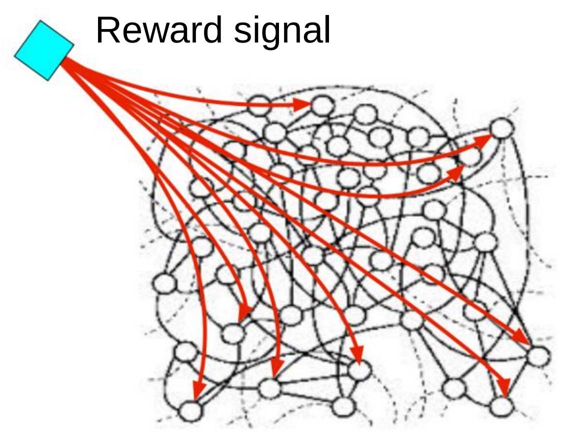
Faster Learning with a Team of Reinforcement Learning Agents

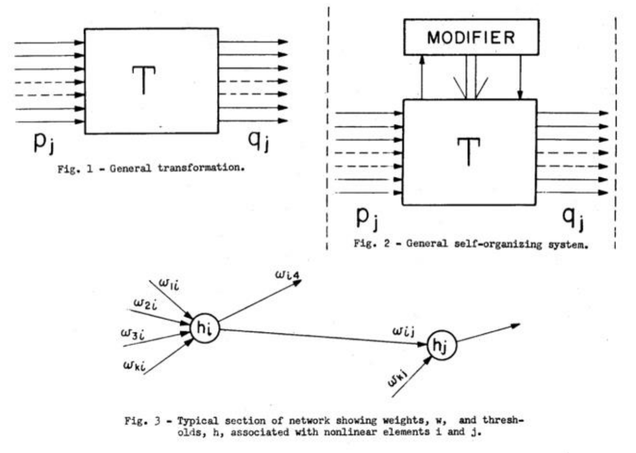
Stephen Chung, Andrew G. Barto

Though backpropagation underlies nearly all deep learning algorithms, it is generally regarded as being biologically implausible. An alternative way to train an artificial neural network is through making each unit stochastic and treating each unit as a reinforcement learning (RL) agent, and thus the network is considered as a team of agents. As such, all units can learn via REINFORCE, a local learning rule modulated by a global reward signal that is more consistent with biologically observed forms of synaptic plasticity. However, this learning method suffers from high variance and thus very low speed of learning. The high variance stems from the lack of effective structural credit assignment. This paper reviews two recently proposed algorithms to facilitate structural credit assignment when all units learn via REINFORCE, namely *MAP Propagation* and *Weight Maximization*. Experiments show that both algorithms can learn significantly faster than a network of units learning via REINFORCE. In contrast to backpropagation, both algorithms retain certain biologically plausible properties of REINFORCE, such as having local learning rules and the ability to be computed asynchronously. Therefore these algorithms may offer insights for understanding possible mechanisms of structural credit assignment in biological neural systems.

Background: Trainingng an ANN without backprop

Though it is common to use backprop to train an artificial neural network (ANN), backprop is generally regarded as biologically implausible. Alternatively, we can **view an ANN as a team of reinforcement learning (RL) agents** by letting each unit implement an RL algorithm that learns via the same global reward signal[1] broadcast uniformly to all the units. In the simplest case, the units explore independently by means of independent random noise injected into their actions. Here we use REINFORCE as the RL algorithm for each unit, and we call this ANN learning method Global REINFORCE. Williams [1] showed that for feedforward ANNs, this method changes weights according to an unbiased estimate of the gradient of the expected global reward signal.




MODIFIER
T
p_j
q_j
Fig. 1 - General transformation.
Fig. 2 - General self-organizing system.
Fig. 3 - Typical section of network showing weights, w, and thresholds, h, associated with nonlinear elements i and j.

Farley and Clark (1954)

A Long History - Farley and Clark (1954) implemented a team of RL units in the first simulation of ANN learning on a digital computer. Tsetlin (1973) introduced teams of RL agents (called learning automata); Narendra and Thathachar (1974, 2012) surveyed following work; Barto (1985, 1986) explored this approach to training multi-layer ANNs; and Barto and Jordan (1987) presented this approach as an alternative to error back propagation, which had appeared that same year. The simplicity of the approach, along with its slow learning rate, led some to call it the "naive method" for training ANNs.

Biological Plausibility - Global REINFORCE, when applied to networks of Bernoulli-logistic units, gives a three-factor learning rule which depends on a reward signal in addition to a unit's input and output signals. This three-factor learning rule is very similar to reward-modulated spike-timing-dependent plasticity (R-STDP) observed biologically.

Slow Learning - However, Global REINFORCE is seldom used in practice due to its large variance and thus the low learning speed. **Thus, the goal of the paper is to review algorithms that improve the learning speed of global REINFORCE** while retaining some degree of biological plausibility.

[1]Note that the reward signal can be replaced with the TD error for RL tasks or the negative loss for supervised learning tasks

Algorithm: Improving Structural Credit Assignment

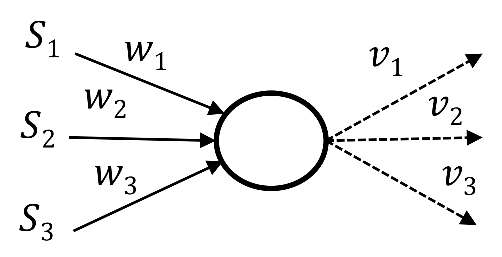
We propose two algorithms, namely ***maximum a posteriori (MAP) propagation*** **(Chung, 2021)** and ***Weight Maximization*** **(Chung, 2022)** , that reduce the variance of global REINFORCE efficiently.

MAP propagation - MAP propagation replaces the actions of hidden agents (i.e., agents that are not in the last layer) with their most probable action conditioned on the action of output agents (i.e., agents in the last layer) and the state, before applying global REINFORCE. Equivalently, defining the energy function as the negative log-probability of agents' action conditioned on the state, MAP propagation minimizes the energy function of the network w.r.t. actions of hidden agents before applying REINFORCE.

Weight Maximization - We define the ***outgoing weight*** of hidden agent i as the vector of weights connecting from that agent to agents in the next layer. Weight Maximization replaces the external reward signal R to each hidden agent by the change in the squared L^2 norm of its outgoing weight. This is based on the heuristic that this norm roughly reflects the contribution of the agent in the network. For example, if the hidden agent on layer $L-1$ is useful in guiding action, then the output agent should learn a large weight associated with it. With the replaced reward signal, each hidden agent is trying to maximize the norm of its outgoing weight, or intuitively, its contribution within the network. This allows a more targeted structural credit assignment by giving each hidden unit a different reward signal. **We prove that Weight Maximization is approximately following gradient ascent in rewards in expectation.**

Example – A single Bernoulli-logistic unit

1. We sample the action from a Bernoulli distribution $A \sim Ber(p)$ where $p = \sigma(\sum_i w_i S_i)$; this action A is multiplied by the outgoing weights v and passed to the outgoing units
2. As the outgoing units learn[2], the outgoing weight is updated from v to v'; we then compute the reward signal R as the change in the squared L^2 norm of the outgoing weight v by: $R = \sum_i v_i'^2 - \sum_i v_i^2$
3. We apply REINFORCE to update the incoming weight $w : w' = w + \alpha R \nabla_w \log Pr(A|S;w) = w + \alpha R(A-p)$



Notation -

S	Input (Vector)
A	Action (Binary)
R	Reward (Scalar)
w	Incoming weight (Vector)
v	Outgoing weight (Vector)
α	Update step size (Scalar)
σ	Sigmoid function

[2]The actual reward signal requires a slight adjustment to stabilize learning; see paper [8] for details

Discussion

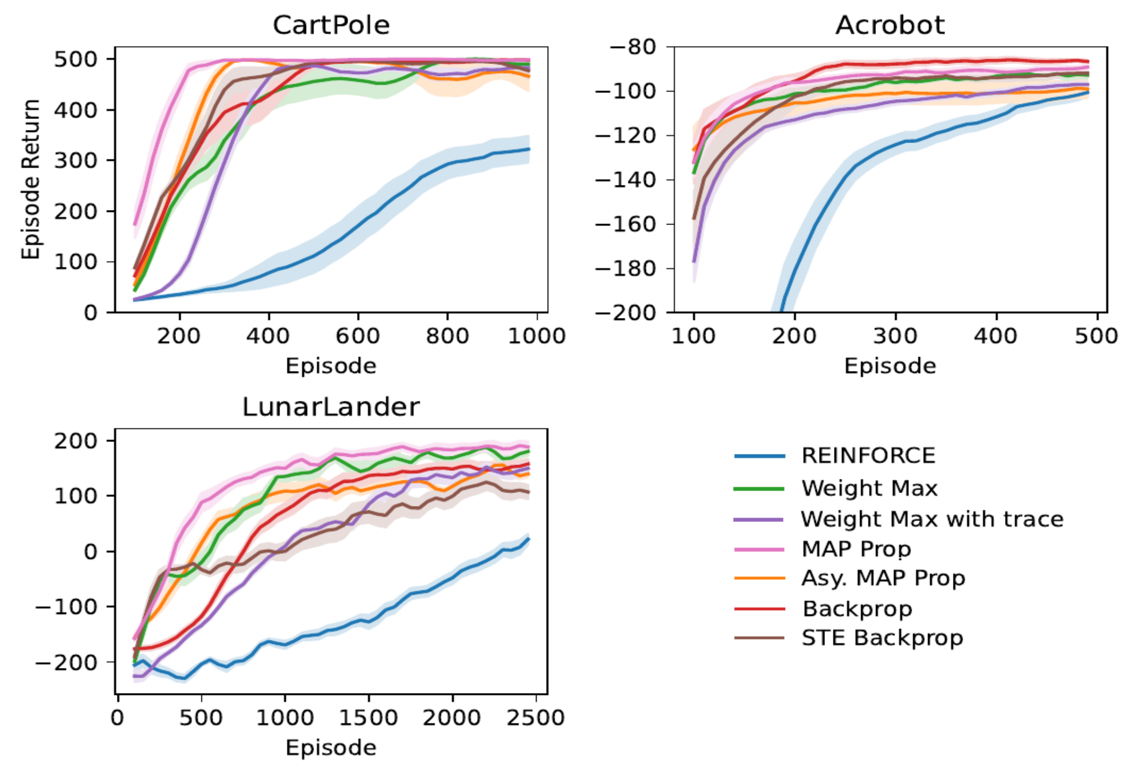
- We also proposed asynchronous versions of MAP Propagation and Weight Maximization, called asynchronous MAP prop and Weight Maximization with traces respectively
- Derived from global REINFORCE, both algorithms retain certain properties relevant to biological plausibility of REINFORCE:

Learning Rule	Local learning rule	No symmetric feedback connections	Asynchronous computation across units
Global REINFORCE	✓	✓	✓
MAP Prop.	✓	✗	✗
Asy. MAP Prop.	✓	✗	✓
Weight Max.	✓	✓	✗
Weight Max. with traces	✓	✓	✓
Backprop	✗	✗	✗

- Computational advantages – agents trained by MAP prop have more effective exploration than when RL relies on backprop for learning policies or value functions; Weight Max. can be applied to train discrete-valued units instead of only differentiable units

Experiment Results

- Applied both algorithms to train an actor-critic network in three RL tasks
- The actor-critic network is a two-hidden-layer ANN with 64 and 32 units in the first and the second hidden layer respectively
- Both algorithms are significantly faster than the Global REINFORCE baseline, and the learning speed is comparable to backprop



Running average returns over the last 100 episodes in four RL tasks. REINFORCE, Weight Max and STE backprop use Bernoulli-logistic units; MAP Prop uses normally distributed units, and backprop uses Rectified Linear Units (ReLU).

References

Barto, A.G. (1985) Learning by statistical cooperation of self-interested neuron-like computing elements. *Human Neurobiology*, 4: 229-256.

Barto, A.G. (1986) Game-theoretic cooperativity in networks of self-interested units. In J. S. Denker (Ed.), *Neural Networks for Computing*, 151(1): 41-46. AIP Publishing. American Institute of Physics, NY.

Barto, A.G., & Jordan, M.I. (1987) Gradient following without back-propagation in layered networks. In *Proceedings of the First IEEE Annual Conference on Neural Networks*, San Diego, CA, pp. II-629-II-636.

Chung, S. (2021) Map propagation algorithm: Faster learning with a team of reinforcement learning agents. In *Advances in Neural Information Processing Systems*, vol. 34.

Chung, S. (2022) Learning by competition of self-interested reinforcement learning agents," in *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*.

Farley, B. G., Clark, W. A. (1954). Simulation of self-organizing systems by digital computer. *IRE Transactions on Information Theory*, 4(4), 76-84.

Narendra, K. S., & Thathachar, M. A. (1974). Learning automata-a survey. *IEEE Transactions on systems, man, and cybernetics*, (4), 323-334.

Narendra, K. S., & Thathachar, M. A. (2012). *Learning automata: an introduction*. Courier corporation.

Tsetlin, M. L. (1973). *Automaton theory and modeling of biological systems*. Academic Press, New York.