Feedback alignment in deep convolutional networks

Theodore H. Moskovitz[1,2,*], Ashok Litwin-Kumar[1], and L.F. Abbott[1]

[1]Zuckerman Mind, Brain and Behavior Institute, Columbia University, New York, NY
[2]Department of Computer Science, Columbia University, New York, NY
[*]t.moskovitz@columbia.edu

Abstract

Ongoing studies have identified similarities between neural representations in biological networks and in deep artificial neural networks. This has led to renewed interest in developing analogies between the backpropagation learning algorithm used to train artificial networks and the synaptic plasticity rules operative in the brain. These efforts are challenged by biologically implausible features of backpropagation, one of which is a reliance on symmetric forward and backward synaptic weights. A number of methods have been proposed that do not rely on weight symmetry but, thus far, these have failed to scale to deep convolutional networks and complex data. We identify principal obstacles to the scalability of such algorithms and introduce several techniques to mitigate them. We demonstrate that a modification of the *feedback alignment* method that enforces a weaker form of weight symmetry, one that requires agreement of weight sign but not magnitude, can achieve performance competitive with backpropagation. Our results complement those of Bartunov et al. (2018) and Xiao et al. (2018b) and suggest that mechanisms that promote alignment of feedforward and feedback weights are critical for learning in deep networks.

1 Introduction

While the hierarchical processing performed by deep neural networks is inspired by the brain, there are a number of fundamental differences between these artificial networks and their biological counterparts. In particular, the use of the backpropagation (BP) algorithm (Rumelhart et al., 1986) to perform gradient descent, which is central to the optimization of artificial networks, requires several assumptions that are difficult to reconcile with biology. Objections include the separation of learning and inference into two separate phases and a requirement of symmetric synaptic connectivity between forward and backward paths through the network, an issue known as the *weight transport problem* (Grossberg, 1987). While feedback connections are common, such symmetry has not yet been observed in the brain.

Feedback alignment (FA), a modification to BP in which this symmetry is broken by replacing the forward weights with randomized connections for the backward pass, avoids the weight transport problem (Lillicrap et al., 2016). While this method exhibits performance competitive with backpropagation in simple fully-connected networks (Lillicrap et al., 2016; Nøkland, 2016), it has performed poorly when applied to deeper convolutional architectures and complex datasets (Liao et al., 2015; Bartunov et al., 2018). In this work, we explore the obstacles that hinder the performance of FA in deeper networks and experiment with restricted excitatory and inhibitory connectivity, presenting modifications that allow these methods to remain competitive with BP.

2 Related work

The weight transport problem for artificial neural networks was identified early on (Grossberg, 1987; Crick, 1989; Zipser and Rumelhart, 1993). While a number of potential solutions had been proposed previously (Crick, 1989; Brandt and Lin, 1996; Hinton, 2003), the FA method generated substantial interest because it required no assumptions on the structure of the feedback weights used to convey error signals, instead taking them to be fixed and random (Lillicrap et al., 2016). Initial work demonstrated that FA was competitive with BP on the MNIST handwritten digit dataset and a random input-output task in multilayer fully-connected networks. In these networks, it was observed that as training progresses, the angle between the BP gradient and the FA error signal converges from approximately orthogonal to roughly 45°, meaning that the FA weight updates are correlated with but not identical to those in BP.

Liao et al. (2015) applied FA to convolutional neural networks (CNNs), testing performance on a variety of tasks including visual and auditory classification. Without additional modifications, the performance of FA was substantially worse than BP in contrast to the earlier experiments on fully-connected networks. To achieve competitive performance, the authors made three modifications to the basic algorithm. The first of these is a technique termed uniform sign-concordant feedback (uSF), in which the feedback matrix B is set to $B = \text{sign}(W)$, where W represents the forward weights. The second is the addition of Batch Normalization (Ioffe and Szegedy, 2015), and the third is a set of techniques termed Batch Manhattan in which the magnitude of the gradient is discarded. Batch Manhattan in particular was introduced to avoid vanishing/exploding gradients arising from the inherent weight asymmetry of FA. In this paper we demonstrate several alternative mechanisms for avoiding vanishing/exploding gradients without discarding the magnitude of the error signal.

Nøkland (2016) introduced *direct feedback alignment* (DFA), in which the output layer error signal is propagated directly to all upstream layers instead of through adjacent layers as in standard BP and FA. This idea was extended by (Baldi et al., 2016) to include residual connections from downstream layers other than the output layer. Here, we further develop these ideas into new gradient methods, as well as demonstrate their success on deeper models than have been used previously.

Recently, Bartunov et al. (2018) presented results testing FA on CIFAR-10 and ImageNet architectures. It is important to note that, in in the interest of biological plausibility, their models eschewed the weight-sharing in convolutional layers that substantially improves generalization performance in most CNNs. Our models do take advantage of weight sharing to improve performance. A number of other approaches seek to further improve biological plausibility through the elimination of a distinct error signal altogether (Le Cun, 1986; Bengio, 2014; Lee et al., 2015) or the use of segregrated dendrites and continuous learning (Guergiuev et al., 2016; Sacramento et al., 2018), but these approaches have also struggled when applied to deep networks and are outside the scope of our investigation.

During the final stages of the preparation of this manuscript, Xiao et al. (2018b) released results similar to ours on even deeper networks. Our results are consistent with theirs, and we extend the analysis by studying networks with fixed excitatory and inhibitory connections as well as several different modifications to FA. Together, these results provide strong evidence that sign-concordant feedback improves performance compared to random feedback.

3 Feedback alignment in convolutional networks

We begin by describing the implementation of FA in convolutional networks, which is similar to its implementation in fully-connected networks. For a convolutional layer l, the affine pre-activation u^l and the ith nonlinear feature map activation I_i^l are calculated as

$$u_i^{l+1} = w_i^{l+1} * I_i^l + b^{l+1},\ I_i^{l+1} = f(u_i^{l+1}), \tag{1}$$

where w_i is the ith kernel, b is the bias term, $*$ denotes convolution, and f is the activation function. Assuming a loss function J, the gradient as used in backpropagation at layer l is

$$\delta_i^l = \frac{\partial J}{\partial u_i^l} = (\overline{w}_i^{l+1} * \delta_i^{l+1}) \odot f'(u_i^l), \tag{2}$$

where $\odot$ is component-wise multiplication and $\overline{w}$ denotes a rotation of w by 180°, i.e. a *flipped* kernel. FA replaces the feedforward kernel with a separate feedback matrix B to produce the error signal

$$\delta_i^l = \frac{\partial J}{\partial u_i^l} = (B_i^{l+1} * \delta_i^{l+1}) \odot f'(u_i^l). \tag{3}$$

In both methods, the parameter updates are calculated similarly, with $\Delta W^l \propto \delta^l (x^{l-1})^T$ in fully-connected networks, and $\Delta W^l \propto \delta^l * \overline{I}^{l-1}$ in CNNs.

3.1 Modifications to the FA algorithm

We now describe modifications to FA that improve performance by reducing the possibility of vanishing or exploding gradients and encouraging angular alignment between the feedforward and

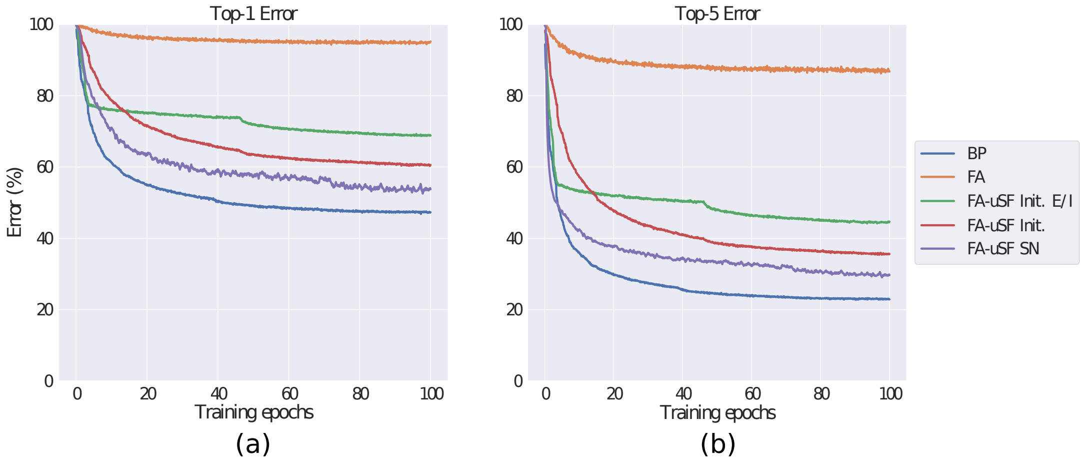


Figure 1: Top-1 and Top-5 test errors on ImageNet for BP, FA without uSF or normalization (FA-no uSF), FA with uSF and the initialization method (FA-init), and FA with uSF and strict normalization (FA-strict-norm). See text for method details.

feedback weights. The reasons for their effectiveness are explored in detail in Section 5. For uSF, the first feedback matrix setting we use is

$$B_t^l = |B_0^l| \odot \text{sign}(W_t^l), \tag{4}$$

where $|\cdot|$ is the element-wise absolute value function and t denotes the training epoch. We call this technique, which uses no information about the magnitude of the current forward weights, the *initialization* (Init.) method. One modification of the initialization method that we also tested was to freeze the sign of the forward weights after a few epochs of training. This imposes a new constraint on the network's synaptic connections, keeping them either excitatory (positive) or inhibitory (negative). Because under uSF the feedback weights equal the sign of the forward weights, this also results in a constant feedback matrix. We call this the *excitatory/inhibitory* (E/I) method.

The next setting we use, which incorporates the norm of the forward weights, is

$$B_t^l = ||W_t^l||_2 \odot \frac{\text{sign}(W_t^l)}{||\text{sign}(W_t^l)||_2}. \tag{5}$$

We call this the *strict normalization* (SN) method. Note also that when uSF is used, some form of explicit normalization is required, as otherwise the magnitude of the feedback weights is ±1 regardless of the magnitude of the forward weights.

4 Experiments and classification results

Each method described above was applied to deep CNNs tested on visual classification. Models were trained on the MNIST handwritten digits dataset, the CIFAR-10 image set, and the ImageNet dataset. On all three datasets, we implemented relatively simple baseline architectures following the basic paradigm established by LeCun et al. (Lecun et al., 1998), namely several convolutional layers each followed by a pooling layer, with one or several fully-connected layers on top. To test the performance of our proposed methods on models with increased depth, we also applied them to more complicated architectures with nine convolutional layers (Springenberg et al., 2014) on CIFAR-10 and ImageNet. The exact architecture details are presented in Supplementary Tables 2 and 3. In addition to FA, we trained models with direct feedback alignment (DFA) and a new method we call *dense* feedback alignment (DenseFA), which adds residual feedback connections from every downstream layer. Unfortunately, the memory requirements of these methods precluded their use on the larger models we trained. To further investigate the relationship between FA and BP, we also investigated the performance of models that were trained with BP but with either added noise (BP + Noise) or with weight matrices that were forced to align with arbitrary random matrices (BP + Alignment). These techniques and their motivations are discussed in greater detail in Section 5.3. On ImageNet, as a means of circumventing the need for normalization altogether, we also experimented with BP (BP Const.) and FA (FA-uSF Const.) models with weight norms constrained to be constant over training (see Section 5.4 for details).

Method	MNIST	CIFAR-10 1	CIFAR-10 2	ImageNet 1	ImageNet 2
BP	0.8	17.2	11.0	79.5	45.5
BP + Noise	0.8	17.4	11.0	79.2	46.0
BP + Alignment	0.9	17.4	11.2	79.4	45.9
FA	1.1	26.6	35.6	95.2	94.5
FA-uSF Init. E/I	0.9	18.9	17.8	86.9	67.8
FA-uSF Init.	0.7	17.6	13.1	79.6	60.1
FA-uSF SN	0.7	17.7	12.6	78.9	54.4
DFA	1.0	28.6	-	-	-
DenseFA	0.7	16.9	-	-	-
BP Const.	-	-	-	-	46.9
FA-uSF Const.	-	-	-	-	51.2

Table 1: Top-1 Test Error (%). The feedback alignment methods are competitive with, and in some cases exceed, backpropagation performance. 'uSF' denotes the use of sign-concordant feedback. 'Init.' denotes the initialization method, and 'SN' denotes the strict normalization method. 'E/I' denotes a model with frozen excitatory and inhibitory connections. These approaches are detailed in Section 3.1. 'Const.' denotes a model for which the L_2-norms of the feedforward weights was fixed at initialization. This approach is described in Section 5.4. Model architectures are described in detail in Supplementary Section 7.2.

The MNIST model was trained for 25 epochs, the LeNet-style CIFAR-10 model was trained for 154 epochs, and the all-convolutional CIFAR-10 architecture was trained for 256 epochs. Both ImageNet models were trained for 100 epochs. In the E/I condition, we froze the signs of the weights (by clipping their values at 0) after 5% of the training time (i.e., 5 epochs on ImageNet). All models were trained with the Adam optimizer Kingma and Ba (2014). Further training details can be found in Supplemental Section 7.1. Test results are summarized in Table 1, with ImageNet test error curves plotted in Figure 1. Our results improve on those reported by Bartunov et al. (2018) and are consistent with those of Xiao et al. (2018b).

5 Scaling to deep networks

We now describe in more detail considerations when extending FA to deeper networks and the modifications we made to improve performance.

5.1 Vanishing and exploding activations and error signals

Consider a neural network with depth L trained to minimize loss J. Assume for simplicity that the layers are sized equally so that each weight matrix is the same size and has a similar distribution of weights. The gradient with respect to the first layer activation x^1 is:

$$\frac{\partial J}{\partial x^1} = \frac{\partial J}{\partial x^L}\frac{\partial x^L}{\partial u^L}\frac{\partial u^L}{\partial x^{L-1}}\frac{\partial x^{L-1}}{\partial u^{L-1}}\frac{\partial u^{L-1}}{\partial x^{L-2}}\cdots\frac{\partial x^2}{\partial u^2}\frac{\partial u^2}{\partial x^1}. \tag{6}$$

Observe that at each layer

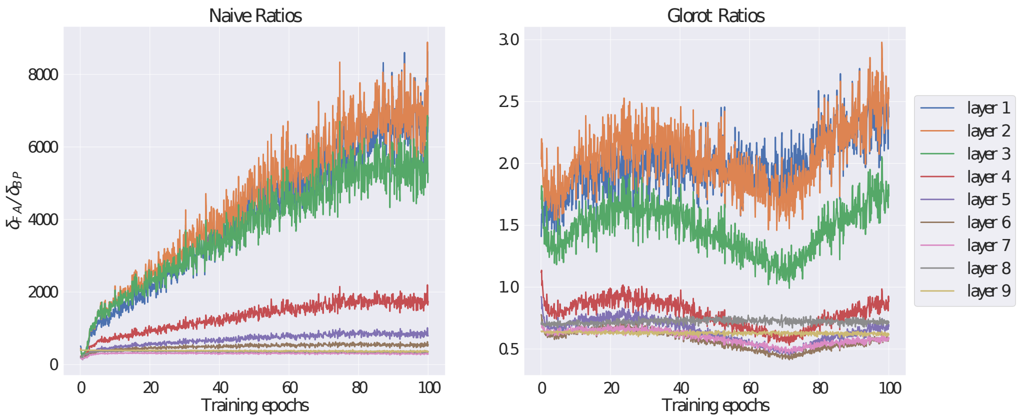
$$\frac{\partial u^l}{\partial x^{l-1}} = \begin{cases}(W^l)^T & \text{if BP}\\ B^l & \text{if FA.}\end{cases} \tag{7}$$

If the weights are assumed to be independent, then

$$\frac{||\nabla^{BP}_{x^1}J||_2}{||\nabla^{FA}_{x^1}J)||_2} \sim ||\prod_{l=1}^{L}\frac{(W^l)^T}{B^l}||_2 \approx \prod_{l=1}^{L}\frac{(||W^l)^T||_2}{||B^l||_2}, \tag{8}$$

where $||\cdot||_2$ denotes the Euclidean norm. Because $||W^l||_2$ changes over the course of training and $||B^l||_2$ remains fixed, a network trained with feedback alignment runs a risk of experiencing vanishing or exploding gradients even if BP does not, depending on the ratio $||B^l||_2/||W^l||_2$. Moreover, this problem is exponential in the depth of the network (Figure 2a).

There are two possible families of approaches to solving this issue. First, careful initialization of the forward and backward weights can dramatically reduce fluctuations in the scale of activations and error signals between layers. Initialization methods designed to control variance from layer to layer as a means of effectively training deep networks are common (Glorot and Bengio, 2010; Xiao et al., 2018a). One example is the initialization strategy introduced by (Glorot and Bengio, 2010)



(a)

(b)

Figure 2: The layer-wise gradient ratios between FA and BP for a naive initialization (a) and using a variance-preserving initialization such as the one devised by Glorot and Bengio (2010) (b). This demonstrates the importance of initialization in controlling for vanishing or exploding gradients as the depth of the network increases.

in which weights are initialized with a variance of $1/[\frac{1}{2}(n_{\text{in}} + n_{\text{out}})]$, with n_{in} the number of input connections from the previous layer to neurons of the subsequent layer and n_{out} the number of output connections. In BP, this method controls the variance of both the forward activations and the backward gradients by averaging the number of incoming and outgoing connections. However, in FA the forward and backward passes are decoupled, and the most effective initialization is therefore to set the variance of the forward weights as $1/n_{\text{in}}$ and the variance of the fixed, backward weights as $1/n_{\text{out}}$. Note that n_{out} is the number of incoming connections that a layer receives during the backward pass. While we found this method to be effective (Figure 2b), depth remains a challenge, as the distribution of the forward weights can drift during training. With careful initialization, the scale of the drift is constant to within an order of magnitude, but there is still an adverse effect on performance.

The second family of approaches instead explicitly manages the sizes of the weights. For example, Liao et al. (2015) introduced a parameter update rule termed *Batch Manhattan*, which discards the magnitude of the error signal completely. That is, whereas the gradient descent weight update is proportional to $\frac{\partial J}{\partial W}$, Batch Manhattan sets it proportional to sign $\left(\frac{\partial J}{\partial W}\right)$. In discarding the magnitude of the gradient, this method ignores the impact that the size of this signal has on the effective learning rate of the network. The use of an adaptive, per-parameter optimization algorithm such as Adam (Kingma and Ba, 2014) can ameliorate this to a degree, but even in this case we found a reduction in performance. We found that a more effective method was either to constrain the norms of the feedback matrices to be equal to that of their feedforward counterparts, as in SN, or to simply scale by the initialized feedback weights, as in the initialization method.

5.2 Gradient signal alignment

When investigating FA in shallow feedforward networks, (Lillicrap et al., 2016) observed that as training progressed, the signal calculated in FA and the gradient used by BP are roughly orthogonal at initialization but that the angle between them converges to approximately 45° by the end of training. While we also observed this convergence in shallow networks, deeper networks exhibited less alignment. More precisely, while we found that the angle converged to some degree for the topmost layers, this was not the case for the lower layers (Figure 3a). In order to maintain alignment, we used the uSF method introduced by (Liao et al., 2015). When uSF is applied, the angle quickly drops below 90°, but then rises slightly and levels off (Figure 3b). This is likely because at the beginning of training, depending on the initialization method, forward weight values are more tightly distributed than at the end of training.

5.3 Modifications to BP

Because of the relationship between alignment and performance, we investigated models that were trained with BP but with modifications that mimic features of FA. These included either adding

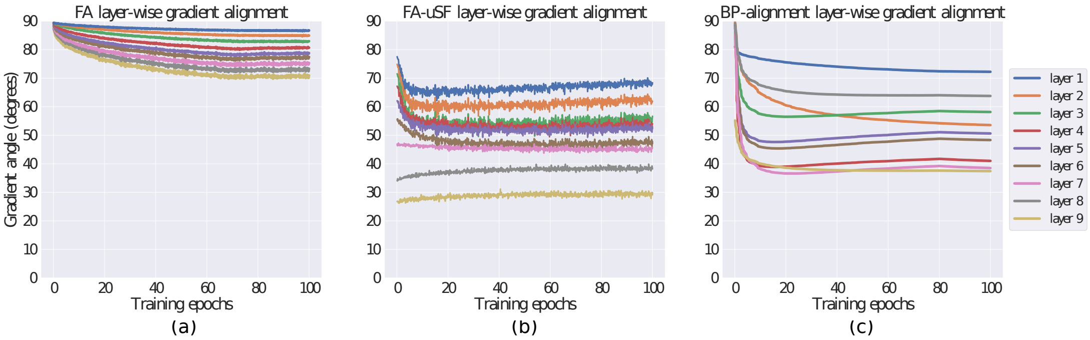


Figure 3: The layer-wise angular alignment between the gradient computed with BP and with (a) FA, (b) FA with uSF, and (c) BP with an alignment constraint. Although the angles are comparable, constrained BP still slightly outperforms FA 1. These results demonstrate that viable solutions can be found without strictly following the gradient.

noise to the error signal or forcing the forward weights to align with an unrelated matrix (see Table 1 for results). In the first condition, we added noise drawn from a normal distribution centered at zero with variance approximately that of the BP gradients. The angle between the noisy learning signals and that of the true gradients remained at roughly 45° throughout training, comparable to the alignment achieved by FA with uSF. This did not substantially reduce performance (Table 1), indicating that the reduction in performance for FA cannot be accounted for by gradient noise that is centered on the BP gradient.

In the second condition, we applied an L_2 penalty to the difference between the model weights $\theta = \{w^1, ..., w^L\}$ and a random set of target matrices $\Lambda = \{v^1, ..., v^L\}$ (where L is the depth of the network) in addition to the cross entropy loss:

$$J(y, \hat{y}, \theta, \Lambda) = -\sum_{k=1}^{K} y \log \hat{y} + \lambda \sum_{l=1}^{L} \sum_{i,j} (w_{ij}^{l} - v_{ij}^{l})^2, \tag{9}$$

where y and $\hat{y}$ are the true and predicted logits, respectively, K is the number of label classes, and λ is the regularization weight. This penalty forces the forward weights to align with a set of fixed random matrices, as in FA, but without using the separate matrix for error propagation (Figure 3). We found that setting $\lambda = 0.001$ was effective at producing an alignment comparable to that of FA. Performance did not noticeably suffer as a result of either of this change. This argues that the constraint of aligning with an arbitrary matrix does not in itself limit the performance of our networks.

5.4 Constraining the magnitude of the weights

While the various normalization methods we have introduced are effective at reducing the effect that the changing magnitude of the feedforward weights has on FA over time, to completely circumvent this issue, we trained several models on ImageNet in which the norm of the forward weights was adjusted after each training epoch. Specifically, the weights were initialized normally, and after each epoch t the weights at each layer l were scaled as follows:

$$w_t^l = ||w_0^l||_2 \frac{w_t^l}{||w_t^l||_2}, \tag{10}$$

where $||w_0^l||_2$ is the L_2-norm of the initial weights. Using a variance-preserving initialization method (Figure 2), this condition allows us to evaluate the effectiveness of FA without the need for additional mechanisms to normalize the backward weights as the forward weights change. The results demonstrate that constraining the norm in this way does not substantially affect the performance of BP, while resulting in improved performance using FA with 51.2% Top-1 error, further narrowing the gap between the two methods (Table 1, Figure 4).

6 Discussion

Taken together, these results extend previous work and demonstrate that modified FA learning algorithms can perform with accuracy competitive with BP even for deep CNNs. Necessary modifications include controlling the magnitude of the error signal, which can be accomplished using the

Method	Top-1	Top-5
BP	46.9	22.6
FA-uSF	51.2	26.6

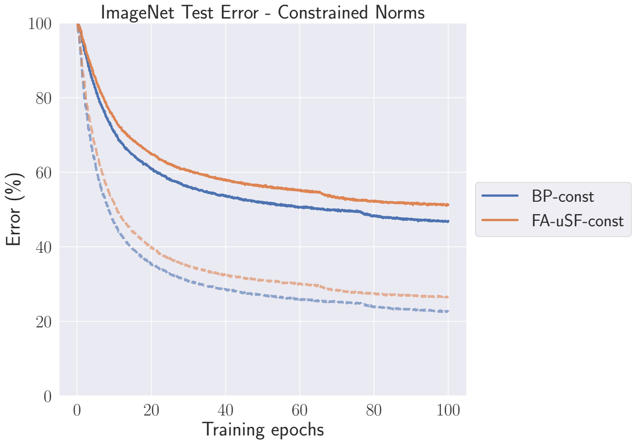


Figure 4: Top-1 (solid) and Top-5 (dotted) ImageNet test results and error curves for models trained with the L_2-norms of their forward weights fixed at the beginning of each training epoch. This condition eliminates the need for the training-time normalization methods, and further narrows the gap between BP and FA.

normalization methods we investigated, and encouraging alignment, which can be accomplished with sign-concordant feedback. These conclusions are consistent with those of Xiao et al. (2018b). In our simulations, we found that networks with forms of these two methods as well as with fixed weight norms reaches an ImageNet Top-1 error of 51.2%, which is competitive with BP. A topic of experimental interest is identifying biological mechanisms with roles analogous to these methods.

Homeostatic mechanisms regulating feedback connections could play a role in normalizing forward and backward synaptic weights (Turrigiano and Nelson, 2004). We studied a variety of mechanisms to accomplish this normalization, from explicitly scaling the feedback weights to match the feedforward weights in norm to constraining the feedforward weights to have an unchanging norm. We found that these methods improved upon methods that ignore gradient norms (Liao et al., 2015). Fixing the feedforward weight norm (Figure 4) led to the best results, suggesting that regulating the forward pass and normalizing the backward weights appropriately may be sufficient to achieve high performance. We focused only on instantaneous normalization, but in the future, it would be valuable to experiment with scaling the weights with a delay that reflects the time constant of homeostatic regulation.

Sign-concordant feedback was also crucial to performance, consistent with the idea that “vanilla” feedback alignment does not scale to deep networks (Bartunov et al., 2018; Xiao et al., 2018b). An intriguing possibility is that the segregation of biological neurons into excitatory and inhibitory subtypes permits cell-type-specific wiring that promotes sign-concordant feedback. The identification of activity-dependent or development processes that promote such wiring would provide support for a backpropagation-like algorithm being plausible in the brain. While our results and those of Xiao et al. (2018b) represent a step toward understanding the plausibility of such an algorithm, many questions remain, including how performance is affected in networks without weight sharing (Bartunov et al., 2018) and how to incorporate continuous learning without separate forward and backward passes (Guergiuev et al., 2016; Sacramento et al., 2018). Further experimental and computational studies are needed to develop clearer analogies between biological and artificial neural network learning.

References

P. Baldi, P. J. Sadowski, and Z. Lu. Learning in the machine: Random backpropagation and the learning channel. *CoRR*, 2016. URL http://arxiv.org/abs/1612.02734.

S. Bartunov, A. Santoro, B. A. Richards, G. E. Hinton, and T. Lillicrap. Assessing the Scalability of Biologically-Motivated Deep Learning Algorithms and Architectures. *ArXiv e-prints*, 2018. URL http://arxiv.org/abs/1807.04587.

Y. Bengio. How auto-encoders could provide credit assignment in deep networks via target propagation. *CoRR*, 2014. URL http://arxiv.org/abs/1407.7906.

R. Brandt and F. Lin. Can supervised learning be achieved without explicit error back-propagation? *Proceedings of International Conference on Neural Networks (ICNN)*, 1996.

F. Crick. The recent excitement about neural networks. *Nature*, 337:129–132, 1989.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256. PMLR, 2010.

S. Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11(1):23–63, 1987.

J. Guergiuev, T. P. Lillicrap, and B. A. Richards. Towards deep learning with segregated dendrites. *ArXiv e-prints*, 2016. URL http://arxiv.org/abs/1610.00161.

G. Hinton. The ups and downs of hebb synapses. *Canadian Psychology/Psychologie canadienne*, 44(1):10–13, 2003.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, 2015. URL http://arxiv.org/abs/1502.03167.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, 2014. URL http://arxiv.org/abs/1412.6980.

Y. Le Cun. Learning process in an asymmetric threshold network. In E. Bienenstock, F. F. Soulié, and G. Weisbuch, editors, *Disordered Systems and Biological Organization*, pages 233–240, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

D.-H. Lee, S. Zhang, A. Fischer, and Y. Bengio. Difference target propagation. In A. Appice, P. P. Rodrigues, V. Santos Costa, C. Soares, J. Gama, and A. Jorge, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 498–515, Cham, 2015. Springer International Publishing.

Q. Liao, J. Z. Leibo, and T. A. Poggio. How important is weight symmetry in backpropagation? *CoRR*, 2015. URL http://arxiv.org/abs/1510.05067.

T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7, 2016.

A. Nøkland. Direct feedback alignment provides learning in deep neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1037–1045. Curran Associates, Inc., 2016.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

J. Sacramento, R. Ponte Costa, Y. Bengio, and W. Senn. Dendritic error backpropagation in deep cortical microcircuits. *ArXiv e-prints*, 2018. URL http://arxiv.org/abs/1801.00062.

J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, 2014. URL http://arxiv.org/abs/1412.6806.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15: 1929–1958, 2014.

G. G. Turrigiano and S. B. Nelson. Homeostatic plasticity in the developing nervous system. *Nature Reviews Neuroscience*, 5(2):97, 2004.

L. Xiao, Y. Bahri, J. Sohl-Dickstein, S. S. Schoenholz, and J. Pennington. Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks. *ArXiv e-prints*, 2018a. URL `http://arxiv.org/abs/1806.05393`.

W. Xiao, H. Chen, Q. Liao, and T. Poggio. Biologically-plausible learning algorithms can scale to large datasets. *ArXiv e-prints*, 2018b. URL `http://arxiv.org/abs/1811.03567`.

D. Zipser and D. Rumelhart. The neurobiological significance of the new learning models. *Computational Neuroscience*, pages 192–200, 1993.

Research was supported by a Burroughs-Wellcome Award (AKL) and by NSF NeuroNex Award DBI-1707398 and the Gatsby Charitable Foundation.

7 Supplemental information

7.1 Training Information

In the MNIST network, the initial learning rate was set to $\eta = 0.0001$, and was reduced to 1×10^{-5} after 20 epochs. A stepped learning rate decay was also used in the CIFAR-10 models. In the LeNet-style model, the initial learning rate was set to $\eta = 0.0005$, and decayed by a factor of 0.8 every 30 epochs. In the all-convolutional model, the initial learning rate was set to $\eta = 0.05$, and multiplied by a factor of 0.1 every 120 epochs. A weight decay with strength $\lambda = 0.001$ was also added to all layers. The batch size was set to $n = 50$ for the MNIST network and $n = 128$ for the CIFAR-10 models. Dropout (Srivastava et al., 2014) was applied with a drop probability of 0.5 after the densely-connected layer in the MNIST network (layer 5). In the all-convolutional architecture, dropout was applied after the downsampling layers (3 and 6) with a drop probability of 0.5, as well after the input layer with a drop probability of 0.2.

7.2 Model architectures

Layer	MNIST	CIFAR-10 Model 1	CIFAR-10 Model 2
Input	$28 \times 28 \times 1$	$24 \times 24 \times 3$*	$32 \times 32 \times 3$
1	5×5 conv. 32 ReLU	5×5 conv. 64 ReLU	3×3 conv. 96 ReLU
2	2×2 max-pool stride 2	2×2 max-pool stride 2	3×3 conv. 96 ReLU
3	5×5 conv. 64 ReLU	5×5 conv. 64 ReLU	3×3 conv. 96 ReLU stride 2
4	2×2 max-pool stride 2	2×2 max-pool stride 2	3×3 conv. 192 ReLU
5	1024 dense ReLU	384 dense ReLU	3×3 conv. 192 ReLU
6	10-way softmax	192 dense ReLU	3×3 conv. 192 ReLU stride 2
7	-	10-way softmax	3×3 conv. 192 ReLU
8	-	-	1×1 conv. 192 ReLU
9	-	-	1×1 conv. 10 ReLU
10	-	-	global average pooling (8×8 dim.)
11	-	-	10-way softmax

Table 2: Model architectures. *CIFAR-10 images were cropped as part of data augmentation to increase the size of the training set.

Layer	ImageNet Model 1	ImageNet Model 2
Input	$299 \times 299 \times 3$	$299 \times 299 \times 3$
1	9×9 conv. 192 ReLU stride 2	9×9 conv. 192 ReLU stride 4
2	2×2 max-pool stride 2	3×3 conv. 192 ReLU stride 2
3	5×5 conv. 192 ReLU stride 2	3×3 conv. 192 ReLU stride 3
4	2×2 max-pool stride 2	5×5 conv. 256 ReLU stride 2
5	512 dense ReLU	3×3 conv. 256 ReLU stride 1
6	512 dense ReLU	3×3 conv. 256 ReLU stride 2
7	1000-way softmax	3×3 conv. 512 ReLU stride 1
8	-	1×1 conv. 512 ReLU stride 1
9	-	1×1 conv. 1000 ReLU stride 1
10	-	global average pooling (4×4 dim.)
11	-	1000-way softmax

Table 3: ImageNet architectures. These are scaled variations of the two CIFAR-10 architectures above.