Error-driven Input Modulation: Solving the Credit Assignment Problem without a Backward Pass

Giorgia Dellaferrera [1 2] **Gabriel Kreiman** [1]

[1] Department of Ophthalmology, Children's Hospital, Harvard Medical School, Boston, MA, United States
[2] Institute of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland
Correspondence to: Giorgia Dellaferrera <giorgia.dellaferrera@childrens.harvard.edu>, Gabriel Kreiman <gabriel.kreiman@childrens.harvard.edu>

Abstract

Supervised learning in artificial neural networks typically relies on backpropagation, where the weights are updated based on the error-function gradients and sequentially propagated from the output layer to the input layer. Although this approach has proven effective in a wide domain of applications, it lacks biological plausibility in many regards, including the weight symmetry problem, the dependence of learning on non-local signals, the freezing of neural activity during error propagation, and the update locking problem. Alternative training schemes — such as sign symmetry, feedback alignment, and direct feedback alignment — have been introduced, but invariably rely on a backward pass that hinders the possibility of solving all the issues simultaneously. Here, we propose to replace the backward pass with a second forward pass in which the input signal is modulated based on the error of the network. We show that this novel learning rule comprehensively addresses all the above-mentioned issues and can be applied to both fully connected and convolutional models. We test this learning rule on MNIST, CIFAR-10, and CIFAR-100. These results help incorporate biological principles into machine learning.

1. Introduction

The backpropagation algorithm (BP) has proven to reach impressive results in training Artificial Neural Networks (ANNs) on a broad range of complex cognitive tasks including speech recognition, image classification (LeCun et al., 2015), and playing games (Silver et al., 2016). However, BP has been criticized for relying on a biologically unrealistic strategy of *synaptic credit assignment*, *i.e.,* estimating how much each parameter has contributed to the output error (Crick, 1989; Whittington & Bogacz, 2019; Lillicrap et al., 2020). In particular, a few aspects of BP appear to be at odds with neurobiology. (i) The same weights are used both in the feedforward and in the feedback pathway, raising the weight symmetry or *weight transport problem*. (ii) The parameter updates depend on the activity of all downstream nodes, while biological synapses learn based on local signals related to the activity of the neurons they connect (Whittington & Bogacz, 2019). (iii) The error gradients are stored separately from the activations (Liao et al., 2016) and do not influence the activities of the nodes produced in the forward pass. Hence, during the backward pass, the network activity is frozen. In the brain, instead, the neural activity is not frozen during plasticity changes and the signals travelling through feedback connections influence the neural activities produced by forward propagation, leading to their enhancement or suppression (Lillicrap et al., 2020). (iv) Input signals cannot be processed in an online fashion, but each sample needs to wait for both the forward and backward computations to be completed for the previous sample. This is referred to as the *update locking problem* (Jaderberg et al., 2017; Czarnecki et al., 2017). These considerations have motivated the development of alternative methods for credit assignment, each proposing solutions to some of these criticisms (Lillicrap et al., 2016; Nokland, 2016; Liao et al., 2016; Frenkel et al., 2019; Nøkland & Eidnes, 2019; Clark et al., 2021). However, none of the training schemes designed so far is able to comprehensively address all the aforementioned challenges.

Here we introduce a novel learning rule that is able to train ANNs on image classification tasks without incurring in the issues described above. We name our scheme PEPITA, Present the Error to Perturb the Input To modulate Activity. PEPITA relies on perturbing the input signal based on error-related information. The difference between the network responses to the input and to its perturbed version is used to compute the synaptic updates. Specifically, the algorithm is implemented by performing two forward computations

for each input sample, avoiding any backward pass and performing the updates in a layerwise feedforward fashion during the second forward pass. By avoiding the backward pass, PEPITA does not suffer from the *weight transport* problem and it partially solves the *update locking* problem. Furthermore, as the error is incorporated in the input signal, PEPITA does not freeze the neural activity to propagate and apply the modulatory signal. Finally, the update rule respects the *locality* constraints. We show that PEPITA can be formulated as a two-factor Hebbian-like learning rule. The error information is used as a global learning signal, which is consistent with biological observations of global neuromodulators that influence synaptic plasticity, and which is similar to several reinforcement learning schemes (Williams, 1992; Mazzoni et al., 1991). As a proof-of-principle, we show that PEPITA can be successfully applied to train both fully connected and convolutional models, leading to performance only slightly worse than BP.

2. Background and related work

We review some of the training schemes proposed to solve the biological unrealistic aspects of backpropagation. We begin by discussing learning rules relying on *unit-specific feedback*, *i.e.,* propagating the error from the output layer to each hidden layer through specific connectivity matrices. Then, we describe recent algorithms which involve delivering learning signals in alternative fashions.

2.1. Credit assignment in conventional networks

The backpropagation algorithm (Rumelhart et al., 1995) relies on a forward and a backward pass. During the forward pass, the input signal is propagated from the input layer to the output layer, where the error is computed by comparing with the target. During the backward pass, the error flows from the top layers to the bottom layers through the same weights used in the feedforward pathway. For each synapse the gradient of the error - and the associated synaptic update - is computed through recursive application of the chain rule. Alternative training principles have been proposed to relax the constraint of symmetric weights. Previous works have shown that effective learning in neural networks can be achieved also when the error is backpropagated through connections that share only the sign and not the magnitude with the feedforward weights (*sign symmetry* algorithm) (Liao et al., 2016; Xiao et al., 2018) or through random fixed connections (*feedback-alignment* algorithm, FA) (Lillicrap et al., 2016). In the latter, the feedback provided by the random matrices is able to deliver useful modulatory information for learning since the forward connections are driven to align with the fixed feedback matrices. As an extension to FA, the *direct feedback alignment* approach (DFA) shows that useful learning information can be propagated directly from the output layer to each hidden unit through random connectivity matrices (Nokland, 2016). In a similar vein, (Akrout et al., 2019) builds on FA and introduces the *weight mirror* neural circuit. Such approach adjusts the initially random feedback weights of FA to improve their agreement with the forward connections, yielding to an improved performance compared to FA.

These biologically inspired training schemes achieve performance close to BP in many pattern recognition tasks without incurring in the *weight transport* problem. However, they do not tackle the issues of non-locality, freezing of neural activity and update locking.

2.2. Credit assignment without random feedback path

Recently, learning techniques based on local error handling have been shown to successfully train ANNs while tackling the *update locking* problem (Nøkland & Eidnes, 2019; Belilovsky et al., 2020; Mostafa et al., 2018). In these approaches, however, each layer is trained independently through auxiliary fixed random classifiers, thereby incurring both in the *weight transport* problem at the level of the classifiers and in a significant computational overhead. With the same goal, the *direct random target projection* (DRTP) algorithm has been proposed to update the parameters of the hidden layers based on the sample labels rather than the network error (Frenkel et al., 2019). Such a strategy overcomes both the *weight transport* and the *update locking* issues without incurring in additional computational requirements. However, it requires that the activity is frozen as the modulatory information on the targets flows through the network. Furthermore, DRTP suffers from a significantly large performance degradation with respect to BP compared to the FA algorithms.

Another original approach to credit assignment known as *global error-vector broadcasting* (GEVB) avoids delivering error information through fixed random connections (Clark et al., 2021). The GEVB learning rule performs parameter updates based on the inner product of the presynaptic activity and a global error vector. This scheme provides a performance almost on par with BP, and solves both the *weight transport* and the *locality* issues. However, it can only operate in a new class of deep neural networks, the vectorized nonnegative networks, which require each node to be represented by a vector unit with the same dimensionality as the output class. This implies a higher computational overhead, which significantly increases for datasets with a large number of classes, *e.g.,* CIFAR-100.

3. Error-driven input modulation

3.1. Overview of the proposed learning rule

Here we introduce a local plasticity rule to train ANNs with supervised learning. The training rules described so far rely on a forward pass, followed by a backward pass during which the error (BP, FA, DFA, GEVB) or the target (DRTP) travels from the output layer to each hidden unit through paths specific to each learning rule. The backward computation leads to at least two biologically implausible aspects. First, the weight updates rely on non-local information: the error coming directly from the output layer (all schemes) as well as downstream feedback connections (BP, FA). Second, during the backward pass the network activity is frozen. We circumvent both these problems by proposing a training scheme that does not involve a backward pass, but rather performs two successive forward passes per input sample. The first pass is similar to any conventional training scheme. In the second forward pass, the error computed during the first pass is used to modify the input. The modulated input travels through the network, eliciting activities slightly different from those of the first pass. Such differences in node activity are used to update the networks parameters. As the output and input dimensionalities are generally different, we use a fixed random matrix F, with zero mean and small standard deviation, to project (*i.e.,* add) the error on the input. We refer to the first and second forward passes as "standard pass" and "modulated pass", respectively. Figure 1 shows a schematic comparison between error propagation in BP, FA and DFA and the novel configuration proposed in this paper, PEPITA.

3.2. The learning rule

Given a fully connected neural network with L layers and an input signal x, in the standard pass the hidden unit and output unit activations are computed as:

$$\begin{aligned} h_1 &= \sigma_1(W_1 x), \\ h_l &= \sigma_l(W_l h_{l-1}) \quad 2 \leq l \leq L. \end{aligned} \tag{1}$$

where σ_l denotes the non-linearity at the output of the l^{th} layer and W_l denotes the matrix of weights between layer $l-1$ and layer l. At the modulated pass, the activations are computed as:

$$\begin{aligned} h_1^{err} &= \sigma_1(W_1(x + Fe)), \\ h_l^{err} &= \sigma_l(W_l h_{l-1}^{err}) \quad 2 \leq l \leq L. \end{aligned} \tag{2}$$

where e denotes the network error and F denotes the fixed random matrix used to project (i.e., add) the error on the input. We compute the error as $e = h_L - target$.
During the two forward passes, the network weights are equal and the difference in the node activities is solely due to the error-driven modulation of the input signal.
The weight updates are computed after (or during, see Discussion) the modulated pass. Each synaptic update depends on the product between a postsynaptic term and a presynaptic term. The postsynaptic term is given by the difference in activation of the postsynaptic node between the standard and the modulated pass. The presynaptic term corresponds to the activity of the presynaptic nodes during the modulated pass.

First layer:

$$\Delta W_1 = (h_1 - h_1^{err}) \cdot (x + F \cdot e)^T \tag{3}$$

Intermediate hidden layers, with $2 \leq l \leq L-1$:

$$\Delta W_l = (h_l - h_l^{err}) \cdot (h_{l-1}^{err})^T \tag{4}$$

Output layer:

$$\Delta W_L = e \cdot (h_{L-1}^{err})^T \tag{5}$$

The weights of the output layer are trained as in BP since the information about the error is directly accessible at the last layer. Furthermore, we tested a modification of the algorithm and replaced the activity of the modulated pass with that of the standard pass in the presynaptic term: $\Delta W_l = (h_l - h_l^{err}) \cdot (h_{l-1})^T$. In the simulations, such modification did not affect the network performance.

Finally, the prescribed synaptic updates are applied depending on the chosen optimizer, as for any gradient-based optimization technique. For example, using the Stochastic Gradient Descent (SGD)-like scheme, with learning rate η:

$$W(t+1) = W(t) - \eta \Delta W \tag{6}$$

The algorithm is reviewed in the pseudocode below.

3.3. Extension to convolutional layers

The same approach can be applied to train convolutional models. However, a modification is required to account for the parameter sharing used in convolutions. As for fully connected models, PEPITA trains the convolutional models through a standard and a modulated forward pass, and the kernels are learnt based on post- and pre-synaptic related terms. For each filter, the update is computed after the modulated pass through the following steps:

1. For each pixel in the output map, first we compute the difference in activity between the standard and modulated pass. Then we multiply such difference by the area of the input map on which the filter was applied to generate the specific output pixel. Each of the computed products has the same dimensionality as the filter.

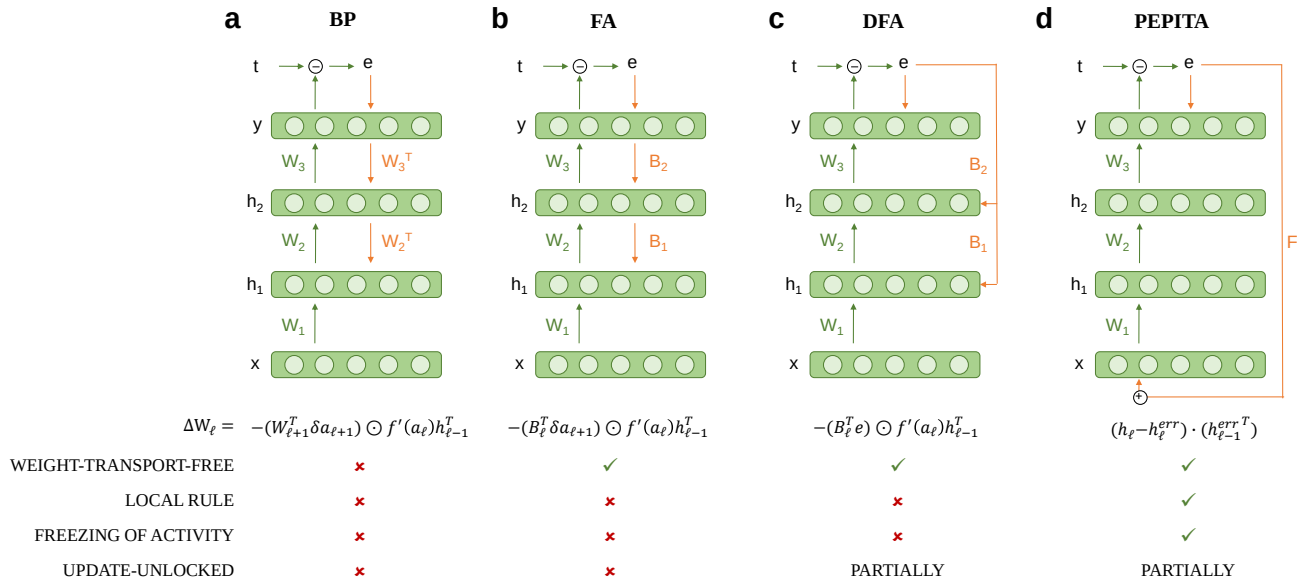

Figure 1. Overview of different error transportation configurations. a) Back-propagation (BP). b) Feedback-alignment (FA). c) Direct feedback-alignment (DFA). d) Present the Error to Perturb the Input To modulate Activity (PEPITA). Green arrows indicate forward paths and orange arrows indicate error paths. Weights that are adapted during learning are denoted as W_i, and weights that are fixed and random are denoted as B_i if specific to a layer (BP, FA, DFA) or F if specific to the input signal (PEPITA).

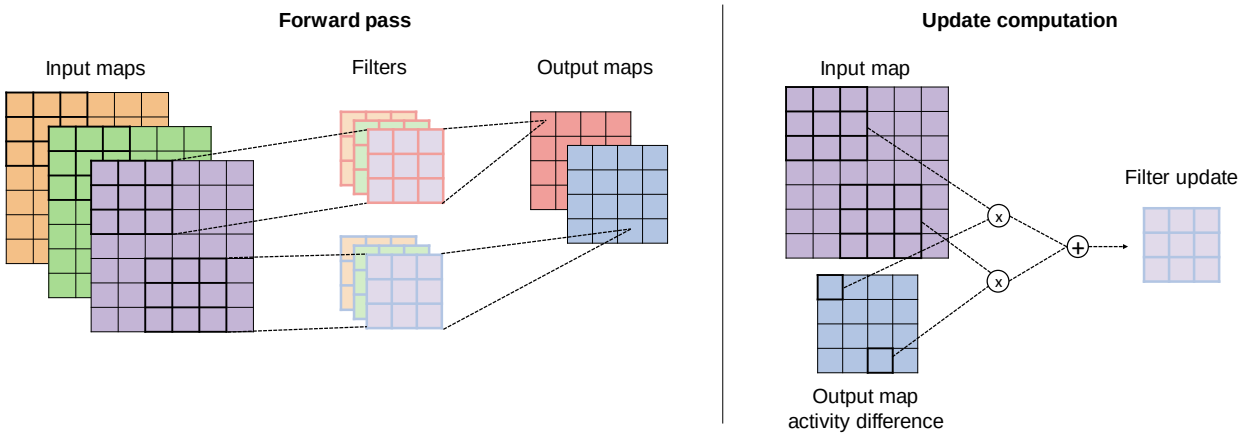

Figure 2. Update scheme for convolutional layers

2. All the products computed in step 1 are summed into a single update term.
3. The resulting term is divided by the number of summed products.
4. The computed update is applied to the filter with the chosen optimization technique, such as Eq. 6.

Figure 2 provides a schematic of the standard forward pass and the computation of the update for a single channel of a filter.

4. Results

4.1. Methods

Inspired by (Liao et al., 2016), we focus on relative differences between algorithms, not absolute performance. Each experiment is a {model, dataset} pair. We tested the PEPITA learning rule on 3 datasets: MNIST (LeCun & Cortes,

Algorithm 1 Implementation of PEPITA

Given: Input (x) and label (*target*)

#standard forward pass

$h_0 = x$

for $\ell = 1, ..., L$

  $h_\ell = \sigma_\ell(W_\ell h_{\ell-1})$

$e = h_L -$ *target*

#modulated forward pass

$h_0^{err} = x + Fe$

for $\ell = 1, ..., L$

  $h_\ell^{err} = \sigma_\ell(W_\ell h_{\ell-1}^{err})$

  if $\ell < L$:

    $\Delta W_\ell = (h_\ell - h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$

  else:

    $\Delta W_\ell = e \cdot (h_{\ell-1}^{err})^T$

2010), CIFAR-10 (Krizhevsky et al., b) and CIFAR-100 (Krizhevsky et al., a). We did not use any data augmentation. For each dataset, we trained both a fully connected and a convolutional model with BP, FA, DRTP and PEPITA. With BP, FA and PEPITA, we used rectified linear unit (ReLU) (Nair & Hinton, 2010) as non-linearity for the hidden layers and softmax for the output layer. With DRTP we used hyperbolic tangent as non-linearity for the hidden layers and sigmoid for the output layer. We used SGD with momentum with hyperparameter 0.9. For BP, FA and DRTP we used cross entropy loss. In the fully connected networks, we introduced dropout with drop rate of 10% after the hidden layer. In the convolutional model we applied Max Pooling after the convolutional layer. We initialized the networks using the He initialization (He et al., 2015). We optimized the learning rate, the learning rate decay schedule, the batch size and the dropout rate separately for each experiment. We used the entire training set for the training and did not use a validation set. The network architectures and simulation details for each dataset and learning rule are reported in Table 2 of the Supplementary Material.

Code for the simulations of fully connected models with PEPITA, BP, FA and convolutional models with PEPITA is provided at: https://drive.google.com/drive/folders/1RQSFttcBKq_d-TLmpB6DZd1ikRzZW2rK?usp=sharing . For the other experiments we used the code from (Frenkel et al., 2019).

4.2. Experimental results

Here, we show that the proposed learning rule successfully trains both fully connected and convolutional networks on image classification tasks. The experimental results are summarized in Table 1. The accuracy obtained by PEPITA for fully connected networks (Table 1, columns 1-3) is close to the accuracy achieved by BP and FA, and superior to that of DRTP for all experiments. Figure 3a shows the learning curve on the test set for the fully connected model trained on MNIST with PEPITA. Figure 4 reports the t-SNE visualization of the representation learned at the hidden layer for the same model. For all datasets we observe an improved performance of the convolutional models trained with PEPITA compared to the fully connected networks (Table 1, columns 4-6), indicating that the convolutional version of PEPITA is able to learn useful two-dimensional filters.

To gain insight into the dynamics of the PEPITA-based learning rule, we considered a two-layer network and measured the alignment of the product between the forward weights ($W_1 W_2$) with the fixed matrix F. As in (Xiao et al., 2018), we flattened the matrices into vectors and computed the angle between the vectors. Figure 3b reports the alignment dynamics of a 1 hidden layer network during training on MNIST. We observe that at initialization the alignment angle is $\simeq 90°$ (*i.e.,* random), and that during training the angle increases, saturating at approximately $120°$. The evolution of the angle alignment finds a plateau as the test accuracy saturates. Interestingly, we observe that our approach encourages a soft 'antialignment' (*i.e.,* the angle increases above $90°$) of the forward matrices with the fixed feedback matrix, in contrast to previously proposed methods (Lillicrap et al., 2016; Liao et al., 2016; Xiao et al., 2018) which promote soft alignment instead (*i.e.,* the angle drops below $90°$). We provide an intuitive explanation of this phenomenon in section 4.3. Furthermore, we have empirically verified that the parameter updates prescribed by PEPITA do not converge with the updates computed with backprop, not even in sign.

4.3. Analytic results

We observed experimentally that the intrinsic dynamics of PEPITA tend to drive the product between the forward weight matrices to 'antialign' with the F^T matrix (Figure 3b). Here we provide a formal proof showing why such phenomenon occurs in the case of a linear fully connected network. We follow the same reasoning presented for feedback alignment in Supplementary Notes 11 and 12 of (Lillicrap et al., 2016) and use the same notation.

We consider a linear network with one hidden layer, where A is the weight matrix from the input to the hidden layer and W is the weight matrix from the hidden to the output layer. Given an input vector x, the hidden layer activation is computed as $h = Ax$ (vector), and the output of the network as $y = Wh$ (vector). Each input x is associated with a desired target $y*$ (vector), which is given by a target linear transformation T, such that $y* = Tx$. The aim of the training is to learn A and W so that the network is functionally equivalent to T. We also define the matrix

Table 1. Test accuracy [%] achieved by BP, FA, DRTP and PEPITA in the experiments. Mean and standard deviation are computed over 10 independent runs.

	FULLY CONNECTED MODELS			CONVOLUTIONAL MODELS		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
BP	98.63±0.03	55.27±0.32	27.58±0.09	98.86±0.04	64.99±0.32	34.20±0.20
FA	98.42±0.07	53.82±0.24	24.61±0.28	98.50±0.06	57.51±0.57	27.15±0.53
DRTP	95.10±0.10	45.89±0.16	18.32±0.18	97.32±0.25	50.53±0.81	20.14±0.68
PEPITA	98.01±0.09	52.57±0.36	24.91±0.22	98.29±0.13	56.33±1.35	27.56±0.60

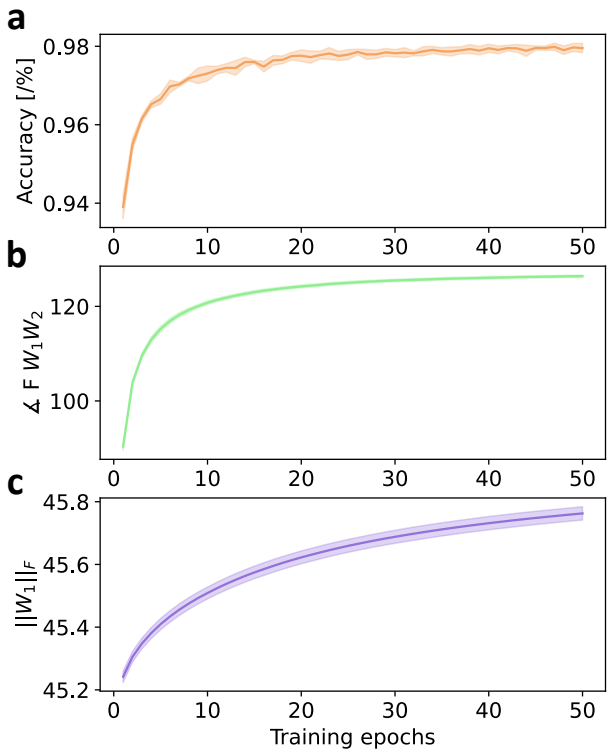

Figure 3. Dynamics of a 1 hidden layer network, with 1024 hidden units, 10% dropout, during training on MNIST with PEPITA. (a) Test accuracy computed after each training epoch (b) Angle between the product of the forward matrices and the F matrix (c) Evolution of the norm of the initial weight matrix. The solid line reports the mean over five independent runs, the shaded area the standard deviation.

$E = T - WA$, so that the error vector can be written as $e = Ex$.

Given a learning rate η, the weight updates prescribed by PEPITA can be written as:

$$\Delta W = \eta e h^T = \eta E x x^T A^T \tag{7}$$

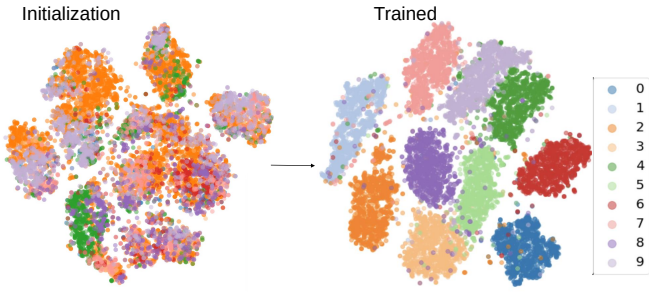

Figure 4. t-SNE embeddings of the MNIST representation at the hidden layer of the fully connected model trained on MNIST, both before (initialization) and after training (trained). Each color corresponds to a different class.

and

$$\begin{aligned}
\Delta A &= \eta(h - h^{err})x^T = && (8)\\
&= \eta(Ax - Ax^{err})x^T = && (9)\\
&= \eta(Ax - A(x + Fe))x^T = && (10)\\
&= \eta(Ax - Ax - FEx))x^T = && (11)\\
&= -\eta AFExx^T. && (12)
\end{aligned}$$

Notice that the prescribed update for A here is computed as $\Delta A = \eta(h - h^{err})x^T$ rather than $\Delta A = \eta(h - h^{err})(x + Fe)^T$. As mentioned earlier, the two rules lead to the same network performance. In this analytical proof for simplicity we consider the first form.

We assume that we train the network with batch learning and that the input samples x are i.i.d. standard normal random variables (*i.e.,* mean 0 and standard deviation 1) and thus $xx^T = I$, where I is the identity matrix. Then the parameter updates can be written as:

$$\Delta W = \eta E x x^T A^T = \eta E A^T \tag{13}$$

and

$$\Delta A = -\eta AFExx^T = -\eta AFE. \tag{14}$$

We note that the minus sign in the update of A does not appear in the update of A using feedback alignment (Lillicrap

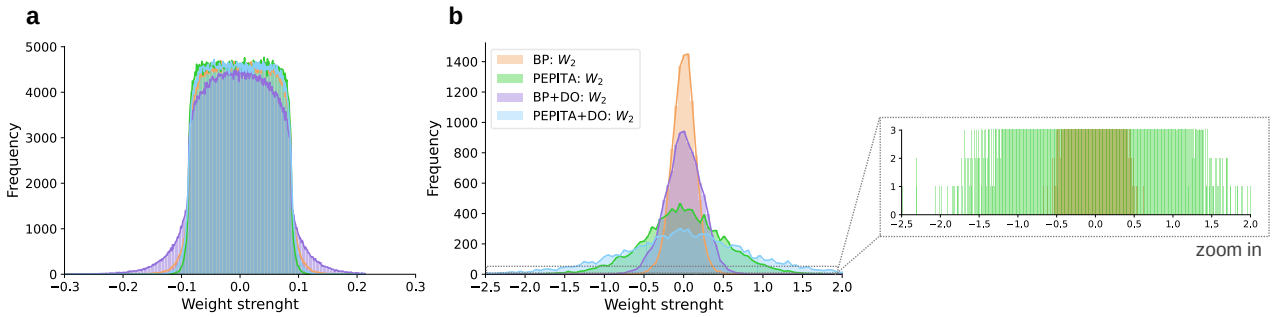

Figure 5. Distribution of the weights trained with BP (orange, purple) and PEPITA (green, blue) on MNIST for a network with one hidden layer with 1024 units, with and without dropout (dropout rate 10%). (a) First hidden layer (b) Output layer

et al., 2016). Therefore, this step of the proof explains the difference in the direction of the alignment.

In the limit of a small η, the discrete time learning dynamics converge to the continuous time dynamical system:

$$\dot{W} = EA^T \tag{15}$$

and

$$\dot{A} = -AFE. \tag{16}$$

where $\dot{A}$ and $\dot{W}$ are the corresponding temporal derivatives. In order to show why the forward matrices and F^T come to antialignment with each other, we need to prove that the time derivative of $tr(FWA)$ is negative. Indeed, the antialignment is defined by the condition $\frac{d}{dt}tr(FWA) < 0$, where $tr(FWA) = [WA]_\downarrow \cdot [F^T]_\downarrow$ and $[\cdot]_\downarrow$ is an operator that flattens a matrix into a vector. To this goal, we decouple the deterministic dynamics of A and W: at first we freeze W and train A, and then we keep A constant and train W.

In the first phase, we impose $\dot{W} = 0$ and let $\dot{A} = -AFE$. In our experiments we observe that the norm of A monotonically increases during training (Figure 3c). Thus $\frac{d}{dt}||A||^2 = \frac{d}{dt}tr(A^TA) > 0$. We can rewrite the time derivative of the norm as:

$$\frac{d}{dt}tr(A^TA) = 2tr(A^T\dot{A}) = \tag{17}$$
$$= 2tr(A^T(-AFE)) = \tag{18}$$
$$= 2tr(-A^TAFE) = \tag{19}$$
$$= 2tr(-||A||^2FE) = \tag{20}$$
$$= -2||A||^2tr(FE). \tag{21}$$

Therefore, we deduce $tr(FE) < 0$.

Next, we consider the second phase, in which $\dot{A} = 0$ and $\dot{W} = EA^T$. Under this condition, we examine the evolution in time of $tr(FWA)$:

$$\frac{d}{dt}tr(FWA) = tr(F\dot{W}A) = \tag{22}$$
$$= tr(FEA^TA) = \tag{23}$$
$$= tr(FE||A||^2) = \tag{24}$$
$$= ||A||^2tr(FE). \tag{25}$$

From the above result $tr(FE) < 0$, we conclude that $\frac{d}{dt}tr(FWA) < 0$. This analytical result for a linear network supports our empirical finding on non-linear models that the product of the forward matrices comes to antialign with F^T. We conjecture that such antialignment enables the error that modulates the input, and consequently the neural activity, to deliver useful learning signals for weight updates in A and W.

4.4. Final weight distribution

We compared the layer-wise weight distribution of networks trained with BP and PEPITA, initialized with the same normal distribution (He et al., 2015). For a two-layer model trained on MNIST, the trained weights in the first layer have similar distributions for BP and PEPITA (Figure 5a), with the weight distribution for BP covering a slightly greater range than for PEPITA. The distribution in the last layer is instead significantly different: both BP and PEPITA lead to bell-shaped distributions with mean close to zero, however the distribution width obtained with PEPITA is significantly wider (Figure 5b). This implies that, while most synaptic connections are weak, there are few sparse and strong weights (zoom of Figure 5b). Interestingly, such heavy-tailed weight distribution reflects the connectivity structure of neurons in biological networks. Indeed, findings of electrophysiological studies indicate that the amplitudes of excitatory post-synaptic potentials between cortical neurons obey a long-tailed pattern, typically lognormal (Buzsáki & Mizuseki, 2014; Song et al., 2005; Iyer et al., 2013). Such fir-

ing pattern implies that some synapses are very strong while many synapses are weak ("strong-sparse and weak-dense" networks) (Teramae & Fukai, 2014). In this vein, (Feldman & Valiant, 2009) demonstrates that a network regime featuring some maximally strong synapses enables sparse and disjoint representation of items. This yields a significantly higher memory capacity compared to a regime supported by weak synapses only. We also remark that previous work (Blundell et al., 2015) has shown that networks regularized with dropout (Hinton et al., 2012) or uncertainty on the weights (*i.e.,* Bayes by Backprop (Blundell et al., 2015)) present a greater range of weights as compared to standard SGD. In conclusion, PEPITA, which is conceived to solve the biological implausible aspects of network training via BP, conveys the network weights resembling more biologically plausible distributions than BP, possibly supporting higher memory capacities as well as more regularized parameters.

5. Discussion

We presented PEPITA, a learning rule which relies on perturbing the input signal through error-related information and updating the network parameters based on local variations of activity. Our results indicate that PEPITA is able to train both fully connected and convolutional neural networks on image classification tasks, with an accuracy comparable to the feedback alignment algorithms. We empirically observe that PEPITA encourages the feedforward weights to evolve towards a soft 'antialignment' with the feedback matrix, and it nudges the weight distribution towards a "strong-sparse and weak-dense" distribution reminiscent of the organization of cortical neurons.

By replacing the backward pass with an additional forward pass, our approach avoids BP's biologically unrealistic aspects of weight transport, non-locality, freezing of neural activity during parameter updates, and update locking. First, PEPITA is not affected by the weight transport problem as it does not send weight-specific feedback signals. Secondly, regarding locality, the prescribed updates of each synapse solely rely on the activity of the nodes it is connected to. The only global signal required by the algorithm is the error projected onto the input. Its role in perturbing the node activity can correspond to a neuromodulator that influences local synaptic plasticity (Mazzoni et al., 1991; Williams, 1992; Clark et al., 2021). Third, during training, PEPITA never requires the network activity to be frozen since the error is propagated together with the input signal. Finally, in PEPITA the updates are performed layer-wise in a feedforward fashion allowing to partially solve the update locking issue. Indeed, the weights of the first layer can be updated right at the beginning of the second forward pass and do not need to wait for the updates of the downstream layers. Hence, the forward computation for the next input sample at the first layer can start in parallel with the update of the second layer from the previous sample, and so on. In the case of very deep networks, such a strategy could substantially reduce the computational time, suggesting that PEPITA could be suitable for edge computing devices requiring fast processing of the input signals.

While our approach addresses BP's issues of biological plausibility, it introduces additional elements with respect to BP and FA. Specifically, PEPITA involves the projection of the error onto the input through a fixed random matrix. We speculate that the role of such a projection matrix is reminiscent of the connectivity matrices supporting the cortico-thalamo-cortical loops throughout neocortex. Furthermore, PEPITA requires storing the activations of the standard pass during the modulated pass. We suggest that this could be implemented in biological neurons through dendritic and somatic activities. Earlier works have proposed models of two-compartmental neurons in which learning is driven by local prediction mismatch between synaptically-driven dendritic activity and somatic output (Asabuki & Fukai, 2020; Sacramento et al., 2018). We propose that if we apply PEPITA to train a network of such two-compartmental neurons, the response to the input signal in the standard forward pass is first integrated in the dendritic compartment and then encoded in the somatic activity. Then, in the modulated forward pass, the dendrites encode the response to the modulated input while the soma is still storing the response to the original input. The mismatch between the dendritic activity encoding the response to the modulated input and the somatic activity encoding the response to the original input drives the synaptic update.

Importantly, PEPITA bears structural similarity with learning rules that are believed to take place in the brain, *i.e.,* Hebbian-like approaches. Indeed, the update rule shown in the pseudocode can be decoupled into two separate successive updates: $\Delta W_\ell^1 = (h_\ell) \cdot (h_{\ell-1}^{err})^T$ and $\Delta W_\ell^2 = (-h_\ell^{err}) \cdot (h_{\ell-1}^{err})^T$. Each of the two updates embodies a fundamental feature of Hebbian learning (Gerstner et al., 2014), *i.e.,* the synaptic strength modification is proportional to both a presynaptic and a postsynaptic signal. Therefore the definition of PEPITA fits in the criteria to be a biologically plausible mechanism.

In conclusion, we have demonstrated that a learning rule that solves the biologically implausible aspects of BP by relying only on forward computations, is able to train both fully connected and convolutional models with a performance close to BP and on par with FA. The proposed algorithm can thus help bridge the gap between neurobiology and machine learning.

References

Akrout, M., Wilson, C., Humphreys, P., Lillicrap, T., and Tweed, D. Deep learning without weight transport. In *NeurIPS*, 2019.

Asabuki, T. and Fukai, T. Somatodendritic consistency check for temporal feature segmentation. *Nature Communications*, 11, 03 2020.

Belilovsky, E., Eickenberg, M., and Oyallon, E. Decoupled greedy learning of cnns, 2020.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1613–1622. JMLR.org, 2015.

Buzsáki, G. and Mizuseki, K. The log-dynamic brain: How skewed distributions affect network operations. *Nature reviews. Neuroscience*, 15:264–278, 02 2014.

Clark, D. G., Abbott, L. F., and Chung, S. Credit assignment through broadcasting a global error vector, 2021.

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

Czarnecki, W. M., Swirszcz, G., Jaderberg, M., Osindero, S., Vinyals, O., and Kavukcuoglu, K. Understanding synthetic gradients and decoupled neural interfaces. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 904–912. JMLR.org, 2017.

Feldman, V. and Valiant, L. G. Experience-induced neural circuits that achieve high capacity. *Neural Comput.*, 21 (10):2715–2754, oct 2009. ISSN 0899-7667.

Frenkel, C., Lefebvre, M., and Bol, D. Learning without feedback: Direct random target projection as a feedback-alignment algorithm with layerwise feedforward training, 2019. Preprint at https://arxiv.org/abs/1909.01311.

Gerstner, W., Kistler, W. M., Naud, R., and Paninski, L. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press, USA, 2014. ISBN 1107635195.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, December 2015.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Improving neural networks by preventing co-adaptation of feature detectors. 2012. Preprint at https://arxiv.org/abs/1207.0580.

Iyer, R., Menon, V., Buice, M., Koch, C., and Mihalas, S. The influence of synaptic weight distribution on neuronal population dynamics. *PLOS Computational Biology*, 9 (10):1–16, 10 2013. doi: 10.1371/journal.pcbi.1003248.

Jaderberg, M., Czarnecki, W. M., Osindero, S., Vinyals, O., Graves, A., Silver, D., and Kavukcuoglu, K. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1627–1635. JMLR.org, 2017.

Krizhevsky, A., Nair, V., and Hinton, G. Cifar-100 (canadian institute for advanced research). a. URL http://www.cs.toronto.edu/~kriz/cifar.html.

Krizhevsky, A., Nair, V., and Hinton, G. Cifar-10 (canadian institute for advanced research). b. URL http://www.cs.toronto.edu/~kriz/cifar.html.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521:436–44, 05 2015. doi: 10.1038/nature14539.

Liao, Q., Leibo, J. Z., and Poggio, T. How important is weight symmetry in backpropagation?, 2016.

Lillicrap, T., Santoro, A., Marris, L., Akerman, C., and Hinton, G. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21, 04 2020. doi: 10.1038/s41583-020-0277-3.

Lillicrap, T. P., Cownden, D., Tweed, D. B., and Akerman, C. J. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7:13276, 2016.

Mazzoni, P., Andersen, R. A., and Jordan, M. I. A more biologically plausible learning rule for neural networks. *Proceedings of the National Academy of Sciences*, 88(10): 4433–4437, 1991. ISSN 0027-8424.

Mostafa, H., Ramesh, V., and Cauwenberghs, G. Deep supervised learning using local errors. *Frontiers in Neuroscience*, 12, 2018. ISSN 1662-453X.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pp. 807–814, Madison, WI, USA, 2010. Omnipress.

Nokland, A. Direct feedback alignment provides learning in deep neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing*

Systems, NIPS'16, pp. 1045–1053, Red Hook, NY, USA, 2016. Curran Associates Inc.

Nøkland, A. and Eidnes, L. H. Training neural networks with local error signals. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 4839–4850, 09–15 Jun 2019.

Rumelhart, D. E., Durbin, R., Golden, R., and Chauvin, Y. *Backpropagation: The Basic Theory*, pp. 1–34. L. Erlbaum Associates Inc., USA, 1995.

Sacramento, J. a., Ponte Costa, R., Bengio, Y., and Senn, W. Dendritic cortical microcircuits approximate the backpropagation algorithm. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

Song, S., Sjöström, P. J., Reigl, M., Nelson, S., and Chklovskii, D. B. Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLOS Biology*, 3 (3):e68, 03 2005.

Teramae, J. and Fukai, T. Computational implications of lognormally distributed synaptic weights. *Proceedings of the IEEE*, 102(4):500–512, 2014.

Whittington, J. and Bogacz, R. Theories of error backpropagation in the brain. *Trends in Cognitive Sciences*, 23:235–250, 03 2019.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992.

Xiao, W., Chen, H., Liao, Q., and Poggio, T. Biologically-plausible learning algorithms can scale to large datasets, 2018.

A. Simulation details.

Table 2. Network architectures and settings used in the experiments. For the convolutional layers A,B,C means A feature maps of size B, with stride C.

	FULLY CONNECTED MODELS			CONVOLUTIONAL MODELS		
	MNIST	CIFAR10	CIFAR100	MNIST	CIFAR10	CIFAR100
INPUTSIZE	28×28×1	32×32×3	32×32×3	28×28×1	32×32×3	32×32×3
LAYER 1	FC1:1024	FC1:1024	FC1:1024	CONV1:32,5,1	CONV1:32,5,1	CONV1:32,5,1
LAYER 2	FC2:10	FC2:10	FC2:100	FC1:10	FC1:10	FC1:100
η BP	0.1	0.01	0.01	0.1	0.01	0.01
η FA	0.1	0.001	0.001	0.1	0.01	0.01
η DRTP	0.01	0.001	0.001	0.01	0.01	0.01
η PEPITA	0.1	0.01	0.01	0.1	0.1	0.1
η DECAY	×0.1 AT 60	×0.1 AT 60,90	×0.1 AT 60,90	×0.1 AT 10,30,60	×0.1 AT 10,30,60	×0.1 AT 10,30,60
BATCH SIZE	64	64	64	100	100	100
F STD	$0.05{\cdot}2\sqrt{\frac{6}{28\cdot28\cdot1}}$	$0.05{\cdot}2\sqrt{\frac{6}{28\cdot28\cdot1}}$	$0.05{\cdot}2\sqrt{\frac{6}{28\cdot28\cdot1}}$	$0.05{\cdot}2\sqrt{\frac{6}{32\cdot32\cdot3}}$	$0.05{\cdot}2\sqrt{\frac{6}{32\cdot32\cdot3}}$	$0.05{\cdot}2\sqrt{\frac{6}{32\cdot32\cdot3}}$
#EPOCHS	100	100	100	100	100	100
DROPOUT	10%	10%	10%	-	-	-