Learning by Competition of Self-Interested Reinforcement Learning Agents

Stephen Chung

An artificial neural network can be trained by uniformly broadcasting a reward signal to units that implement a REINFORCE learning rule. Though this presents a biologically plausible alternative to backpropagation in training a network, the high variance associated with it renders it impractical to train deep networks. The high variance arises from the inefficient structural credit assignment since a single reward signal is used to evaluate the collective action of all units. To facilitate structural credit assignment, we propose replacing the reward signal to hidden units with the change in the L^2 norm of the unit's outgoing weight. As such, each hidden unit in the network is trying to maximize the norm of its outgoing weight instead of the global reward, and thus we call this learning method ***Weight Maximization***. We prove that Weight Maximization is approximately following the gradient of rewards in expectation. In contrast to backpropagation, Weight Maximization can be used to train both continuous-valued and discrete-valued units. Moreover, Weight Maximization solves several major issues of backpropagation relating to biological plausibility. Our experiments show that a network trained with Weight Maximization can learn significantly faster than REINFORCE and slightly slower than backpropagation. Weight Maximization illustrates an example of cooperative behavior automatically arising from a population of self-interested agents in a competitive game without any central coordination.

Background & Motivation

Training an ANN by REINFORCE 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 injecting stochastic noise to each unit in the ANN and treating each unit as an RL agent that explores independently. For example, in an RL task, when learning the policy function parametrized by an ANN, we can view the ANN as the policies of multiple RL agents instead of a single agent:

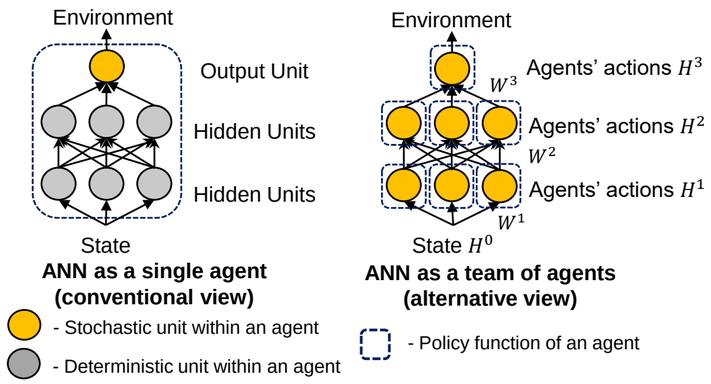

Fig 1 Illustration of viewing an ANN as a team of agents. In the alternative view, we treat each unit in an ANN as an agent. To each agent, all other agents are part of the black-box environment. For example, to the agents on the middle layer, the actions of agents on the first layer are considered as the state.

From this alternative view, we can pass the same reward from the environment to all agents and train them by REINFORCE [1,2]. Denoting the reward[2] by R and the step size by α, the learning rule becomes:

$$W^l \leftarrow W^l + \alpha R \nabla_{W^l} \log Pr\left(H^l \middle| H^{l-1}; W^l\right),$$

where $1 \leq l \leq L$ and L is the number of layers. We call this learning method global REINFORCE. Global REINFORCE gives an unbiased estimate of the gradient of reward [1,3], and is biologically plausible due to its similarity with reward-modulated STDP, a learning rule that is observed biologically.

[1]*Special thanks to Andrew G. Barto, who inspired this research and provided valuable comments on the research*

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

Weight Maximization

Reducing the variance of global REINFORCE

Despite its biological plausibility, 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 reduce the variance of global REINFORCE while retaining biological plausibility.

Key Ideas of Weight Maximization The parameter update in global REINFORCE has a large variance since a scalar signal R is used to evaluate the activation values of all units. Is it possible to replace this scalar reward signal R with another signal that is directly influenced by and tailored to individual unit?

Let define the ***outgoing weight*** of hidden unit i on layer l by the vector $W_{:,i}^{l+1}$, i.e. the weights connecting from that unit to units on the next layer[3]. A heuristic is to use the ***change in the L^2 norm of the unit's outgoing weight*** as the reward signal (note that the next layer is also learning, so the outgoing weight changes on every step). This is motivated by the idea that the norm of a unit's outgoing weight roughly reflects the contribution of the unit in the network (imagine that you are asked to remove a neuron from a network, and there is a neuron with a zero outgoing weight).

Details of Weight Maximization The new reward signal to hidden unit i on layer l can thus be expressed as (ΔW^{l+1} denotes the change in W^{l+1} before multiplying the step size α):

$$||W_{:,i}^{l+1} + \alpha \Delta W_{:,i}^{l+1}||_2^2 - ||W_{:,i}^{l+1}||_2^2 = 2\alpha \Delta W_{:,i}^{l+1} \cdot W_{:,i}^{l+1} + O(\alpha^2).$$

We propose to ignore $O(\alpha^2)$ since the step size α is usually very small. This leads to the following learning rule of Weight Maximization for all layer l and unit i:

$$R_i^l = \begin{cases} 2\alpha \Delta W_{:,i}^{l+1} \cdot W_{:,i}^{l+1} & for\ l \in \{1, 2, \dots, L-1\}, \\ R & for\ l = L, \end{cases}$$

$$\Delta W_{i,:}^l = R_i^l \nabla_{W_{i,:}^l} log Pr\left(H_i^l \middle| H^{l-1}; W^l\right),$$

$$W^l \leftarrow W^l + \alpha \Delta W^l.$$

Note that we do not change the reward signal to the output unit. In the paper, we prove that Weight Maximization is approximately following gradient ascent in R in expectation.

[3] We use $A_{i,:}$ and $A_{:,j}$ to denote the i^{th} row and j^{th} column of the matrix A resp. For all l, we assume W^l is a matrix, and $Pr\left(H_i^l \middle| H^{l-1}; W^l\right) = \prod_i f(H_i^l, W_{i,:}^l H^{l-1})$ for some differentiable function f.

Properties of Weight Maximization

- Can be combined with eligibility traces to remove the iteration requirement and enable the algorithm to be implemented asynchronously across units (note that Weight Maximization requires iterating backward from the top layer when computing the reward signal);
- Can be applied to any discrete units (even black box discrete units) and continuous units, though experiments show that Weight Maximization works better with discrete units;
- The approximation error in gradient ascent can be mitigated by weight regularization / weight decay.

Learning Rule	Local learning rules	No symmetric feedback connections	Asynchronous computation across units
Backprop	×	×	×
Weight Max.	✓	✓	×
Weight Max. with traces	✓	✓	✓
Global REINFORCE	✓	✓	✓

Table 1. Comparison of the properties relating to biological plausibility.

Experiment Results

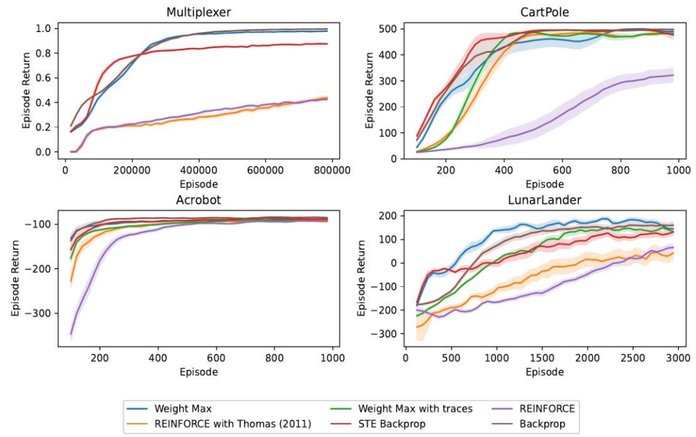

Fig 2 Episode returns in different RL tasks. All networks have 64 units on the first hidden layer and 32 units on the second hidden layer. Except backprop which uses standard ReLu units, all other algorithms use Bernoulli-logistic units.

- Paradoxical - every units maximizing their own interest (the norm of outgoing weight) also approximately maximize the interest of the whole network;
- Invisible hand in economy: society's interest as a whole can be maximized when individuals seek to maximize their own interest;
- Another relationship with biological neurons - when a brain area is damaged, the upstream neurons to that brain area will try to seek new target neurons to innervate.

References

[1] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement

learning. Machine learning, 8(3-4):229–256, 1992.

[2] Andrew G Barto. Learning by statistical cooperation of self-interested neuron-like computing elements. Human Neurobiology, 4(4):229–256, 1985.

[3] Philip S Thomas. Policy gradient coagent networks. In Advances in Neural Information Processing Systems, pages 1944–1952, 2011.