A More Biologically Plausible Learning Rule for Recurrent Neural Networks

Allen Wang
allenwang@utexas.edu

Max Gray
maxwellhgray@utexas.edu

Sarbartha Banerjee
sarbartha@utexas.edu

***Abstract*—Back-propagation is traditionally used to facilitate updating of weights in artificial neural networks. While successful, it is not biologically plausible in the brain. Feedback alignment and direct feedback alignment take steps towards formulating a more biologically plausible learning rule that supports learning in artificial neural networks. An autapse in the brain is capable of integrating its own activations. It therefore should be plausible that a feedback network with recurrent connections could use the integrated activity to re-weight its errors. We propose a new learning rule that re-weights the error in feedback alignment with its self-integrated activations.**

***Index Terms*—Feedback alignment, Direct feedback alignment, biological plausibility**

I. Introduction

For supervised learning in machine learning tasks, back-propagation [1] has traditionally been used to train neural networks. While recently successful in multiple tasks, it has been observed to not be biologically plausible in the human brain. Specifically, if a separate network is used to propagate errors downstream based on a forward propagated activation, back-propagation makes a strong assumption that the feedback neurons would have knowledge of all of the feedforward weights.

Various more biologically plausible alternatives to back-propagation including feedback alignment (FA) [3] and direct feedback alignment (DFA) [4] have been proposed. They have actually been shown to support learning in neural networks.

FA and DFA were shown to support learning in feedforward network applications, but to our knowledge, has not been explored in a recurrent neural network. Building upon the ideas in FA and DFA, we propose a new learning rule based on the characteristics of an autapse in pursuit of a more biologically plausible learning rule.

II. Related Work

A recurrent neural network can be "unfolded" over time, as if it were a feedforward network. However, in this "unfolded" network, the weight matrices must be fixed. Given this restriction, the gradient of a recurrent neural network can then be represented as

$$\Delta W_{ij}^{(t)} = \frac{1}{t}\sum_{l=1}^{t} \Delta W_{ij}^{(l)} \tag{1}$$

That is, given a sample at time t, an error is calculated, i.e., $\boldsymbol{e}^{(t)} = \hat{\boldsymbol{y}}(t) - \boldsymbol{y}(t)$, if the loss is Mean Squared Error, where $\hat{\boldsymbol{y}}(t)$ represents the output activation of the network and $\boldsymbol{y}$ represents the actual result. Various algorithms for calculating the individual gradients between unfolded layers is discussed in this section.

Back-propagation is the algorithm that is traditionally used to calculate the gradient of an error function with regards to the weights in a layer of a neural network. It is derived from the chain rule in Calculus. Its exact derivation is excluded from this discussion, but takes the form

$$\Delta W_{ij}^{(l)} = \eta \delta_i^{(l)} x_j^{(l)} \tag{2}$$

where

$$\delta_i^{(l)} = \boldsymbol{W}^T \delta_i^{(l+1)} \odot f'(\boldsymbol{a}^{(l)}) \tag{3}$$

where $\delta^{(l+1)}$ refers to the loss propagated from the next layer, $\boldsymbol{a}$ refers to the input to the layer, i.e., some $\boldsymbol{a}^{(l)} = \boldsymbol{W}\boldsymbol{x}(l) + \boldsymbol{b}(l)$, $x^{(l)}$ refers to the post-synaptic activity, and $\delta^{(t)}$ would refer to the error at the output layer, e.g., $\delta(t) = \boldsymbol{e}$.

Back-propagation, while being very effective in training neural networks for specific tasks, is not biologically plausible in the brain. Specifically, notice that back-propagating the errors requires $\boldsymbol{W}^T$, meaning that feedback neurons would need to know all the forward synaptic weights $\boldsymbol{W}$. This is known as the weight transport problem. [2]

Feedback Alignment [3], proposed by Lillicrap et al., has been shown to support learning in neural networks as an alternative to traditional back-propagation. It avoids the weight transport problem by instead propagating the errors through a random matrix, denoted $\boldsymbol{B}$. It was shown to support learning in neural networks, with the restriction that $\boldsymbol{e}\boldsymbol{W}^T\boldsymbol{B}\boldsymbol{e}$, in other words, $\boldsymbol{W}$ and $\boldsymbol{B}$ are in phase. It was shown that using this learning rule, $\boldsymbol{W}$ actually learns to become in phase with $\boldsymbol{B}$. Formally, feedback alignment takes the form of

$$\Delta W_{ij}^{(l)} = \eta \delta_i^{(l)} x_j^{(l)} \tag{4}$$

where

$$\delta_i^{(l)} = \boldsymbol{B}^T \delta_i^{(l+1)} \odot f'(\boldsymbol{a}^{(l)}) \tag{5}$$

Direct Feedback Alignment [4] is another learning rule that was proposed, building off the work in feedback alignment. The key idea is that instead of propagating the errors back from the next layer, (specifically the $\delta_i^{(l+1)}$ term) the error at the output layer is directly propagated to each layer. Formally,

$$\Delta W_{ij}^{(l)} = \eta \left(\boldsymbol{B}^T \boldsymbol{e}\right)_i \odot f'(\boldsymbol{a}^{(l)}) x_j^{(l)} \tag{6}$$

This paper details another learning rule, specifically for recurrent neural networks that is biologically inspired.

III. Biological Motivation and Method

The motivation of our learning rule results from biological neurons with recurrent weights. Consider an autapse. Its activation can be denoted as

$$\tau \frac{dk_i}{dt} + k_i = x_i \tag{7}$$

Solving for k, we get

$$k_i(t) = \frac{1}{\tau} \int_0^t \exp\left(-\frac{t-t'}{\tau}\right) x_i(t') dt' \tag{8}$$

In other words, the autapse is performing as a low pass filter of its own activations. In the case of a feedback network used for error propagation, it should therefore be plausible for such a network to reweight its error signals using its integrated activations. Augmenting this to feedback alignment, we get a new propagated error:

$$\boldsymbol{e}_i^{(t)} = (\boldsymbol{B}\boldsymbol{e}(t))_i \, k_j(t) \tag{9}$$

We therefore result in a new learning rule:

$$\Delta W_{ij}^t = -\eta \boldsymbol{e}'_{ij}(t) \tag{10}$$

$$= -\frac{\eta}{\tau} (B\boldsymbol{e}(t))_i \int_0^t \exp\left(-\frac{t'-t}{\tau}\right) \boldsymbol{x}_j(t') \; dt' \tag{11}$$

IV. Experiments and Results

Without a standard dataset for Recurrent Neural Networks such as MNIST, we decided to simulate a ball bouncing. The ball is fixed in a bounded box, i.e., the ball will bounce off of the walls and not leave the box and follows regular physics dynamics. We task a recurrent neural network to predict the position of the ball in the next n time steps. Formally, given $\boldsymbol{x}(t) = \{x_1(t), x_2(t)\}$, the network is meant to predict $\boldsymbol{y}(t) = \boldsymbol{x}(t+n)$. We compare the performance of back-propagation, feedback alignment, direct feedback alignment, and our new learning rule on a recurrent neural network.

The construction of the recurrent neural network is only one layer with n neurons. The input $\boldsymbol{x}(t)$ is projected into the n space by some random matrix $\boldsymbol{U} \in \mathbb{R}^{n \times 2}$. The outputs of the network is projected onto the 2 space by a random vector $\boldsymbol{V} \in \mathbb{R}^{2 \times n}$. More formally, the input to the network is $\boldsymbol{U}\boldsymbol{x}(t)$ and the output of the network is $\boldsymbol{V}\boldsymbol{o}(t)$, where $\boldsymbol{o}(t)$ refers to the activation of the output in the network at time t. $\boldsymbol{U}, \boldsymbol{V}$ are both fixed, where $\boldsymbol{W}$ is the only matrix that is updated. In the case that $n = 2$, i.e, where there is an equal number of neurons as there are inputs and outputs, then $\boldsymbol{U}, \boldsymbol{V}$ become the identity matrix.

We noticed a few things in our simulations that led to inconclusive results. First, we noticed that the predictions from the network learned the trajectories - however, it had a problem with scaling and translation. This was universal across all learning rules. Figure 1 shows this for trajectories with 100 time steps, when $\boldsymbol{U}, \boldsymbol{V}$ were the identity matrix. While only one trajectory is included, we noticed this pretty consistently. Note that each predicted trajectory was almost exactly the same, the only difference being how far off it was scaled or translated. Each network was initialized to the same $\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V}$, and $\boldsymbol{B}$ matrices as well. We believe that this may be due to the fact that this was only a one layer recurrent network. If we had included a trainable output layer (for instance, if $\boldsymbol{V}$ was updated) then the network would probably have been able to scale properly. Since we were exploring RNN specific learning rules, however, it was unclear how to approach updating $\boldsymbol{V}$.

This observation, however, makes the results in MSE plots inconclusive. Since scaling and translations directly affect the calculated MSE, then it isn't a good measure for whether or not a network is actually learning the dynamics. Figure 2 shows the training and validation MSEs corresponding to the networks that generated figure 1. In this figure, we notice that back-propagation (bptt) and feedback alignment (fa) generally learn at the same rate, while our learning rule (modified) seems to run slower. Direct feedback alignment (dfa), meanwhile, seems to have diverged. However, looking at the trajectories, we can see that the learning rules did learn the trajectories. Therefore comparing the rates or how well the networks learned the trajectories is inconclusive, but we assert that all learning rules are capable of learning the dynamics of the system. Finally, we note that the MSE plot generated in 2 is inconsistent. Sometimes we noticed that DFA would converge fastest, followed by modified (and DFA would not diverge). Other times, we noticed that modified was the fastest. MSE plots are therefore misleading until we are able to construct a network that correctly scales and translates the predictions.

Note that the figures generated are only using the case when $\boldsymbol{U}, \boldsymbol{V}$ are the identity matrix. This means that the state layer size was 2. When we increased the state layer size, then the predictions were scaled terribly with large MSEs. Specifically, we were noticing that the predictions were not in the range of $0, 1$, so we were able to consistently visualize the predictions.

Finally, we wanted to provide some comments on the initialization of the network for our modified learning rule. First, we noticed that if $\boldsymbol{B}$ had eigenvalues that were strictly positive, then the MSE would decrease. However, if $\boldsymbol{B}$ had negative eigenvalues, the MSE would begin to increase and

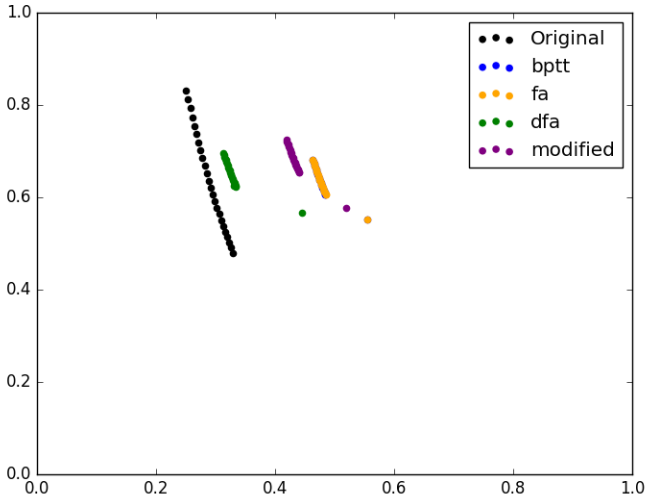

Fig. 1. Predicted trajectory for different learning rules

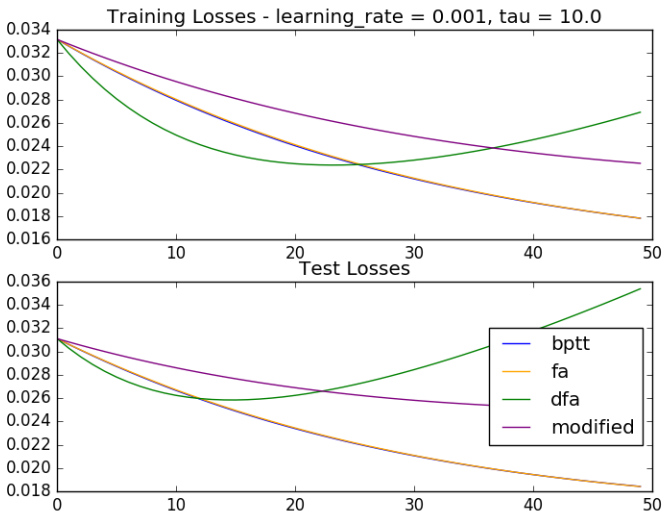

Fig. 2. Training and Validation MSEs for different learning rules

continually increase. We did not plot trajectories for these cases, so we are unable to comment whether the network is actually learning in this case. Additionally, τ can be thought of as a decay term. Therefore this introduces another hyperparameter in the network. We only experimented with a few values (specifically $\tau = 5, 10, 15$), but we believe that more work can be done in tuning this parameter. Finally, due to time constraints, we were unable to test different activations in the recurrent neural network - all of these results were from a network with sigmoid activation.

V. Conclusion

Despite inconclusive results, we believe that our work demonstrates that this learning rule is capable of supporting learning in a recurrent neural network. With our implementation, since our modified learning rule did not depend on back-propagation through time (the averaging term in the other learning rules), it was able to compute much faster. Furthermore, our learning rule did not require a derivative term of the activation unit, which is another step towards a biologically plausible learning rule. Ultimately, without such a network being able to scale and translate predictions properly, we are unable to conclusively say whether or not our learning rule performs better or worse than other learning rules that exist. Perhaps further work can be done to validate the rule on simpler tasks than the one we used.

REFERENCES

[1] R. J. Williams D. E. Rumelhart, G. E. Hinton. Learning internal representations by error propagation. *Nature*, 323:533536, 1986.

[2] Grossberg, S. Competitive learning: from interactive activation to adaptive resonance. *Cogn. Sci.* 11, 2363 (1987).

[3] Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random feedback weights support learning in deep neural networks. *CoRR*, abs/1411.0247, 2014

[4] Arild Nkland. Direct Feedback Alignment Provides Learning in Deep Neural Networks. In *Advances In Neural Information Processing Systems*, pp. 10371045, 2016.