# Latent Equilibrium: Arbitrarily fast computation with arbitrarily slow neurons

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

The response time of physical computational elements is finite, and neurons are no exception. In hierarchical models of cortical networks each layer thus introduces a response lag. This inherent property of physical dynamical systems results in delayed processing of stimuli and causes a timing mismatch between network output and instructive signals, thus afflicting not only inference, but also learning. We introduce Latent Equilibrium, a new framework for inference and learning in networks of slow components which avoids these issues by harnessing the ability of biological neurons to phase-advance their output with respect to their membrane potential. This principle enables quasi-instantaneous inference independent of network depth and avoids the need for phased plasticity or computationally expensive network relaxation phases. We jointly derive disentangled neuron and synapse dynamics from a prospective energy function that depends on a network's generalized position and momentum. The resulting model can be interpreted as a biologically plausible approximation of error backpropagation in deep cortical networks with continuous-time, leaky neuronal dynamics and continuously active, local plasticity. We demonstrate successful learning of standard benchmark datasets, achieving competitive performance using both fully-connected and convolutional architectures, and show how our principle can be applied to detailed models of cortical microcircuitry. Furthermore, we study the robustness of our model to spatio-temporal substrate imperfections to demonstrate its feasibility for physical realization, be it in vivo or in silico.

## 1 Introduction

Physical systems composed of large collections of simple, but intricately connected elements can exhibit powerful collective computational properties. A prime example are animals' nervous systems, and most prominently the human brain. Its computational prowess has motivated a large, cross-disciplinary and ongoing endeavor to emulate aspects of its structure and dynamics in artificial substrates, with the aim of being ultimately able to replicate its function. The speed of information processing in such a system depends on the response time of its components; for neurons, for example, it can be the integration time scale determined by their membrane time constant.

If we consider hierarchically organized neural networks composed of such elements, each layer in the hierarchy causes a response lag with respect to a changing stimulus. This lag introduces two related critical issues. For one, the inference speed in these systems decreases with their depth. In turn, this induces timing mismatches between instructive signals and neural activity, which disrupts learning. For example, recent proposals for bio-plausible implementations of error backpropagation (BP) [1–4] in the brain all require some form of relaxation, both for inference and during learning [5–11]. Notably, this also affects some purely algorithmic methods involving auxiliary variables [12]. To

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

deal with this inherent property of physical dynamical systems, two approaches have been suggested: either phased plasticity that is active only following a certain relaxation period, or long stimulus presentation times with small learning rates. Both of these solutions entail significant drawbacks: the former is challenging to implement in asynchronous, distributed systems such as cortical networks or neuromorphic hardware, while the latter results, by construction, in slow learning. This has prompted the critique that any algorithm requiring such a settling process is too slow to describe complex brain function, particularly when involving real-time responses [13]. To the best of our knowledge this fundamental problem affects all modern models of approximate BP in biological substrates [5–11].

To overcome these issues, we propose a novel framework for fast computation and learning in physical substrates with slow components. In particular, this provides a biologically plausible approximation of BP in deep cortical networks with continuous-time, leaky neuronal dynamics and local, continuous plasticity. Moreover, our model is easy to implement in both software and hardware and is well-suited for distributed, asynchronous systems.

In our framework, inference can be arbitrarily fast (up to finite simulation resolution or finite communication speed across physical distances) despite a finite response time of individual system components; downstream responses to input changes thus become effectively instantaneous. Conversely, responses to instructive top-down input that generate local error signals are also near-instantaneous, thus effectively removing the need for any relaxation phase. This allows truly phase-free learning from signals that change on much faster time scales than the response speed of individual network components. Similarly to other approaches [5, 8, 9], we derive neuron and synapse dynamics from a joint energy function. However, our energy function is designed to effectively disentangle these dynamics, thus removing the disruptive co-dependencies that otherwise arise during relaxation. This is achieved by introducing a simple, but crucial new ingredient: neuronal outputs that try to predict their future state based on their current information, a property we describe as "prospective". Thereby, our framework also constructs an intimate relationship between such "slow" neuronal networks and artificial neural network (ANN)[1], thus enabling the application of various auxiliary methods from deep learning.

## 2 The problems of slow components

To illustrate the issue with relaxation, we consider a simple network of two model neurons arranged in a chain (Fig. 1a). Biological neuronal membrane potentials $u$ are conventionally modeled as leaky integrators of their input $I$: $C_\mathrm{m}\dot{u} = -g_\mathrm{l}u + I$, where the membrane capacitance $C_\mathrm{m}$ and leak conductance $g_\mathrm{l}$ determine the membrane time constant $\tau^\mathrm{m} := C_\mathrm{m}/g_\mathrm{l}$ and thereby its response speed. These dynamics attenuate and delay input, resulting in significant differences between neuronal output rates and those expected in an instantaneous system (Fig. 1b). This mismatch increases with every additional layer; a feedforward network with $n$ layers thus has an effective relaxation time constant of approximately $n\tau^\mathrm{m}$.

Besides slow inference, this delayed response leads to critical issues during learning from downstream instructive signals. Consider the typical scenario where such a target signal is present in the output layer and plasticity is continuously active (and not phased according to some complicated schedule). If the system fulfills its task correctly, the target signal corresponds to the output of the relaxed system and no learning should take place. However, due to delays in neuronal responses, the output signal differs from the target during relaxation, which causes plasticity to adapt synaptic weights in an effort to better match the target. As a result, the system "overshoots" during early relaxation and has to undo these synaptic changes through further learning in the late relaxation phase (Fig. 1c). Also, since output errors need inputs to propagate forward through the entire network before propagating backwards themselves, the input layer only observes correct errors after about $2n\tau^\mathrm{m}$. We refer to this issue as the "relaxation problem". In the following, we present a solution for this problem, combining prospective neuron dynamics with continuously active, local synaptic plasticity.

## 3 Fast computations in slow substrates

Inspired by the canonical coordinates from classical mechanics, we describe the state of a neuronal network by its position in phase space $(\boldsymbol{u}, \dot{\boldsymbol{u}})$, with the generalized position $\boldsymbol{u}$ and the generalized

[1] To differentiate between biologically plausible, leaky neurons and abstract neurons with instantaneous response, we respectively use the terms "neuronal" and "neural".

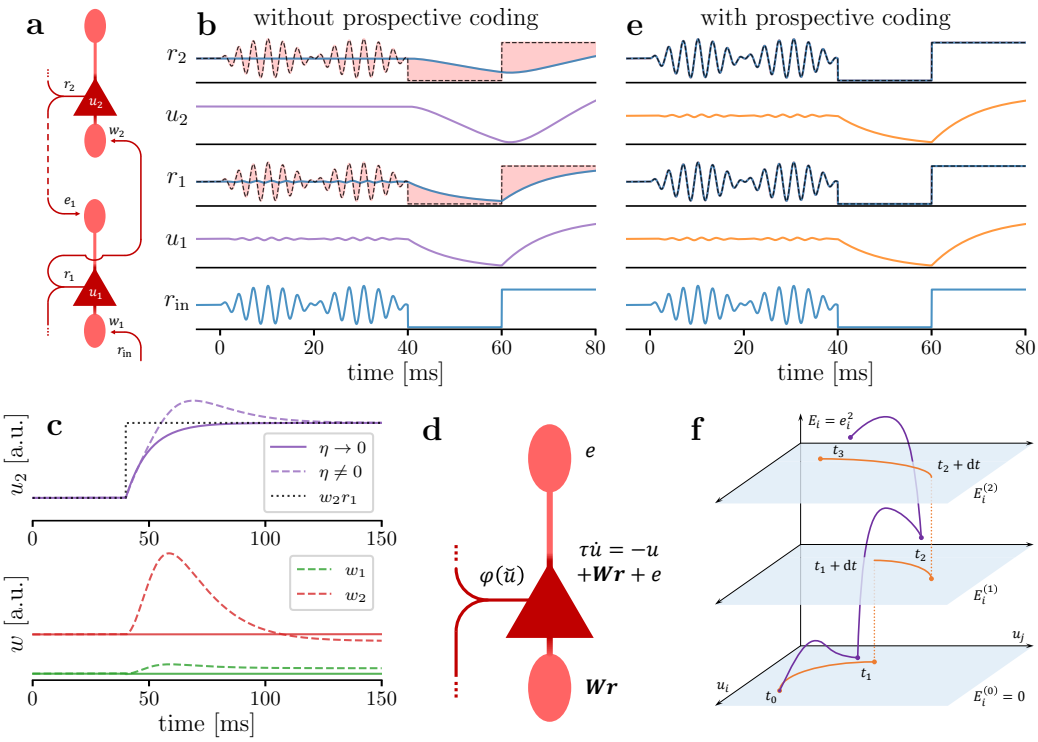


**Figure 1: Prospective coding solves the relaxation problem. (a)** A simple, functionally feedforward network of two neurons. Note the recurrence induced by the propagation of both bottom-up signals and top-down errors. **(b)** Neuronal firing rates (blue) and membrane potentials (purple) for an input sequence consisting of a quickly but smoothly changing stimulus followed by two constant stimuli. Dashed black lines denote instantaneous (relaxed) rates. Red shading highlights the mismatch between instantaneous and actual firing rates that disrupts both inference and learning. **(c)** Continuous learning during relaxation without prospective coding in the network from (a). Dotted black: target membrane potential. Solid lines: trajectories for vanishing learning rates. Dashed lines: trajectories for nonzero learning rates. Purple: membrane potentials. Note the overshoot when learning is continuously active. Green/red: presynaptic weights of 1st/2nd neuron. **(d)** Sketch of neuron model derived from Latent Equilibrium (LE). The membrane potential still reacts slowly to input, but the output firing rate uses the prospective membrane potential $\breve{u}^{\mathrm{m}}$. **(e)** Same as (b), but with LE. Note the instantaneous reaction to input changes of the second neuron. **(f)** Sketch of the mismatch energy $E_i$ for a hidden neuron during learning for an input with several jumps. In LE (orange), the energy itself jumps with the input, but then remains constant while neuron dynamics $(u_i, u_j)$ evolve. Without LE (purple), the energy changes transiently and plasticity follows incorrect gradients before relaxation. The trajectories for $E = 0$ correspond to the scenario from (c). Planes of constant energy are drawn for visual guidance and do not represent constant-energy manifolds.

momentum $\dot{\boldsymbol{u}}$. The relation between these two components describes the physics of the system. To obtain the leaky integrator dynamics that characterize biological neurons, we first define the abstract network state $\breve{\boldsymbol{u}}^{\mathrm{m}}$ as

$$\breve{\boldsymbol{u}}^{\mathrm{m}} := \boldsymbol{u} + \tau^{\mathrm{m}}\dot{\boldsymbol{u}} \; . \tag{1}$$

We next define an energy function $E$ that characterizes this state:

$$E(\breve{\boldsymbol{u}}^{\mathrm{m}}) := \frac{1}{2}\left\|\breve{\boldsymbol{u}}^{\mathrm{m}} - \boldsymbol{W}\boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\mathrm{m}}) - \boldsymbol{b}\right\|^2 + \beta\mathcal{L}(\breve{\boldsymbol{u}}^{\mathrm{m}}) \; . \tag{2}$$

This energy represents a global view of the network from which neuron and weight dynamics will be derived. Here, $\boldsymbol{W}$ and $\boldsymbol{b}$ represent the weight matrix and bias vector, $\varphi$ is the neuronal activation function and the loss $\mathcal{L}$ is scaled by a constant $\beta$; we use bold font for matrices, vectors, and vector-valued functions. This formulation of the energy function is very generic and can, by appropriate choice of parameters, apply to different network topologies, including multilayer perceptrons, convolutional or recurrent networks. Note that this energy can be written as a neuron-wise sum over mismatch energies $E_i(\breve{u}_i^{\mathrm{m}}, \boldsymbol{r}_{i,\mathrm{pre}}) = (\breve{u}_i^{\mathrm{m}} - \boldsymbol{W}_i\boldsymbol{r}_{i,\mathrm{pre}} - b_i)^2$, where $\boldsymbol{r}_{i,\mathrm{pre}} := \boldsymbol{\varphi}(\breve{\boldsymbol{u}}_{i,\mathrm{pre}}^{\mathrm{m}})$ is the presynaptic input vector for the $i$th neuron in the network.

We can now derive neuronal dynamics as extrema of the energy function from Eqn. 2:

$$\nabla_{\breve{\boldsymbol{u}}^{\mathrm{m}}} E = 0 \quad \Longrightarrow \quad \tau^{\mathrm{m}} \dot{\boldsymbol{u}} = -\boldsymbol{u} + \boldsymbol{W}\boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\mathrm{m}}) + \boldsymbol{b} + \boldsymbol{e} \ , \tag{3}$$

with presynaptic bottom-up input $\boldsymbol{W}\boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\mathrm{m}})$ and top-down error signals

$$\boldsymbol{e} = \boldsymbol{\varphi}'(\breve{\boldsymbol{u}}^{\mathrm{m}})\boldsymbol{W}^{\mathrm{T}} \left[\breve{\boldsymbol{u}}^{\mathrm{m}} - \boldsymbol{W}\boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\mathrm{m}}) - \boldsymbol{b}\right] \tag{4}$$

for hidden neurons and $\boldsymbol{e} = -\beta \nabla_{\breve{\boldsymbol{u}}^{\mathrm{m}}} \mathcal{L}$ for neurons which directly contribute to the loss. Using Eqn. 3, it is easy to see that in hierarchical networks these errors can be expressed recursively over layers $\ell$, $\boldsymbol{e}_\ell = \boldsymbol{\varphi}'(\breve{\boldsymbol{u}}^{\mathrm{m}})\boldsymbol{W}_{\ell+1}^{\mathrm{T}}\boldsymbol{e}_{\ell+1}$, thus instantiating a variant of BP. Eqn. 3 can be interpreted as the dynamics of a structured pyramidal neuron receiving presynaptic, bottom-up input via its basal dendrites and top-down input via its apical tree (Fig. 1d). We provide a more detailed description of our model's biophysical implementation in Section 5. We note that our approach is in contrast to previous work that introduced neuron dynamics via gradient descent on an energy function, such as [6, 8]. Indeed, this difference is crucial for solving the relaxation problem, as discussed below. Since for a given input our network moves, by construction, within a constant-energy manifold, we refer to our model as Latent Equilibrium (LE).

We can now revisit our choice of $\breve{\boldsymbol{u}}^{\mathrm{m}}$ from a functional point of view. Instead of the classical output rate $\varphi(u)$, our neurons fire with $\varphi(\breve{u}^{\mathrm{m}})$, which depends on both $u$ and $\dot{u}$ (Eqn. 1). This represents a known, though often neglected feature of biological neurons [14–16] and has also been considered in other models of bio-plausible BP derived from a stationary action [17, 18]. As neuron membranes are low-pass filters (Eqn. 3), $\breve{u}^{\mathrm{m}}$ can be viewed as a prospective version of $u$: when firing, the neuron tries to look into the future and predict the state of its membrane potential after relaxation. The prospective nature of $\breve{u}^{\mathrm{m}}$ also holds in a strict mathematical sense: the breve operator $\breve{\ }^{\mathrm{m}} := (1 + \tau^{\mathrm{m}}\mathrm{d}/\mathrm{d}t)$ is the exact inverse of an exponential low-pass filter (see SI). While neuronal membranes continue to relax slowly towards their steady states, neuronal outputs use membrane momenta to compute a correct instantaneous reaction to their inputs, even in the case of jumps (Fig. 1e). Thus, information can propagate instantaneously throughout the network, similarly to an ANN, counterintuitively even when membrane dynamics are never in equilibrium. The activity of the output layer hence reflects arbitrarily fast changes in the input – even on time scales smaller than the neuronal time constant $\tau^{\mathrm{m}}$ – rather than responding with a significant time lag and attenuation as in previous, gradient-based models.

Note that, even for functionally feedforward networks, our resulting network structure is recurrent, with backward coupling induced by error inputs to the apical tree. As a non-linear recurrent network, it cannot settle instantaneously into the correct state; rather, in numerical simulations, it jumps quickly towards an estimated stationary activity state and reaches equilibrium within several such jumps (of infinitesimal duration). In practice, saturating activation functions can help avoid pathological behavior under strong coupling. Moreover, we can introduce a very short exponential low-pass filter $\tau^{\mathrm{s}}$ on top-down signals, slightly larger than the temporal resolution of the simulation. Thus, in physical systems operating in continuous time, $\tau^{\mathrm{s}}$ can effectively become infinitesimal as well and does not affect the speed of information propagation through the network. In particular, as we discuss below, the perpetual consistency between input and output allows our model to continuously learn to reduce the loss, obviating the need for network relaxation phases and the associated global control of precisely timed plasticity mechanisms.

## 4 Fast learning in slow substrates

Based on our prospective energy function (Eqn. 2), we define synaptic weight dynamics, i.e., learning, as time-continuous stochastic gradient descent with learning rate $\eta_{\mathrm{w}}$:

$$\dot{\boldsymbol{W}} \propto \nabla_{\boldsymbol{W}} E \quad \Longrightarrow \quad \dot{\boldsymbol{W}} = \eta_{\mathrm{w}} \left[\breve{\boldsymbol{u}}^{\mathrm{m}} - \boldsymbol{W}\boldsymbol{r} - \boldsymbol{b}\right] \boldsymbol{r}^{\mathrm{T}} \ . \tag{5}$$

Thus, weights evolve continuously in time driven by local error signals without requiring any particular schedule. Neuronal biases are adapted according to the same principle. Note that this rule only uses quantities that are available at the locus of the synapse (see also Section 5). Intuitively, this locality is enabled by the recurrent nature of the network: errors in the output units spread throughout the system, attributing credit locally through changes in neuronal membrane potentials. These changes are then used by synapses to update their weight in order to reduce the network loss. However, our learning rule is not an exact replica of BP, although it does approximate it in the limit

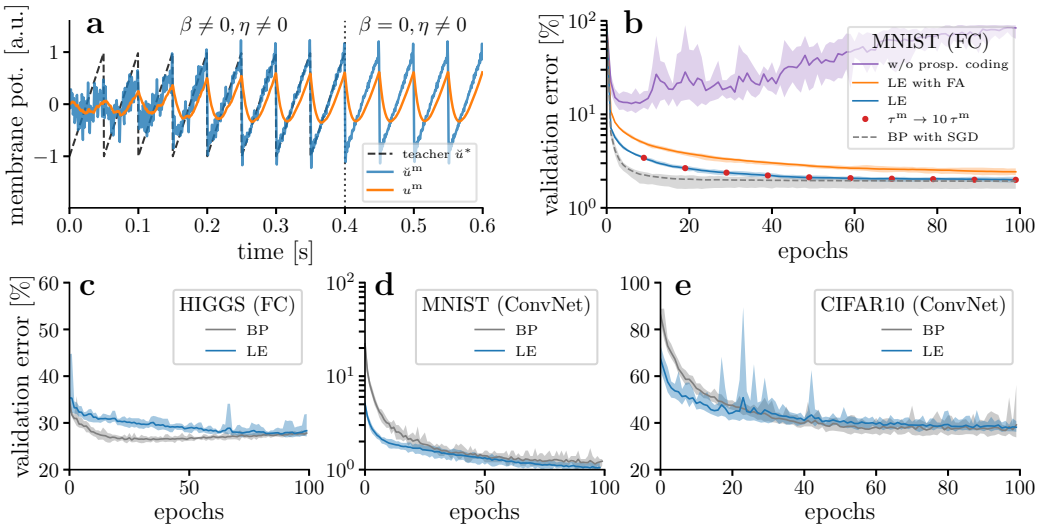


**Figure 2: Learning with LE. (a)** Fourier synthesis with fully connected (FC) network of size 50-30-1, receiving sinusoidal inputs with equidistant frequencies between 10 and 510 Hz. Teaching signal only present for the first 0.4 s, plasticity is continuously active. **(b)** MNIST dataset with FC network of size 784-300-100-10, presentation times $T_{\text{pres}} = 1\,\text{ms} = 0.05\,\tau^{\text{m}}$ per sample. Since variability for runs with different $\tau^{\text{m}}$ is statistically indistinguishable, we only show one set of corresponding min/max values. For comparison, we also show the performance with feedback alignment (FA) [22]. **(d)** MNIST with convolutional network (ConvNet) LeNet-5 [21]. **(c)** HIGGS dataset with FC network of size 28-300-300-300-1. **(e)** CIFAR-10 dataset (ConvNet: LeNet-5). For all examples, standard ANNs with the same topology trained with classical BP are shown for comparison.

of infinitely weak supervision $\beta \to 0$ (often referred to as "nudging"); strictly speaking, it minimizes the energy function $E$, which implicitly minimizes the loss $\mathcal{L}$. This form of credit assignment can be related to previous models which similarly avoid a separate artificial backward pass ()as necessary in classical BP) by allowing errors to influence neuronal activity [19]. Plasticity in the weights projecting to output neurons depends on the choice of $\mathcal{L}$; for example, for an $L^2$ loss, plasticity in the output layer corresponds to the classical delta rule [20]: $\dot{\boldsymbol{W}}_N = -\eta\beta\left[\boldsymbol{r}_N^* - \boldsymbol{r}_N\right]\boldsymbol{r}_{N-1}^{\mathrm{T}}$.

Despite similarities to previous work, learning in our framework does not suffer from many of the shortcomings that we have already noted. Since activity propagates quasi-instantaneously throughout the network, our plasticity can be continuously active without disrupting learning performance. This is true by construction and most easily visualized for a sequence of (piecewise constant) input patterns: following a change in the input, membrane dynamics take place in a constant-energy manifold (Eqn. 3) across which synaptic weight dynamics remain unchanged, i.e., they equally and simultaneously pull downward on all points of this manifold (Fig. 1f). This disentangling of membrane and synaptic weight dynamics constitutes the crucial difference to previous work, where the neuronal mismatch energies $E_i$ change as dynamics evolve and thus can not represent the true errors in the network before reaching a fixed point. We further note that LE also alleviates a second form of catastrophic forgetting in these other models: due to the misrepresentation of errors during relaxation, continuous learning changes synaptic weights even in perfectly trained networks (Fig. 1c).

To illustrate time-continuous learning on continuous data streams, we first consider a simple Fourier-synthesis task in which a network driven by oscillating inputs learns to approximate a periodic (here: sawtooth) target signal. Despite the wavelengths of both input and target being much smaller than the time constant of the neuron, continuous plasticity in our model allows the output neuron to approximate the target well (Fig. 2a). The remaining differences merely reflect the fundamental limitation of a finite set of input frequencies and hidden units.

We next turn the more traditional paradigm of classification tasks to demonstrate the scalability of our model. We train feedforward networks using LE on several standard benchmark datasets (see SI for details) using sample presentation times much shorter than the membrane time constant. We first learn MNIST [23] and HIGGS [24] with a fully connected architecture. After 100 epochs, our model reaches classification test errors (mean $\pm$ std) of $(1.98 \pm 0.11)\,\%$ (MNIST) and $(27.6 \pm 0.4)\,\%$

(HIGGS), on par with a standard ANN trained using stochastic gradient descent (SGD) and reaching $(1.93 \pm 0.14)\,\%$ (MNIST) and $(27.8 \pm 0.4)\,\%$ (HIGGS) with the same architecture, i.e., with the same number of learnable parameters (Fig. 2b,c). To illustrate the indifference of LE with respect to neuronal time constants, we repeated MNIST experiments with significantly slower neuronal dynamics. In contrast, a network following the classical paradigm without prospective rates performs poorly: for membrane time constants of $\tau^{\mathrm{m}} = 10\,\mathrm{ms}$ and presentation times of $T = 100\,\mathrm{ms}$ per sample, the network does not exceed $90\%$ accuracy on MNIST. For shorter presentation times, the performance of such models quickly degrades further.

Since our model does not assume any specific connectivity pattern, we can easily integrate different deep network topologies. Here we demonstrate this by introducing convolutional architectures on both MNIST and the more challenging CIFAR10 [25] datasets. On these datasets, our LE networks achieve test performances of $(1.1 \pm 0.1)\,\%$ (MNIST) and $(38.0 \pm 1.3)\,\%$ (CIFAR10), again on par with ANNs with identical structure at $(1.08 \pm 0.07)\,\%$ (MNIST) and $(39.4 \pm 5.6)\,\%$ (CIFAR10) (Fig. 2d,e). These results demonstrate that LE enables time-continuous learning on arbitrarily short presentation time scales in hierarchical networks of leaky integrator neurons to achieve results competitive with standard BP in ANNs.

## 5 Fast computation and learning in cortical microcircuits

Due to the simplicity of their implementation, the principles of LE can be applied to models of approximate BP in the brain in order to alleviate the issues discussed above. Here we demonstrate how a network of hierarchically organized dendritic microcircuits [8] can make use of our theoretical advances to significantly increase both inference and training speed, thus removing several critical shortcomings towards its viability as a scalable model of cortical processing. The resulting dynamical system represents a detailed and biologically plausible version of BP, with real-time dynamics, and phase-free, continual local learning able to operate on effectively arbitrary sensory timescales.

The fundamental building block of this architecture is a cortical microcircuit model consisting of pyramidal cells and interneurons (Fig. 3a,b). Pyramidal cells have three compartments: a basal dendrite receiving bottom-up input from lower areas, an apical dendrite receiving top-down input from higher areas and lateral input from interneurons, and a somatic compartment that integrates dendritic information and generates the neuronal output. Interneurons consist of two compartments: a basal dendrite receiving input from pyramidal cells in the same layer, and a somatic compartment that receives input from pyramidal cells in higher layers. Pyramidal somatic compartments are leaky integrators of input from neighboring compartments (see SI for the full set of equations):

$$C_{\mathrm{m}} \dot{u}_i^{\mathrm{som}} = g_{\mathrm{l}} \left(E_{\mathrm{l}} - u_i^{\mathrm{som}}\right) + g^{\mathrm{bas}} \left(v_i^{\mathrm{bas}} - u_i^{\mathrm{som}}\right) + g^{\mathrm{api}} \left(v_i^{\mathrm{api}} - u_i^{\mathrm{som}}\right) \ , \tag{6}$$

where $i$ is the neuron index, $E_{\mathrm{l}}$ the leak potential, $v_i^{\mathrm{bas}}$ and $v_i^{\mathrm{api}}$ the basal and apical membrane potentials, respectively, and $g^{\mathrm{bas}}$ and $g^{\mathrm{api}}$ the dendro-somatic couplings. Due to the conductance-based interaction between compartments, the effective time constant of the soma is $\tau^{\mathrm{eff}} := C_{\mathrm{m}}/(g_{\mathrm{l}} + g^{\mathrm{bas}} + g^{\mathrm{api}})$. For somatic membrane potentials, assuming that apical dendrites encode errors (see below) and basal dendrites represent the input, this corresponds directly to Eqn. 3. Following [26], plasticity in basal synapses is driven by the local error signal given by the discrepancy between somatic and dendric membrane potentials $u_i^{\mathrm{som}}$ and $v_i^{\mathrm{bas}}$:

$$\dot{w}_{ij} = \eta_{\mathrm{w}} \left[\varphi(u_i^{\mathrm{som}}) - \varphi(\alpha v_i^{\mathrm{bas}})\right] \varphi(u_j^{\mathrm{som}}) \ , \tag{7}$$

which is analogous to Eqn. 5 up to a monotonic transformation on the voltages (see SI).

In this architecture, plasticity serves two purposes. For pyramidal-to-pyramidal feedforward synapses, it implements error-correcting learning as a time-continuous approximation of BP. For pyramidal-to-interneuron synapses, it drives interneurons to mimic their pyramidal partners in the layers above (see also SI). Thus, in a well-trained network, apical compartments of pyramidal cells are at rest, reflecting zero error, as top-down and lateral inputs cancel out. When an output error propagates through the network, these two inputs can no longer cancel out and their difference represents the local error $e_i$. This architecture does not rely on the transpose of the forward weight matrix, improving viability for implementation in distributed asynchronous systems. Here, we keep feedback weights fixed, realizing a variant of feedback alignment [22]. In principle, these weights could also be learned in order to further improve the local representation of errors [8, 27, 28].

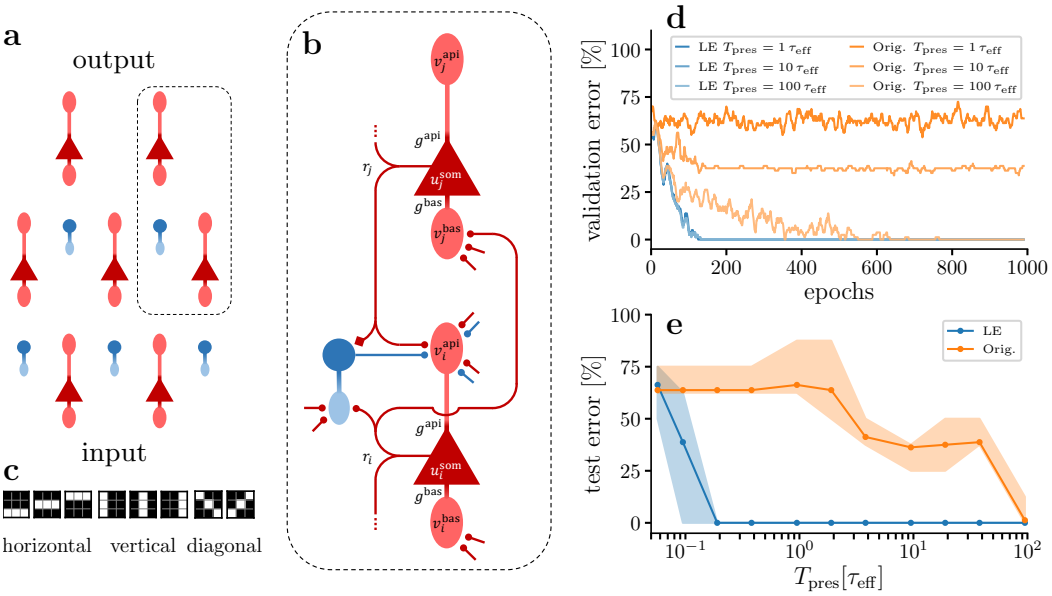


**Figure 3: LE in cortical microcircuits. (a,b)** Model architecture following [8]. Red: pyramidal cells, blue: interneurons; somatic compartments have darker colors. Each soma sends out a single axon that transmits either $\varphi(u)$ (original model) or $\varphi(\breve{u}^{\text{m}})$ (prospective rates). Except for top-down synapses, all synapses are plastic. **(c)** Synthetic dataset with 8 images grouped in 3 classes used for training a 3-layer network with 9-30-3 pyramidal cells. **(d)** Model performance during learning with (blue) and without (orange) LE. Note the similarity in performance gains at the beginning of training, before the disruptive effects of relaxation begin to dominate. For a better visualization of this effect, fluctuations are smoothed with a sliding window over 10 epochs. **(e)** Model performance (min, max and average over 10 seeds) after 1000 epochs for different input presentation times.

Incorporating the principles of LE is straightforward and requires only that neurons fire prospectively: $\varphi(u) \to \varphi(\breve{u}^{\text{eff}})$. While we have already addressed the evidence for prospective neuronal activity, we note that our plasticity also uses these prospective signals, which constitutes an interesting prediction for future in-vivo experiments. We can now compare the behavior and performance of our LE-augmented model to its original archetype. Since large networks using the original model require prohibitively long training times when simulated with full dynamics rather than just their steady state [8], we use small networks and a small synthetic dataset as a benchmark (Fig. 3c).

Our microcircuit model can learn perfect classification even for very short presentation times. In contrast, the original model without the prospective mechanism stagnates at high error rates even for this simple task. As discussed earlier, this can be traced back to the learning process being disrupted during relaxation. Without prospective dynamics, the model requires presentation times on the order of $100\,\tau^{\text{eff}}$ to achieve perfect accuracy (Fig. 3e). In contrast, LE only degrades for presentation times below $0.1\,\tau^{\text{eff}}$, which is due to the limited resolution of our numerical integration method. Thus, incorporating LE into cortical microcircuits can bring the required presentation times into biologically plausible regimes, allowing networks to deal with rich sensory data.

## 6 Robustness to substrate imperfections

Computer simulations often assume perfectly homogeneous parameters across the network. Models can hence inadvertently rely on this homogeneity, resulting in unpredictable behavior and possibly fatal dysfunction when faced with the physics of analog substrates which are characterized by both heterogeneity in their components as well as temporal perturbation of their dynamics. Therefore, we consider robustness to spatio-temporal noise to represent a necessary prerequisite for any mathematical model aspiring to physical implementation, be it biological or artificial.

Spatial noise reflects the individuality of cortical neurons or the heterogeneity arising from device mismatch in hardware. Here, we focus on the heterogeneity of time constants; in contrast to, for

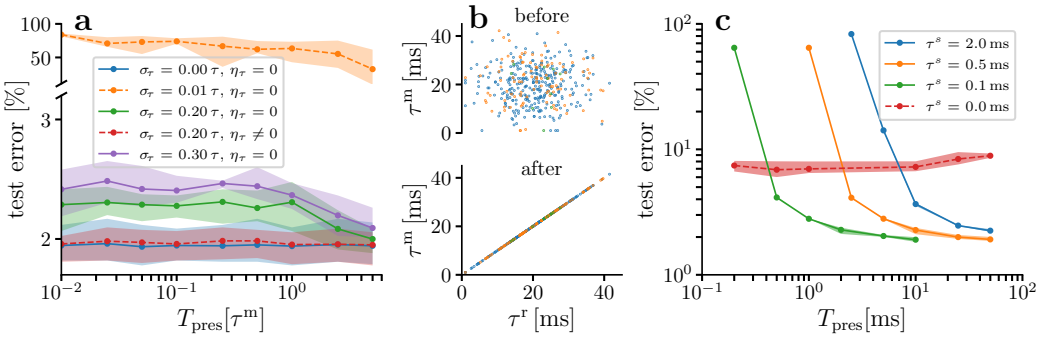


**Figure 4: Robustness of LE.** Model performance on MNIST (min, max and average over 4 seeds) after 100 epochs for different input presentation times. **(a)** Effects of heterogeneity of time constants using either $\breve{u}^{\text{m}}$ (solid) or $\breve{u}^{\text{r}}$ (dashed) to calculate the mismatch error. $\eta_\tau \neq 0$ indicates learning after the developmental phase. **(b)** Integration and prospective time constant pairs $(\tau^{\text{m}}, \tau^{\text{r}})$ for individual neurons within a network before (top) and after (bottom) development. Colors encode the layer to which the neurons belong. **(c)** Effects of temporal noise of width $\sigma_{\text{r}} = 0.2\, r_{\max}$ with (solid) and without (dashed) synaptic filtering for different synaptic time constants (min, max and average over 3 seeds).

example, variability in synaptic parameters or activation functions, these variations can not be "trained away" by adapting synaptic weights. The two time constants that govern neuron dynamics in our model, namely integration (Eqn. 3) and prospective coding (Eqn. 1), previously assumed to be identical, are affected independently by such variability. To differentiate between the two, we assign the prospective dynamics their own time constant: $\breve{u}^{\text{r}} := u + \tau^{\text{r}}\dot{u}$. We can now model heterogeneity as independent, multiplicative Gaussian noise on all time constants: $\tau^{\text{m/r}} \to (1+\xi)\tau^{\text{m/r}}$, with $\xi \sim \mathcal{N}(0, \sigma_\tau^2)$; we use multiplicative noise to emphasize that our model is agnostic to absolute time scales, so only relative relationships between specific time constants matter.

Due to the resulting mismatches between the timing of neuronal input and output, neuronal outputs suffer from exponential transients, leading to relaxation issues similar to the ones we already addressed in detail. However, depending on the transmission of top-down signals, the effects on learning performance can be very different. According to the formal theory, backward errors use the correct prospective voltages: $\boldsymbol{e} \propto [\breve{\boldsymbol{u}}^{\text{m}} - \boldsymbol{W}\boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\text{r}})]$ (Eqns. 4 and 5); this allows robust learning even for relatively large perturbations in the forward signals (Fig. 4a). In contrast, in biophysical implementations such as the microcircuits discussed above, neurons can only transmit a single output signal $\varphi(\breve{u}_i^{\text{r}})$, which consequently also affects the errors: $\boldsymbol{e} \propto [\breve{\boldsymbol{u}}^{\text{r}} - \boldsymbol{W}\boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\text{r}})]$. Since deviations due to a mismatch between integration and prospective time constants persist on the time scale of $\tau^{\text{m}}$ (see SI), even small amounts of mismatch can lead to a significant loss in performance (Fig. 4a).

Here we address this issue by introducing a neuron-local adaptation mechanism that corrects the difference between prospective voltages $\breve{\boldsymbol{u}}^{\text{m}}$ and $\breve{\boldsymbol{u}}^{\text{r}}$ induced by mismatching time constants:

$$\dot{\boldsymbol{\tau}}^{\text{m}} = \eta_\tau \left(\breve{\boldsymbol{u}}^{\text{r}} - \boldsymbol{W}\boldsymbol{r} - \boldsymbol{b}\right) \dot{\boldsymbol{u}} \,. \tag{8}$$

In biological substrates, this could, for example, correspond to an adaptation of the transmembrane ion channel densities, whereas on many neuromorphic substrates, neuronal time constants can be adapted individually [29]. Before training the network we allow it to go through a "developmental phase" in which individual neurons are not affected by top-down errors and learn to match their prospective firing to their membrane dynamics (Fig. 4b), thus recovering the performance of networks with perfectly matched time constants (Fig. 4a). This developmental phase merely consists of presenting input samples to the network, for a duration that depends on the required matching precision. Here, we achieved mismatches below 1 ‰ within the equivalent of 20 training epochs. Note that, as observed, for example, in vivo, neuronal time constants remain diverse after this phase, but are matched in a way that enables fast reaction of downstream areas to sensory stimuli – an ability that is certainly helpful for survival.

We next model additive temporal noise on neuronal outputs as might be induced, for example, by noisy transmission channels: $r \to r + \xi$, with $\xi \sim \mathcal{N}(0, \sigma_{\text{r}}^2)$. Formally, this turns membrane dynamics into a Langevin process. These perturbations add up over consecutive layers and can also accumulate over time due to recurrent interactions in the network. This introduces noise to the weight updates

that can impair learning performance. Due to their slow integration of inputs, traditional neuron models filter out this noise, but our prospective mechanism effectively removes this filter. We thus need an additional denoising mechanism, for which we again turn to biology: by introducing synaptic filtering with a short time constant $\tau^{\mathrm{s}}$, synaptic input is denoised before reaching the membrane; formally, $\boldsymbol{r} = \boldsymbol{\varphi}(\breve{\boldsymbol{u}}^{\mathrm{m}})$ is replaced by a low-pass-filtered $\overline{\boldsymbol{r}}^{\mathrm{s}}$ in the energy function (Eqn. 2, see also SI). This narrow filter also mitigates possible numerical instabilities, as discussed in Section 3.

Networks equipped with synaptic filters learn reliably even in the presence of significant noise levels (Fig. 4c). However, introducing this filter affects the quasi-instantaneity of computation in our networks, which then require longer input presentation times in the presence of synaptic filtering. Even so, these presentation times need only be "long" with respect to the characteristic time constant of relaxation mismatches – in this case, $\tau^{\mathrm{s}}$. Thus, for the typical scenario of white noise described above, minuscule $\tau^{\mathrm{s}}$ on and even below biological time scales can achieve effective denoising, without significantly affecting the advantages conferred by prospective coding. In conclusion, the demonstrated robustness of our model to spatial and temporal substrate imperfections introduced by simple, biologically inspired mechanisms, make it a promising candidate for implementation in analog physical systems.

# 7 Conclusion

We have introduced a new framework for inference and learning in physical systems composed of computational elements with finite response times. Our model rests on four simple axioms: prospective coding (Eqn. 1), neuronal mismatch energy (Eqn. 2), energy conservation under neuronal dynamics (Eqn. 3) and gradient descent on the energy under synaptic plasticity (Eqn. 5). In particular, incorporating the simple, biologically inspired mechanism of prospective coding allows us to avoid critical issues and scalability bottlenecks inherent to many current models of approximate BP in cortex. Furthermore, we have demonstrated robustness of the resulting implementations to substrate imperfections, a prerequisite for deployment in analog neuronal systems, both biological and artificial.

Our framework carries implications both for neuroscience and for the the design of neuromorphic hardware. The prospective mechanism described here would allow biological circuits to respond much faster than previously assumed. Furthermore, our framework suggests that both inference and learning take place on prospective, rather than instantaneous neuronal quantities. From a hardware perspective, this lifts the previously perceived limitations of slow analog components (as compared to digital ones) without relinquishing their power efficiency.

# 8 Broader impact

The physical interpretation of our model not only offers a biologically plausible implementation of continuous-time, continuously active neuro-synaptic dynamics, but also outlines a specific path towards mixed-signal (analog/digital) in-silico implementation. Even with existing technologies, such neuromorphic systems harbor the potential of surpassing their biological archetypes with respect to both energy efficiency and speed [30]. In conjunction with the inherent ability of our framework to support the processing of continuous data streams, the reduced power consumption of such devices makes them a prime candidate for the construction of autonomous, embodied learning machines.

While obviously beneficial for research and commercial deployment, one should be aware that improved training efficiency carries the risk of deploying ever more intransparent models [31]. Furthermore, along with its obvious benefits, improved, and in particular autonomous AI entails a plethora of far-reaching societal consequences that are the subject of ongoing academic and public debate [32]. Future progress hence needs to be considered carefully and responsibly, and, in particular, properly reflected in public policy.

On the path towards understanding and replicating biological intelligence, a corollary benefit for the scientific community may emerge. Modern machine learning requires enormous amounts of compute, thus largely limiting cutting-edge developments to institutions with the corresponding resources. The envisioned bio-inspired yet also bio-transcendent hardware systems have the potential to drastically increase the overall efficiency of custom-designed computational platforms. The resulting decrease in operation costs could thus significantly expedite the democratization of AI research.

# NeurIPS Paper Checklist

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) Have you read the ethics review guidelines and ensured that your paper conforms to them?
   Yes.
   (c) Did you discuss any potential negative societal impacts of your work?
   Yes, see Section 8.
   (d) Did you describe the limitations of your work?
   Yes, see Section 6.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results?
   Yes.
   (b) Did you include complete proofs of all theoretical results?
   Yes, see SI.
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.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)?
   Yes, see SI.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)?
   Yes.
   (d) Did you include the amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)?
   Yes, see SI.
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.
   (b) Did you mention the license of the assets?
   No since we included references for all datasets including the relevant information.
   (c) Did you include any new assets either in the supplemental material or as a URL? No.
   (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.

# References


1. Linnainmaa, S. Taylor Expansion of the Accumulated Rounding Error. *BIT Numerical Mathematics* **16,** 146–160 (1976).
2. Werbos, P. J. in *System Modeling and Optimization* 762–770 (Springer, 1982).
3. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. Learning Representations by Back-propagating Errors. *Nature* **323,** 533–536 (1986).
4. Ivakhnenko, A. G. & Lapa, V. G. *Cybernetic Predicting Devices* (1966).
5. Whittington, J. C. R. & Bogacz, R. An Approximation of the Error Backpropagation Algorithm in a Predictive Coding Network with Local Hebbian Synaptic Plasticity. *Neural Computation* **29,** 1229–1262 (2017).
6. Scellier, B. & Bengio, Y. Equilibrium propagation: Bridging the Gap between Energy-Based Models and Backpropagation. *Frontiers in Computational Neuroscience* **11,** 24 (2017).
7. Guerguiev, J., Lillicrap, T. P. & Richards, B. A. Towards Deep Learning with Segregated Dendrites. *eLife* **6,** e22901 (2017).
8. Sacramento, J., Ponte Costa, R., Bengio, Y. & Senn, W. Dendritic Cortical Microcircuits Approximate the Backpropagation Algorithm. *Advances in Neural Information Processing Systems* **31,** 8721–8732 (2018).
9. Song, Y., Lukasiewicz, T., Xu, Z. & Bogacz, R. Can the Brain do Backpropagation? – Exact Implementation of Backpropagation in Predictive Coding Networks. *Advances in Neural Information Processing Systems* **33,** 22566 (2020).
10. Millidge, B., Tschantz, A. & Buckley, C. L. Predictive Coding Approximates Backprop along Arbitrary Computation Graphs. arXiv: `2006.04182` (2020).
11. Millidge, B., Tschantz, A., Buckley, C. L. & Seth, A. Activation Relaxation: A Local Dynamical Approximation to Backpropagation in the Brain. arXiv: `2009.05359` (2020).
12. Choromanska, A. et al. Beyond Backprop: Online Alternating Minimization with Auxiliary Variables. *International Conference on Machine Learning* **97,** 1193–1202 (2019).
13. Bartunov, S. et al. Assessing the Scalability of Biologically-motivated Deep Learning Algorithms and Architectures. arXiv: `1807.04587` (2018).
14. Hodgkin, A. L. & Huxley, A. F. A Quantitative Description of Membrane Current and Its Application to Conduction and Excitation in Nerve. *The Journal of Physiology* **117,** 500–544 (1952).
15. Plesser, H. E. & Gerstner, W. Escape Rate Models for Noisy Integrate-and-Free Neurons. *Neurocomputing* **32-33,** 219–224 (2000).
16. Köndgen, H. et al. The Dynamical Response Properties of Neocortical Neurons to Temporally Modulated Noisy Inputs In Vitro. *Cerebral Cortex* **18,** 2086–2097 (2008).
17. Dold, D. et al. *Lagrangian dynamics of dendritic microcircuits enables real-time backpropagation of errors* in (2019).
18. Kungl, A. F., Dold, D., Riedler, O., Petrovici, M. A. & Senn, W. *Deep reinforcement learning for time-continuous substrates* in (2020).
19. Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J. & Hinton, G. Backpropagation and the Brain. *Nature Reviews Neuroscience* **21,** 335–346 (2020).
20. Widrow, B. & Hoff, M. E. *Adaptive switching circuits* tech. rep. (Stanford Univ Ca Stanford Electronics Labs, 1960).
21. LeCun, Y. et al. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation* **1,** 541–551. ISSN: 0899-7667. `https://doi.org/10.1162/neco.1989.1.4.541` (2021) (Dec. 1, 1989).
22. Lillicrap, T. P., Cownden, D., Tweed, D. B. & Akerman, C. J. Random Synaptic Feedback Weights Support Error Backpropagation for Deep Learning. *Nature Communications* **7,** 1–10 (2016).
23. LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* **86,** 2278–2324 (1998).
24. Baldi, P., Sadowski, P. & Whiteson, D. Searching for Exotic Particles in High-energy Physics with Deep Learning. *Nature Communications* **5,** 1–9 (2014).
25. Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images (2009).

26. Urbanczik, R. & Senn, W. Learning by the Dendritic Prediction of Somatic Spiking. *Neuron* **81,** 521–528 (2014).
27. Amit, Y. Deep Learning with Asymmetric Connections and Hebbian Updates. *Frontiers in Computational Neuroscience* **13,** 18 (2019).
28. Akrout, M., Wilson, C., Humphreys, P. C., Lillicrap, T. & Tweed, D. Deep learning without weight transport. arXiv: `1904.05391` (2019).
29. Thakur, C. S. et al. Large-scale Neuromorphic Spiking Array Processors: A Quest to Mimic the Brain. *Frontiers in Neuroscience* **12,** 891 (2018).
30. Marković, D., Mizrahi, A., Querlioz, D. & Grollier, J. Physics for Neuromorphic Computing. *Nature Reviews Physics* **2,** 499–510 (2020).
31. Bender, E. M., Gebru, T., McMillan-Major, A. & Shmitchell, S. *On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?* in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (2021), 610–623.
32. Futurium (European Commission). *Ethics Guidelines for Trustworthy AI* `https://ec.europa.eu/futurium/en/ai-alliance-consultation`.