Towards Biologically Plausible Learning: Feedback Alignment and Convolutional Neural Networks

Jake Lance and Alexander Shchokin

Introduction

Error backpropagation (BP) is a fundamental machine learning technique that has proven incredibly effective across a wide variety of domains. While indeed suitable for training deep neural networks, it has come under scrutiny by computational neuroscientists for its biological implausibility. Termed the weight transport problem [1], the requirement for each neuron to have access to its downstream synaptic weights W^T is highly implausible for an explanatory account of human cognition. In recent years, alternative learning algorithms without the weight transport requirement have been proposed. In this project, we investigate a biologically plausible alternative to the backpropagation algorithm termed Feedback Alignment (FA), originally introduced by Lillicrap et al. [2]. Following a trial implementation of these algorithms, as well as a novel modification to the FA algorithm (mFA), we compare the performance of these algorithms in the context of two distinct convolutional neural network (CNN) architectures.

Literature Review

In 2016, Lillicrap and colleagues introduced the Feedback alignment algorithm [2]. This algorithm represented a novel learning algorithm that performs comparably to backpropagation in a wide variety of domains. Using fixed feedback weights to reduce the error, the authors demonstrated that networks can effectively learn without the requirement for neurons to have access to their precise downstream weights. The original paper tested the algorithm with the MNIST data set using a feedforward neural network, comparing the algorithm to backpropagation-based learning and other forms like shallow learning and node perturbation. While backprop remained superior, feedback alignment outperformed all other techniques tested. Soon after the release of this paper, a key variation in FA was been introduced: Direct Feedback Alignment (DFA). In this variant, errors from the last layer are propagated directly to each layer [3]. In this paper, it was also noticed that convolutional neural networks lead to far worse performance with FA and DFA than with BP. These findings were confirmed in further papers [4]. While DFA techniques remain a popular modification to the FA algorithm, it is not entirely clear how pure FA techniques will perform in a convolutional neural network learning task.

Problem Formulation

Part 1: Algorithmic Foundations

In this paper, we sought to compare the performance of Backpropagation, Feedback Alignment, and our own variant of Feedback Alignment. Backpropagation is a fundamental machine learning algorithm where blame is assigned to neurons by multiplying error signals with the entirety of a neuron's downstream synaptic weights [2].

Consider a neural network with L layers, loss function $\mathcal{L}$, and parameters $\theta = \{W_1, W_2, \ldots, W_L, b_1, \ldots, b_L\}$. Each of the three learning algorithms aim to minimize $\mathcal{L}$ by updating network weights using gradients. All three algorithms have an identical forward pass. That is, for each layer l, we compute $\mathbf{y}_l = f(W_l \mathbf{y}_{l-1} + b_l)$, for some activation function f, with loss computed as some function $\mathcal{L}(\mathbf{y}_L, \mathbf{y}_{\text{true}})$.

For the backward pass, each algorithm differs in how the error signals are propagated, and how the weights are updated. In backpropagation, the error signal for the output layer is computed as the gradient of the loss with respect to the output with $\delta_L = \frac{\partial \mathcal{L}}{\partial \mathbf{y}_L} f'(\mathbf{z}_L)$. The error is then propagated backwards using the transposed weights $\delta_{l-1} = W_l^\top \delta_l f'(\mathbf{z}_{l-1})$, and the gradients and weights are then updated.

The key difference between the two algorithms is that while the backward pass for backpropagation makes use of the true transposed weights $W_l^\top$, feedback alignment replaces this matrix with random feedback weights $\hat{W}_l^\top$, which we call WB in our code. Interestingly, such a matrix can still support learning despite being random and fixed.

For our modified FA algorithm, we use a partial alignment approach. After receiving the final layer's error signal, gradients for the last layer's weights and biases are computed, as usual. We then define a hyperparameter p to select a proportion of gradient values to update as usual. All other values remain unchanged. Specifically, we use a binary mask to select a proportion p of the largest (absolute) gradient values to update from the computed weights and bias gradients.

Part 2: Learning Simulations

Our simulation process consisted of two phases: a preliminary phase and the convolutional phase.

In the preliminary phase, the three algorithms were tested in a simple environment. The primary purpose of this environment was to serve as a sanity check before implementing the convolutional neural network architectures with MNIST data. A simple feed-forward neural network was devised with $5 \to 30 \to 2$ neurons in three layers. The training process was divided into 300 independent simulations and further into 1000 discrete time steps t. At the start of each iteration, some input vector $\mathbf{x}$ is initialized, as well as the target, a randomly generated quadratic function of the form $y^* = (B\mathbf{x} + 1)^2$ for some fixed random matrix B. Note that in this case the superscript 2 represents element-wise exponentiation. Several architectures were evaluated including single layer networks and deep networks with multiple hidden layers. In all cases, ReLU activation was used for the hidden layers, and an affine transformation was applied for the output layers to produce a prediction.

In the second phase, we used the three learning algorithms in parallel to train convolutional networks to recognize the handwritten digits in the MNIST data set. Two separate architectures were investigated. The first network was a 4-layer mapnet with the two middle layers as 28-by-28 maps with field-fractions of 0.25, representing the fraction of neurons in the following layer that each neuron projects to. The second network was similarly a 4-layer mapnet with identical middle layers, but with an additional connection introduced from the second to fourth layer, with $W^{(4)}$ defined as $\begin{bmatrix} y^{(2)} \\ y^{(3)} \end{bmatrix}$. That is, the we define the weight matrix of the fourth layer as the matrix activations of layer two stacked on the matrix of activations of layer three. In both cases, the connections from either of the middle layers to the final layer were defined as dense. In the middle layers, we use a logarithmic variation of ReLU activation function defined as $y_i^{(l)} = \max(0, \quad \text{sign}(v_i^{(l)}) \cdot \log(1 + |v_i^{(l)}|))$. The final layer of the network is defined as affine, and a softmax function is then applied: $y_i = \frac{\exp(y_i^{(4)})}{\sum_j \exp(y_j^{(4)})}$. The resulting y vector is compared with the one-hot label vector y^*, and our error and loss are then computed as $e = y - y^*$ and $L = \frac{1}{2} e^\top e$.

We argue that introducing a skip connection introduces an extra degree of complexity and has several interesting implications for comparing FA and BP. Firstly, it is known that in general, skip connections tend to assist backpropagation by mitigating exploding gradients [6]. Moreover, we were incredibly curious about the role skip connections would have on feedback alignment, given their improved biological plausibility. That is, neurons in human brains are known to wire in unique patterns, including direct wiring to upstream neurons [7].

Simulations for each of the convolutional architectures were run four times independently and the results averaged. For each simulation, we tested the three learning algorithms in parallel across 10 epochs. That is, we ran 10 passes through all 60,000 training examples with 600 minibatches in each epoch. At the start of each epoch, the data was shuffled to aid learning, supplying a different series of 600 minibatches across different epochs. At the end of each epoch, the network was run on the entire test of 10,000 testing examples and the number of incorrect guesses was reported in the algorithm's respective 10-element error data collection vector k_{BP}, k_{FA}, k_{mFA}, where the ith element represents the number of incorrect guesses following the ith epoch. For optimization, the well-known ADAM optimizer was leveraged [5].

All simulations were implemented in MATLAB R2023b (Update 5) and the relevant code has been provided with our submission, with exception of the 20MB MNIST training data, as this surpasses the MarkUs upload limit. It is available from the authors upon request.

Results

Preliminary Phase: BP, FA, and modified FA were tested in a quadratic target function context. While several neural architectures were considered, all reported figures for the quadratic target function task derive from a 3-layer network of 5, 30, and 2 neurons in order. BP was compared in parallel with FA, and then in parallel with modified FA.

As expected, all algorithms demonstrated an ability to reduce an error measure (Normalized Squared Error) over time. This was our primary objective for this subsection. Figures 1a) and 1b) in the Appendix demonstrate FA and mFA's comparability to BP respectively. It can be noticed that these algorithms exhibit very similar learning behaviour.

Given this similarity, we devised a novel metric to better visualize their differences. In our simulation, an error vector is computed at each time step, for each learning algorithm, based on the difference between the predicted and true output values. As a metric for comparison, we then compute an element-wise sum of these error vectors for each of the learning algorithms. That is, for each learning algorithm A, we compute a simulation-wide error vector T_A. We then take the difference $T_{FA} - T_{BP}$ (or $T_{mFA} - T_{BP}$) and plot the distribution of resulting differences across several independent simulations as a histogram. Figures 2a) and 2b) in the Appendix represent the spread of these differences for FA and mFA respectively, versus backpropagation.

This plot provides several key insights. Firstly, we see that there tends to be high variability in the error differences at the end of each simulation. This can be well explained by the random nature of the target function that is initialized for each simulation. Importantly, we also notice that the mean of the distribution shifts from $\mu = 10.409$ to $\mu = 2.604$ when our modified feedback alignment algorithm is leveraged. This indicates a potentially effective modification to the algorithm, which is further explored in the convolutional phase.

Convolution Phase: BP, FA, and mFA were tested using two convolutional neural network architectures and the MNIST data set. For each of the two neural network architectures, four independent simulations of the training and testing process were carried out. For each simulation, three error vectors T_{BP}, T_{FA}, T_{mFA} were computed with the ith entry in each vector representing the number of incorrect guesses (out of $10,000$) during the testing phase of the ith epoch for the specified learning technique. As intended, all three algorithms managed to demonstrate learning on the data set in the convolutional setting.

As explained in the Problem Specification section, the first architecture was a 4-layer CNN with no skip connections, with the two middle layers as 28-by-28 maps with field-fractions of 0.25. In this setting, backpropagation performed optimally across all epochs, when averaged across all four independent simulations (Figure 3a, Appendix). In this case, our modified feedback alignment algorithm performed similarly to standard feedback alignment, unlike the preliminary phase. Over time, the difference between backpropagation and the feedback alignment errors began to shrink, but backpropagation remained optimal throughout.

To further our investigation, we compared the performance of the three algorithms on the previous CNN to a similar CNN with an introduced skip connection. We found that the introduction of a skip connection greatly improves all three learning algorithms, especially across early epochs (Figure 3b, Appendix). Over time, these differences become less pronounced.

Conclusion

While feedback alignment demonstrates potential for training convolutional neural networks with biologically inspired learning algorithms, its performance still lags behind the precision and effectiveness of traditional backpropagation methods in minimizing error. The incorporation of skip connections, however, highlights an exciting intersection of neuroscience and machine learning, showing that even simple architectural adjustments can enhance the learning process for both traditional and biologically plausible algorithms. While our results indicate an advantage for skip connections with Feedback Alignment learning, our work has several limitations. First, we have not examined performance on more challenging tasks like CIFAR-10 to see if this performance scales up. Moreover, these findings were only reflected between two distinct architectures. Next steps could include investigating if this advantage persists for a wider variety of domains and skip vs. no-skip architectures.

Moving forward, while backpropagation remains the most effective technique in many contexts, the continued exploration of alternative learning methods such as Feedback Alignment, particularly in conjunction with biologically inspired modifications like skip connections, may open new avenues for more efficient and human-inspired neural network training.

References

[1] Grossberg, Stephen. "Competitive learning: From interactive activation to adaptive resonance." Cognitive science 11.1 (1987): 23-63.

[2] Lillicrap, Timothy P., et al. "Random feedback weights support learning in deep neural networks." arXiv preprint arXiv:1411.0247 (2014).

[3] Nøkland, Arild. "Direct feedback alignment provides learning in deep neural networks." Advances in neural information processing systems 29 (2016).

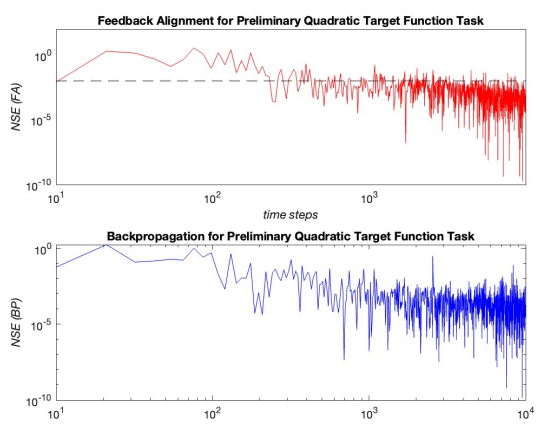
[4] Liao, Qianli, Joel Leibo, and Tomaso Poggio. "How important is weight symmetry in backpropagation?." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 30. No. 1. 2016.

[5] Kingma, Diederik P. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

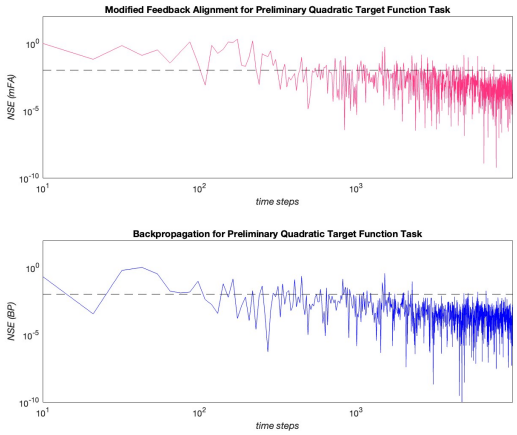
[6] Xu, Guoping, et al. "Development of Skip Connection in Deep Neural Networks for Computer Vision and Medical Image Analysis: A Survey." arXiv preprint arXiv:2405.01725 (2024).

[7] Larkum, Matthew E., et al. "A perspective on cortical layering and layer-spanning neuronal elements." Frontiers in neuroanatomy 12 (2018): 56.

Appendix

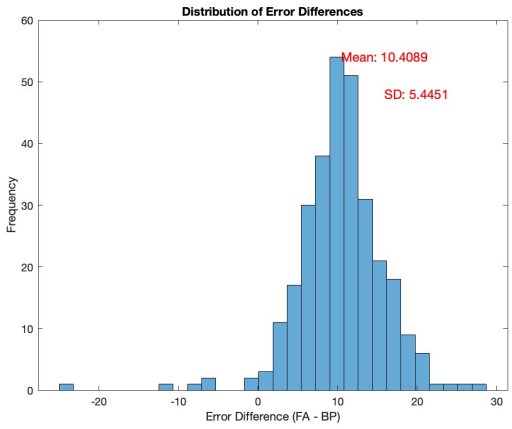


(a) FA vs BP

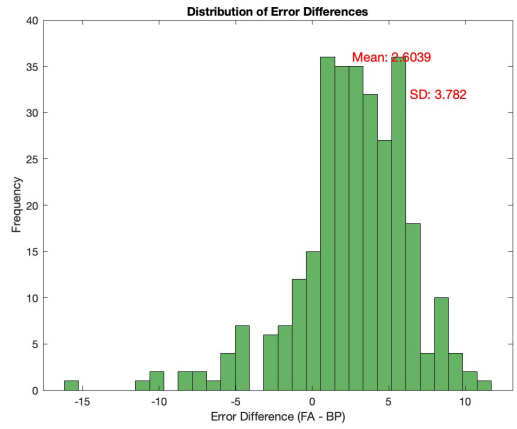


(b) mFA vs BP

Figure 1: Feedback Alignment (Red) and Modified Feedback Alignment (Pink) both versus Backpropagation (Blue) on Phase 1 Learning Task

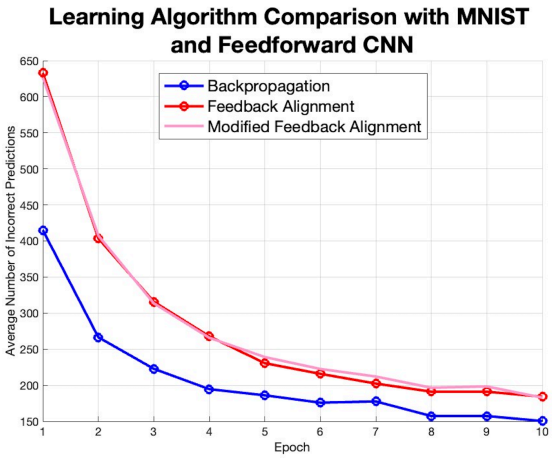


(a) Feedback Alignment and Backprop Difference Plot

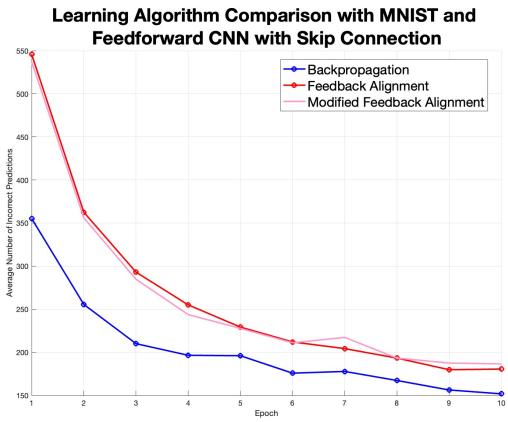


(b) Modified Feedback Alignment and Backprop Difference Plot

Figure 2: Difference Plots for Feedback Alignment (blue) and Modified Feedback Alignment (green)



(a) No Skip Connection



(b) With Skip Connection

Figure 3: MNIST CNN Learning Evaluation with No Skip Connections (Left) and with Skip Connection (Right)