Kickback cuts Backprop's red-tape:
Biologically plausible credit assignment in neural networks

David Balduzzi
david.balduzzi@vuw.ac.nz
Victoria University of Wellington

Hastagiri Vanchinathan
hastagiri@inf.ethz.ch
ETH Zurich

Joachim Buhmann
jbuhmann@inf.ethz.ch
ETH Zurich

Abstract

Error backpropagation is an extremely effective algorithm for assigning credit in artificial neural networks. However, weight updates under Backprop depend on lengthy recursive computations and require separate output and error messages – features not shared by biological neurons, that are perhaps unnecessary. In this paper, we revisit Backprop and the credit assignment problem.

We first decompose Backprop into a collection of interacting learning algorithms; provide regret bounds on the performance of these sub-algorithms; and factorize Backprop's error signals. Using these results, we derive a new credit assignment algorithm for nonparametric regression, Kickback, that is significantly simpler than Backprop. Finally, we provide a sufficient condition for Kickback to follow error gradients, and show that Kickback matches Backprop's performance on real-world regression benchmarks.

Introduction

The discovery of error backpropagation was hailed as a breakthrough because it solved the main problem of distributed learning – the spatial credit assignment problem (Werbos 1974; Rumelhart, Hinton, and Williams 1986). Decades later, Backprop is the workhorse underlying most deep learning algorithms, and a major component of the state-of-the-art in supervised learning.

Since Backprop's introduction, there has been tremendous progress improving the performance of neural networks. An enormous amount of effort has been expended exploring the effects of: the activation functions of nodes; network architectures (e.g. number of layers and number of nodes); regularizers such as dropout (Srivastava et al. 2014); modifications to accelerate gradient descent; and unsupervised methods for pre-training to find better local optima.

However, it was known from the start that Backprop is not biologically plausible (Crick 1989). Implementing Backprop requires that neurons produce two distinct signals – outputs and errors – whereas only one has been observed in cortex (Lamme and Roelfsema 2000; Roelfsema and van Ooyen 2005).

It is therefore remarkable that almost no attempts have been made to rethink the core algorithm – backpropagation – and the problem that it solves – credit assignment. This paper revisits the credit assignment problem and takes a fresh look at the signaling architecture that underlies Backprop.

Outline. Our starting point is to decompose Backprop into local learning algorithms, Theorem 1. Nodes under Backprop are modeled as agents that minimize their losses. Backprop ensures that nodes cooperate, towards the shared goal of minimizing the output layer's error, by gluing together their loss functions using recursively computed error signals.

Reformulating Backprop as local learners immediately suggests modifying the signaling architecture (the glue) whilst keeping the learners. In this paper, we aim to simplify Backprop's error signals.

Theorem 2 lays the groundwork, by providing a regret bound for local learners that holds for any scalar feedback – and not just the error signals used by Backprop.

The next step is to show that, when a neural network has 1-dimensional outputs (e.g. nonparametric regression), Backprop's error signals factorize into two components, Theorem 3. The first component is a scalar error computed at the output layer that is analogous to a neuromodulatory signal; the second is a complicated sum over paths to the output layer that has no biological analog.

Our proposed algorithm, *Kickback*, modifies Backprop by truncating the second component. Kickback is *not* gradient descent on the output error. Nevertheless, Theorem 4 provides a simple sufficient condition, coherence, for Kickback to follow the error gradient.

It turns out that many of the components of Kickback have close neurophysiological analogs. We discuss Kickback's biological significance by relating it to a recently developed, discrete-time model neuron (Balduzzi and Besserve 2012).

Finally, we present experiments demonstrating that Kickback matches Backprop's performance on standard benchmark datasets.

Synopsis. Our contribution is twofold. Firstly, we provide a series of simple, fundamental theorems on Backprop, one of the most heavily used learning algorithms. In particular, Theorem 1 suggests that ideas from multi-agent learning and mechanism design have a role to play in deep learning.

Secondly, we propose Kickback, a stripped-down variant of Backprop that simultaneously performs well and ties in nicely with the signaling architecture of cortical neurons.

Related work. The idea of building learning algorithms out of individual learning agents dates back to at least (Selfridge 1958). More recent approaches include REINFORCE (Williams 1992), the hedonistic neurons in (Seung 2003), and the neurons modeled using online learning in (Hu et al. 2013). None of these approaches have led to algorithms that are competitive on benchmarks.

The algorithm closest to Kickback is attention-gated reinforcement learning (AGREL), which also eliminates the error signals from Backprop (Roelfsema and van Ooyen 2005). AGREL and Kickback are analogous at a high level, however the details differ markedly. In terms of results, the main differences are as follows. Firstly, we implement Kickback for networks with 2 and 3 hidden layers; whereas AGREL was only implemented for 1 hidden layer. Indeed, as discussed in (Roelfsema and van Ooyen 2005), extending AGREL to multiple hidden layers is problematic. Secondly, AGREL achieved comparable performance to Backprop on toy datasets: XOR, counting, and a mine detection dataset containing ± 200 inputs; whereas Kickback matches Backprop on much larger, real-world nonparametric regression problems. Finally, AGREL converges 1.5 to 10 times slower than Backprop, whereas Kickback's convergence is essentially identical to Backprop.

Error Backpropagation

Recent work has shown that using rectilinear functions instead of sigmoids can significantly improve the performance of neural networks. We restrict to rectifiers because they perform well empirically (Jarrett et al. 2009; Nair and Hinton 2010; Glorot, Bordes, and Bengio 2011; Krizhevsky, Sutskever, and Hinton 2012; Zeiler et al. 2013; Dahl, Sainath, and Hinton 2013; Maas, Hannun, and Ng 2013), are more realistic models of cortical neurons than sigmoid units (Glorot, Bordes, and Bengio 2011), and are universal function approximators (Leshno et al. 1993).

Denote the positive and negative rectifiers by $P(a) := \max(0, a)$ and $N(a) := -\max(0, a)$ respectively. Rectifiers are continuous everywhere and differentiable everywhere except at 0. The subgradients are:

$$\nabla P(a) := \begin{cases} 1 & a > 0 \\ 0 & \text{else} \end{cases} \qquad \nabla N(a) := \begin{cases} -1 & a > 0 \\ 0 & \text{else.} \end{cases}$$

Let $S(a)$ denote either $P(a)$ or $N(a)$; the notation is useful when discussing positive and negative rectifiers simultaneously. Similarly, let $\mathbb{1}$ denote either subgradient. The subgradient $\mathbb{1}$ acts as a *signed* indicator function.

The output of node j is $S_{\mathbf{w}_j}(\mathbf{x}) := S(\langle \mathbf{w}_j, \mathbf{x} \rangle)$. We say that node j fires if $\langle \mathbf{w}_j, \mathbf{x} \rangle > 0$; the firing rate is $|S_{\mathbf{w}_j}(\mathbf{x})|$.

From global to local learning. Under Backprop the entire neural network optimizes a single objective function using gradient descent on the network's error. The partial derivatives with respect to weights are computed via the chain rule.

In more detail, suppose a neural network has error function $E(\mathbf{x}, y)$ that depends on the output layer $\mathbf{x}_o$ and labels y. Backprop recursively updates weight vectors using the chain rule. For nodes in the output layer, $\delta_o := \frac{\partial E}{\partial x_o}$. For hidden node j, the error signal is derived via

$$\delta_j := \sum_{\{k \mid j \to k\}} w_{jk} \mathbb{1}_k \delta_k. \tag{1}$$

Our first result is that, when the hidden nodes are rectilinear, Backprop decomposes into many interacting learning algorithms that maximize local objective functions.

Consider the following setup.

Definition 1 (rectilinear loss)**.** *A node with a rectilinear activation function $S_{\mathbf{w}}(\bullet)$ receives input $\mathbf{x}$ and incurs **rectilinear loss***

$$\boldsymbol{\ell}_{RL}(\mathbf{w}, \mathbf{x}, \varphi) := \varphi \cdot S_{\mathbf{w}}(\mathbf{x}) = \begin{cases} \pm\varphi \cdot \langle \mathbf{w}, \mathbf{x} \rangle & \textit{if } \langle \mathbf{w}, \mathbf{x} \rangle > 0 \\ 0 & \textit{else} \end{cases}$$

that depends on an externally provided scalar φ.

If the node fires then the rectilinear loss is the linear loss $\boldsymbol{\ell}_L(\mathbf{w}, \varphi \cdot \mathbf{x}) := \langle \mathbf{w}, \varphi \cdot \mathbf{x} \rangle$, which has been extensively analyzed in online learning (Cesa-Bianchi and Lugosi 2006). If the node does not fire then the rectilinear loss is zero.

Theorem 1 (Backprop decomposes into local learners)**.** *The weight updates induced by Backprop on rectilinear hidden node j are the same as gradient descent on the rectilinear loss:*

$$\nabla_{\mathbf{w}_j} \boldsymbol{\ell}_{RL}(\mathbf{w}_j, \mathbf{x}, \delta_j) = \nabla_{\mathbf{w}_j} E(\mathbf{x}_o, y).$$

The rectilinear loss resembles the hinge loss. However, it is not convex since, even if the node has a positive rectifier, φ is not necessarily positive.

Proof Sketch. Let $a_j = \langle \mathbf{w}_j, \mathbf{x} \rangle$ and $x_j = S(a_j)$. Weight updates under Backprop are

$$\Delta w_{ij} \propto -\frac{\partial E}{\partial w_{ij}} = -\frac{\partial E}{\partial a_j} \frac{\partial a_j}{\partial w_{ij}} = -\delta_j \cdot x_i \cdot \mathbb{1}_j.$$

Weight updates for gradient descent on the rectilinear loss are

$$\Delta w_{ij} \propto -\frac{\partial \boldsymbol{\ell}_{RL}}{\partial w_{ij}} = -\varphi \cdot x_i \cdot \mathbb{1}_j. \tag{2}$$

Substituting $\varphi \leftarrow \delta_j$ yields the theorem. □

Backprop is thus a collection of local optimizations glued together by the recursively computed error signals.

A regret bound. Since the rectilinear loss has not been previously studied, our second result is a guarantee on the predictive performance of the local learners.

Theorem 2 (regret bound for local learners)**.** *Suppose that weights are projected into a compact convex set $\mathcal{K}$ at each time step. Let $F := \{t \mid S_{\mathbf{w}^t}(\mathbf{x}^t) > 0\}$ denote the time-points when the node fired.*

The following guarantee holds for any sequence of inputs and scalar feedback when $|F| \geq 1$:

$$\frac{1}{|F|}\left[\sum_{t\in F}\ell_{RL}(\mathbf{w}^t,\mathbf{x}^t,\varphi^t) - \inf_{\mathbf{w}\in\mathcal{K}}\sum_{t\in F}\ell_{RL}(\mathbf{w},\mathbf{x}^t,\varphi^t)\right] \leq \sqrt{\frac{8DE}{|F|}}$$

where $D = \max_{t\in F}\left\{\|\varphi^t\cdot\mathbf{x}^t\|_2^2\right\}$ *and* $E = \max_{\mathbf{w}\in\mathcal{K}}\|\mathbf{w}\|_2^2 - \|\mathbf{w}_1\|_2^2$.

Theorem 2 shows that the loss incurred by rectifiers on *the inputs that cause them to fire* converges towards the loss of the best weight-vector in hindsight. The theorem is shown for hard constraints (i.e. projecting into $\mathcal{K}$); similar results hold for convex regularizers.

The result holds for arbitrary sequences of inputs and feedbacks, including adversarial. It is therefore more realistic than the standard *i.i.d.* assumption. Indeed, even if a network's inputs are *i.i.d.*, the inputs to nodes in deeper layers are not – due to weight-updates within the network.

Proof Sketch. Standard results on online learning do not directly apply, since the rectilinear loss is not convex. To adapt these results, observe that, by (2), nodes only learn from the inputs that cause them to fire.

Clearly, $S_{\mathbf{w}^t}(\mathbf{x}^t) = \langle\mathbf{w}^t,\mathbf{x}^t\rangle$ for all $t \in F$. That is, a node's output is linear on the inputs for which it fires. Further, the rectilinear loss is linear on F. The theorem follows from a well-known result on gradient descent for the linear loss, see (Hazan 2012). □

Theorem 2 is not restricted to Backprop's error signals; it holds for any sequence of scalars $\{\varphi^t\}$. This suggests exploring alternate ways of gluing together local learners.

Kickback: truncated error backpropagation

Backprop has two unfortunate properties. Firstly, the error signals δ_j are computationally expensive: they depend on the activity and weights of all downstream nodes in the network. Secondly, nodes produce two distinct signals: outputs that are fed forward and errors that are fed back. In contrast, cortical neurons communicate with only one signal type, spikes, which are sent in all directions. This suggests that it may be possible to make do with less.

Viewed from a distance, Backprop is a single distributed optimization, performing gradient descent on the network's error. Zooming in, via Theorem 1, reveals that Backprop is a collection of local learners *glued together* by recursively computed error signals. We thus have a framework for experimenting with alternate feedback signals (Balduzzi 2014).

Kickback takes the same local learners as Backprop but weakens the glue that binds them, thereby reducing communication complexity and increasing biological plausibility.

Factorizing Backprop's error signals. It is necessary to distinguish between global and local error signals. Local errors signals are the recursively computed signals δ_j. The global error is the derivative of the network's error function with respect to the activity of the output layer.

Definition 2 (influence). *The influence of node j on node k is $\tau_{jk} := w_{jk}\mathbb{1}_k$. The **influence** of node j on the next layer is $\tau_j := \sum_{\{k|j\to k\}}\tau_{jk}$. The **total influence** of node j on downstream nodes is*

$$\pi_j := \left(\sum_{\{k|j\to k\}}\tau_{jk}\left(\sum_{\{l|k\to l\}}\tau_{kl}\left(\sum_{\{m|l\to m\}}\cdots\right)\right)\right), \tag{3}$$

the sum over all paths from j to the output layer.

Our third result is that Backprop's error signals factorize whenever a neural network has 1-dimensional outputs.

Theorem 3 (error signal factorization). *Suppose neural network N has scalar output and let $\beta = \frac{\partial E}{\partial x_o}$ be the global error. Then, the error signal of a hidden node j factorizes as*

$$\delta_j = \beta\cdot\pi_j = \big(\textit{global error}\big)\cdot\big(\textit{total influence}_j\big). \tag{4}$$

The theorem holds in the setting of nonparametric regression. Multi-label classification is excluded.

Proof Sketch. Backprop recursively updates weight vectors using the chain rule, recall (1). When the output is one-dimensional, x_o contributes β to the recursive computation of π_j over hidden nodes. □

Kickback. We are now ready to introduce Kickback.

Algorithm 1 (Kickback). *The **truncated feedback** ϵ_j at node j is*

$$\epsilon_j := \beta\cdot\tau_j = \big(\textit{global error}\big)\cdot\big(\textit{influence}_j\big). \tag{5}$$

*Under **Kickback**, hidden nodes perform gradient descent on the rectilinear loss with truncated feedback:*

$$\Delta w_{ij} \propto -\nabla_{w_{ij}}\ell_{RL}(\mathbf{w}_j,\mathbf{x},\epsilon_j) = -\beta\cdot\tau_j\cdot x_i\cdot\mathbb{1}_j. \tag{6}$$

Kickback and Backprop are contrasted in Figure 1 and in equations (4) versus (5). Importantly, Kickback eliminates the need for nodes to communicate error signals – as distinct from their outputs.

Kickback as time-averaged Backprop. Truncating the feedback signal, from (4) to (5), preserves more information than appears at first glance. The truncated signal received by node j *explicitly* depends on j's influence on the next layer. However, Kickback *implicitly* incorporates information about the influence of multiple layers.

For simplicity, suppose there is no regularizer and that the learning rate η is constant. Then, summing over the updates in (2), a weight at time T is $w_{ij}^T = \eta\sum_{t\in F_j}\varphi_j^t x_i^t$. In the specific case of Kickback, the weight is

$$w_{ij}^T = \eta\sum_{t\in F_j}\left(\beta^t x_i^t \tau_j^t\right) = \eta\sum_{t\in F_j}\Big(\beta^t x_i^t\sum_{\{k|j\to k\}}w_{jk}^t\mathbb{1}_k\Big).$$

The weight w_{ij}^T thus implicitly incorporates the effect of interactions $\tau_{jk}^t = w_{jk}^t\mathbb{1}_k^t$ in the next layer down, and so on recursively.

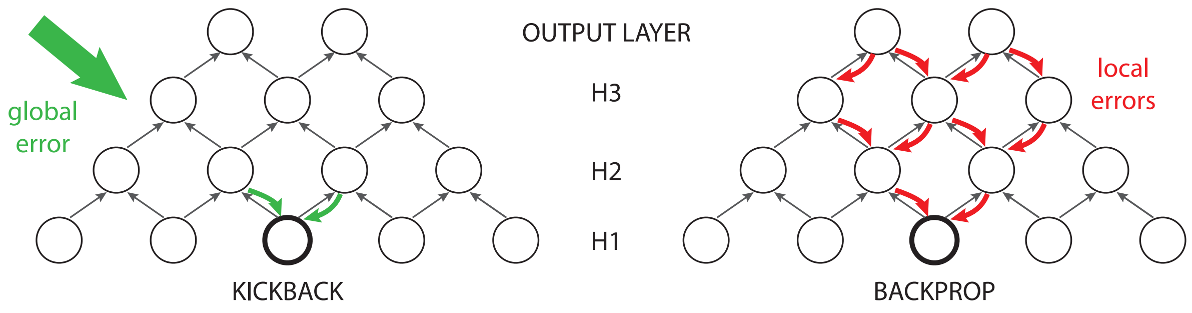


Figure 1: **Schematic comparison of Kickback and Backprop.** Black arrows represent feedforward conenctivity. Colored arrows depict paths used to compute the bold node's feedback under each algorithm.

Coherence. With a small enough learning rate, gradient descent will tend towards a local minimum. Kickback does not perform gradient descent on the error function since it uses modified feedback signals. Thus, without further assumptions, it is not guaranteed to improve performance. Our fourth result is to provide a sufficient condition.

Definition 3 (coherence)**.** *Node j is **coherent** when $\tau_j > 0$. A network is coherent when all its nodes are coherent.*

Example 1 (signed coherence)**.** *An easy way to guarantee coherence for every node is to impose the purely* local *condition that all connections targeting positive nodes have positive weights, and similarly that all connections targeting negative nodes have negative weights.*

If a network is coherent, then increasing a positive node's firing rate increases the average (signed) activity in the next layer and *all downstream layers*. Increasing the activity of negative nodes has the opposite effect.

On the other hand, if a network is not coherent, then nothing can be said in general about how the activity of nodes in one layer affects other layers.

Coherence thus enforces *interpretability*: it ensures that a node's influence on the next layer is indicative of its *total* influence on all downstream layers.

Theorem 4 (coherence $\implies$ Kickback reduces error)**.** *If a network is coherent then weight updates under Kickback, with a sufficiently small learning rate, improve performance.*

Proof Sketch. It suffices to show that the feedback has the same sign under Backprop, $\delta_j = \beta \cdot \pi_j$, and Kickback, $\epsilon_j = \beta \cdot \tau_j$ for an arbitrary hidden node j.

If j is coherent then $\tau_j > 0$. If, furthermore, all downstream nodes are coherent, then unraveling (3) obtains that $\pi_j > 0$. The result follows. □

Under Backprop, each node's total influence is computed explicitly. Kickback makes do with less information: a node "knows" its influence on the next layer, but does not "know" its total influence.

Biological relevance

There is a direct link from Kickback to neurobiology provided by the *selectron*: a simplified model neuron (Balduzzi and Besserve 2012). The selectron is derived from standard models of neural dynamics and learning – the Spike Response Model (SRM) and Spike-Timing Dependent Plasticity (STDP) – by taking the so-called "fast-time constant limit" to go from continuous to discrete time.

Theorem 5 (selectron)**.** *The fast time-constant limit of the SRM (Gerstner and Kistler 2002) is a node that outputs 1 if $\langle \mathbf{w}, \mathbf{x} \rangle > 0$ and 0 otherwise.*

Weight updates in the fast time-constant limit of neuromodulated STDP (Song, Miller, and Abbott 2000) are

$$\Delta w_{ij} \propto \nu \cdot x_i \cdot \mathbb{1}_j = \begin{cases} \nu \cdot x_i & \text{if } \langle \mathbf{w}_j, \mathbf{x} \rangle > 0 \\ 0 & \text{else,} \end{cases} \tag{7}$$

where ν is a global, scalar-valued neuromodulatory signal.

The weight updates in (7) *are gradient* ascent *on*

$$\mathit{Reward}(\mathbf{w}, \mathbf{x}, \nu) := \nu \cdot P_{\mathbf{w}}(\mathbf{x}) = \begin{cases} \nu \langle \mathbf{w}, \mathbf{x} \rangle & \text{if } \langle \mathbf{w}_j, \mathbf{x} \rangle > 0 \\ 0 & \text{else.} \end{cases}$$

Setting $\varphi := -\nu$ in *Reward*$(\mathbf{w}, \mathbf{x}, \nu)$ recovers the rectilinear loss in Definition 1. The selectron thus maximizes a *rectilinear reward* via the same weight updates used to minimize the rectilinear loss. The difference between the two models is that the selectron has 0/1-valued outputs (spikes), whereas nodes have real-valued outputs (firing rates).

Proof. (Balduzzi and Besserve 2012). □

Kickback's weight updates are $\Delta w_{ij} \propto -\beta \cdot \tau_j \cdot x_i \cdot \mathbb{1}_j$. Each factor has a biological analog. The global error, β, corresponds to neuromodulators, such as dopamine, that have been experimentally observed to signal prediction errors for future rewards (Schultz, Dayan, and Montague 1997). The *kickback* term, τ_j, corresponds to NMDA backconnections that have a multiplicative effect on synaptic updates, proportional to the weighted sum of downstream activity (Vargas-Caballero and Robinson 2003; Roelfsema and van Ooyen 2005). The feedforward term, x_i, corresponds to presynaptic spiking activity (Song, Miller, and Abbott 2000). Finally, the signed indicator function $\mathbb{1}_j$, ensures that only active nodes update their weights – thereby playing the role of post-synaptic activity in STDP.

The regret bound in Theorem 2 is also biologically significant. Synapses incur a significant metabolic cost (Tononi and Cirelli 2014). Regularizing synaptic weights provides a way to quantify metabolic costs. Indeed, limits on the physical size and metabolic budget of synapses suggest that synaptic weights may be constrained to an ℓ_1-ball (Balduzzi and Besserve 2012).

To the best of our knowledge, Theorem 2 is the first *adversarial* generalization bound for a biologically derived model. The generalization bound for the selectron in (Balduzzi and Besserve 2012) assumes that inputs are *i.i.d.* Moving beyond the *i.i.d.* assumption is important because biological organisms face adversarial environments.

The final ingredient is coherence. Investigating biologically plausible mechanisms that ensure coherence (or some other sufficient condition) is deferred to future work.

Experiments

Goals. Our primary aim is to compare Kickback's performance to Backprop. We present results on two robotics datasets, SARCOS[1] and Barrett WAM[2]. Kickback's performance across multiple hidden layers is of particular interest, since it truncates errors. Results for 3 hidden layers are reported; results for 1 and 2 hidden layers were similar.[3] A secondary aim is to investigate the effect of coherence.

Competing on the datasets tackled by deep learning algorithms is not yet feasible. Further work is required to adapt Kickback to multiclass learning.

Architecture. Experiments were performed on a 5-layer network with 2 output nodes, 10, 100 and 200 nodes in three hidden layers, and with the input layer directly drawn from the data. Experiments were implemented in Theano (Bergstra et al. 2010). All nodes are rectifiers. We set half of nodes as positive and half as negative. Output nodes perform rectilinear regression, see below, whereas hidden nodes minimize the rectilinear loss on feedback implementing either Kickback or Backprop.

Training was performed in batch sizes of 20. Lower batchsizes yield better performance at the cost of longer training times. We chose 20 as a reasonable compromise.

Rectilinear regression. Recently, (Glorot, Bordes, and Bengio 2011) introduced an ℓ_1 penalty on firing rates, which encourages sparsity and can improve performance. Here, we consider an ℓ_2-penalty: $\boldsymbol{\ell}_{RL}(\mathbf{w}, \mathbf{x}, \varphi) - \frac{1}{2}S_{\mathbf{w}}(\mathbf{x})^2$. Weight updates under gradient descent are

$$\Delta\mathbf{w} \propto \begin{cases} (\varphi - \langle\mathbf{w},\mathbf{x}\rangle)\mathbf{x} & \text{if } \langle\mathbf{w},\mathbf{x}\rangle > 0 \\ 0 & \text{else.} \end{cases} \tag{8}$$

Notice that the penalty $\langle\mathbf{w},\mathbf{x}\rangle$ in (8) is the firing rate. Comparing with the gradient $(\varphi - \langle\mathbf{w},\mathbf{x}\rangle)\mathbf{x}$ of the mean-squared error $\frac{1}{2}(\varphi - \langle\mathbf{w},\mathbf{x}\rangle)^2$ shows that the ℓ_2-activation penalty leads nodes to *perform linear regression on the inputs that cause them to fire* (Balduzzi 2013). A regret bound analogous to Theorem 2 holds for rectilinear regression, with a faster convergence rate of $O(\frac{\log|F|}{|F|})$.

Training error is the MSE of the output node with the correct sign[4]; test error is the sum of the output nodes' MSEs.

Initialization and coherence. No pretraining was used. We consider two network initializations. The first is **uniform**: draw weights uniformly at random from an interval symmetric about 0, without regard to coherence. The second initialization, **signed** is taken from Example 1: draw weights uniformly, then change their signs so that connections targeting positive nodes have positive weights and conversely for negative nodes. **Signed** guarantees coherence at initialization. Although it is possible to impose coherence during training, we found that doing so was unnecessary in practice.

Results are plotted under both initializations for Kickback – excepting Panel (e), where **uniform** failed to converge. For Backprop, the initialization that yielded the *better* performance is reported.

Results. We report the normalized mean-squared errors for six datasets. To directly compare the behavior of the two algorithms, we report individual runs rather than averages. Performance was robust to small changes in parameters.

Each SARCOS dataset consists of 44,484 training and 4,449 test points; Barrett split as 12,000 and 3,000. Parameters were tuned via grid-search with 5-fold cross-validation. Backprop's only parameter is the learning rate. Kickback was implemented with a learning rate tuned for Backprop. Kickback has two additional parameters that rescale the feedback to hidden layers 1 & 2. We observed that tuning via cross-validation typically set the rescaling factors such that the truncated errors are rescaled to about same magnitude, on average, as Backprop's feedback.

Kickback and Backprop are competitive with nonparametric methods such as kernel regression, e.g. (Kpotufe and Boularias 2013). Kickback performs best with **signed** initialization, as expected from Theorem 4. With **signed** initialization, Kickback almost exactly matches Backprop in all 6 datasets. Importantly, Kickback continues to reduce the MSE after 100s of epochs; following the correct gradient even when the error is small.

The comparison between Backprop and Kickback is not completely fair: Kickback's additional parameters cause it to outperform Backprop in panel (b). We have endeavored to keep the comparison as level as possible.

The effect of coherence. Kickback's performance was better than expected: coherence was not imposed after initialization under **signed**; and no guarantees are applicable to **uniform**. A possible explanation is that Kickback preserves or increases coherence.

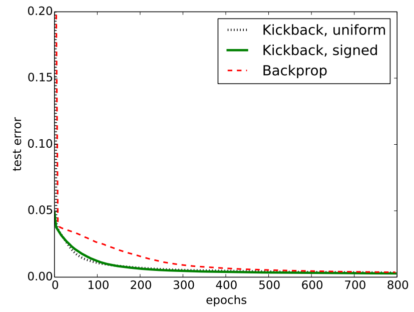
To test this hypothesis, we quantified the coherence of layer α as $\text{coh}(L_\alpha) = \frac{\sum_{j\in L_\alpha}\tau_j}{\sum_{j\in L_\alpha}|\tau_j|}$, which lies in $[-1, 1]$.

[1]Taken from `www.gaussianprocess.org/gpml/data/`.

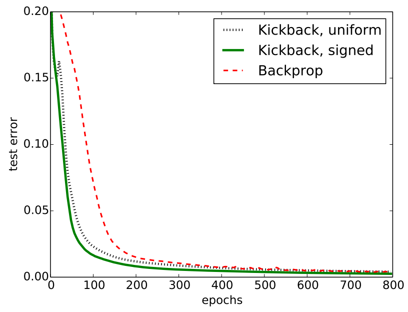
[2]Taken from `http://www.ias.tu-darmstadt.de/Miscellaneous/Miscellaneous`.

[3]In short: the performance of both Kickback and Backprop is worse, but still comparable, with fewer layers.

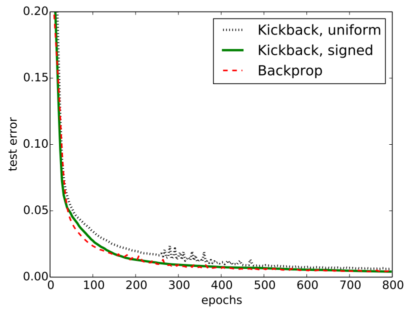
[4]Recall there is one positive and one negative output rectifier.



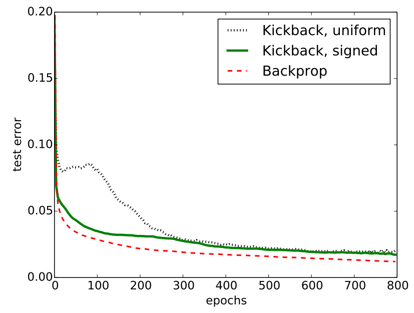
(a) Barrett 2



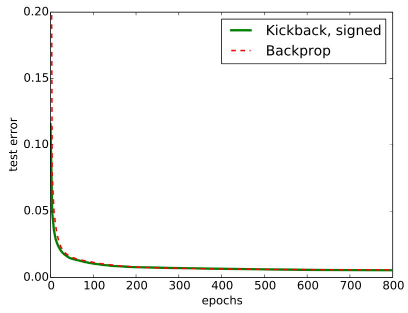
(b) Barrett 3



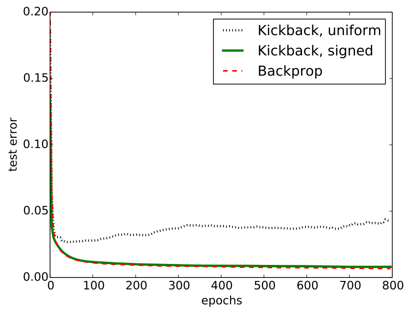
(c) Barrett 4



(d) SARCOS 1



(e) SARCOS 3



(f) SARCOS 7

Figure 2: **Mean-squared test error per epoch for Kickback and Backprop.**

With **signed** initialization, coherence consistently remained above 0.9 under Kickback; but exhibited considerable variability under Backprop. With **uniform** initialization, Kickback increased the coherence of hidden layers 2 & 3, from 0 to > 0.5, with the exception of panel (c). Backprop did not alter coherence in any consistent way.

Barrett 4 is the only dataset where nodes become incoherent $(\text{coh} < 0)$ on average. The oscillations in Panel (c) for **uniform** arise because Kickback is not guaranteed to follow the training error gradient in the absence of coherence. It is surprising the network learns at all. Note that oscillations do not occur when networks are given a **signed** initialization.

Conclusion

A necessary step towards understanding how the brain assigns credit is to develop a minimal working model that fits basic constraints.

Backprop solves the credit assignment problem. It is one of the simplest and most effective methods for learning representations. In combination with various tricks and optimizations, it continues to yield state-of-the-art performance. However, it flouts a basic constraint imposed by neurobiology: it requires that nodes produce error signals that are distinct from their outputs.

Kickback is a stripped-down version of Backprop motivated by theoretical (Theorems 1–4) and biological (Fig. 1 and Theorem 5) considerations. Under Kickback, nodes perform gradient descent, or ascent, on the representation – that is, the kicked back activity – produced by the next layer. The sign of the global error determines whether nodes follow the gradient downwards, or upwards.

Kickback is the first competitive algorithm with biologically plausible credit assignment. Previous proposals were not competitive and did not scale to more than one hidden-layer (Kickback works well for 1, 2 and 3 hidden-layers; we have yet to test 4 or more). Kickback's reduced communication complexity makes it well-suited to hardware implementations (Indiveri et al. 2011; Nere et al. 2012).

An important outcome of the paper is a new formulation of Backprop in terms of interacting local learners, that may connect deep learning to recent developments in multi-agent systems (Seuken and Zilberstein 2008; Sutton et al. 2011) and mechanism design (Balduzzi 2014).

Kickback's rescaling factors (1 per hidden layer) are a loose-end that require addressing in future work.

Perhaps the most important direction is to extend Kickback to multiclass learning. For this, it is necessary to consider multidimensional outputs, in which case the derivative of the energy function with respect to the output layer is not a scalar. A natural approach to tackle this setting is to use more sophisticated global error signals. Indeed, modeling the neuromodulatory system as producing scalar outputs is a vast oversimplification (Dayan 2012).

Finally, reinforcement learning is a better model of how an agent adapts to its environment than supervised learning (Veness et al. 2010). A natural avenue to explore is how Kickback, suitably modified, performs in this setting.

Acknowledgements. We thank Jacob Abernethy and Satinder Singh for useful conversations. This research was supported in part by SNSF grant 200021_137971.

References

Balduzzi, D., and Besserve, M. 2012. Towards a learning-theoretic analysis of spike-timing dependent plasticity. In *Advances in Neural Information Processing Systems (NIPS)*.

Balduzzi, D. 2013. Randomized co-training: from cortical neurons to machine learning and back again. *Randomized Methods for Machine Learning Workshop, Neural Inf Proc Systems (NIPS)*.

Balduzzi, D. 2014. Cortical prediction markets. In *Proc. 13th Int Conf on Autonomous Agents and Multiagent Systems (AAMAS)*.

Bergstra, J.; Breuleux, O.; Bastien, F.; Lamblin, P.; Pascanu, R.; Desjardins, G.; Turian, J.; Warde-Farley, D.; and Bengio, Y. 2010. Theano: A CPU and GPU Math Expression Compiler. In *Proc. Python for Scientific Comp. Conf. (SciPy)*.

Cesa-Bianchi, N., and Lugosi, G. 2006. *Prediction, Learning and Games*. Cambridge University Press.

Crick, F. 1989. The recent excitement about neural networks. *Nature* 337(12):129–132.

Dahl, G. E.; Sainath, T. N.; and Hinton, G. 2013. Improving deep neural networks for LVCSR using rectified linear units and dropout. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Dayan, P. 2012. Twenty-Five Lessons from Computational Neuromodulation. *Neuron* 76:240–256.

Gerstner, W., and Kistler, W. 2002. *Spiking Neuron Models*. Cambridge University Press.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Deep Sparse Rectifier Neural Networks. In *Proc. 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Hazan, E. 2012. The convex optimization approach to regret minimization. In Sra, S.; Nowozin, S.; and Wright, S. J., eds., *Optimization for machine learning*. MIT Press.

Hu, T.; Towfic, Z. J.; Pehlevan, C.; Genkin, A.; and Chklovskii, D. B. 2013. A Neuron as a Signal Processing Device. In *Asilomar Conference on Signals, Systems and Computers*.

Indiveri, G.; Linares-Barranco, B.; Hamilton, T. J.; van Schaik, A.; Etienne-Cummings, R.; Delbruck, T.; Liu, S.-C.; Dudek, P.; Häfliger, P.; and *et al*. 2011. Neuromorphic silicon neuron circuits. *Front. Neurosci* 5(73).

Jarrett, K.; Kavukcuoglu, K.; Ranzato, M.; and LeCun, Y. 2009. What is the Best Multi-Stage Architecture for Object Recognition? In *Proc. International Conference on Computer Vision (ICCV)*.

Kpotufe, S., and Boularias, A. 2013. Gradient Weights help Nonparametric Regressors. In *Advances in Neural Information Processing Systems (NIPS)*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*.

Lamme, V., and Roelfsema, P. 2000. The distinct modes of vision offered by feedforward and recurrent processing. *Trends in Neurosci.* 23(11):571–579.

Leshno, M.; Lin, V. Y.; Pinkus, A.; and Schocken, S. 1993. Multilayer Feedforward Networks With a Nonpolynomial Activation Function Can Approximate Any Function. *Neural Networks* 6:861–867.

Maas, A. L.; Hannun, A. Y.; and Ng, A. 2013. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*.

Nair, V., and Hinton, G. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*.

Nere, A.; Olcese, U.; Balduzzi, D.; and Tononi, G. 2012. A neuromorphic architecture for object recognition and motion anticipation using burst-STDP. *PLoS One* 7(5):e36958.

Roelfsema, P. R., and van Ooyen, A. 2005. Attention-gated reinforcement learning of internal representations for classification. *Neural Comput* 17(10):2176–2214.

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

Schultz, W.; Dayan, P.; and Montague, P. 1997. A neural substrate of prediction and reward. *Science* 275(1593-1599).

Selfridge, O. G. 1958. Pandemonium: a paradigm for learning. In *Mechanisation of Thought Processes: Proceedings of a Symposium Held at the National Physics Laboratory*.

Seuken, S., and Zilberstein, S. 2008. Formal models and algorithms for decentralized decision making under uncertainty. *Auton Agent Multi-Agent Syst* 17(2):190–250.

Seung, H. S. 2003. Learning in Spiking Neural Networks by Reinforcement of Stochastic Synaptic Transmission. *Neuron* 40(1063-1073).

Song, S.; Miller, K. D.; and Abbott, L. F. 2000. Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nat Neurosci* 3(9).

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *JMLR* 15:1929–1958.

Sutton, R.; Modayil, J.; Delp, M.; Degris, T.; Pilarski, P. M.; White, A.; and Precup, D. 2011. Horde: A Scalable Real-time Architecture for Learning Knowledge from Unsupervised Motor Interaction. In *Proc. 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*.

Tononi, G., and Cirelli, C. 2014. Sleep and the Price of Plasticity: From Synaptic and Cellular Homeostasis to Memory Consolidation and Integration. *Neuron* 81(1):12–34.

Vargas-Caballero, M., and Robinson, H. P. 2003. A slow fraction of Mg2+ unblock of NMDA receptors limits their contribution to spike generation in cortical pyramidal neurons. *J Neurophysiol* 89(5):2778–83.

Veness, J.; Ng, K. S.; Hutter, M.; and Silver, D. 2010. Reinforcement Learning via AIXI Approximation. In *Proc. 24th AAAI Conference on Artificial Intelligence (AAAI)*.

Werbos, P. J. 1974. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Ph.D. Dissertation, Harvard.

Williams, R. J. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8:229–256.

Zeiler, M. D.; Ranzato, M.; Monga, R.; Mao, M.; Yang, K.; Le, Q. V.; Nguyen, P.; Senior, A.; Vanhoucke, V.; Dean, J.; and Hinton, G. 2013. On Rectified Linear Units for Speech Processing. In *IEEE Int Conf on Acoustics, Speech and Signal Proc (ICASSP)*.