Local plasticity rules can learn deep representations using self-supervised contrastive predictions

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning in the brain is poorly understood and learning rules that respect biological constraints, yet yield deep hierarchical representations, are still unknown. Here, we propose a learning rule that takes inspiration from neuroscience and recent advances in self-supervised deep learning. Learning minimizes a simple layer-specific loss function and does not need to back-propagate error signals within or between layers. Instead, weight updates follow a local, Hebbian, learning rule that only depends on pre- and post-synaptic neuronal activity, predictive dendritic input and widely broadcasted modulation factors which are identical for large groups of neurons. The learning rule applies contrastive predictive learning to a causal, biological setting using saccades (i.e. rapid shifts in gaze direction). We find that networks trained with this self-supervised and local rule build deep hierarchical representations of images, speech and video.

1 Introduction

Synaptic connection strengths in the brain are thought to change according to 'Hebbian' plasticity rules [Hebb, 1949]. Such rules are local and depend only on the recent state of the pre- and post-synaptic neurons [Sjöström et al., 2001, Caporale and Dan, 2008, Markram et al., 2011], potentially modulated by a third factor related to reward, attention or other high-level signals [Kuśmierz et al., 2017, Gerstner et al., 2018]. Therefore, one appealing hypothesis is that *representation learning in sensory cortices emerges from local and unsupervised plasticity rules*.

However, there seems to be a substantial gap between the rich hierarchical representations observed in the cortex and the representations emerging from local plasticity rules implementing principal/independent component analysis [Oja, 1982, Hyvärinen and Oja, 1998], sparse coding [Olshausen and Field, 1997, Rozell et al., 2008, Kohonen, 2012] or slow-feature analysis [Földiák, 1991, Wiskott and Sejnowski, 2002, Sprekeler et al., 2007]. Hebbian rules seem to struggle especially when 'stacked', i.e. when asked to learn deep, hierarchical representations.

This performance gap is puzzling because there are learning rules, relying on back-propagation (BP), that *can* build hierarchical representations similar to those found in visual cortex [Yamins et al., 2014, Zhuang et al., 2021]. Although some progress towards biologically plausible implementations of back-propagation has been made [Lillicrap et al., 2016b, Guergiuev et al., 2016, Sacramento et al., 2018, Payeur et al., 2020], most models rely on a neuron-specific error signal that needs to be transmitted by a separate error network [Crick, 1989, Lillicrap et al., 2020, Kunin et al., 2020]. Algorithms like contrastive divergence [Hinton, 2002], contrastive Hebbian learning [Xie and Seung, 2003] or equilibrium propagation [Scellier and Bengio, 2017] use local activity exclusively to calculate updates, but they require to wait for convergence to an equilibrium which is not appropriate for online learning from quickly varying inputs.

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

The present paper demonstrates that deep representations can emerge from a local, biologically plausible and unsupervised learning rule, by integrating two important insights from neuroscience: First, we focus on self-supervised learning from temporal data – as opposed to supervised learning from labelled examples – because this comes closest to natural data, perceived by real biological agents, and because the temporal structure of natural stimuli is a rich source of information. In particular, we exploit the typical, self-generated changes of gaze direction ('*saccades*') to distinguish input from a moving object during fixation from input arriving after a saccade towards a new object. Second, we notice that electrical signals stemming from segregated apical dendrites can modulate synaptic plasticity in biological neurons [Major et al., 2013, Körding and König, 2001].

Algorithmically, our approach takes inspiration from deep self-supervised learning algorithms that seek to contrast, cluster or predict stimuli in the context of BP [Van den Oord et al., 2018, Caron et al., 2018, Zhuang et al., 2019, Löwe et al., 2019]. Interestingly, Löwe et al. [2019] demonstrated that such methods even work if end-to-end BP is partially interrupted. We build upon this body of work and suggest the *Contrastive, Local And Predictive Plasticity* (CLAPP) model which avoids BP completely, yet still builds hierarchical representations.

2 Main goals and related work

In this paper, we propose a local plasticity rule that learns deep representations. To describe our model of synaptic plasticity, we represent a cortical area by the layer l of a deep neural network. The neural activity of this layer at time t is represented by the vector $\boldsymbol{z}^{t,l} = \rho(\boldsymbol{a}^{t,l})$, where ρ is a non-linearity and $\boldsymbol{a}^{t,l} = \boldsymbol{W}^l \boldsymbol{z}^{t,l-1}$ is the vector of the respective summed inputs to the neurons through their basal dendrites $\boldsymbol{W}^l$ (the bias is absorbed into $\boldsymbol{W}^l$). To simplify notation, we write the pre-synaptic input as $\boldsymbol{x}^{t,l} = \boldsymbol{z}^{t,l-1}$ and we only specify the layer index l when it is necessary.

Our plasticity rule exploits the fact that the temporal structure of natural inputs affects representation learning [Li and DiCarlo, 2008]. Specifically, we consider a scenario where an agent first perceives a moving object at time t (e.g. a flying eagle in Figure 1 a), and then spontaneously decides to change gaze direction towards another moving object at time $t + \delta t$ (e.g. *saccade* towards the elephant in Figure 1 a). We further assume that the visual pathway is 'self-aware' of saccades due to saccade-specific modulation of processing [Ross et al., 2001].

Following classical models of synaptic plasticity, we assume that a weight change is implemented by a biologically plausible, *Hebbian*-like learning rule [Hebb, 1949, Markram et al., 2011] which is local in space and time: updates ΔW_{ji}^t of a synapse, connecting neurons i and j, can only depend on the current activity of the pre-synaptic and post-synaptic neurons at time t, or slightly earlier at time $t - \delta t$, and one or several widely broadcasted modulating factors [Urbanczik and Senn, 2009, Gerstner et al., 2018].

Furthermore, we allow the activity of another neuron k to influence the weight update ΔW_{ji}, as long as there is an explicit connection W_{jk}^{pred} from k to j. The idea is to overcome the representational limitations of classical Hebbian learning by including dendritic inputs, which are thought to predict the future somatic activity [Körding and König, 2001, Urbanczik and Senn, 2014] and take part in the plasticity of the post-synaptic neuron [Larkum et al., 1999, Dudman et al., 2007, Major et al., 2013]. Hence we assume that each neuron j in a layer l may receive dendritic inputs $(\boldsymbol{W}^{\text{pred}} \boldsymbol{c}^{t,l})_j$ coming either from the layer above ($\boldsymbol{c}^{t,l} = \boldsymbol{z}^{t,l+1}$) or from lateral connections in the same layer ($\boldsymbol{c}^{t,l} = \boldsymbol{z}^{t,l}$). For simplicity, we assume that these dendritic inputs influence weight updates ΔW_{ji} of the post-synaptic neuron j but they do not influence its activity z_j^t.

Given these insights from neuroscience, we gather the essential factors that influence synaptic plasticity in the following learning rule prototype:

$$\Delta W_{ji} \quad \propto \quad \underbrace{\text{modulators}}_{\text{broadcast factors}} \cdot \underbrace{(\boldsymbol{W}^{\text{pred}} \boldsymbol{c}^{t_1})_j}_{\text{dentritic prediction}} \cdot \underbrace{\text{post}_j^{t_2} \cdot \text{pre}_i^{t_2}}_{\text{local-activity}} \,. \tag{1}$$

The modulating broadcast factors are the same for large groups of neurons, for example all neurons in the same area, or even all neurons in the whole network. $\text{post}_j^{t_2}$ and $\text{pre}_i^{t_2}$ are functions of the pre- and post- synaptic activities. At this point, we do not specify the exact timing between t_1 and t_2, as this will be determined by our algorithm in section 3.

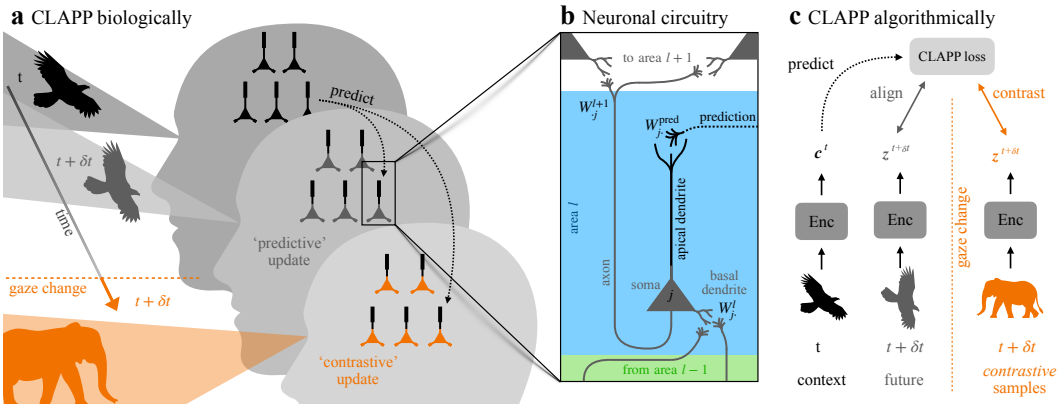

Figure 1: Contrastive, local and predictive plasticity (CLAPP). **a** Perceiving a moving object (e.g. an eagle) at times t and $t + \delta t$ leads to neural responses in the visual cortex. After a gaze change ('saccade'), a different object (elephant) is seen. **b** (zoom) At each time step, pyramidal neurons integrate input activity at the basal dendrites (matrix $\boldsymbol{W}^l$ of feedforward weights) and pass on their response to downstream areas ($\boldsymbol{W}^{l+1}$). At any point in time, neurons predict future neural responses through recurrent connections $\boldsymbol{W}^{\text{pred}}$. These inputs target the apical dendrites and modulate ongoing synaptic plasticity through 'predictive' updates. Information about a saccade is transmitted by a broadcast signal triggered at the moment of saccade initiation, which leads to 'contrastive' updates. As no external supervision or reward signals are provided, learning is self-supervised and local in time and space ('Hebbian'). **c** Algorithmically, an encoder network (Enc) produces a 'context' representation $\boldsymbol{c}^t$ at time t. Given $\boldsymbol{c}^t$, CLAPP tries to *predict* the encoding of the future input $\boldsymbol{z}^{t+\delta t}$. In case of a gaze change between t and $t + \delta t$, CLAPP seeks to keep the prediction as different as possible from the encoding of the upcoming *contrastive* sample.

Related work Many recent models of synaptic plasticity fit an apparently similar learning rule prototype [Roelfsema and Holtmaat, 2018, Lillicrap et al., 2016a, Nøkland, 2016, Nøkland and Eidnes, 2019, Lillicrap et al., 2020, Pozzi et al., 2020] if we interpret the top-down signals emerging from the BP algorithm as the dendritic signal. However, top-down error signals in BP are not directly related to the activity $\boldsymbol{c}^t$ of the neurons in the main network during processing of sensory input. Rather, they require a separate linear network mirroring the initial network and feeding back error signals (see Figure 2 a and Lillicrap et al. [2020]). Our model is fundamentally different, because in our case, the dendritic signal onto neuron j is strictly $(\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^t)_j$ which is a weighted sum of the main network activity and there is no need of a (linear) feedback network transmitting exact error values across many layers.

Moreover, we show in simulations in section 4, that the dendritic signal does not have to come from a layer above but that the prediction fed to layer l may come from the same layer. This shows that our learning rule works even in the complete absence of downward signaling from $l + 1$ to l. This last point is a significant difference to contrastive divergence [Hinton, 2002] and equilibrium propagation [Scellier and Bengio, 2017]: they also calculate updates using only activities of the main network, but they require (1) tuned top-down connections to propagate signals downwards in the network hierarchy [Kunin et al., 2020] and (2) convergence to an equilibrium state for each input.

Most previous learning rules which include global modulating factors interpret it as a reward prediction error [Schultz et al., 1997, Gerstner et al., 2018, Pozzi et al., 2020]. In this paper, we address self-supervised learning and view global modulating factors as broadcasting signals, modeling the self-awareness that something has changed in the stimulus (e.g. because of a saccade). In vision, we use a broadcast factor to model the strong, saccade-specific activity patterns identified throughout the visual pathway [Kowler et al., 1995, Leopold and Logothetis, 1998, Ross et al., 2001, McFarland et al., 2015]. In other sensory pathways, like audition, this broadcast factor may model attention signals arising when changing focus on a new input source [Fritz et al., 2007]. The main function of the broadcast factor in our model is to identify contrastive inputs, which avoids a common pitfall for self-supervised learning models: 'trivial' or 'collapsed' solutions, where the model produces a constant output, which is easily predictable, but useless for downstream tasks.

Our theory takes inspiration from the substantial progress seen in unsupervised machine learning in recent years and specifically from contrastive predictive coding (CPC) [Van den Oord et al., 2018].

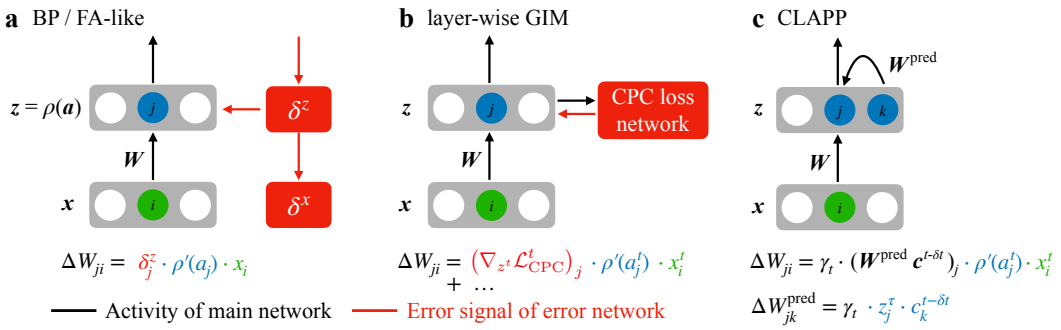

Figure 2: Comparison of weight updates **a** Networks trained with back-propagation (BP) or Feedback Alignment (FA)-like methods require separate error networks (red) for computing weight updates. **b** Layer-wise GIM, with one layer per gradient-isolated module, does not transmit error signals across layers (i.e. modules) but requires (1) the transmission of information other than the network activity (red) and (2) a perfect replay of negative samples. Thus, the resulting update computation needs a separate loss network and cannot be interpreted as a local learning rule. **c** Contrastive Local and Predictive Plasticity (CLAPP) calculates updates using locally and temporally available information: pre- and post-synaptic activity and predictive recurrent input onto the apical dendrite $\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^{t-\delta t}$. Global broadcasting factors γ_t modulate plasticity depending on the presence or absence of a saccade.

CPC trains a network (called *encoder*) to make *predictions* of its own responses to future inputs, while keeping this prediction as different as possible to its responses to *fake* inputs (*contrasting*). A key feature of CPC is that predicting and contrasting happens in latent space, i.e. on the output representation of the encoder network. This avoids modeling a generative model for perfect reconstruction of the input and all its details (e.g. green, spiky). Instead the model is forced to focus on extracting high-level information (e.g. cactus). In our notation, CPC evaluates a prediction $\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^t$ such that a score function $u_t^\tau = \boldsymbol{z}^{\tau\top}\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^t$ becomes larger for the true future $\tau = t + \delta t$ (referred to as positive sample) than for any other vector $\boldsymbol{z}^{t'}$ taken at arbitrary time points t' elsewhere in the entire training set (referred to as negative samples in CPC). This means, that the prediction should align with the future activity $\boldsymbol{z}^{t+\delta t}$ but not with the negative samples. Van den Oord et al. [2018] formalizes this as a softmax cross-entropy classification, which leads to the traditional CPC loss:

$$\mathcal{L}_{\text{CPC}}^t = -\log \frac{\exp u_t^{t+\delta t}}{\sum_{\tau \in \mathcal{T}} \exp u_t^\tau}\,, \tag{2}$$

where $\mathcal{T} = \left\{t^{t+\delta t}, t'_1 \ldots t'_N\right\}$ comprises the positive sample and N negative samples. The learned model parameters are the elements of the matrix $\boldsymbol{W}^{\text{pred}}$, as well as the weights of the encoder network. The loss function $\mathcal{L}_{\text{CPC}}^t$ is then minimized by stochastic gradient descent on these parameters using BP. Amongst the many recent variations of this learning scheme [He et al., 2019, Chen et al., 2020, Xiong et al., 2020], we focus here on CPC [Van den Oord et al., 2018] because of (1) its algorithmic simplicity and (2) its more local variant Greedy InfoMax [Löwe et al., 2019].

Greedy InfoMax (GIM) [Löwe et al., 2019] is a variant of CPC which makes a step towards local, BP-free learning: the main idea is to split the encoder network into a few gradient-isolated modules to avoid back-propagation between these modules. As the authors mention in their conclusion, "*the biological plausibility of GIM is limited by the use of negative samples and within-module back-propagation*". This within-module back-propagation still requires a separate feedback network to propagate prediction errors (Figure 2 a), but can be avoided in the most extreme version of GIM, where each gradient-isolated module contains a single layer (*layer-wise GIM*). However, the gradients of layer-wise GIM, derived from Equation 2, still cannot be interpreted as synaptic plasticity rules because the gradient computation requires (1) the transmission of information other than the network activity (see Figure 2 b), and (2) perfect memory to replay the negative samples $\boldsymbol{z}^{t'}$, as mentioned in the above quote (see Appendix A for details). Overall it is not clear how this weight update of layer-wise GIM could be implemented with realistic neuronal circuits. The CLAPP rule, proposed in this paper, solves the above mentioned implausibilities and allows a truly local implementation in space and time.

3 Theoretical results: contrastive, local and predictive plasticity

We now suggest a simpler contrastive learning algorithm which solves the issues encountered with layer-wise GIM and for which a gradient descent update is naturally compatible with the learning rule prototype from Equation 1. The most essential difference compared to CPC or GIM is, that we do not require the network to simultaneously access the true future activity $\boldsymbol{z}^{t+\delta t}$ and recall (or imagine) the network activity $\boldsymbol{z}^{t'}$ seen at some other time. Rather, we consider the naturalistic time-flow illustrated in Figure 1 a, where an agent fixates on a moving animal for a while and then changes gaze spontaneously. In this way, the prediction $\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^t$ is expected to be meaningful during fixation, but inappropriate right after a saccade. In our simulations, we model this by feeding the network with subsequent frames from the same sample (e.g. different views of an eagle), and then abruptly changing to frames from another sample (e.g. different views of an elephant).

Derivation of the CLAPP rule from a self-supervised learning principle Rather than using a loss function for multi-class classification to separate the true future from multiple negative samples, as in CPC, we consider here a binary classification problem: we interpret the score function $u_t^{t+\delta t} = \boldsymbol{z}^{t+\delta t\top}\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^t$ as the agent's guess whether it performed a fixation or a saccade. As mentioned in section 2, information about the presence or absence of a saccade between two time points is available in the visual processing stream and is modeled here by the variable $y^t = -1$ and $y^t = +1$, respectively. We interpret y^t as the label of a binary classification problem, characterized by the Hinge loss, and define the CLAPP loss as:

$$\mathcal{L}^t_{CLAPP} = \max\left(0, 1 - y^t \cdot u_t^{t+\delta t}\right) \quad \text{with} \quad \begin{cases} y^t = +1 & \text{for fixation} \\ y^t = -1 & \text{for saccade} \end{cases} \tag{3}$$

We now derive the gradients of Equation 3 with respect to the feedforward weights and show that gradient descent on this loss function is compatible with the learning rule prototype suggested in Equation 1. Since we chose to formalize the binary classification with a Hinge loss, the gradient vanishes when the classification is already correct: high score $u_t^{t+\delta t} > 1$ during fixation ($y^t = +1$), or a low score $u_t^{t+\delta t} < -1$ after a saccade ($y^t = -1$). Otherwise, it is $-\nabla u_t^{t+\delta t}$ during a fixation or $\nabla u_t^{t+\delta t}$ after a saccade. In the 'predicted layer' $\boldsymbol{z}$, i.e. the target of the prediction, let W_{ji} denote the feedforward weight from neuron i in the previous layer (with activity x_i^t) to neuron j, with summed input a_j^t and activity z_j^t. Similarly, in the 'predicting layer' $\boldsymbol{c}$, i.e. the source of the prediction, let $W^{\boldsymbol{c}}_{kl}$ denote the feedforward weight between the neuron l in the previous layer (with activity $x_l^{\boldsymbol{c},t}$) and neuron k, with summed input $a_k^{\boldsymbol{c},t}$ and activity c_k^t. We then find the gradients with respect to these weights as:

$$\frac{\partial \mathcal{L}^t_{CLAPP}}{\partial W_{ji}} = \pm(\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^t)_j \, \rho'(a_j^{t+\delta t}) \, x_i^{t+\delta t} \tag{4}$$

$$\frac{\partial \mathcal{L}^t_{CLAPP}}{\partial W^{\boldsymbol{c}}_{kl}} = \pm(\boldsymbol{W}^{\text{pred}\top}\boldsymbol{z}^{t+\delta t})_k \, \rho'(a_k^{\boldsymbol{c},t}) \, x_l^{\boldsymbol{c},t} \, , \tag{5}$$

where the sign is negative during fixation and positive after a saccade. To change these equations into online weight updates, we consider the gradient descent update delayed by δt, such that $\Delta W_{ji}^t = -\eta \frac{\partial \mathcal{L}^{t-\delta t}_{CLAPP}}{\partial W_{ji}}$, where η is the learning rate. Let us define a modulating factor $\gamma_t = y^t \cdot H^t$, where $y^t = \pm 1$ is a network-wide broadcast signal (self-awareness) indicating a saccade (-1) or a fixation ($+1$) and $H^t \in \{0, \eta\}$ is a layer-wide broadcast signal indicating whether the saccade or fixation was correctly classified as such. In this way, Equation 4 becomes a weight update which follows strictly the ideal learning rule prototype from Equation 1:

$$\Delta W_{ji}^t = \underbrace{\gamma_t}_{\text{broadcast factors}} \cdot \underbrace{(\boldsymbol{W}^{\text{pred}}\boldsymbol{c}^{t-\delta t})_j}_{\text{dendritic prediction}} \cdot \underbrace{\rho'(a_j^t) x_i^t}_{\text{local activity}} \, . \tag{6}$$

For the updates of the connections onto the neuron c_k^t, which emits the prediction rather than receiving it, our theory in Equation 5 requires the opposite temporal order and the transmission of

the information in the opposite direction: from $\boldsymbol{z}^t$ back to $\boldsymbol{c}^t$. Since connections in the brain are unidirectional [Lillicrap et al., 2016b], we introduce another matrix $\boldsymbol{W}^{\text{retro}}$ which replaces $\boldsymbol{W}^{\text{pred}\top}$ in the final weight update. Given the inverse temporal order, we interpret $\boldsymbol{W}^{\text{retro}}\boldsymbol{z}^t$ as a retrodiction rather than a prediction. We show in Appendix C, that using $\boldsymbol{W}^{\text{retro}}$ still minimises a CLAPP loss function as in Equation 3. The resulting weight update satisfies the learning rule prototype from Equation 1, as it can be written:

$$\Delta W_{kl}^{\boldsymbol{c},t} = \underbrace{\gamma_t}_{\text{broadcast factors}} \cdot \underbrace{(\boldsymbol{W}^{\text{retro}}\boldsymbol{z}^t)_k}_{\text{dendritic retrodiction}} \cdot \underbrace{\rho'(a_k^{\boldsymbol{c},t-\delta t})x_l^{\boldsymbol{c},t-\delta t}}_{\text{local activity}} . \tag{7}$$

The prediction and retrodiction weights, $\boldsymbol{W}^{\text{pred}}$ and $\boldsymbol{W}^{\text{retro}}$, respectively, are also plastic. By deriving the gradients of $\mathcal{L}_{CLAPP}^t$ with respect to $\boldsymbol{W}^{\text{pred}}$, we find an even simpler Hebbian learning rule for these weights:

$$\Delta W_{jk}^{\text{pred}} = \Delta W_{kj}^{\text{retro}} = \underbrace{\gamma_t}_{\text{broadcast factors}} \cdot \underbrace{z_j^t \cdot c_k^{t-\delta t}}_{\text{pre and post}} , \tag{8}$$

where neuron k in the predicting layer $\boldsymbol{c}$ is pre-synaptic (post-synaptic) and neuron j in the predicted layer $\boldsymbol{z}$ is post-synaptic (pre-synaptic) for the prediction weights W_{jk}^{pred} (retrodiction weights W_{kj}^{retro}). Note that the update rules for W_{jk}^{pred} and W_{kj}^{retro} are reciprocal, a method that leads to mirrored connections, given small enough initialisation [Burbank, 2015, Pozzi et al., 2020].

We emphasize that all information needed to calculate the above CLAPP updates (Equations 6 – 8) is spatially and temporally available, either as neuronal activity at time t, or as traces of recent activity $(t - \delta t)$ [Gerstner et al., 2018]. In order to implement Equation 6, the dendritic prediction has to be retained during δt. However, we argue that dendritic activity can outlast (50-100 ms) somatic neuronal activity (2-10 ms) [Major et al., 2013], which makes predictive input from several time steps in the past $(t - \delta t)$ available at time t.

Generalizations While the above derivation considers fully-connected feedforward networks, we apply analogous learning rules to convolutional neural networks (CNN) and recurrent neural networks (RNN). Analyzing the biological plausibility of the standard spatial weight sharing in CNNs is beyond the scope of the current work and we use spatial weight sharing for faster convergence. Furthermore, we allow gradients to flow through *single* static max-pooling layers.

To obtain local learning rules even for RNNs, we combine CLAPP with the e-prop theory [Bellec et al., 2020], which provides a biologically plausible alternative to BP through time: gradients can be propagated forward in time through the intrinsic neural dynamics of a neuron using eligibility traces. The propagation of gradients across recurrently connected units is forbidden and disabled. This yields a biologically plausible and self-supervised plasticity update in GRU units, as explained in Appendix C.

4 Empirical results

Building hierarchical representations We first demonstrate numerically, that CLAPP yields deep hierarchical representations, despite using a local plasticity rule compatible with Equation 1. We report here the results for $\boldsymbol{c}^{t,l} = \boldsymbol{z}^{t,l}$, i.e. the dendritic prediction in Equation 1 is generated from lateral connections and the representations in the same layer. We note, however, that we obtained comparable results with $\boldsymbol{c}^{t,l} = \boldsymbol{z}^{t,l+1}$ (i.e. the dendritic prediction is generated from one layer above), suggesting that top-down signaling is neither necessary for, nor incompatible with, our algorithm.

We first consider the STL-10 image dataset [Coates et al., 2011]. To simulate a time dimension in these static images, we follow Hénaff et al. [2019] and Löwe et al. [2019]: each image is split into 16×16 patches and the patches are viewed one after the other in a vertical order (one time step is one patch). Other hyper-parameters and data-augmentation are taken from Löwe et al. [2019], see Appendix B. We then train a 6-layer VGG-like [Simonyan and Zisserman, 2015] encoder (VGG-6) using the CLAPP rule (Equations 6 – 8). Training is performed on the unlabelled part of the STL-10 dataset for 300 epochs. We use 4 GPUs (`NVIDIA Tesla V100-SXM2 32 GB`) for data-parallel training, resulting in a simulation time of around 4 days per run.

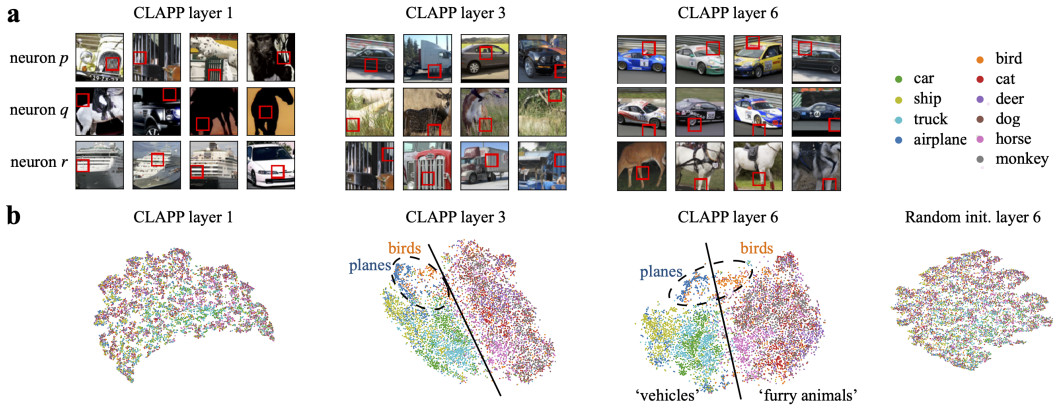

Figure 3: Hierarchical representations learned by CLAPP. **a** Red boxes in STL-10 images indicate patches that best activate a specific neuron (rows) in a network trained with CLAPP. Layer 1 extracts simple features like gratings or uniform patches, higher layers extract richer features like parts of objects. **b** 2-dimensional t-SNE projection of neuronal activities at different layers unveils increasing representational structure in higher layers (every dot represents one input image). Note that CLAPP has not seen any class labels during training.

In order to study how neuronal selectivity changes over layers, we select neurons randomly and show image patches which best activate these neurons the most (rows in Figure 3 a). As expected for a visual hierarchy, first-layer neurons (first column in Figure 3 a) are selective to horizontal or vertical gratings, or homogeneous colors. In the third layer of the network (second column), neurons start to be selective to more semantic features like grass, or parts of vehicles. Neurons in the last layer (third column) seem to be selective to specific parts of objects (e.g. the contact point of a road and a wheel).

To get an idea of the learned representation manifold, we visualise in Figure 3 b the encodings of the (labeled) STL-10 test set using the non-linear dimension reduction technique t-SNE [Van der Maaten and Hinton, 2008]. The representation in the first layer is mostly unrelated to the underlying class. In the third layer, a coherent clustering emerges yielding an almost perfect separation between furry animals and vehicles. The imperfection, that birds are positioned on the 'wrong' side of this separation, is corrected in the sixth layer: we can draw a straight line between vehicle and animal clusters, but the semantic proximity between birds and planes is preserved. This clustered representation is remarkable since the network has never seen class labels during CLAPP training and was never instructed to separate classes. The t-SNE point cloud obtained with the same architecture, but without training (Random init.), shows that a convolutional architecture alone does not yield semantic clustering.

To produce a more quantitative measurement of the quality of learned representations, we follow the methodology of Van den Oord et al. [2018] and Löwe et al. [2019]: we freeze the trained encoder weights and train a linear classifier to recognize the class labels from each individual layer (Figure 4). As expected for a deep representation, the classification accuracy increases monotonically with the layer number and only saturates at layers 5 and 6. The accuracies obtained with layer-wise GIM are almost indistinguishable from those obtained with CLAPP. It is only at the last two layers, that layer-wise GIM performs slightly better than CLAPP; yet GIM has multiple biologically implausible features that are removed by CLAPP.

As a further benchmark, we also plot the accuracies obtained with an encoder trained with greedy supervised training. This method trains each layer independently using a supervised classifier at each layer, without BP between layers, which results in an almost local update (see Löwe et al. [2019] and Appendix B). We find that accuracy is overall lower and saturates already at layer 4. On this dataset, with many more unlabelled than labelled images, greedy supervised accuracy is almost 10% below the accuracy obtained with CLAPP. Again, we see that a convolutional encoder alone does not yield deep representations, as performance decreases at higher layers for a randomly initialized encoder.

Comparing CPC and CLAPP Since CLAPP can be seen as a simplification of CPC (or GIM) we study four algorithmic differences between CPC and CLAPP individually. They are: (1) Gradients in CLAPP (layer-wise GIM) cannot flow from a layer to the next one, as opposed to BP in CPC, (2)

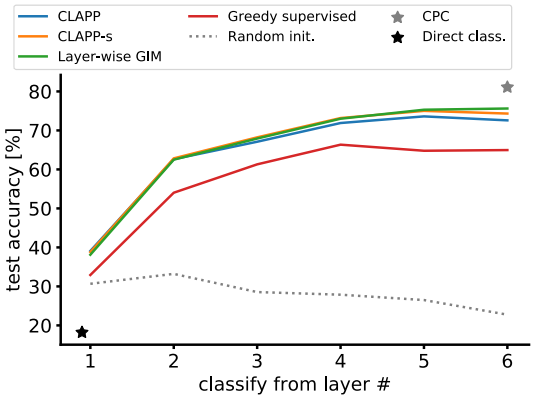

Figure 4: CLAPP stacks well: representations after stacking up to 5 layers increase performance of a linear classifier on STL-10, despite the local learning rule (blue and orange lines), while performance decreases for a randomly initialized convolutional network (dotted). Greedy supervised training (see Appendix B) also stacks, but already saturates at layer 4 and shows overall lower performance. Direct linear classification on image pixels (black star) and CPC performance after 6 layers (gray star) serve as upper and lower performance bounds, respectively.

CLAPP performs a binary comparison (fixation vs. saccade) with the Hinge loss, whereas CPC does multi-class classification with the cross entropy loss, (3) CLAPP processes a single input at a time, whereas CPC uses many positive and negative samples synchronously, and (4) we introduced $\boldsymbol{W}^{\text{retro}}$ to avoid the weight transport problem in $\boldsymbol{W}^{\text{pred}}$.

We first study features (1) and (2) but relax the constraints from (3) and (4). That means, in this paragraph, we allow a fixation and $N = 16$ synchronous saccades and set $\boldsymbol{W}^{\text{retro}} = \boldsymbol{W}^{\text{pred},\top}$. We refer to *Hinge Loss CPC* as the algorithm minimizing the CLAPP loss (Equation 3) but using end-to-end BP. *CLAPP-s* (for *synchronous*) applies the Hinge Loss to every layer, but with gradients blocked between layer. We find that the difference between the CPC loss and Hinge Loss CPC is less than 1%, see Table 1. In contrast, additional blocking of gradients between layers causes a performance drop of almost 5% for both loss functions. To investigate how gradient blocking influences performance, we conducted a series of simulations where we split the 6 layers of the network into two or three gradient isolated modules, exploring the transition from Hinge CPC to CLAPP-s. The performance drops monotonously but not catastrophically, as the number of gradient blocks increases (Table 1).

CLAPP's temporal locality allows the interpretation that an agent alternates between fixations and saccades, rather than perfectly recalling negative samples and processing synchronously, as required by CPC. To study the effect of temporal locality, we now apply features (2) and (3) and relax the constraints (1) and (4). The algorithm combining temporal locality and the CLAPP loss function is referred to as *time-local Hinge CPC*. We find that the temporal locality constraint decreases accuracy by 1.2% compared to Hinge Loss CPC. The last feature introduced for biological plausibility is using the matrix $\boldsymbol{W}^{\text{retro}}$ and we observe almost no difference in classification accuracy with this alternative (the accuracy decreases by 0.1%). Conversely, omitting the update in Equation 7 entirely, i.e. setting the retrodiction $\boldsymbol{W}^{\text{retro}} = 0$, compromises accuracy by 2% compared to vanilla Hinge Loss CPC.

When combining all features (1) to (4), we find that the fully local CLAPP learning rule leads to an accuracy of 73.6% at layer 5. We conclude from the analysis above, that the feature with the biggest impact on performance is (1): blocking the gradients between each layer. However, despite the performance drop caused by blocking the gradients, CLAPP still stacks well and leverages the depth of the network (Figure 4). All other features (2) - (4), introduced to derive a weight update compatible with our prototype (Equation 1), only caused a minor performance loss.

Applying CLAPP to speech and video We now demonstrate that CLAPP is applicable to other modalities like the LibriSpeech dataset of spoken speech [Panayotov et al., 2015] and the UCF-101 dataset containing short videos of human actions [Soomro et al., 2012]. To increase computational efficiency, we study CLAPP-s on speech and video. Based on the image experiments, we expect similar results for CLAPP, given enough time to converge. We use the same performance criteria as for STL-10 and summarize our results in Table 1, for details see Appendix B.

For the audio example, we use the same architecture as Van den Oord et al. [2018] and Löwe et al. [2019]: multiple temporal convolution layers and one recurrent GRU layer on top. We compare the performance of the algorithms on phoneme classification (41 classes) using labels provided by Van den Oord et al. [2018]. In this setting, layer-wise training lowers performance by only 0.4% for layer-wise GIM, and by 1.1% for CLAPP-s. Implemented as such, CLAPP-s still relies on BP through

Table 1: CLAPP performs best among methods that are local in space and time. Linear classification test accuracy [%] on STL-10, phone classification on LibriSpeech, and video human action recognition on UCF-101 using features from the encoder trained with different methods. On STL-10, performance degrades gracefully with the number of gradient-isolated modules in the VGG-6 encoder (at fixed number of encoder layers). Greedy supervised training uses BP in auxiliary classifier networks ('almost' local in space). For LibriSpeech, BP through time is used (can be avoided, see Appendix C). Values with * are taken from Löwe et al. [2019]. For simulation details, see Appendix B.

Method	local in ... space?	time?	STL-10	LibriSpeech	UCF-101
Chance performance			10.0	2.4	0.99
Random init.	✓	✓	21.8	27.7*	30.5
MFCC	✓	✓	-	39.7*	-
Greedy supervised	(✓)	✓	66.3	73.4*	-
Supervised	✗	✓	73.2	77.7*	51.5
CPC	✗	✗	81.1	64.3	35.7
Layer-wise GIM	✗	✗	75.6	63.9	41.2
Hinge Loss CPC (ours)	✗	✗	80.3	62.8	36.1
CLAPP-s (2 modules of 3 layers)	✗	✗	77.6	-	-
CLAPP-s (3 modules of 2 layers)	✗	✗	77.4	-	-
CLAPP-s (ours)	✓	✗	75.0	61.7	41.6
time-local Hinge CPC (ours)	✗	✓	79.1	-	-
CLAPP (ours)	✓	✓	73.6	-	-

time (BPTT) to train the GRU layer. Using CLAPP-s with biologically plausible e-prop [Bellec et al., 2020], instead of non-local BPTT, reduces performance by only 3.1 %, whereas omitting the GRU layer altogether compromises performance by 9.3 %, see Appendix C.

Applying CLAPP to videos is especially interesting because their temporal sequence of images perfectly fits the scenario of Figure 1 a. In this setting, we take inspiration from Han et al. [2019], and use a VGG-like stack of 2D and 3D convolutions to process video frames over time. On this task (101 classes), we found layer-wise GIM and CLAPP-s to achieve higher downstream classification accuracy than their end-to-end counterparts CPC and Hinge Loss CPC (see Table 1), in line with the findings on STL-10 in Löwe et al. [2019]. On the other hand, we found that CLAPP requires many more negative samples on videos than on STL-10 and LibriSpeech. Under the constraint of temporal locality in fully local CLAPP, this leads to prohibitively long convergence times in the current setup.

5 Discussion

We introduced CLAPP, a self-supervised and biologically plausible learning rule that yields deep hierarchical representations in neural networks. CLAPP integrates neuroscientific evidence on the dendritic morphology of neurons and takes the temporal structure of natural data into account. Algorithmically, CLAPP minimises a layer-wise contrastive predictive loss function and stacks well on different task domains like images, speech and video – despite the locality in space and time.

While the performance loss due to layer-wise training is a clear limitation of the current model, the stacking property is preserved and preliminary results suggest improved versions that stack even better (e.g. using adaptive receptive field sizes or predictions from higher layers). Note however, that CLAPP models self-supervised learning of cortical hierarchies and does *not* provide a general credit assignment method, such as BP. CLAPP in its current form is data- and compute-intensive, however, CLAPP runs on unlabelled data with quasi infinite supply, and is eligible for neuromorphic hardware, potentially decreasing energy consumption by orders of magnitude [Wunderlich et al., 2019].

Classical predictive coding models alter neural activity at inference time, e.g. by cancelling predicted future activity [Rao and Ballard, 1999, Keller and Mrsic-Flogel, 2018]. Here, we suggest a different, perhaps complementary, role of predictive coding in synaptic plasticity, where dendritic activity predicts future neural activity, but directly enters the learning rule [Körding and König, 2001, Urbanczik and Senn, 2014]. CLAPP currently does not model certain features of biological neurons, e.g. spiking activity or long range feedback, and requires neurons to transmit signals with precise value and timing. We plan to address these topics in future work.

References

G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass. A solution to the learning dilemma for recurrent networks. *Nature communications*, 11, 2020.

K. S. Burbank. Mirrored stdp implements autoencoder learning in a network of spiking neurons. *PLoS computational biology*, 11(12):e1004566, 2015.

N. Caporale and Y. Dan. Spike timing–dependent plasticity: a hebbian learning rule. *Annu. Rev. Neurosci.*, 31:25–46, 2008.

M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *Int. Conf. Mach. Learn.*, 2020. URL https://github.com/google-research/simclr.

A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

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

J. T. Dudman, D. Tsay, and S. A. Siegelbaum. A Role for Synaptic Inputs at Distal Dendrites: Instructive Signals for Hippocampal Long-Term Plasticity. *Neuron*, 56(5):866–879, dec 2007. ISSN 08966273. doi: 10.1016/j.neuron.2007.10.020.

P. Földiák. Learning Invariance from Transformation Sequences. *Neural Comput.*, 3(2): 194–200, 1991. ISSN 0899-7667. doi: 10.1162/neco.1991.3.2.194. URL http://www.mitpressjournals.org/doi/10.1162/neco.1991.3.2.194.

J. B. Fritz, M. Elhilali, S. V. David, and S. A. Shamma. Auditory attention - focusing the searchlight on sound. *Curr. Opin. Neurobiol.*, 17(4):437–455, aug 2007. ISSN 09594388. doi: 10.1016/j.conb.2007.07.011.

W. Gerstner, M. Lehmann, V. Liakoni, D. Corneil, and J. Brea. Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules. *Frontiers in neural circuits*, 12:53, 2018.

J. Guergiuev, T. P. Lillicrap, and B. A. Richards. Deep learning with segregated dendrites. *arXiv Prepr.*, 1610(00161):1–29, 2016. URL https://arxiv.org/abs/1610.00161.

T. Han, W. Xie, and A. Zisserman. Video representation learning by dense predictive coding. *arXiv preprint*, 2019.

K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *Conf. Comput. Vis. Pattern Recognit.*, pages 9726–9735. IEEE Computer Society, nov 2019. URL http://arxiv.org/abs/1911.05722.

D. O. Hebb. *The Organization of Behavior*. 1949. ISBN 0805843000.

O. J. Hénaff, A. Razavi, C. Doersch, S. M. A. Eslami, and A. van den Oord. Data-Efficient Image Recognition with Contrastive Predictive Coding. *arXiv Prepr.*, 2019.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

A. Hyvärinen and E. Oja. Independent component analysis by general nonlinear Hebbian-like learning rules. *Signal Processing*, 64:301–313, 1998.

G. B. Keller and T. D. Mrsic-Flogel. Predictive Processing: A Canonical Cortical Computation. *Neuron*, 100(2):424–435, 2018. ISSN 10974199. doi: 10.1016/j.neuron.2018.10.003. URL https://doi.org/10.1016/j.neuron.2018.10.003.

T. Kohonen. *Self-organizing maps*, volume 30. Springer Science & Business Media, 2012.

P. K. Körding and P. König. Neurons with Two Sites of Synaptic Integration Learn Invariant Representations. *Neural Comput.*, 13:2823–2849, 2001.

E. Kowler, E. Anderson, B. Dosher, and E. Blaser. The role of attention in the programming of saccades. *Vision Res.*, 35(13):1897–1916, 1995. ISSN 00426989. doi: 10.1016/0042-6989(94)00279-U. URL https://pubmed.ncbi.nlm.nih.gov/7660596/.

D. Kunin, A. Nayebi, J. Sagastuy-Brena, S. Ganguli, J. M. Bloom, and D. L. K. Yamins. Two Routes to Scalable Credit Assignment without Weight Symmetry. In *ICML*, 2020. URL http://arxiv.org/abs/2003.01513.

Ł. Kuśmierz, T. Isomura, and T. Toyoizumi. Learning with three factors: modulating Hebbian plasticity with errors. *Curr. Opin. Neurobiol.*, 46:170–177, oct 2017. ISSN 18736882. doi: 10.1016/j.conb.2017.08.020. URL http://dx.doi.org/10.1016/j.conb.2017.08.020.

M. E. Larkum, J. J. Zhu, and B. Sakmann. A new cellular mechanism for coupling inputs arriving at different cortical layers. *Nature*, 398(6725):338–341, 1999. ISSN 00280836. doi: 10.1038/18686.

D. A. Leopold and N. K. Logothetis. Microsaccades differentially modulate neural activity in the striate and extrastriate visual cortex. *Experimental Brain Research*, 123(3):341–345, 1998.

B. Li and J. J. DiCarlo. Unsupervised Natural Experience Rapidly Alters Invariant Object Representation in Visual Cortex. *Science (80-.).*, 321(September):1502–1508, 2008.

T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nat. Commun.*, 7:13276, 2016a. ISSN 2041-1723. doi: 10.1038/ncomms13276.

T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1):1–10, 2016b.

T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton. Backpropagation and the brain. *Nat. Rev. Neurosci.*, 21(6):335–346, 2020. ISSN 14710048. doi: 10.1038/s41583-020-0277-3.

S. Löwe, P. O'Connor, and B. S. Veeling. Putting An End to End-to-End: Gradient-Isolated Learning of Representations. *Advances in neural information processing systems*, 2019.

G. Major, M. E. Larkum, and J. Schiller. Active properties of neocortical pyramidal neuron dendrites. *Annual review of neuroscience*, 36:1–24, jul 2013. ISSN 1545-4126. doi: 10.1146/annurev-neuro-062111-150343. URL http://www.ncbi.nlm.nih.gov/pubmed/23841837.

H. Markram, W. Gerstner, and P. J. Sjöström. A history of spike-timing-dependent plasticity. *Frontiers in synaptic neuroscience*, 3:4, 2011.

J. M. McFarland, A. G. Bondy, R. C. Saunders, B. G. Cumming, and D. A. Butts. Saccadic modulation of stimulus processing in primary visual cortex. *Nature communications*, 6(1):1–14, 2015.

A. Nøkland. Direct feedback alignment provides learning in deep neural networks. In *Advances in neural information processing systems*, pages 1037–1045, 2016.

A. Nøkland and L. H. Eidnes. Training Neural Networks with Local Error Signals. *International Conference on Machine Learning*, 2019. doi: arXiv:1901.06656v1.

E. Oja. A simplified neuron model as a principal component analyzer. *J. Math. Biol.*, 1:267–273, 1982.

B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Res.*, 37(23):3311–3325, 1997. ISSN 00426989. doi: 10.1016/S0042-6989(97)00169-7.

V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

A. Payeur, J. Guerguiev, F. Zenke, B. Richards, and R. Naud. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *bioRxiv*, 2020.

I. Pozzi, S. M. Bohté, and P. R. Roelfsema. Attention-Gated Brain Propagation: How the brain can implement reward-based error backpropagation. In *NeurIPS*, 2020.

R. P. Rao and D. H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nat. Neurosci.*, 2(1), 1999. URL http://neurosci.nature.com.

P. R. Roelfsema and A. Holtmaat. Control of synaptic plasticity in deep cortical networks. *Nat. Rev. Neurosci.*, 19, 2018. doi: 10.1038/nrn.2018.6. URL www.nature.com/nrn.

J. Ross, M. C. Morrone, M. E. Goldberg, and D. C. Burr. Changes in visual perception at the time of saccades. *Trends in neurosciences*, 24(2):113–121, 2001.

C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural computation*, 20(10):2526–2563, 2008.

J. Sacramento, R. P. Costa, Y. Bengio, and W. Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. In *Advances in neural information processing systems*, pages 8721–8732, 2018.

B. Scellier and Y. Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience*, 11:24, 2017.

W. Schultz, P. Dayan, and P. R. Montague. A neural substrate of prediction and reward. *Science*, 275 (5306):1593–1599, 1997.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. URL http://www.robots.ox.ac.uk/.

P. J. Sjöström, G. G. Turrigiano, and S. B. Nelson. Rate, timing, and cooperativity jointly determine cortical synaptic plasticity. *Neuron*, 32(6):1149–1164, 2001.

K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

H. Sprekeler, C. Michaelis, and L. Wiskott. Slowness: An Objective for Spike-Timing–Dependent Plasticity? *PLoS Comput. Biol.*, 3(6):e112, 2007. ISSN 1553-734X. doi: 10.1371/journal.pcbi.0030112. URL http://dx.plos.org/10.1371/journal.pcbi.0030112.

R. Urbanczik and W. Senn. Reinforcement learning in populations of spiking neurons. *Nature neuroscience*, 12(3):250–252, 2009.

R. Urbanczik and W. Senn. Learning by the Dendritic Prediction of Somatic Spiking. *Neuron*, 81 (3):521–528, 2014. ISSN 08966273. doi: 10.1016/j.neuron.2013.11.030. URL http://dx.doi.org/10.1016/j.neuron.2013.11.030.

A. Van den Oord, Y. Li, and O. Vinyals. Representation Learning with Contrastive Predictive Coding. *arXiv Prepr.*, 2018.

L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

L. Wiskott and T. J. Sejnowski. Slow Feature Analysis : Unsupervised Learning of Invariances. *Neural Comput.*, 770:715–770, 2002.

T. Wunderlich, A. F. Kungl, E. Müller, A. Hartel, Y. Stradmann, S. A. Aamir, A. Grübl, A. Heimbrecht, K. Schreiber, D. Stöckel, et al. Demonstrating advantages of neuromorphic computation: a pilot study. *Frontiers in neuroscience*, 13:260, 2019.

X. Xie and H. S. Seung. Equivalence of backpropagation and contrastive hebbian learning in a layered network. *Neural computation*, 15(2):441–454, 2003.

Y. Xiong, M. Ren, and R. Urtasun. LoCo: Local Contrastive Representation Learning. *Advances in neural information processing systems*, 2020.

D. L. Yamins, H. Hong, C. F. Cadieu, E. A. Solomon, D. Seibert, and J. J. DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014.

C. Zhuang, A. L. Zhai, and D. Yamins. Local aggregation for unsupervised learning of visual embeddings. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6002–6012, 2019.

C. Zhuang, S. Yan, A. Nayebi, M. Schrimpf, M. C. Frank, J. J. DiCarlo, and D. L. Yamins. Unsupervised neural network models of the ventral visual stream. *Proceedings of the National Academy of Sciences*, 118(3), 2021.

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) Did you describe the limitations of your work? [Yes] See Discussion section 5
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] Our work proposes a new general training algorithm for neural networks. We believe that discussing potential positive and negative societal impacts of such a general algorithm is premature at this stage.
   (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 Appendix A and Appendix C
   (b) Did you include complete proofs of all theoretical results? [Yes] In Appendix A and Appendix C we provide full calculations and proofs for all theoretical results and claims in the paper.
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] All simulation details are given in section 4 and Appendix B. We will release all relevant code on github and provide URL and DOI.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We specify all training details in section 4 and Appendix B.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Our main simulations are compute and time intensive, which makes averaging over many random seeds costly. For some simulations on STL-10, LibriSpeech and UCF-101 we performed multiple runs and obtained consistent results, i.e. the variance does not influence the drawn conclusions.
   (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] We gave the compute requirements for the most costly simulations (STL-10) in section 4.
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] All used assets are open sourced. We cite the creators of all assets we use or mention in our paper at the point we mention them for the first time.

(b) Did you mention the license of the assets? [No] For all assets we refer to the original source where the respective license can be found.

(c) Did you include any new assets either in the supplemental material or as a URL? [No] We will release code as URL upon acceptance or reviewer request.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] All used assets were open sourced.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We used established data sets for which no labels revealing person identity are present.

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]