'$(k > n)$, for each observation the RNN model has corresponding predictions of future states $\hat{S}_{t+k}^{(j=n)}..\hat{S}_{t+k}^{(j=2)}, \hat{S}_{t+k}^{(j=1)}$ where 'k' is the future time-step at which we are predicting observation and 'j' is the number of prediction dependent transitions(see below),for true observations $'j = 0'$, 'n' is the number of steps involved in temporal error calculation (*eligibility trace*).
so any observation trajectory is represented by $S_{t=1}^{(0)}, S_{t=2}^{(0)}..S_{t=T}^{(0)}$ and corresponding predictions for any time-step by $\hat{S}_t^{(1)}, \hat{S}_t^{(2)}, \hat{S}_t^{(3)}, ..., \hat{S}_t^{(n)}$,where $S_t^{(0)}$ represents the true observation from the world.
And let *T* be the total number of time steps in a trajectory.

RNN as internal model of environment We consider a setup similar to (Chiappa, Racanière, Wierstra, & Mohamed, 2017),etc. where an RNN is used to learn environment model and hence it allows us to have not just single-step predictions but multi-step predictions. It is important to point out there are two kinds of transitions which can happen in RNNs one is an open loop transition and other is closed loop, i.e. at any given time step depending on whether RNN has access to true observation or if it uses its own prediction as input for transition/rollout in multi-step prediction setting RNN performs prediction dependent transitions. These two kinds of transitions are illustrated in fig(2), fig(3).
Let h_t be the hidden state of the recurrent network, at any given time-step it takes previous hidden state, current observation and predicts the next observation.

Two kinds of transitions in RNN Let *f(.)* be the RNN,then Observation dependent transitions are

$$\hat{S}_{t+k}^{(j=1)}, h_{t+k} = f(\theta, h_{t+k-1}, S_{t+k-1}^{(j=0)}) \tag{1}$$

Prediction dependent transition are

$$\hat{S}_{t+k}^{(j=n)}, h_{t+k} = f(\theta, h_{t+k-1}, \hat{S}_{t+k-1}^{(j=n-1)}) \tag{2}$$

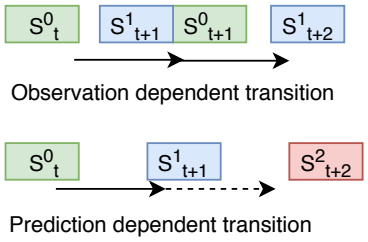

Figure 2: Different kinds of transitions

Objective Function:

Multi-step prediction or TD(1) Loss: It is slightly different to traditional monte carlo or TD(1) update we apply in RL where we only have access to target at the end of each episode, here similar to (Venkatraman, Hebert, & Bagnell, 2015),(Bengio, Vinyals, Jaitly, & Shazeer, 2015) we have access to target for each of multi-step prediction.

$$loss = \sum_{t=1}^{T} \sum_{j=1}^{n} ||S_t^{(0)} - \hat{S}_t^{(j)}|| \tag{3}$$

This can be considered as a case where $\lambda = 1$ in TD(λ) setting.

Temporal Difference or TD(0) Loss: Temporal error models or TD(0) setup is exactly similar to traditional setup in RL, but instead of value prediction here we predict the next or future observations given current observation

$$loss = \sum_{t=1}^{T} \sum_{j=1}^{n} ||\hat{S}_t^{(j-1)} - \hat{S}_t^{(j)}|| \tag{4}$$

This can be considered as a case where $\lambda = 0$ in TD(λ) setting.
The red arrow in fig(3) represents a TD error where a 3-step rollout is acting as a TD target to 4-step rollout, and the difference between h_{t5}^3, h_{t5}^4 is temporal difference(TD) or temporal error for each hidden state(neuron) in RNN, hence is a local learning rule.

Training objective: Combining TD(0) and TD(1) For simplicity we will consider a special case of combining returns(TD targets) in TD(λ) setting where we are averaging for $\lambda \in (0,1)$ but we would not be considering other values of 'n' in n-step TD(λ) setting, this can also be interpreted as one-step TD regularization added to traditional multi-step objective.γ is the discount factor for recency. Please note that the value of 'n' is the length of eligibility trace which enables us to solve TCA.

$$loss = \sum_{t=1}^{T} \sum_{j=1}^{n} \gamma^{j-1} (\lambda ||S_t^{(0)} - \hat{S}_t^{(j=1)}|| + (1-\lambda) ||\hat{S}_t^{(j-1)} - \hat{S}_t^{(j)}||) \tag{5}$$

Experiments

We consider a simple task where a single layer RNN with 300 nodes is trained to learn dynamics of a simple pendulum(2 dimensional), the input to the RNN is position and velocity of the pendulum and each trajectory/episode length is 1500 time-steps. In all our experiments during training and evaluation, we have a warm-start period of 50 time-steps where the RNN has access to true observations at each time-step and learning is not applied.

An important point to note is that we are using back-propagation to train the RNN from input to output per time step i.e. though we do not perform BPTT and use TD learning as a framework to solve TCA we still use BP to update parameters of the RNN w.r.t overall objective,we allow gradient to back-propagate only one time-step by detaching all previous variables in pytorch framework(as illustrated by red line in fig(3)). Blue, green arrows in fig(3) are explicit(supervised) errors for each hidden node (in our experiments we directly get the back-propagated gradient from the output layer to account for explicit errors). We use $\lambda = 0.5$ for all experiments

Evaluations *reported in Table(1)* We evaluate the RNN on two different multi-step prediction settings.

The first one is traditional multi-step setting, where RNN would have access to information of all the observations until the current state and then we evaluate for different step-lengths how accurate are predictions, step-length varies from traditional one-step prediction to 50 step prediction and these are reported by 'length of prediction' in the below table(1).

The second set of evaluations is on increasing complexity of state estimation problem i.e. RNN has access to true observation at some frequency i.e. if 'input sampling frequency' $= 1$ then it is traditional one-step prediction but if sampling frequency is '5' then RNN looks at true observation only once in 5 steps and hence would have $1-step, 2-step, 3-step, 4-step, 5-step$ predictions. This set of evaluations can be considered as the geometric mean of different step-lengths in multi-step predictions. We vary the sampling frequency from $'1' to '100'$, in the extreme case of 100 RNN has access to only 15 observations in the entire 1500 time-step trajectory after warm-start.

'k1' in table(1) of TBPTT represents number of time steps gradient is back-propagated,*we would like to point out that there isn't lot of hyper parameter tuning done w.r.t λ, γ and etc.*

Summary of results: Here we focus on long term prediction,sampling frequency is low,and also there seems to inherent tradeoff between shor-term and long-term prediction accuracy.

- n-step(n=5) performs better than n=1 for both combined loss, TD(1) and hence it shows evidence that eligibility trace improves TCA.
- TD(0)+TD(1) performs better than TBPTT potentially because of vanishing gradient problem.
- TBPTT with scheduled sampling is the best performing model as is also the gold standard, but for very long term prediction or very low sampling frequency TD(0)+TD(1) works better potentially because of self consistency and variance reduction property of TD methods.
- The proposed approach obtains an analytical gradient instead of numerical gradient which was focus of most previous work as they were based on energy based models.

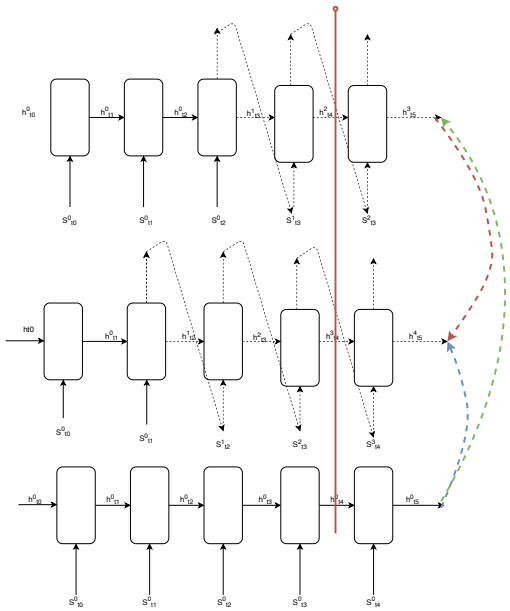

Figure 3: Different roll-outs of RNN with varying input information

Discussion and Future work

In this work, we applied TD learning for TCA for generic prediction problem on one layer RNN, on a simple domain. As the proposed approach encodes error locally, does not require a backward pass in time, hence it can support stochastic communication. We would like to extend this framework to hierarchical RNNs and also evaluate on more complex domains to evaluate the scalability of this method. It is also important

Table 1: Mean square error(MSE) for different objectives and training setups

	TBPTT(k1=1)	TD(1),n=5	TD(0)+TD(1),n=5	TBPTT(k1=5)	TBPTT(k1=5)+TD(1)
Length of prediction	MSE	MSE	MSE	MSE	MSE
1	7	4	8	2	1
3	26	12	22	7	2
5	58	24	34	13	4
10	225	74	55	40	9
20	1003	273	80	144	24
30	2119	624	102	317	53
50	3465	1834	136	814	188
input sampling frequency	MSE	MSE	MSE	MSE	MSE
1	8	4	8	2	1
3	25	9	18	6	2
5	53	16	26	12	4
10	238	53	42	38	11
20	1122	329	65	183	41
30	1741	596	86	541	91
50	2650	1092	142	1110	272
100	3599	2559	282	1766	1218

to understand and analyze the computational and theoretical implications of the proposed approach. An important problem with learned computational models of the world is poor long term prediction and we believe this could be a useful step towards that.There is a need for energy efficient hardware for deep learning, and neuromorphic computing with spiking neural networks are one of the promising directions but currently we cannot implement BP on those platforms because BP needs end-end gradient computation, so research on biologically plausible approximations of BP may enable ubiquitous use of such hardware.

References

Bartunov, S., Santoro, A., Richards, B. A., Marris, L., Hinton, G. E., & Lillicrap, T. P. (2018). Assessing the scalability of biologically-motivated deep learning algorithms and architectures. In *Proceedings of the 32nd international conference on neural information processing systems* (pp. 9390–9400). USA: Curran Associates Inc.

Bengio, S., Vinyals, O., Jaitly, N., & Shazeer, N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th international conference on neural information processing systems - volume 1* (pp. 1171–1179). Cambridge, MA, USA: MIT Press.

Chiappa, S., Racanière, S., Wierstra, D., & Mohamed, S. (2017). Recurrent environment simulators. *CoRR*, *abs/1704.02254*.

Friston, K. J., & Kiebel, S. J. (2009). Predictive coding under the free-energy principle. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, *364 1521*, 1211-21.

Gardner, M. P., Schoenbaum, G., & Gershman, S. J. (2018). Rethinking dopamine as generalized prediction error. *bioRxiv*. doi: 10.1101/239731

Kumaran, D., Hassabis, D., & McClelland, J. L. (2016). What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in Cognitive Sciences*, *20*(7), 512 - 534.

Lee, D.-H., Zhang, S., Fischer, A., & Bengio, Y. (2015). Difference target propagation. In *Ecml/pkdd.*

Lillicrap, T. P., & Santoro, A. (2019). Backpropagation through time and the brain. *Current Opinion in Neurobiology*, *55*, 82 - 89. (Machine Learning, Big Data, and Neuroscience)

Rao, R. P. N., & Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, *2*, 7-8.

Richards, B. A., & Lillicrap, T. P. (2019). Dendritic solutions to the credit assignment problem. *Current Opinion in Neurobiology*, *54*, 28 - 36. (Neurobiology of Learning and Plasticity) doi: https://doi.org/10.1016/j.conb.2018.08.003

Scellier, B., & Bengio, Y. (2017). Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, *11*, 24. doi: 10.3389/fncom.2017.00024

Sutton, R. S. (1988, Aug 01). Learning to predict by the methods of temporal differences. *Machine Learning*, *3*(1), 9–44.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. USA: A Bradford Book.

Venkatraman, A., Hebert, M., & Bagnell, J. A. (2015). Improving multi-step prediction of learned time series models. In *Proceedings of the twenty-ninth aaai conference on artificial intelligence* (pp. 3024–3030). AAAI Press.

Whittington, J. C., & Bogacz, R. (2019). Theories of error back-propagation in the brain. *Trends in Cognitive Sciences*, *23*(3), 235 - 250.