Single-phase deep learning in cortico-cortical networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

The error-backpropagation (backprop) algorithm remains the most common solution to the credit assignment problem in artificial neural networks. In neuroscience, it is unclear whether the brain could adopt a similar strategy to correctly modify its synapses. Recent models have attempted to bridge this gap while being consistent with a range of experimental observations. However, these models are either unable to effectively backpropagate error signals across multiple layers or require a multi-phase learning process, neither of which are reminiscent of learning in the brain. Here, we introduce a new model, bursting cortico-cortical networks (BurstCCN), which solves these issues by integrating known properties of cortical networks namely bursting activity, short-term plasticity (STP) and dendrite-targeting interneurons. BurstCCN relies on burst multiplexing via connection-type-specific STP to propagate backprop-like error signals within deep cortical networks. These error signals are encoded at distal dendrites and induce burst-dependent plasticity as a result of excitatory-inhibitory topdown inputs. First, we demonstrate that our model can effectively backpropagate errors through multiple layers using a single-phase learning process. Next, we show both empirically and analytically that learning in our model approximates backprop-derived gradients. Finally, we demonstrate that our model is capable of learning complex image classification tasks (MNIST and CIFAR-10). Overall, our results suggest that cortical features across sub-cellular, cellular, microcircuit and systems levels jointly underlie single-phase efficient deep learning in the brain.

1 Introduction

For effective learning, synaptic modifications throughout the brain should result in improved behavioural function. This requires a process by which credit should be assigned to synapses given their contribution to behavioural output [1–3]. In multilayer networks, credit assignment is particularly challenging as the impact of changing a synaptic connection depends on its downstream brain areas. Classical local Hebbian plasticity rules, even when coupled with global neuromodulatory factors, are unable to communicate enough information for effective credit assignment through multiple layers of processing [3]. In machine learning, the error-backpropagation (backprop) algorithm is the most successful solution to the credit assignment problem. However, it relies on a number of biologically implausible assumptions to compute gradient information used for synaptic updates. Previous work has attempted to address these implausibilities but important issues remain open in terms of mapping backprop to the neuronal physiology. Earlier attempts relied on mapping backprop to the brain by using single-compartment neuron models [4, 5]. This poses a problem as single-compartment neurons are unable to simultaneously store the necessary inference and credit assignment signals. One solution is to model neurons with an apical dendritic compartment that separately stores credit information [5, 6], supported by the electrotonic separation of the soma and apical dendrites [7]. These distal credit signals can then be communicated to the soma through non-linear dendritic events that trigger bursting at the soma [8], thereby inducing long-term synaptic plasticity [9]. In particular,

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

two recent approaches, Error-encoding Dendritic Networks (EDNs) [6] and Burstprop [10], have demonstrated how such multi-compartment neuron models can be used to construct networks capable of backprop-like credit assignment. EDNs encode credit signals at apical dendrites resulting from the mismatch between dendritic-targeting interneuron activity and downstream activity. Burstprop uses the mechanism of bursting, controlled by dendritic excitability, to communicate credit signals. However, these models still have major issues, such as the inability to effectively backpropagate error signals through many layers (EDNs) and the requirement for a multi-phase learning process (Burstprop).

Here, we propose a new model called the Bursting Cortico-Cortical Network (BurstCCN) as a solution to the credit assignment problem which addresses several outstanding issues of current biologically plausible backprop research. Our model builds upon prior multi-compartment neuron models [6, 10]: it encodes credit signals in distal dendritic compartments which trigger bursting activity at the somatic level to drive backprop-like synaptic updates. We demonstrate that combining well-established properties of cortical neurons such as bursting activity, short-term plasticity (STP) and dendrite-targeting interneurons provides a biologically plausible mechanism for performing credit assignment. In contrast to previous models, BurstCCN is highly effective at backpropagating credit signals in multi-layer architectures while only requiring a single-phase learning process. We implement multiple versions of the BurstCCN at different levels of abstraction in order to demonstrate some of its key properties and empirically confirm our theoretically motivated claims.

First, we used a spike-based implementation of the BurstCCN to demonstrate its ability to learn without the need for multiple phases. We further show the importance of this single-phase learning by training a continuous-time rated-based version of the BurstCCN on an continuous-time non-linear regression task. Next, we show empirically and analytically that our model's dynamics result in learning that approximately follows backprop-derived gradients. Finally, we use a simplified discrete-time BurstCCN implementation to demonstrate that the model achieves good performance in non-trivial image classification tasks (MNIST and CIFAR-10), even in the presence of random feedback synaptic weights.

2 Bursting Cortico-Cortical Networks

2.1 Burst Ensemble Multiplexing

Burst Ensemble Multiplexing (BEM) [11] refers to the idea that ensembles of cortical neurons are capable of simultaneously representing multiple distinct signals within the patterns of their spiking activity. Typically, pyramidal cells receive top-down and bottom-up signals into their apical and basal dendrites, respectively. Bottom-up basal inputs affect the rate of spiking and top-down apical inputs convert these somatically induced spikes into high-frequency bursts. Postsynaptic populations can then use STP to decode these distinct signals from the overall spiking activity.

The BurstCCN uses the concept of BEM in a similar way to Burstprop [10] in which ensembles of cells encode both feedforward inference signals and feedback error signals. The model encodes these signals as the rates of *events* and *bursts*, respectively, across the ensembles. Here, the specific definition of a burst is a collection of spikes with interspike intervals less than 16ms and an event is either a burst or a single isolated spike (i.e. a spike not followed or preceded by another within 16ms). The burst probability of an ensemble is defined as the probability that an event at a given time is a burst and is computed as a ratio of the event rate ($\mathbf{e}$) and burst rate ($\mathbf{b}$): $\mathbf{p} = \mathbf{b}/\mathbf{e}$.

2.2 Rate-based BurstCCN

In our discrete-time implementation of the rate-based BurstCCN, example input-output pairs are processed independently in discrete timesteps. For each example, the event rates of the input layer, $\mathbf{e}_0$, encode the input stimulus. The model then computes each subsequent layer's activities, equivalent to that of a standard feedforward artificial neural network (Fig. 1A). Specifically, somatic potentials are computed by integrating basal input as $\mathbf{v}_l = \mathbf{W}_l \mathbf{e}_{l-1}$ where $\mathbf{W}_l$ are short-term depressing (STD) feedforward weights from layer $l-1$ to layer l. The STD nature of these weights ensures that only event rate information propagates forwards. Each layer's event rates are then computed by applying a non-linear activation function, f, to the somatic potentials, $\mathbf{e}_l = f(\mathbf{v}_l)$. These linear and nonlinear

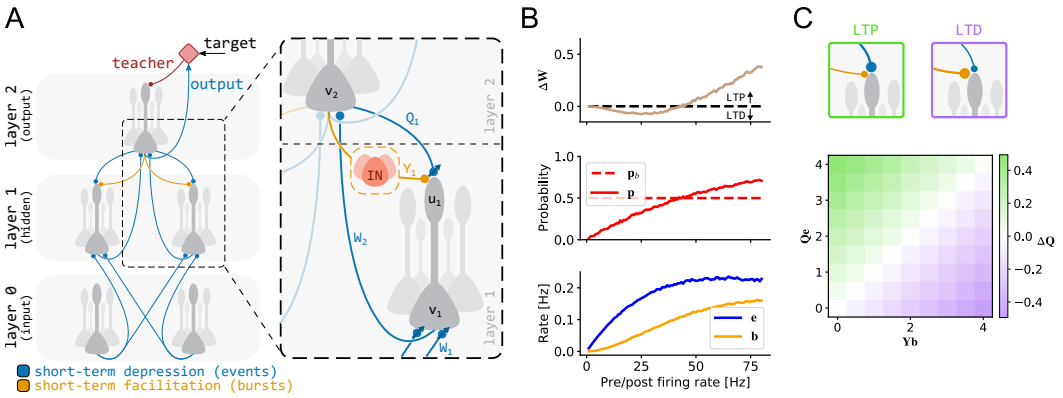

Figure 1: **Bursting cortico-cortical networks (BurstCCN) for credit assignment through bursting activity.** (**A**) Schematic of network with neuron ensembles and STP-specific connection types. Events from the input are propagated forward through short-term depressing (STD) connections, $\mathbf{W}$. Output event rates are compared to a target value which generates a teaching signal presented to the output layer apical dendrites. This acts as an error signal and appears as a deflection in the dendritic potential from its resting potential which causes changes to bursting activity from its baseline. The error-carrying bursting signals are propagated back through short-term facilitating connections, $\mathbf{Y}$, which we interpret as being communicated by populations of dendrite-targeting interneurons. Events are also propagated backwards via STD connections, $\mathbf{Q}$, to provide a means of cancelling baseline bursting activity. The difference in activity from these two feedback connections results in changes to dendritic excitability that lead to burst-dependent synaptic plasticity. (**B**) Burst-dependent plasticity rule. Simple setup of a single connection between a pre- and post-synaptic cell that are both modelled with Poisson spike trains with equal rates. As the firing rates increase, (top) plasticity of the synaptic weight switches from long-term depression (LTD) to long-term potentiation (LTP) (middle) when the burst probability increases above the baseline value. (Bottom) The magnitude of the weight change is scaled by the event rate. (**C**) Homeostatic plasticity rule for $\mathbf{Q}$ weights. Difference between the signal through $\mathbf{Q}$ and $\mathbf{Y}$ dictates the direction and magnitude of synaptic plasticity.

operations are repeated for each layer in the network to ultimately obtain the output layer event rates, $\mathbf{e}_L$, where L denotes the total number of layers.

The desired target output of the network, $\mathbf{e}_{target}$, is compared to the output layer event rates to produce a signed error, $\mathbf{e}_{target} - \mathbf{e}_L$, which is used as a teaching signal. This error information is then propagated backwards through each layer in the network by altering the apical dendritic compartment potential and, as a result, the burst probability of each pyramidal ensemble. At the output layer, the burst probability is computed directly as $\mathbf{p}_L = \mathbf{p}_L^b + \mathbf{p}_L^b \odot (\mathbf{e}_{target} - \mathbf{e}_L) \odot h(\mathbf{e}_L)$ where $\odot$ denotes the element-wise product, $\mathbf{p}_L^b$ represents the baseline burst probability in the absence of any teaching signal and $h(\mathbf{e}_l) = f'(\mathbf{v}_l) \odot \mathbf{e}_l^{-1}$. These burst probabilities are used at the output layer ($l = L$) to compute the burst rates as $\mathbf{b}_l = \mathbf{e}_l \odot \mathbf{p}_l$ which are decoded and sent backwards to layer $l-1$ apical dendrites by a set of short-term facilitating (STF) feedback weights, $\mathbf{Y}_{l-1}$. The event rates are also send backwards through a separate set of STD feedback weights, $\mathbf{Q}_{l-1}$, leading to a apical dendritic potential in the previous layer of $\mathbf{u}_{l-1} = \mathbf{Q}_{l-1}\mathbf{e}_l - \mathbf{Y}_{l-1}\mathbf{b}_l$. This determines the layer's burst probabilities which are computed as $\mathbf{p}_{l-1} = \bar{\sigma}(\mathbf{u}_{l-1} \odot h(\mathbf{e}_{l-1}))$ where $\bar{\sigma}$ denotes the sigmoid function, σ, with scaling and offset parameters, $\bar{\sigma}(x) = \sigma(\alpha x + \beta)$ ([10]; see SM, Section B). The same process is repeated backwards for each layer until the input layer to obtain their dendritic potentials and burst probabilities. We interpret the STF feedback connections as being provided via a type of dendrite-targeting interneuron and STD feedback as direct connections in line with recent experimental studies [12–17].

After the error information has been propagated backwards, feedforward synaptic weight changes are computed using a burst-dependent synaptic plasticity rule:

$$\Delta \mathbf{W}_l = \eta_l^{(\mathbf{W})} \left((\mathbf{p}_l - \mathbf{p}_l^b) \odot \mathbf{e}_l \right) \mathbf{e}_{l-1}^T \tag{1}$$

where $\eta^{(\mathbf{W})}$ is a learning rate and ^{T} is the transpose operation. Importantly, the learning rule depends on the change in burst probability from the predefined layer-wise baseline burst probability,

$\mathbf{p}_l^b = p_l^b(1, \ldots, 1)^T$, which represents the signed error signal required for backprop-like learning. Consequently, when we make both pre- and postsynaptic cells fire following Poisson statistics we obtain long-term depression and long-term potentiation for low and high firing rates, respectively (Fig. 1B). This is in line with a large number of experimental studies at cortical synapses [18, 19]. It can be shown that the updates produced by this learning rule approximate those obtained by the backpropagation algorithm in the weak-feedback case (see Section 3.3.1 and SM, Section B).

In the absence of a teaching signal, it is important for pyramidal ensembles to produce a baseline level of bursting such that no weight changes occur (cf. Eq. 1). This holds true for the output layer as there are no other inputs onto the apical dendrites. However, for the hidden layers the event rate signals through $\mathbf{Q}$ and the burst rate signals through $\mathbf{Y}$ need to exactly cancel each other out such that the apical dendritic potentials are at rest (i.e. $\mathbf{u} = 0$). For any $\mathbf{Y}$ weights, there is always an optimal set of $\mathbf{Q}$ weights that will produce this exact cancellation regardless of the event rates propagating through the network. Specifically, they must be set as $\mathbf{Q}_l = p_l^b \mathbf{Y}_l$ which we refer to as the weights being in a *Q-Y symmetric* state. As it is not biologically plausible for the $\mathbf{Q}$ synapses to have direct knowledge of $\mathbf{Y}$, inspired by earlier work [6, 20] we use a learning rule for $\mathbf{Q}$ that provides this cancellation:

$$\Delta \mathbf{Q}_l = -\eta_l^{(\mathbf{Q})} \, \mathbf{u}_l \, \mathbf{e}_{l+1}^T \tag{2}$$

which explicitly aims to silence the apical potentials (Fig. 1C). In the absence of a teaching signal at the output layer, all $\mathbf{Q}$ weights will eventually converge to their optimal values and achieve a symmetric state (i.e. $\mathbf{u} = 0$) under reasonable assumptions (see SM, Section B.2). Note that we could similarly have added this learning rule on the $\mathbf{Y}$ feedback weights to cancel the $\mathbf{Q}$ weights, which produces similar results (Fig. S1).

When teaching signals are applied at the output layer, it is important to note that only the bursting activity propagated through the $\mathbf{Y}$ connections changes because the event rates through $\mathbf{Q}$ are unaffected by the dendritic activity. This enables single-phase learning as the symmetry in the two feedback connection types ($\mathbf{Q}$ and $\mathbf{Y}$) can be exploited to directly compare *without teacher* signals (i.e. at baseline) to *with teacher* signals.

Details of the continuous time implementation can be found in the Supplementary Materials.

2.3 Spiking BurstCCN

For a spiking implementation of BurstCCN, we adapted our burst-dependent synaptic plasticity rule in Equation 1 (see SM, Eq. 12). Unlike the two rate-based implementations, the spiking BurstCCN more accurately models the internal neuron spiking dynamics instead of abstracting these details away and only considering the ensemble-level behaviour. Neurons are modelled with two compartments corresponding to the soma and apical dendrites and spikes are generated when a somatic threshold potential is met (see Section A.2 for more details).

3 Results

3.1 BurstCCN can learn with a single learning phase

A key motivation for developing the BurstCCN was to design a model capable of learning without the need for separate learning phases, while being consistent with a range of cortical features across multiple levels. To demonstrate that our model can perform single-phase learning, we trained the spiking version of our model on the XOR classification task and contrasted it with Burstprop, which requires a two-phase learning process (Fig. 2). In both single- and two-phase learning regimes, the input stimulus is presented for a total of 8s before the next example is shown. The two-phase learning regime has an initial prediction phase, lasting 7.2s for each input presentation, where plasticity is switched off throughout the network and the output neurons do not receive any teaching signals (Fig. 2A). This is followed by a teacher phase for the remaining 0.8s where plasticity is restored and teaching signals are delivered at the output. The single-phase regime removes the initial prediction phase and extends the teacher phase to the full duration of the input stimulus (Fig. 2B).

Our results show that both models were capable of successfully learning the task in the two-phase regime as indicated by the high output event rates in response to the $(0, 1)$ and $(1, 0)$ inputs and low event rates for the $(0, 0)$ and $(1, 1)$ inputs (Fig. 2C). However, when training in the single-phase

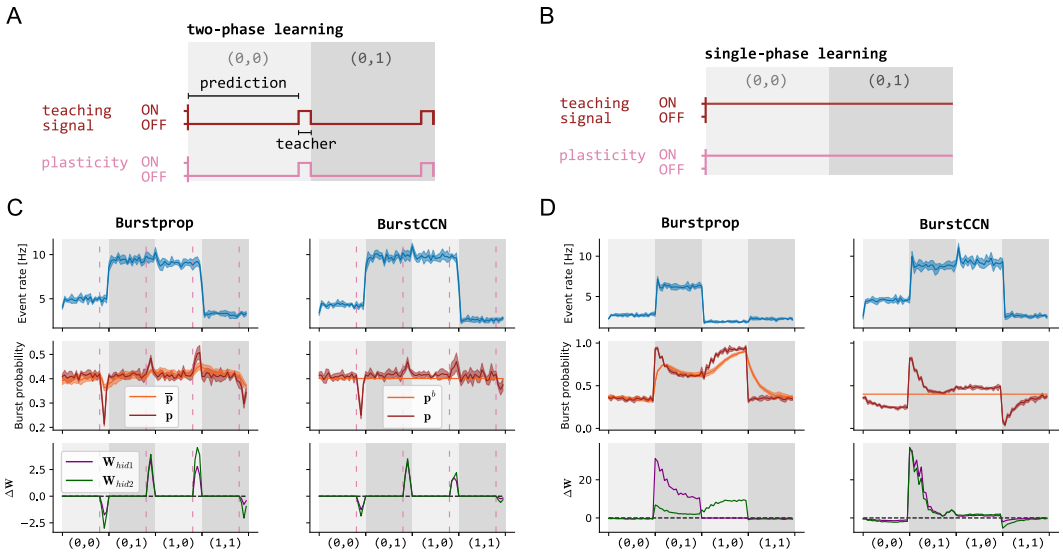

Figure 2: **Spiking BurstCCN does not require multi-phase learning to solve the XOR classification task.** Schematic of the (A) two-phase and (B) single-phase learning settings. (**A**) For each input during two-phase learning, networks are given a 7.2s prediction period during which teaching signals and plasticity are turned OFF, followed by a 0.8s learning period where both teaching signals and plasticity are turned back ON. (**B**) During single-phase learning, both the teaching signals and plasticity remained ON throughout training. (**C, D**) Top: event rate ($\mathbf{e}$) of the output layer; Middle: burst probability ($\mathbf{p}$) for the output layer and the baseline or moving average of the burst probability ($\mathbf{p}^b$ or $\bar{\mathbf{p}}$) for BurstCCN and Burstprop, respectively; Bottom: the resulting weight updates for connections from hidden layer neurons. Model results represent mean $\pm$ standard error (n = 5).

regime, only BurstCCN was able to learn the task (Fig. 2D). The inability of Burstprop to learn the task can be explained by comparing the moving average of the burst probability ($\bar{\mathbf{p}}$) with the actual burst probability ($\mathbf{p}$) which determines the sign of synaptic weight updates (Fig. 2D). Burstprop failed to learn in the single-phase learning setup due to the need to have a stable representation of $\bar{\mathbf{p}}$ without a teaching signal, which becomes problematic in this case.

3.2 BurstCCN can learn with dynamic input-output

Typically, studies that have attempted to solve the credit assignment problem with biologically plausible implementations of backprop make an implicit assumption that during learning there is a period where the continuous-time input stream is fixed [6, 10]. This is required in most cases to allow the network to stabilise its activities before learning can take place. With single-phase learning, we can relax this assumption which enables learning in conditions where the inputs and their corresponding teaching signals are dynamically changing over time. We assessed this ability by training the continuous-time BurstCCN (see SM, Section A.1) on an online non-linear regression task (Fig. 3). This task consisted of three sinusoidal inputs, $x_i(t) = sin(\alpha_i t + \beta_i)$, with random frequencies $\alpha_i \sim U(0, \frac{\pi}{2})$ and phase offsets $\beta_i \sim U(0, 2\pi)$ (Fig. 3A). The network had a single output unit for which a non-trivial target was obtained by passing the same inputs to a 3-25-1 artificial neural network (ANN). This approximates a setting in which a given cortical area learns to regress its input onto the activity of another cortical area. The ANN weights were randomly initialised with $w^1_{ij} \sim \mathcal{U}(-\sqrt{3}, \sqrt{3})$ for the first layer and $w^2_{ij} \sim \mathcal{U}(-0.6, 0.6)$ for the second layer. Despite the BurstCCN initially producing outputs that were significantly different to the target (Fig. 3C), the results showed that over training it learned to produce output patterns that closely matched the non-linear and dynamic target (Fig. 3B,D). This highlights that the BurstCCN is capable of adequately backpropagating useful error signals when both inputs and teaching signals are constantly changing.

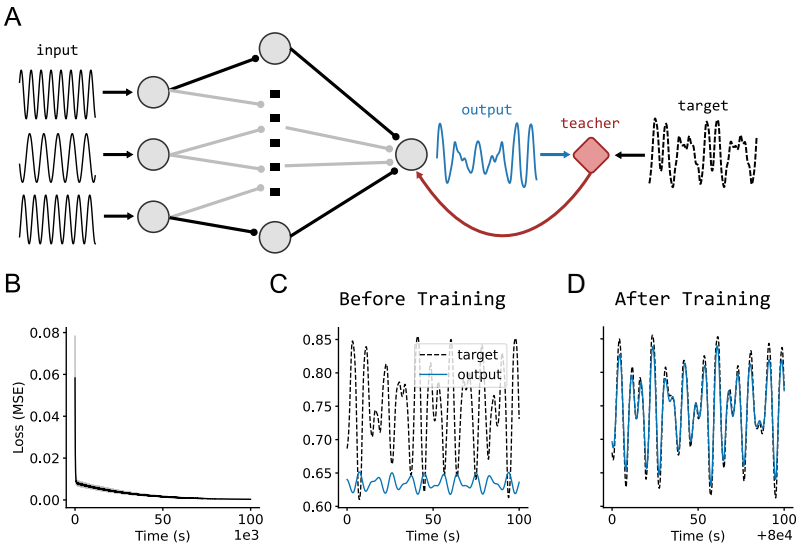

Figure 3: **BurstCCN can learn a dynamic non-linear regression task.** (**A**) Schematic of the task. Three sinusoidal waves with random frequencies are given as inputs. The network needs to learn to match the target pattern which is obtained by passing the same inputs through a fixed, randomly initialised ANN. (**B**) Learning curve for the (continuous-time) BurstCCN. (**C, D**) Example output traces for (C) before and (D) after training. Model results represent mean $\pm$ standard error (n = 5).

3.3 Feedback plasticity rule facilitates alignment to backprop updates

Next, we wanted to understand how well our model approximates backprop. As stated above, the purpose of the learning rule for the feedback $\mathbf{Q}$ STD connections (Eq. 2) is to silence the apical compartment of every ensemble by cancelling activity through the feedback $\mathbf{Y}$ STF connections. When a teaching signal is applied, this becomes important for computing the correct local error signal that is used for learning and backpropagated to previous layers. Here we show analytically and empirically using the discrete version of the model how the errors computed by our model relate to backprop.

3.3.1 BurstCCN with weak feedback approximates backpropagation algorithm

Under some small assumptions, we analytically show that the feedback pathway of BurstCCN is approximately communicating the same error gradients that are computed by backprop. Specifically, we assume that the feedback weights are optimally aligned (i.e. $\mathbf{Q}_l = p_l^b \mathbf{Y}_l$) and focus on the change in burst rate, $\delta \mathbf{b}_l := (\mathbf{p}_l - \mathbf{p}_l^b) \odot \mathbf{e}_l$. If we let $E^{\text{task}} = ||\mathbf{e}_L - \mathbf{e}_{target}||^2$ define the task error then, by construction, the change in burst rate at the output layer is equivalent to the negative error gradient, $\delta \mathbf{b}_L = -\frac{\partial E^{\text{task}}}{\partial \mathbf{v}_L}$. For the hidden layers, we derive the following iterative relationship (see SM, Section B):

$$\delta \mathbf{b}_l = f'(\mathbf{v}_l) \odot (-\mathbf{Y}_l) \delta \mathbf{b}_{l+1} + \mathcal{O}(\mathbf{u}_l^3). \tag{3}$$

This approximates the same relationship present in backprop up to a third-order[1] term with respect to the apical potentials $\mathbf{u}_l$ if the feedback weights are set to be symmetric with the feedforward weights (i.e. $\mathbf{Y}_l = -\mathbf{W}_{l+1}^T$). We refer to this as the *W-Y symmetric* state. The link between weight updates from simply performing gradient descent with backprop and the BurstCCN can be seen clearly:

$$\Delta \mathbf{W}_l^{\text{BurstCNN}} = \eta_l^{(\mathbf{W})} \, \delta \mathbf{b}_l \, \mathbf{e}_{l-1}^T \tag{4}$$

$$\Delta \mathbf{W}_l^{\text{backprop}} = -\eta_l^{(\mathbf{W})} \frac{\partial E^{\text{task}}}{\partial \mathbf{v}_l} \, \mathbf{e}_{l-1}^T \tag{5}$$

It remains to be shown that the apical potentials, $\mathbf{u}_l$, of every layer are indeed appropriately small (so that the approximation error, $||\mathbf{u}_l^3||$, is small). Under the assumption $\mathbf{u}_{l+1}$ is small, we can derive

[1]Here we use abuse of notation $\mathbf{u}_l^3 = \left(u_{l,1}^3, u_{l,2}^3, \dots\right)^T$ to represent the element-wise cubic of $\mathbf{u}_l$

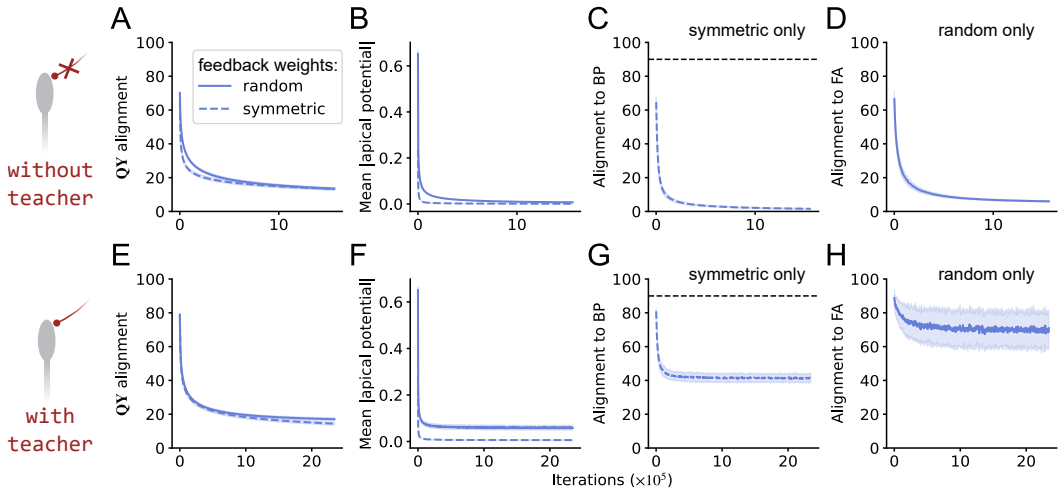

Figure 4: **Feedback learning rule enables a close alignment with backprop and feedback alignment.** The network is a randomly initialised 5-layer discrete-time BurstCCN with random (solid line) or symmetric (dashed line), fixed $\mathbf{W}$ and $\mathbf{Y}$ weights. The $\mathbf{Q}$ weights are updated in the presence of (**A-D**) no teaching signal or (**E-H**) a teaching signal. (A,E) Alignment between $\mathbf{Q}$ and $\mathbf{Y}$ connections, (B,F) the mean absolute value of the apical potentials, (C,G) the alignment to backprop (BP) and (D,H) feedback alignment (FA) as $\mathbf{Q}$ weights learn to silence apical dendrite potential. Updates below 90° marked by the black dashed line are considered useful as they still follow the direction to backprop on average. Model results represent mean ± standard error (n = 5).

the recursive relationship $\mathbf{u}_l \approx f'(\mathbf{v}_{l+1}) \odot (-\mathbf{Y}_l)\mathbf{u}_{l+1}$ (see SM, Section B). We show that if f' is bounded (as is the case for sigmoid and many activation functions) and the weights $\mathbf{Y}_l$ are reasonably small then $||\mathbf{u}_l|| \leq ||\mathbf{u}_{l+1}||$. This means that if the error gradient at the output layer, $\frac{\partial E^{\text{task}}}{\partial \mathbf{e}_N}$, is appropriately small then, by induction, $\mathbf{u}_l$ is small for every layer and $\Delta\mathbf{W}_l^{\text{BurstCNN}} \approx \Delta\mathbf{W}_l^{\text{backprop}}$.

3.3.2 Learning Q feedback connections better approximates backprop-derived gradients

We empirically evaluated our feedback plasticity rule by updating *only* the $\mathbf{Q}$ weights of a randomly initialised 5-layer discrete-time BurstCCN with all other weight types ($\mathbf{W}$ and $\mathbf{Y}$) fixed. We used multiple initialisations and training regimes to understand how the plasticity rule behaves in different scenarios. The network was either initialised in the $\mathbf{W}$-$\mathbf{Y}$ symmetric state or with random feedback weights (where $\mathbf{Y}_l \neq -\mathbf{W}_{l+1}^T$). We computed the angle between the update calculated from the feedforward plasticity rule (Eq. 1) and either backprop or feedback alignment [4] for the symmetric and random configurations, respectively. We examined both cases: in the theoretically ideal case for learning $\mathbf{Q}$, where no teaching signal is present (Fig. 4A-D), and with a teaching signal at the output layer (Fig. 4E-H).

In all cases, as the alignment between the $\mathbf{Q}$ and $\mathbf{Y}$ connections improved (Fig. 4A,E), the apical potential decreased (Fig. 4B,F) and this resulted in updates that more closely aligned to backprop (Fig. 4C,G) and feedback alignment (Fig. 4D,H). In the absence of a teaching signal, this alignment angle to both backprop and feedback alignment eventually became very small which shows that the model approximates backprop updates supporting our analytical results (Fig. 4C-D). Despite producing less aligned feedforward updates in the presence of a teaching signal, the updates computed were still informative since they were consistently well below 90° of the direction of steepest descent (Fig. 4G).

3.4 BurstCCN learns image classification tasks with multiple hidden layers

3.4.1 MNIST

Next, to test whether our model can indeed perform backprop-like deep learning, we trained a number of (discrete-time) BurstCCN architectures on the MNIST handwritten digit classification task [21].

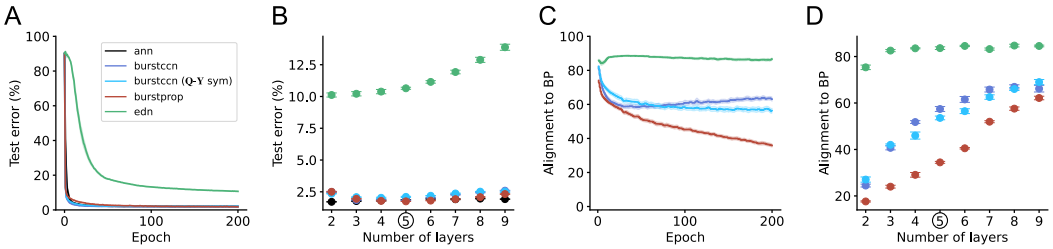

Figure 5: **BurstCCN learns to classify handwritten digits (MNIST) with deep networks.** (**A**) Learning curve and (**C**) alignment to backprop of 5-layer BurstCCN (blue), BurstCCN ($\eta^{(\mathbf{Q})} = 0$) (light blue), Burstprop (red) and EDN (green). (**B**) Different number of hidden layers across all models. (**D**) Alignment to backprop (BP) across number of hidden layers. The black circle indicates that the hyperparameters for each model were optimised for 5-layer networks. Model results represent mean $\pm$ standard error (n = 5).

We compared the BurstCCN with Burstprop [10] and EDNs [6] using similar architectures (see SM, Section C.3.3). We focus on the more biologically plausible case of using random fixed feedback weights (i.e. feedback alignment [4]; see Fig. S2 for symmetric feedback weight case) with the remaining connection types updated using their own plasticity rules as given by the different models. To be able to test our model in its idealised case we also tested a model for which the feedback STD weights ($\mathbf{Q}$) were fixed in the $\mathbf{Q}$-$\mathbf{Y}$ symmetric state (see Section 2.2), we denote this model as "BurstCCN ($\mathbf{Q}$-$\mathbf{Y}$ sym)".

Using 5-layer networks, the BurstCCN obtained a test error of 1.84$\pm$0.01%, comparable to that of Burstprop with 1.75$\pm$0.01% and significantly outperforming the EDN with 10.65$\pm$0.09% (Fig. 5A). As the network depth increased, both BurstCCN and Burstprop retained high performances but the EDN showed a substantial decay in performance with deeper networks (Fig. 5B). In an idealised case for the EDN, the disparity in performance and the effect of depth is less evident (Fig. S3). We then compared the alignment between the models and backprop. For the 5-layer networks, Burstprop's updates were most closely aligned to backprop, followed by the two BurstCCN models which all vastly outperformed the EDN (Fig. 5C). As expected, the BurstCCN with $\mathbf{Q}$-$\mathbf{Y}$ symmetry could better propagate error signals. By increasing the network depth, we demonstrate that it was more difficult to produce updates that were closely aligned to backprop. However, we show that the BurstCCN was still capable of backpropagating useful error signals in relatively deep networks (Fig. 5D).

3.4.2 CIFAR-10

Finally, we investigated the capabilities of the BurstCCN on more challenging tasks that are commonly tested in deep learning. We constructed a deep network consisting of three convolutional layers followed by a fully-connected hidden layer and output layer (Fig. 6A). We trained BurstCCN and Burstprop models using this network architecture on the CIFAR-10 image classification task [22]. BurstCCN ($\mathbf{Q}$-$\mathbf{Y}$ sym) was trained in the $\mathbf{Q}$-$\mathbf{Y}$ symmetric regime whereas BurstCCN was initialised in this state and $\mathbf{Q}$ weights were then updated using the corresponding plasticity rule. All model types were tested with two feedback weight regimes: $\mathbf{W}$-$\mathbf{Y}$ symmetric and random fixed $\mathbf{Y}$ feedback weights (i.e. feedback alignment).

After training, we observed a test error of 38.99$\pm$0.18% for BurstCCN with random feedback weights, similar to performances achieved by feedback alignment with an ANN (36.30$\pm$0.16%) and Burstprop (41.32$\pm$0.14%) (Fig. 6B). For the $\mathbf{W}$-$\mathbf{Y}$ symmetric regime which most resembles backprop, BurstCCN (22.92$\pm$0.03%) performed significantly better than all feedback alignment setups and, once again, obtained a similar error to the symmetric ANN (22.62$\pm$0.10%) and Burstprop (24.15$\pm$0.17%) cases. The increased performance seen in the symmetric setups can be explained by the improvement in alignment to backprop updates (Fig. 6C).

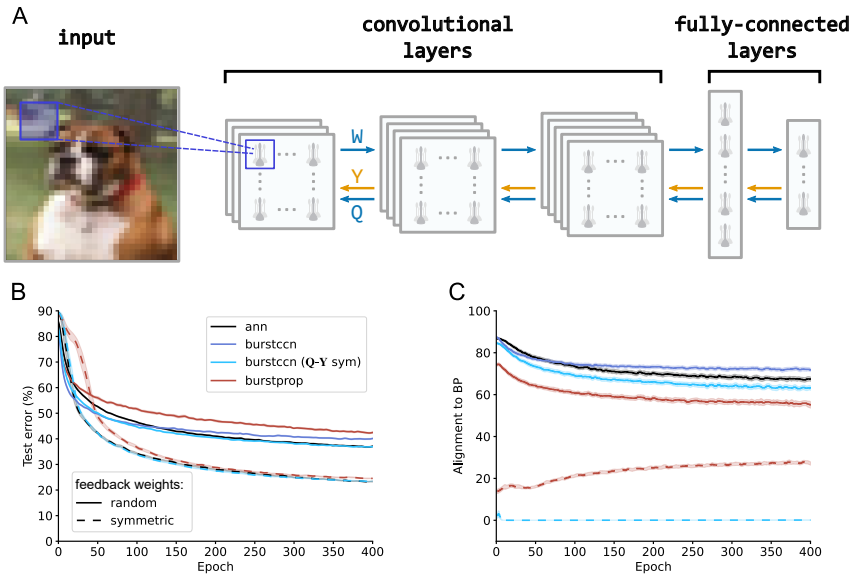

Figure 6: **BurstCCN with convolutional layers learns to solve natural image classification task (CIFAR-10).** (**A**) Schematic of networks trained consisting of an input layer, three convolutional layers, a fully-connected hidden layer and output layer. For the BurstCCN, each layer was connected with a set of feedforward weights, $\mathbf{W}$, and feedback weights, $\mathbf{Y}$ and $\mathbf{Q}$ (see main text for details). (**B**) Learning curve and (**C**) alignment to backprop of the different models with random (solid lines) and symmetric (dashed lines) feedback weight regimes. Model results represent mean $\pm$ standard error (n = 5).

4 Conclusions and discussion

We introduced a new model capable of backprop-like credit assignment by integrating known properties of cortical networks. We have shown that by combining specific biological mechanisms such as bursting, STP and dendrite-targeting inhibition it is possible to construct a model that learns effectively in a continuous setting that is reminiscent of learning in the brain. Moreover, we have demonstrated that such a model can learn complex image classification tasks in deep networks.

Our model proposes specific STP dynamics on the feedforward and feedback connections. It requires STD on cortico-cortical projections onto pyramidal cells in line with experimental evidence [12–16]. In addition, it suggests a key role for dendrite-targeting interneurons such as SST-positive Martinotti cells in the feedback pathway. There is evidence that these interneurons receive STF top-down connections whereas top-down projections onto pyramidal cells exhibit STD dynamics as required by our model [12–17]. In future work, it would be interesting to model the specific neuron types for each connection to satisfy Dale's law and further increase biological plausibility.

A prediction from our model is that manipulations of these interneurons with STF connections would lead to disruptions in burst decoding from the layer (brain area) above thereby obstructing learning in the brain area below. Additionally, as error signals alter the level of bursting in the network, the model predicts that the variance in bursting activity and the distal dendritic potentials would correlate with the level of error in the network during learning.

Although our model captures a wide range of biological features, some biological implausibilities remain. Currently, we use feedback alignment to provide a solution to the *weight transport problem* [23] but this has a substantial impact on performance, particularly in more challenging tasks. Therefore, it would be important to explore some of the recently introduced plausible feedback learning rules [24–26] which could be used in conjunction with our proposed learning rules to outperform feedback alignment [4].

Overall, our work provides a novel solution to the credit assignment problem and suggests that a range of cortical features from sub-cellular to the systems level jointly underlie single-phase, efficient deep learning in the brain.

References

[1] Pieter R Roelfsema and Anthony Holtmaat. Control of synaptic plasticity in deep cortical networks. *Nature Reviews Neuroscience*, 19(3):166–180, 2018.

[2] Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019.

[3] Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

[4] Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1):1–10, 2016.

[5] Blake A Richards and Timothy P Lillicrap. Dendritic solutions to the credit assignment problem. *Current opinion in neurobiology*, 54:28–36, 2019.

[6] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. In *Advances in Neural Information Processing Systems*, pages 8721–8732, 2018.

[7] Stephen R Williams and Greg J Stuart. Dependence of epsp efficacy on synapse location in neocortical pyramidal neurons. *Science*, 295(5561):1907–1910, 2002.

[8] Ning-long Xu, Mark T Harnett, Stephen R Williams, Daniel Huber, Daniel H O'Connor, Karel Svoboda, and Jeffrey C Magee. Nonlinear dendritic integration of sensory and motor input during an active sensing task. *Nature*, 492(7428):247–251, 2012.

[9] P Jesper Sjostrom, Ede A Rancz, Arnd Roth, and Michael Hausser. Dendritic excitability and synaptic plasticity. *Physiological reviews*, 88(2):769–840, 2008.

[10] Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A Richards, and Richard Naud. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature neuroscience*, 24(7):1010–1019, 2021.

[11] Richard Naud and Henning Sprekeler. Sparse bursts optimize information transmission in a multiplexed neural code. *Proceedings of the National Academy of Sciences*, 115(27):E6329–E6338, 2018.

[12] Soohyun Lee, Illya Kruglikov, Z Josh Huang, Gord Fishell, and Bernardo Rudy. A disinhibitory circuit mediates motor integration in the somatosensory cortex. *Nature neuroscience*, 16(11): 1662–1670, 2013.

[13] Amanda K Kinnischtzke, Daniel J Simons, and Erika E Fanselow. Motor cortex broadly engages excitatory and inhibitory neurons in somatosensory barrel cortex. *Cerebral cortex*, 24 (8):2237–2248, 2014.

[14] Iraklis Petrof, Angela N Viaene, and S Murray Sherman. Properties of the primary somatosensory cortex projection to the primary motor cortex in the mouse. *Journal of neurophysiology*, 113(7):2400–2407, 2015.

[15] Timothy A Zolnik, Julia Ledderose, Maria Toumazou, Thorsten Trimbuch, Tess Oram, Christian Rosenmund, Britta J Eickholt, Robert NS Sachdev, and Matthew E Larkum. Layer 6b is driven by intracortical long-range projection neurons. *Cell reports*, 30(10):3492–3505, 2020.

[16] Shovan Naskar, Jia Qi, Francisco Pereira, Charles R Gerfen, and Soohyun Lee. Cell-type-specific recruitment of gabaergic interneurons in the primary somatosensory cortex by long-range inputs. *Cell reports*, 34(8):108774, 2021.

[17] Luis E Martinetti, Kelly E Bonekamp, Dawn M Autio, Hye-Hyun Kim, and Shane R Crandall. Short-term facilitation of long-range corticocortical synapses revealed by selective optical stimulation. *Cerebral cortex*, 32(9):1932–1949, 2022.

[18] Elie L Bienenstock, Leon N Cooper, and Paul W Munro. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience*, 2(1):32–48, 1982.

[19] Per Jesper Sjöström, Gina G Turrigiano, and Sacha B Nelson. Rate, timing, and cooperativity jointly determine cortical synaptic plasticity. *Neuron*, 32(6):1149–1164, 2001.

[20] Tim P Vogels, Henning Sprekeler, Friedemann Zenke, Claudia Clopath, and Wulfram Gerstner. Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks. *Science*, 334(6062):1569–1573, 2011.

[21] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

[22] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[23] Francis Crick. The recent excitement about neural networks. *Nature*, 337(6203):129–132, 1989.

[24] Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 498–515. Springer, 2015.

[25] Mohamed Akrout, Collin Wilson, Peter C Humphreys, Timothy Lillicrap, and Douglas Tweed. Using weight mirrors to improve feedback alignment. *arXiv preprint arXiv:1904.05391*, 2019.

[26] Nasir Ahmad, Marcel A van Gerven, and Luca Ambrogioni. Gait-prop: A biologically plausible learning rule derived from backpropagation of error. *Advances in Neural Information Processing Systems*, 33:10913–10923, 2020.

[27] Jordan Guerguiev, Timothy P Lillicrap, and Blake A Richards. Towards deep learning with segregated dendrites. *ELife*, 6:e22901, 2017.

[28] Richard Naud, Nicolas Marcille, Claudia Clopath, and Wulfram Gerstner. Firing patterns in the adaptive exponential integrate-and-fire model. *Biological cybernetics*, 99(4):335–347, 2008.

[29] Friedemann Zenke and Wulfram Gerstner. Limits to high-speed simulations of spiking neural networks using general-purpose computers. *Frontiers in neuroinformatics*, 8:76, 2014.

[30] Lukas Biewald. Experiment tracking with weights and biases, 2020. URL `https://www.wandb.com/`. Software available from wandb.com.

[31] Chiayu Q Chiu, James S Martenson, Maya Yamazaki, Rie Natsume, Kenji Sakimura, Susumu Tomita, Steven J Tavalin, and Michael J Higley. Input-specific nmdar-dependent potentiation of dendritic gabaergic inhibition. *Neuron*, 97(2):368–377, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See SM and main text.
   (b) Did you include complete proofs of all theoretical results? [Yes] See SM.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See SM.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See SM.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Its also detailed in the captions how many seeds we used and what the error bars represent.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See SM.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See SM.
   (b) Did you mention the license of the assets? [Yes] See SM.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]