Biologically Plausible Deep Learning: A Critical Review

Robert T. Lange [*]
Einstein Center for Neurosciences Berlin
`robert.lange17@imperial.ac.uk`
`www.rob-lange.com`

1 Introduction

Backpropagation [7] provides a biologically implausible solution to the synaptic credit assignment problem in Deep Learning (see e.g. LeCun et al. [4], Schmidhuber [9]). While computational graphs and the chain rule successfully provide approximate gradients in deep layered structures, the mere empirical success does not imply that the brain is capable of implementing such a procedure.In this report we review different proposed solutions to such problems that render backpropagation biologically implausible. More specifically, we focus on an approach which implements local plasticity rules in a neural network architecture with dendritic compartments [2]. Previously it has been argued that such an architecture overcomes multiple points of critique while accomplishing similar strong results. Our robustness checks reveal that such a claim is not justified. Deep Learning (DL) has rightfully been the poster child of Machine Learning success in the 21st century. It has dominated competitions and research across all domains (computer vision, natural language processing, robotics as well as computational neuroscience).

Rob: Restructure Introduction!

Kriegeskorte/DiCarlo/RSA - DNNs outperform alternative frameworks in accurately reproducing activity patterns in cortex - What does this mean? Is DL just extremely flexible/expressive?

The report is structured as follows: First, we set the stage by introducing notation as well as fundamental problems with backpropagation. We briefly review the literature trying to solve such. Afterwards, we introduce the local compartmental learning paradigm introduced in Guerguiev et al. [2]. We show how a simplified model of plateau potentials generated at the apical shaft can be used to obtain local objective functions. Furthermore, we empirically analyze the proposed framework. We conduct experiments with different complexities of datasets as well as analyze the convergence behavior and hyperparameter robustness. Finally, discuss problems with the proposed approach and discuss recent contributions to the literature as well as ideas to solve such problems.

[*]This progress report was submitted as part of the final project of the "Models of Neural Systems" (Winter Term 2018/2019) computer practical course taught and organized by Prof. Richard Kempter (Bernstein Center for Computational Neuroscience, Berlin).

2 Credit Assignment in Deep Layered Structures

Arguably, Deep Learning's most simple layered architecture is the Multi-Layer Perceptron (MLP). A MLP composes layers $\{h_l\}_{l=1}^L$ of non-linear and affine transformations:

$$h_l := f(h_{l-1}; \theta_l) = \sigma_l(W_l h_{l-1} + b_l)$$

where $h_0 = x$ and $\theta_l = \{W_l, b_l\}$. In a classification task the final output layer h_L represents the output distribution over the possible labels. In order to train such a composition one has to define a loss function. A standard classification loss function is given by the cross-entropy between the actual labels distribution, $q(y|x)$, and the output distribution of the network, $p(y|h_L)$:

$$\mathcal{L}(h_L) = -\sum_y q(y|x) logp(y|h_L)$$

In order to train the parameters $\Theta := \{\theta\}_{l=1}^L$ of a network one makes use of powerful auto-differentiation tools and stochastic/batch gradient descent methods. More specifically, the classical backpropagation algorithm is based around the following equations:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta_l} &= \left(\frac{dh_l}{d\theta_l}\right)^T \frac{\partial \mathcal{L}}{\partial h_l} = \left(\frac{dh_l}{d\theta_l}\right)^T \left(\frac{dh_{l+1}}{dh_l}\right)^T \frac{\partial \mathcal{L}}{\partial h_{l+1}} \\
&= \left(\frac{dh_l}{d\theta_l}\right)^T \underbrace{\left(W_{l+1} diag\left(\sigma'_{l+1}(W_{l+1}h_l + b_{l+1})\right)\right)^T}_{:=\delta_{l+1}} \frac{\partial \mathcal{L}}{\partial h_{l+1}} \\
&= \left(\frac{dh_l}{d\theta_l}\right)^T \delta_{l+1}\delta_{l+2} \frac{\partial \mathcal{L}}{\partial h_{l+2}} = \cdots = \left(\frac{dh_l}{d\theta_l}\right)^T \left(\prod_{i=l+1}^{L} \delta_i\right) \frac{\partial \mathcal{L}}{\partial h_L}
\end{aligned}$$

In order to compute the gradient with respect to the parameters of a specific layer l, one has to first compute a forward pass of the network to obtain the hidden units $\{h_l\}_{l=0}^L$. Afterwards, one is able to compute a loss-based error signal, $e := \frac{\partial \mathcal{L}}{\partial h_L}$.

Rob: Add gradient wrt to data as well as basic SGD update rule

There are multiple problems rendering backpropagation biologically implausible.

1. Weight transport Problem: Downstream errors are fed back to upstream neurons via exact symmetric copy of downstream synaptic weight matrix - neuron "deep" within network has to have precise knowledge of all downstream synapses!
2. Global signed error signals: The error computed via forward propagation e has to be accessible at every layer. Physiologically, it is not clear how this can be achieved without a separate pathway.
3. Costly matrix transposition: Depending on the memory constraints, matrix transposition is a costly permutation operation. This might be circumvented by accessing the synaptic weights in a different order. Both options do not seem particularly plausible.
4. Feedback information propagation does not influence the activity of the feedforward network. This does not align with any neuropyhsiological observation.

Based on these observations, the neuroscience community has mostly dismissed the hypothesis of the brain being involved in something akin to deep learning. Still, there remain some efforts to overcome such weaknesses. In the following section we will review such approaches.

3 Literature Review

Feedback Alignment - Lillicrap et al. [6]: Introduce a first feedback alignment approach to solve the weight transport problem of backpropagation. Forward and backward weights are modeled separately - backward weights align with weight matrix transpose through learning process. Argument follows from positive definiteness of weight and random matrix product and a rotation line of thought.

* Possible solutions: 1. Retrograde transmission of info along axons - problem of slow timescale 2. Feedback of errors via second network - problem of symmetry assumption of feedforward and feedback connections 3. Here: Show that even fixed random connections can allow for learning - symmetry not required! Instead implicit dynamics lead to soft alignment between forward and backward weights

* Observations: * Feedback weights does not have to be exact: $B \approx W^T$ with $e^T W B e > 0$. rotation within 90 degrees of backprop signal. Learning speed depends on degree! * Alignment of B and W^T via adjustment of W (and B) possible

* Feedback alignment: * Modulator signal (error-FA) does not impact forward pass post-synaptic activity bu acts to alter plasticity at the forward synapses. * FA may encourage W to align with Moore-Penrose pseudoinverse of B - approximate functional symmetry * Inference vs learning - towards bayesian approaches * Use random weights in backward pass to deliver info to earlier layers * Still requires delivery if signed error via distinct pathway * Direct/Broadcast FA - connect feedback from output layer directly to all previous ones

* Experiments: * Learns linear function with single hidden layer - learning not slower than backprop * Sigmoid nonlinearity and classification task - altered function of post-synaptic activity - learned also to communicate info when 50* More layers 3 hidden layers - as well as backprop and making use of depth - froze layers and trained alternatingly - positive/negative phase? * Neurons that integrate activity over time and spike stochastically - synchronous pathways

* Possible Extensions: * Fixed spike thresholds/refractory period * Dropout/stochasticity

Lee et al. [5], Bartunov et al. [1] - Simplified Difference

General Content: Extent the target propagation algorithm to not require exact gradient at penultimate layer. Test alternative learning rules in more complicated settings (CIFAR/ImageNet) and differentiate between locally and fully connected architectures. Very good review but not much additional innovation. Behavioral + Physiological Realism

* Contrastive Hebbian Learning/Generalized Recirculation: Use top-down feedback connections to influence neural activity and differences to locally approx gradients * Positive/negative phase - need settling process - Likely to slow for brain to compute in real time

* Target Propagation: Trains distinct set of feedback connections defining backward activity propagation

Rob: Write paragraphs for Target Propagation

* Connections trained to approximately invert feedforward connections to compute target activites for each each layer by successive inversion - decoders * Reconstruction + Forward loss * Different target constructions * Vanilla TP: Target computation via propagation from higher layers' targets backwards through layer-wise inverses * Difference TP: Standard delta rule with additional stabilization from prev reconstruction error. Still needs explicit grad comp at final layer * Not tested on data more complex than MNIST

* Simplified Difference Target Propagation: Computation also for penultimate layer with help of correct label distribution - removes implausible gradient communication * Need diversity in targets - problem of low entropy of classification targets * Need precision in targets - poor inverse learned * Combat both problems/weakness of targets with help of auxiliary output resembling random features from penultilmate hidden layer * Parallel vs alternating inverse training - simultaneous more plausible

Rob: Write paragraphs for SD Target Propagation

* Experiments - Mostly negative results: 1. None of existing algos is able to scale up - Good performance MNIST/Somewhat reasonable on CIFAR/Horrible on ImageNet - Seems like weight-sharing is not key to success 2. Need for behavioral realism - judged by performance on difficult tasks 3. Hyperparameter Sensitivity * First fix "good" architeture and then optimize * Use hyperbolic tanh instead of ReLu - work better

4 Local Synaptic Learning Rules with Dendritic Integration

Körding and König [3] postulated that the brain might solve the credit assignment problem with the help of electrical segregation. Inspired by the physiological observation of multi-compartment pyramidal neurons, the derive a basic local learning rule which integrate top-down feedback at the distal apical dendrites with bottom-up sensory input from the basal dendrites.

Rob: Add figure of dendrites

Guerguiev et al. [2] extend this intuition to the assignment of credit in MLPs. More specifically, they formalize the idea of plateau potentials driving synaptic plasticity in pyramidal neurons. The apical compartment does not constant communicate with the somatic compartment. Instead, voltage-gated Ca^{2+} channels provide feedback information to the nucleus. Plateau potentials correspond to a long-lasting increase in membrane potential due to these events in the apical shaft.

A Hidden layer is described by a set of m three compartmental neurons:

Rob: Add text to the equations

$$\begin{aligned} \textit{Apical: } \mathbf{V^{0a}}(t) &= [V_1^{0a}(t), \dots, V_m^{0a}(t)] \\ \textit{Basal: } \mathbf{V^{0b}}(t) &= [V_1^{0b}(t), \dots, V_m^{0b}(t)] \\ \textit{Somatic: } \mathbf{V^{0}}(t) &= [V_1^{0}(t), \dots, V_m^{0}(t)] \end{aligned}$$

The dynamics of the somatic membrane potential are given by

- $\rightarrow$ 3 Compartment Hidden Layer: $\mathbf{V}^{0a}(t), \mathbf{V}^{0b}(t), \mathbf{V}^{0}(t) \in \mathbb{R}^m$

$$\tau \frac{dV_i^0(t)}{dt} = -V_i^0(t) + \frac{g_b}{g_l}\left(V_i^{0b}(t) - V_i^0(t)\right) + \frac{g_a}{g_l}\left(V_i^{0a}(t) - V_i^0(t)\right)$$

$$V_i^{0b} = \sum_{j=1}^{l} W_{ij}^0 s_j^{input}(t) + b_i^0 \text{ and } V_i^{0a} = \sum_{j=1}^{n} Y_{ij} s_j^1(t)$$

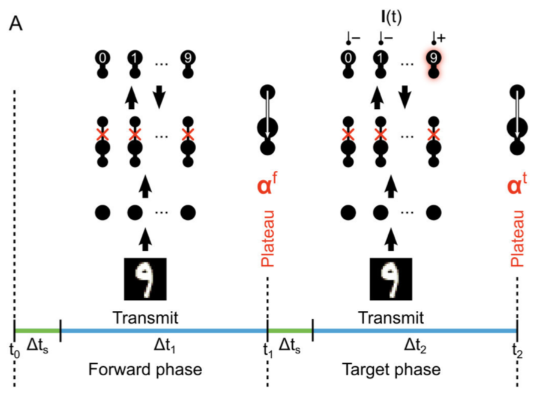
- $\rightarrow$ 2 Compartment Output Layer: $\mathbf{V}^{1b}(t), \mathbf{V}^{1}(t) \in \mathbb{R}^n$

$$\tau \frac{dV_i^1(t)}{dt} = -V_i^1(t) + \frac{g_d}{g_l}\left(V_i^{1b}(t) - V_i^1(t)\right) + I_i(t)$$

$$V_i^{1b} = \sum_{j=1}^{l} W_{ij}^1 s_j^0(t) + b_i^1$$

- $\rightarrow$ $s_j^{input}(t) = \sum_k \kappa(t - t_{jk}^{input})$ with κ response kernel
- $\rightarrow$ **Forward** $(t_0 + \Delta t_s \rightarrow t_1)$: $I_i(t) = 0, \ \forall i = 1, \dots, n$
  - At t_1: $\alpha_i^f = \sigma\left(\frac{1}{\Delta t_1}\int_{t_1-\Delta t_1}^{t_1} V_i^{0a}(t)dt\right)$
- $\rightarrow$ **Target** $(t_1 + \Delta t_s \rightarrow t_2)$: $I_k(t) = \phi_{max}$ for $y_{sample} = k$
  - At t_2: $\alpha_i^t = \sigma\left(\frac{1}{\Delta t_2}\int_{t_2-\Delta t_2}^{t_2} V_i^{0a}(t)dt\right)$

- $\rightarrow$ How is learning defined in such a model? Forward phase dynamics $\Leftrightarrow$ Target phase dynamics
- $\Rightarrow$ Somatic compartments generate Poisson process spikes:
  - Rates-of-fire: $\phi_i^l(t) = \phi_{max}\sigma(V_i^l(t))$
- $\Rightarrow$ Output Layer:
  - Target firing rates: $\phi_i^{1\star} = \frac{1}{\Delta t_2}\int_{t_1+\Delta t_s}^{t_2} \phi_i^1(t)dt$
  - Loss function: $L^1 = ||\phi_i^{1\star} - \bar{\phi}_i^{1f}||_2^2 = ||\frac{1}{\Delta t_2}\int_{t_1+\Delta t_s}^{t_2} \phi_i^1(t)dt - \frac{1}{\Delta t_1}\int_{t_0+\Delta t_s}^{t_1} \phi_i^1(t)dt||_2^2$



⇒ Hidden Layer:

- Target firing rates: $\phi_i^{0\star} = \bar{\phi}_i^{0f} + \alpha_i^t - \alpha_i^f$
- Loss function: $L^0 = ||\phi_i^{0\star} - \bar{\phi}_i^{0f}||_2^2 = ||\alpha^t - \alpha^f||_2^2$

⇒ Local error minimization via SGD

Rob: Write down gradients explicitly

5 Empirical Investigations

Scalability Across Datasets

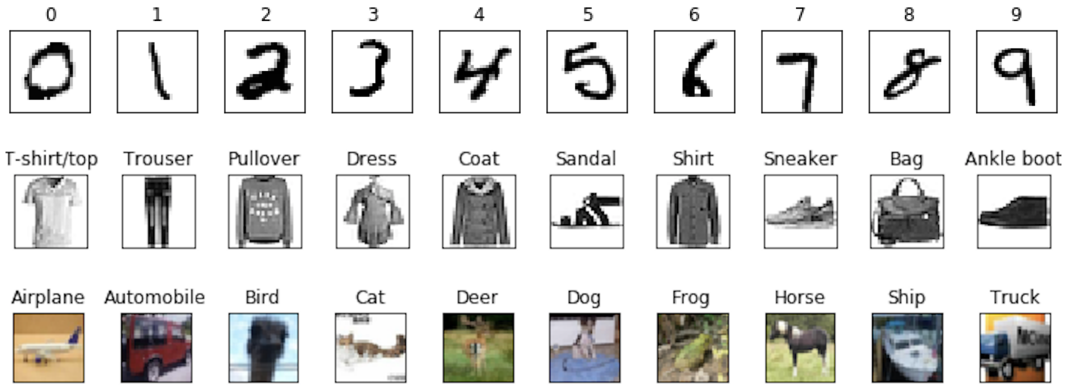


Figure 1: Illustration of the 10 different classes/labels of the analyzed datasets. **Top Row:** MNIST dataset. Data format: $70000 \times 1 \times 28 \times 28$. **Middle Row:** Fashion-MNIST dataset. Data format: $70000 \times 1 \times 28 \times 28$. **Bottom Row:** CIFAR-10 dataset. Data format: $60000 \times 3 \times 32 \times 32$. From top to bottom the intra-class variability/entropy increases significantly. We normalize the pixel values to lie within $[0, 1]$ and reshape the images into vector format (e.g. $X \in [0, 1]^{784}$) before training the classifiers. This helps dealing with erratic gradient behavior.

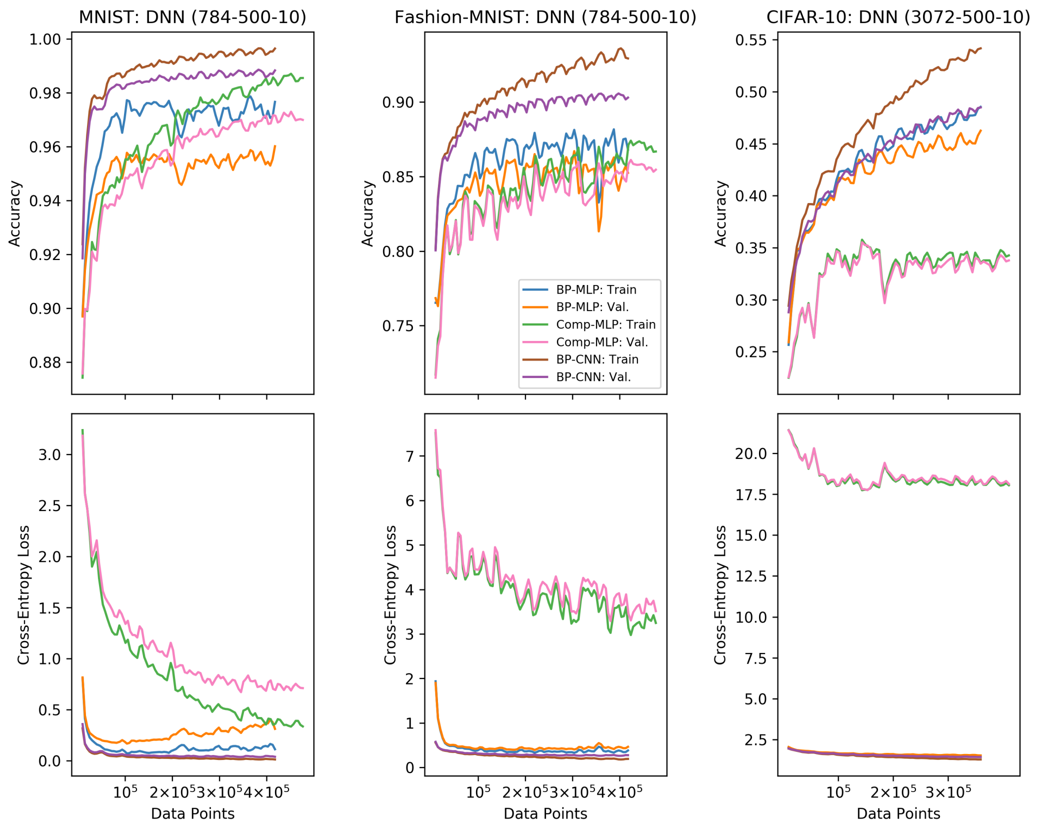


Figure 2: Illustration of the learning performance.

Rob: Add description of learning performance comparison.

Learning Dynamics

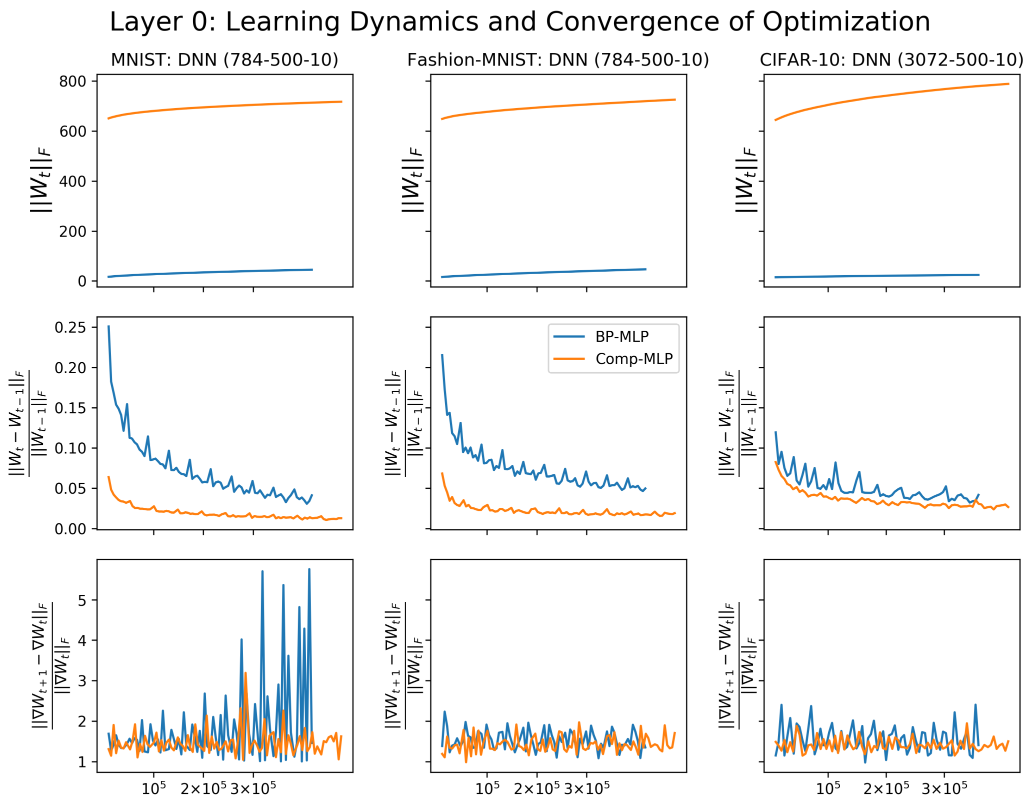


Figure 3: Illustration of the learning performance.

Rob: Add description of learning dynamics comparison.

Hyperparameter Robustness

Bayesian Optimization is a probabilistic technique commonly used to optimize the parameters of complex functions which are costly to evaluate. Computing cross-validated test accuracies of deep networks is one such costly function. Instead of randomly searching through the hyperparameter space, one approximates the loss function $\mathcal{L}^{k-fold}(\theta|X, y)$ with the help of a Gaussian Process (GP).

Rob: Add BO classic reference

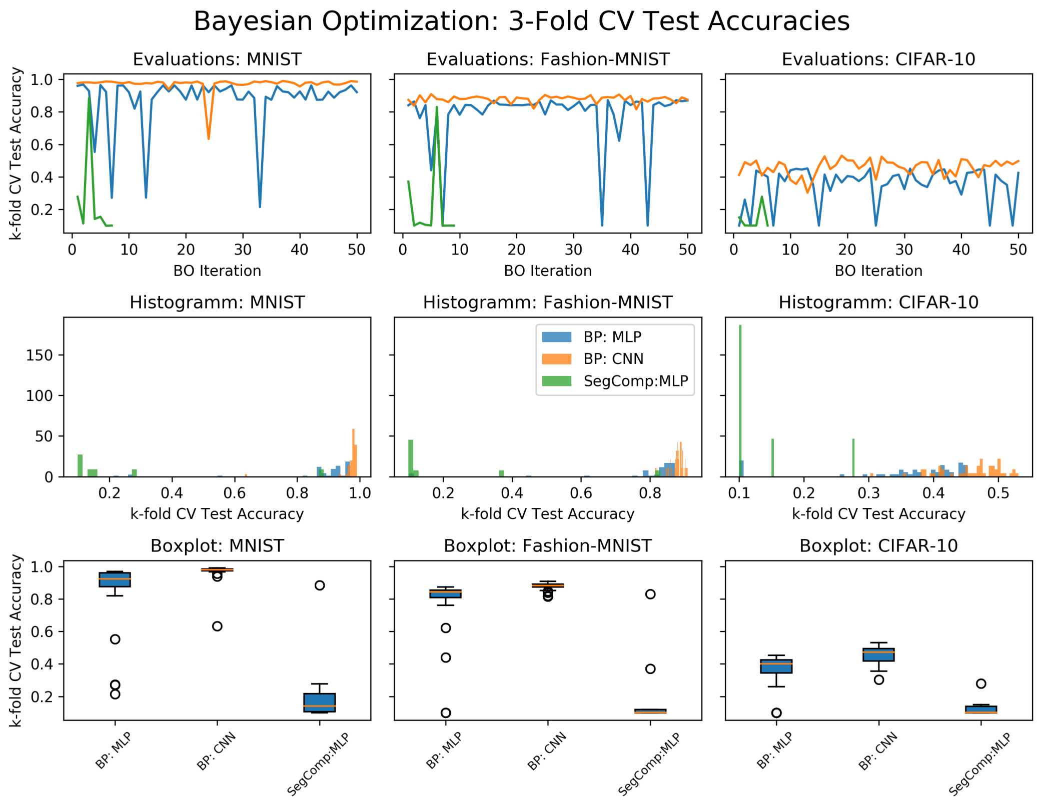


Figure 4: Illustration of the learning performance.

Rob: Add description of Baysian optimization comparison.

6 Outlook and Conclusion

Sacramento et al. [8] - Dendritic Microcircuits

General Content: MLP with simplified dendritic compartments learned in local PE plasticity fashion. No separate phases needed. Errors represent mismatch between pre input from lateral interneurons and top-down feedback. First cortical microcircuit approach. Analytically derive that such a setup/learning rule approximates backprop weight updates and proof basic performance on MNIST. Hypothesis: Pred errors are encoded at distal dendrites of pyramidal neurons - receive input from downstream neurons - in model: error arise from mismatch of lateral local interneuron inputs (SST - somatostatin) - Learning via local plasticity

Relation Guergiev: View apical dendrites as integration zones - temp difference between activity of apical dendrite in presence/absence of teaching input = error inducing plasticity at forward synapses. Used directly for learning b-u synapses without influencing somatic activity. HERE: apical dendrite has explicit error represnentation by sim integration of t-d excitation and lateral inhibition - No need for separate temporal phases - continuous operation with plasticity always turned on

* Main Results/Experiments: * Analytic derivation: Somatic MP at layer k integrate feedforward predictions (basal dendritic potentials) and backprop errors (apical dendritic potentials) * Analytic derivation: Plasticity rule converges to backprop weight change with weak feedback limit * Random/Fixed t-d weights = FA * Learned t-d weights minimizing inverse reconstruction loss = TP * Experiments: * Non-Linear regression task: Use soft rectifying nonlinearity as transfer fct - Tons of hyperparameters - injected noise current (dropout/regularization effect?) * MNIST - Deeper architectures: Use convex combination of learning/nudging * Different interneuron types (PV = parvalbumin-positive) - different types of errors (generative)

In this report we have empirically investigated the robustness and learning dynamics of an alternative learning rule in deep layered structures. We first reviewed and formalized the classical backpropagation algorithm. Afterwards, we put on computational neuroscience googles and highlighted several short-comings such as the weight transport problem as well the necessity to propagate signed errors. In Section 3 of this report we then introduced the methodology outlined by Guerguiev et al. [2] which intends to overcome such limitations. Inspired by dendritic compartments and information integration at different sites, the algorithm solves the weight transport problem. In Section 4 we reviewed more current approaches and compared their benefits and limitations. Thereby, we highlight the difference between behavioral and neurophysiological realism. Furthermore, we discuss the differences between learning and architecture complexity across the different approaches. Afterwards, we implement the approach by Guerguiev et al. [2] and compare model selection as well as hyperparameter robustness across different popular datasets. Our experiments reveal major performance decreases. This brings up the following question: Why should the brain implement a suboptimal **and** non-robust learning rule on a neurophysiological level? A simple answer to this is the flexibility that such an alternative architecture comes with.

Deep learning with ensemble multiplexing

Rob: Add ideas from Blake ICLR 2018 talk

Segregated compartments generate local targets that act as credit assignment signals in a physiologically plausible manner:

- Signal can be used to exploit depth in near-continuous time
- **Computational** Problems
  - $\rightarrow$ Huge hyperparameter space $\rightarrow$ most likely not robust!
- **Physiological** Problems
  - $\rightarrow$ How is the teaching signal internally generated?
  - $\rightarrow$ 2 global phases? - Length sampled from inverse Gaussian
  - $\rightarrow$ Stoch. gen. of plateau potentials - apical calcium spikes

$\Rightarrow$ Sacramento et al. [8]: Neocortical micro-circuits and inhibitory interneurons might act synchronizing.

Todo-List for Rob

References

[1] Bartunov, S., A. Santoro, B. Richards, L. Marris, G. E. Hinton, and T. Lillicrap (2018): "Assessing the scalability of biologically-motivated deep learning algorithms and architectures," in *Advances in Neural Information Processing Systems*, 9389–9399.

[2] Guerguiev, J., T. P. Lillicrap, and B. A. Richards (2017): "Towards deep learning with segregated dendrites," *ELife*, 6, e22901.

[3] Körding, K. P. and P. König (2001): "Supervised and unsupervised learning with two sites of synaptic integration," *Journal of computational neuroscience*, 11, 207–215.

[4] LeCun, Y., Y. Bengio, and G. Hinton (2015): "Deep learning," *nature*, 521, 436.

[5] Lee, D., S. Zhang, A. Biard, and Y. Bengio (2014): "Target Propagation," *CoRR*, abs/1412.7525.

[6] Lillicrap, T. P., D. Cownden, D. B. Tweed, and C. J. Akerman (2016): "Random synaptic feedback weights support error backpropagation for deep learning," *Nature communications*, 7, 13276.

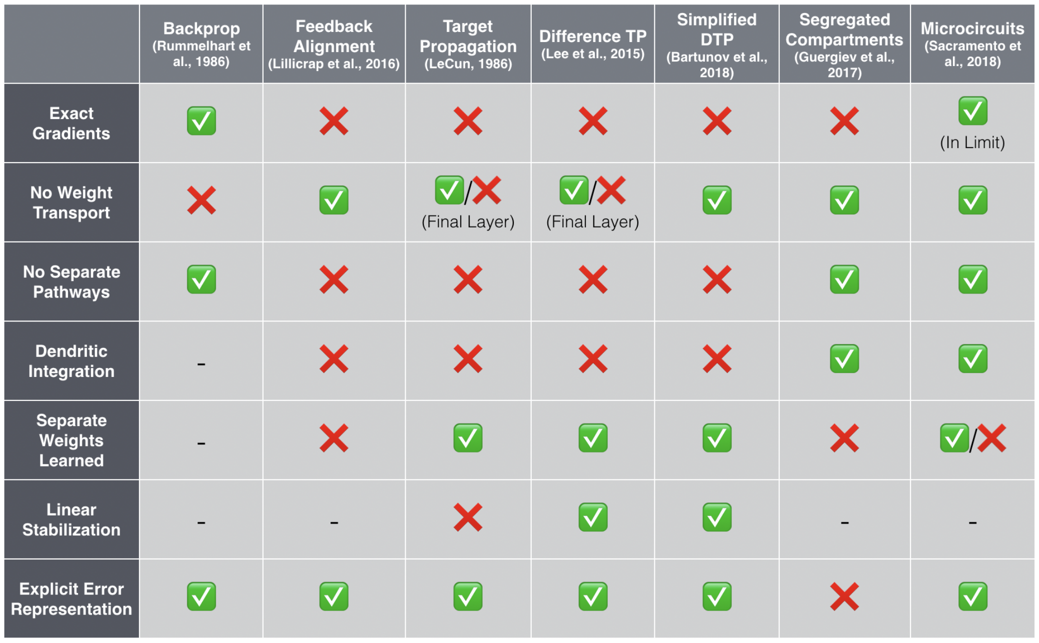
[7] Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986): "Learning representations by back-propagating errors," *Nature*, 323, 533.

[8] Sacramento, J., R. P. Costa, Y. Bengio, and W. Senn (2018): "Dendritic cortical microcircuits approximate the backpropagation algorithm," in *Advances in Neural Information Processing Systems*, 8735–8746.

[9] Schmidhuber, J. (2015): "Deep learning in neural networks: An overview," *Neural networks*, 61, 85–117.

Supplementary Material

A Literature Review



	Backprop (Rummelhart et al., 1986)	Feedback Alignment (Lillicrap et al., 2016)	Target Propagation (LeCun, 1986)	Difference TP (Lee et al., 2015)	Simplified DTP (Bartunov et al., 2018)	Segregated Compartments (Guergiev et al., 2017)	Microcircuits (Sacramento et al., 2018)
Exact Gradients	✅	❌	❌	❌	❌	❌	✅ (In Limit)
No Weight Transport	❌	✅	✅/❌ (Final Layer)	✅/❌ (Final Layer)	✅	✅	✅
No Separate Pathways	✅	❌	❌	❌	❌	✅	✅
Dendritic Integration	-	❌	❌	❌	❌	✅	✅
Separate Weights Learned	-	❌	✅	✅	✅	❌	✅/❌
Linear Stabilization	-	-	❌	✅	✅	-	-
Explicit Error Representation	✅	✅	✅	✅	✅	❌	✅

Figure 5: Literature Review.

Bayesian Optimization Hyperparameter Spaces

Rob: Add hyperparameter descriptions

MLP Hyperparameter Search Space:

Hyperparameter	Range	Description
Batchsize	Integer: $[10, 500]$	Number of data points in mini-batch
Learning Rate	Float: $[0.0001, 0.05]$	SGD learning rate
# Hidden Layers	Integer: $[1, 6]$	
# Hidden Layer Units	Integer: $[30, 500]$	

CNN Hyperparameter Search Space:

Hyperparameter	Range	Description
Batchsize	Integer: $[10, 500]$	Number of data points in mini-batch
Learning Rate	Float: $[0.0001, 0.05]$	SGD learning rate
# Hidden Layers	Integer: $[1, 6]$	
Channels	Integer: $[3, 64]$	
Kernels	Integer: $[2, 10]$	
Stride	Integer: $[1, 3]$	
Padding	Integer: $[1, 3]$	

Compartmental DNN Hyperparameter Search Space:

Hyperparameter	Range	Description
Sparse Feedback	Boolean	Dropout 80% of the weights
Conductances	Boolean	Conductances between soma and dendrites
Broadcast	Boolean	Feedback output to all hidden layers
Spiking Feedback	Boolean	
Spiking Feedforward	Boolean	
Symmetric Weights	Boolean	Enforce symmetry
Noisy Symmetric W.	Boolean	Add noise to symmetric weights
Update Feedback W.	Boolean	Sparse feedback weights
Apical conductances	Boolean	Attenuated conductances apical to soma
Weight Optimization	Boolean	Optimize initial weights
Feedback Bias	Boolean	Biases in feedback weights
dt	Float: [0, 1]	Integration time step
Spike Memory	Integer: $[0, 10]$	
Length Forward Train	Integer: $[2, 50]$	
Length Target Phase	Integer: $[2, 50]$	
Length Forward Test	Integer: $[50, 100]$	
λ_{max}	Float: $[0.2, 0.5]$	Max spike rate per time step
τ_s	Float: $[1, 5]$	Synaptic time constant
τ_L	Float: $[7, 13]$	Leak time constant
g_B	Float: $[0.3, 0.9]$	Basal conductance
g_A	Float: $[0.02, 0.1]$	Apical conductance
E_E	Float: $[5, 12]$	Excitatory Reversal Potential
E_I	Float: $[-12, -5]$	Inhibitory Reversal Potential
Forward Learning R.	Float: $[0.01, 0.05]$	SGD learning rate for forward weights
Backward Learning R.	Float: $[0, 0.05]$	SGD learning rate for backward weights
# Hidden Layers	Integer: $[1, 6]$	
# Hidden Layer Units	Integer: $[30, 500]$	

Notes on Reproduction

Please clone the repository `https://github.com/RobertTLange/Bio-Plausible-DeepLearning` and follow the instructions outlined below:

Rob: Update readme and make ready for submission

Repository Structure

```
Bio-Plausible-DeepLearning
+- workspace.ipynb: Main workspace notebook - Execute for replication
```

How to use this code

1. Clone the repo.

```
git clone https://github.com/RobertTLange/Bio-Plausible-DeepLearning
cd Bio-Plausible-DeepLearning
```

2. Create a virtual environment (optional but recommended).

```
virtualenv -p python BPDL
```

Activate the env (the following command works on Linux, other operating systems might differ):

```
source BPDL/bin/activate
```

3. Install all dependencies:

```
pip install -r requirements.txt
```

4. Run the main notebook:

```
jupyter notebook workspace.ipynb
```