A VARIATIONAL FRAMEWORK FOR LOCAL LEARNING WITH PROBABILISTIC LATENT REPRESENTATIONS

David Kappel[1], **Khaleelulla Khan Nazeer**[2], **Cabrel Teguemne Fokam**[1], **Christian Mayr**[2,3], **Anand Subramoney**[4]

[1] Institute for Neural Computation, Ruhr University Bochum, Germany
[2] Faculty of Electrical and Computer Engineering, and
[3] Centre for Tactile Internet with Human-in-the-Loop (CeTI),
Technische Universität Dresden, Germany
[4] Royal Holloway, University of London, United Kingdom
{david.kappel,cabrel.teguemnefokam}@ini.rub.de
{khaleelulla.khan_nazeer,christian.mayr}@tu-dresden.de
anand.subramoney@rhul.ac.uk

ABSTRACT

We propose a new method for distributed learning by dividing a deep neural network into blocks and introducing a feedback network that propagates information from the targets backward to provide auxiliary local losses. Forward and backward propagation can operate in parallel and with different sets of weights, addressing the problems of locking and weight transport. Our approach derives from a statistical interpretation of training that treats output activations of network blocks as parameters of probability distributions. The resulting learning framework uses these parameters to evaluate the agreement between forward and backward information. Error backpropagation is then performed locally within each block, leading to "block-local" learning. We present preliminary results on a variety of tasks and architectures, demonstrating state-of-the-art performance using block-local learning. These results provide a new principled framework for distributed asynchronous learning.

1 INTRODUCTION

The error backpropagation algorithm, that dominates today's deep learning methods, requires an alternation of interdependent forward and backward phases, each requiring sequential computation. This introduces a locking problem because each phase must wait for the other (Jaderberg et al., 2016). Furthermore, the two phases rely on the same weight matrices to compute updates, which makes it impossible to separate memory spaces, known as the weight transport problem (Grossberg, 1987; Lillicrap et al., 2014a). Locking and weight transport problems, make efficient parallelization of machine learning models extremely difficult, especially for low resource settings.

We propose a new method to address these problems to distribute a globally defined optimization algorithm across computing devices using only local updates. Our approach is derived from variational inference that provides auxiliary local targets through a separate feedback network that back-propagates information backward from the targets. Messages are communicated forward and backwards in parallel. Updates use local losses calculated using the targets provided by the feedback messages. In contrast to previous results, optimizing these local losses does not require a contrastive step where different positive and negative samples are propagated through the network. Within each block, conventional error backpropagation is performed locally ("block local"), thus mitigating the locking problem, and solving the weight transport problem. Our method provides a new principled method for distributing the training of networks across multiple computing devices.

Several models that used random feedback weights to provide local targets have been proposed previously, such as feedback alignment (Lee et al., 2015; Jaderberg et al., 2017; Lillicrap et al., 2020), target propagation (Lee et al., 2015; Meulemans et al., 2020; Frenkel et al., 2021) and related approaches (Akrout et al., 2019; Nøkland, 2016; Samadi et al., 2017). Some previous methods are also based on probabilistic

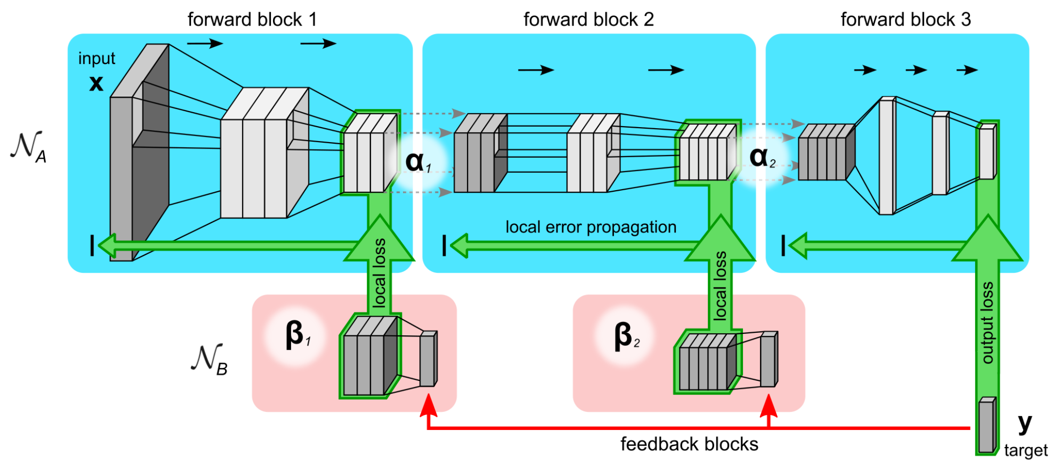

Figure 1: Block-local representations as learning signals. A deep neural network architecture $\mathcal{N}_A$ is split into multiple blocks (forward blocks) and trained on an auxiliary local loss. Targets for local losses are provided by a feedback network $\mathcal{N}_B$.

or energy-based cost functions, e.g. Contrastive learning (Chen et al., 2020; Oord et al., 2019; Xiong et al., 2020; Illing et al., 2021), equilibrium propagation (Scellier and Bengio, 2017) or forward propagation (Hinton, 2022; Zhao et al., 2023). Furthermore, (Belilovsky et al., 2019; Löwe et al., 2019; Nøkland and Eidnes, 2019) have used greedy local, block- or layer-wise optimization, achieved good results by combining different local losses. In contrast to these methods, we do not rely solely on local greedy optimization but provide a principled way to combine local losses with trained feedback connecitons without locking and weight transport across blocks and without contrastive learning.

2 A PROBABILISTIC FORMULATION OF BLOCK-LOCAL DISTRIBUTED LEARNING

We introduce a distributed learning framework that allows us to split a deep neural network into blocks that are trained locally and in parallel. The architecture is illustrated in Fig. 1. The neural network $\mathcal{N}_A$, that forward-propagates inputs $\mathbf{x}$ in the conventional fashion, is split into $N+1$ blocks. Intermediate outputs α_k, at each block k are matched against auxiliary targets β_k provided through feedback blocks $\mathcal{N}_B$. The feedback blocks only use the target vector $\mathbf{y}$ as input and can therefore be computed in parallel and independently. Error back-propagation is used locally within each forward block, i.e. block local learning (BLL). Backward blocks can be updated using convex optimization on each training batch, adding only small compute overhead. In experiments we demonstrate that BLL reaches competitive performance on smaller-scale training tasks making it an interesting alternative to end-to-end training for distributed low-energy applications, such as edge computing.

To arrive at this result we use a probabilistic interpretation of deep learning. The problem of minimizing any loss function $\mathcal{L}$ in a deep neural networks can be reformulated in terms of maximum likelihood, by defining the probability of data samples $(\mathbf{x},\mathbf{y})$ as probability[1] $p(\mathbf{y}|\mathbf{x}) \propto e^{-\mathcal{L}}$. The equivalent learning problem is to minimize the negative log-likelihood $\mathcal{L} = -\log p(\mathbf{y}|\mathbf{x})$ with respect to the network parameters $\boldsymbol{\theta}$ (Ghahramani, 2015). This probabilistic interpretation of deep learning can be used to define block-local losses and distribute learning across the network.

Splits of the network into blocks k can be interpreted in the maximum likelihood formalism by introducing latent variables $\mathbf{z}_k$ such that $p(\mathbf{y}|\mathbf{x}) = \mathbb{E}[p(\mathbf{y}|\mathbf{z}_k)p(\mathbf{z}_k|\mathbf{x})]$. In our framework, at no point does the network produce samples of the implicit random variables $\mathbf{z}_k$, they are introduced here only to conceptualize the mathematical framework. Instead, at block k, the network outputs the parameters of a probability distribution $\alpha_k(\mathbf{z}_k) = p(\mathbf{z}_k|\mathbf{x})$ (e.g., means and variances if α_k is Gaussian). The network thus translates $\alpha_{k-1} \to \alpha_k \to \ldots$ by outputting the statistical parameters of the conditional distribution $\alpha_k(\mathbf{z}_k)$ and taking the $\alpha_{k-1}(\mathbf{z}_{k-1})$ parameters as input.

[1]Learning the prior $p(\mathbf{x}) = \sum_{\mathbf{y}} p(\mathbf{x},\mathbf{y})$ over input samples $\mathbf{x}$ is often of lower relevance and is ignored here for brevity. Therefore, we focus on directly finding optimal parameters for $p(\mathbf{y}|\mathbf{x})$.

More specifically, the network implicitly calculates the following expectation

$$\alpha_k(\mathbf{z}_k) = p(\mathbf{z}_k|\mathbf{x}) = \mathbb{E}[p_k(\mathbf{z}_k|\mathbf{z}_{k-1})\alpha_{k-1}(\mathbf{z}_{k-1})] = f_k(\alpha_{k-1}, \boldsymbol{\theta}_k)\,, \tag{1}$$

with state transition probability p_k. Eq. (1) is an instance of the belief propagation algorithm for the chain of latent variables $\mathbf{z}_k, \mathbf{z}_{k+1}, ...$ Consequently, the whole network realizes a conditional probability distribution $p(\mathbf{y}|\mathbf{x})$ as in the maximum-likelihood formulation above, where $\mathbf{x}$ and $\mathbf{y}$ are network inputs and outputs, respectively. If all blocks have rich enough expressive power (e.g. sufficient number of hidden layers) an accurate representation of the mappings between distributions α_k can be represented in the block-local network parameters $\boldsymbol{\theta}_k$.

2.1 Distributed variational learning

We construct and use an upper bound $\mathcal{F}$ on the log-likelihood loss $\mathcal{L}$ for training the model. $\mathcal{F}$ uses the backward network $\mathcal{N}_B$ to introduce a variational distribution q. If constructed in a suitable way the variational upper bound can replace $\mathcal{L}$ with local losses. The variational posterior $\rho_k(\mathbf{z}_k)$ can be computed up to normalization by propagating forward messages $\alpha_k(\mathbf{z}_k)$ and feedback messages $\beta_k(\mathbf{z}_k)$ forward and backward through the network. Importantly, $\alpha_k(\mathbf{z}_k)$ and $\beta_k(\mathbf{z}_k)$ can have separated parameter spaces, which we denote by $\boldsymbol{\theta}_k$ and $\boldsymbol{\phi}_k$, respectively. We used a single linear layers to back-propagate β-messages. The variational posterior is then given by

$$\rho_k(\mathbf{z}_k) = q_k(\mathbf{z}_k|\mathbf{x}, \mathbf{y}) \quad \propto \quad p(\mathbf{z}_k|\mathbf{x})q(\mathbf{y}|\mathbf{z}_k) = \alpha_k(\mathbf{z}_k)\beta_k(\mathbf{z}_k)\,. \tag{2}$$

However, the true posterior distribution $p_k(\mathbf{z}_k|\mathbf{x}, \mathbf{y})$ for any latent variable $\mathbf{z}_k$ is not tractable in general because it would require us to invert the neural network. Instead we consider a tractable variational upper bound $\mathcal{F}$ to the log-likelihood loss $\mathcal{L}$

$$\mathcal{F} = -\log p(\mathbf{y}|\mathbf{x}) + \frac{1}{N}\sum_{k=1}^{N} \mathcal{D}_{KL}(q_k|p_k) \quad \geq \mathcal{L}\,, \tag{3}$$

where p_k and q_k are true and variational posterior distributions, and the Kullback-Leibler divergence $\mathcal{D}_{KL}(p|q) = \mathbb{E}_p[\log p - \log q]$ measures their mismatch. Conveniently, using the Markov property of the network blocks outlined above we can rewrite this form and identify an upper bound on $\mathcal{L}$, that only consists of block-local losses ℓ_k at all network blocks k, and that can be constructed using the forward and feedback networks $\mathcal{N}_A$ and $\mathcal{N}_B$. The local loss ℓ_k can be written as

$$\ell_k(p_k, \beta_k|\alpha_{k-1}) = \mathcal{D}_{KL}(q_k|\alpha_k) + \mathcal{H}(p_k|\alpha_{k-1})\,, \tag{4}$$

where the first term measures the mismatch between the variational posterior ρ_k and the forward message α_k. The second term is the entropy loss of the forward network $\mathcal{H}(p_k|\alpha_{k-1}) = -\mathbb{E}[\alpha_{k-1}(\mathbf{z}_{k-1})\log p_k(\mathbf{z}_k|\mathbf{z}_{k-1})]$.

The loss in Eq. (4) is local in the sense that it is completely determined by the information available at block k, i.e., the forward message from the previous block α_{k-1}, and the feedback β_k. Furthermore, the loss is local with respect to learning, i.e. it doesn't require global signals to be communicated to each block. In this sense, our approach differs from previous contrastive methods that need to distinguish between positive and negative samples. In our approach, any sample that passes through a block can be used directly for weight updating and is treated in the same way. Hence, an upper bound on the global loss $\mathcal{L}$ can be minimized by local greedy optimization of the losses (4). For linear backward networks, the loss (4) is convex such that optimal $\boldsymbol{\phi}_k$ can be directly computed over a mini-batch, adding little extra compute for feedback messages. Forward network parameters $\boldsymbol{\theta}_k$ are optimized by conventional error back-propagation locally and in parallel within each block.

2.2 Variational greedy block-local learning

To derive concrete losses and update rules for the forward and backward networks, we assume that α_k's and β_k's are channel-wise independent univariate Gaussian distributions with known variance for simplicity. However, the framework outlined above can easily be generalized to more complex probability distribution, such as the exponential family, if the variational posterior (2) can be expressed in closed form. A feed-forward DNN $\mathcal{N}_A : \mathbf{x} \to \mathbf{y}$, can be split into $N+1$ blocks by introducing implicit latent variables $\mathbf{z}_k : \mathbf{x} \to \mathbf{z}_k \to \mathbf{y}$, and generating the respective parameters of the probability distribution. Blocks can be separated after any arbitrary layer, but some splits may turn out more natural for a particular network

```
for x,y in the training data set do
    $\alpha_0 \leftarrow \mathbf{x}$
    for $1 \leq k \leq N$ do
        $\beta_k \leftarrow g_k(\mathbf{y}, \boldsymbol{\phi}_k)$                          ▷ Feedback
        $\alpha_k \leftarrow f_k(\alpha_{k-1}, \boldsymbol{\theta}_k)$                      ▷ Forward
        $\boldsymbol{\phi}_k \leftarrow \text{argmin}_{\boldsymbol{\phi}_k} \ell_k(p_k, \beta_k | \alpha_{k-1})$
        $\boldsymbol{\theta}_k \leftarrow \boldsymbol{\phi}_k + \eta \nabla_{\boldsymbol{\theta}_k} \ell_k(p_k, \beta_k | \alpha_{k-1})$
```

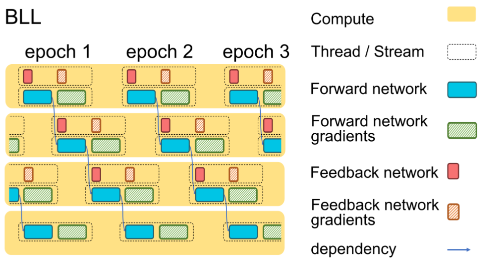

Figure 2: Left: Pseudo code of the BLL training algorithm. f_k and g_k are the transfer functions of forward and feedback blocks, respectively. The *for*-loops can be interleaved and run in parallel. Right: Timeline of execution for BLL.

	Fashion-MNIST ResNet-18	Fashion-MNIST ResNet-50	CIFAR-10 ResNet-50
BP	92.7	93.4	94.0
FA	87.9	83.1	70.3
Pred-Sim	93.9	**94.3**	92.4
BLL	**94.2**	**94.3**	**92.6**

Table 1: Classification accuracy (% correct) on vision tasks. BP: end-to-end backprop, FA: Feedback Alignment, Sim Loss: Local learning with similarity matching loss (Nøkland and Eidnes, 2019), BLL: block local learning. Best out of 5 test accuracies are reported.

architecture. If channel-wise forward and backward messages α_{kj} and β_{kj} are means of the univariate Gaussian distributions, where j is the channel index, then ρ_{kj}'s are simply $\frac{1}{2}(\alpha_{kj} + \beta_{kj})$.

In summary, the BLL algorithm is shown in Figure 2. The two *for*-loops can be interleaved and parallelized by pipelining the propagation of data samples through the network. Updates can be computed as soon as propagation through a given block is complete. There is no locking, since only the data labels are needed to compute the output of the backward network. Furthermore, there is no weight transport problem since parameter spaces are separated and updates computed only locally.

BLL shares many similarities with earlier methods. In particular, feedback alignment or target propagation that propagate feedback through random weights to create local learning objectives and can therefore be seen as a special case of BLL where feedback weights are kept fixed and the number of blocks equals the number of layers in the model. The loss term that emerges in BLL also shows some similarity with the predictive loss proposed in Nøkland and Eidnes (2019), but losses are derived here from a probabilistic variational framework and used to simultaneously learn the forward and feedback networks.

3 Results

3.1 Block-local learning of vision benchmark tasks

We evaluated the BLL algorithm on two vision tasks: Fashion-Mnist and CIFAR-10. Its performance is compared on ResNet18 and ResNet50 architectures with that of Backpropagation (BP), Feedback Alignment (Lillicrap et al., 2014b) (FA) and Local learning using similarity matching loss, Pred-Sim (Nøkland and Eidnes, 2019). The ResNet architectures were divided into four blocks for BLL and Pred-Sim. The splits were introduced after residual layers by grouping subsequent layers into blocks. We included the predictive loss as suggested in (Nøkland and Eidnes, 2019) as additional target in our BLL method. Gradient flow was stopped at block boundaries for all losses.

Group sizes in the blocks were (4,5,4,5) for ResNet-18 and (12,13,12,13) for ResNet-50. Backward networks for BLL were constructed as linear layers with label size as input and the output size equal to the number of channels in the corresponding ResNet block output. The kernels of ResNet-18/ResNet-50 used by FA architectures during backpropagation were fixed and uniformly initialised following the Kaiming He et al. (2015) initialisation method.

The results are summarized in Table 1, best test accuracies over 5 runs are shown for all methods. BLL performs on par or slightly better than Pred-Sim overall for all tasks and architectures. FA was outperformed by BLL, where the gap was becoming wider as the task and model complexity increased (Bartunov et al., 2018). BLL performs comparably to end-to-end backpropagation for Fashion-MNIST, but performance is somewhat degraded for CIFAR-10 using ResNet50, hinting at insufficient information being sent to the blocks through the linear feedback network. This becomes more evident as the number of local blocks is further increased (Fig. 3A).

3.2 Block-local transformer architecture for sequence-to-sequence learning

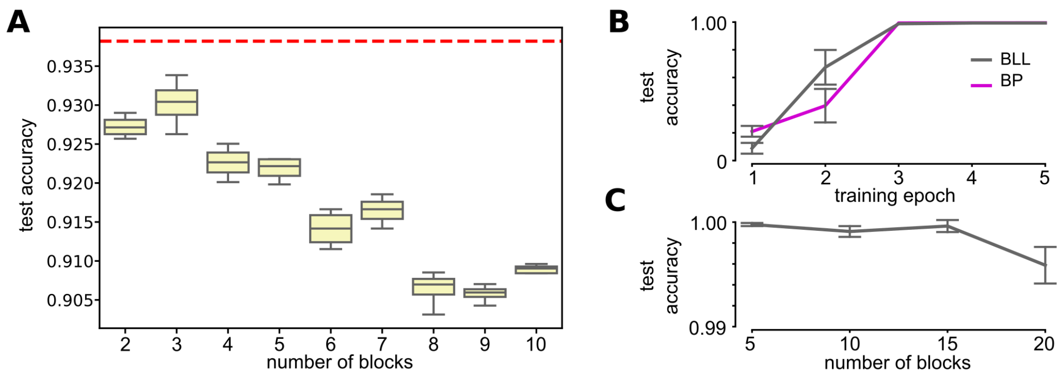

Figure 3: Scaling behavior of BLL. **A:** Test accuracy for different number of blocks for CIFAR-10 on ResNet-50. Dashed line shows BP baseline. **B:** Learning curves for S2S task. **C:** Test accuracy vs. number of blocks for S2S task. Error bars show standard deviations over independent runs.

Next, we demonstrate a proof-of-concept result for a sequence-to-sequence (S2S) task. We used a transformer model with 20 layers with a single attention head each. Block local losses were added after each layer and trained locally. The transformer was trained for 5 epochs on a simple S2S task, where a random permutation of numbers 0..9 was presented and had to be re-generated in reverse order. BLL achieves close-to perfect performance and a convergence speed that is comparable to that of end-to-end BP on this task. Fig. 3 B shows learning curves of BLL and BP. Both algorithms converge after around 3 epochs to nearly 100% accuracy. BLL also achieved good performance for a wide range of network depths. Fig. 3 C shows the performance after 5 epochs for different transformer architectures. Using only 5 transformer blocks yields performance of around 99.9% (average over five independent runs). The test accuracy on this task for the 20 block transformer was 99.6%. These first results suggest that BLL is in principle applicable to transformer architectures and S2S tasks.

4 Discussion

This work addresses an important open problem of modern ML: How can ML models be efficiently distributed and horizontally scaled over many compute nodes for training models too large to fit on one node. Doing so allows us to train ML models more efficiently and to distribute computation over many smaller, energy-efficient devices rather than a single large compute node. This makes our method especially well suited for new energy efficient hardware for ML, such as edge devices. We have demonstrated a probabilistic framework for rigorously defining block-local losses for deep architectures.

Our initial results suggest that this new method performs on par or slightly better than previous related block-local learning approaches for small-scale tasks. Depending on the task, performance begins to degrade when many splits are introduced into the deep network, suggesting that local losses may not provide sufficient feedback information in these cases. However, the theoretical framework presented here is flexible and allows the introduction of complex, multi-layer feedback networks if the convexity of the model is abandoned. Preliminary results for sequence-to-sequence tasks show the potential to apply the method to a wider range of architectures and tasks, but further experimental evidence with complex large-scale transformer models is needed. In summary, we believe that the proposed framework provides an interesting direction for exploring new models for block-local parallel training of deep networks with many potential applications in energy-constrained, distributed settings.

Acknowledgments

We acknowledge the use of Fenix Infrastructure resources, which are partially funded from the European Union's Horizon 2020 research and innovation programme through the ICEI project under the grant agreement No. 800858. The authors gratefully acknowledge the GWK support for funding this project by providing computing time through the Center for Information Services and HPC (ZIH) at TU Dresden. DK and CTF are funded by the German Federal Ministry of Education and Research (BMBF) within the project EVENTS (16ME0733). KKN is funded by the German Federal Ministry of Education and Research (BMBF) within the KI-ASIC project (16ES0996). CM receives funding from the German Research Foundation (DFG, Deutsche Forschungsgemeinschaft) as part of Germany's Excellence Strategy – EXC 2050/1 – Project ID 390696704 – Cluster of Excellence "Centre for Tactile Internet with Human-in-the-Loop" (CeTI) of Technische Universität Dresden. DK would like to thank Laurenz Wiskott and Sen Cheng for institutional support.

References

Mohamed Akrout, Collin Wilson, Peter Humphreys, Timothy Lillicrap, and Douglas B Tweed. Deep learning without weight transport. *Advances in neural information processing systems*, 32, 2019.

Sergey Bartunov, Adam Santoro, Blake A. Richards, Luke Marris, Geoffrey E. Hinton, and Timothy P. Lillicrap. Assessing the scalability of biologically-motivated deep learning algorithms and architectures. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 9390–9400, Red Hook, NY, USA, 2018. Curran Associates Inc.

Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Greedy layerwise learning can scale to ImageNet. In *Proceedings of the 36th International Conference on Machine Learning*, pages 583–593. PMLR, 2019. URL `https://proceedings.mlr.press/v97/belilovsky19a.html`.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Charlotte Frenkel, Martin Lefebvre, and David Bol. Learning without feedback: Fixed random learning signals allow for feedforward training of deep neural networks. *Frontiers in Neuroscience*, 15, 2021. ISSN 1662-453X. URL `https://www.frontiersin.org/articles/10.3389/fnins.2021.629892`.

Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553): 452–459, 2015.

Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63, 1987.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

Bernd Illing, Jean Ventura, Guillaume Bellec, and Wulfram Gerstner. Local plasticity rules can learn deep representations using self-supervised contrastive predictions. In *Advances in Neural Information Processing Systems*, volume 34, pages 30365–30379. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/feade1d2047977cd0cefdafc40175a99-Abstract.html`.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled Neural Interfaces using Synthetic Gradients. *arXiv:1608.05343 [cs]*, August 2016. URL `http://arxiv.org/abs/1608.05343`.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1627–1635. PMLR, 2017. URL `https://proceedings.mlr.press/v70/jaderberg17a.html`.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15*, pages 498–515. Springer, 2015.

Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random feedback weights support learning in deep neural networks. *arXiv:1411.0247 [cs, q-bio]*, November 2014a. URL http://arxiv.org/abs/1411.0247.

Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random feedback weights support learning in deep neural networks. *arXiv preprint arXiv:1411.0247*, 2014b.

Timothy P. Lillicrap, Adam Santoro, Luke Marris, Colin J. Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020. ISSN 1471-0048. doi: 10.1038/s41583-020-0277-3. URL https://www.nature.com/articles/s41583-020-0277-3.

Sindy Löwe, Peter O' Connor, and Bastiaan Veeling. Putting an end to end-to-end: Gradient-isolated learning of representations. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/851300ee84c2b80ed40f51ed26d866fc-Abstract.html.

Alexander Meulemans, Francesco Carzaniga, Johan Suykens, João Sacramento, and Benjamin F Grewe. A theoretical framework for target propagation. *Advances in Neural Information Processing Systems*, 33:20024–20036, 2020.

Arild Nøkland. Direct Feedback Alignment Provides Learning in Deep Neural Networks. *arXiv:1609.01596 [cs, stat]*, September 2016. URL http://arxiv.org/abs/1609.01596.

Arild Nøkland and Lars Hiller Eidnes. Training neural networks with local error signals. In *International conference on machine learning*, pages 4839–4850. PMLR, 2019.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019. URL http://arxiv.org/abs/1807.03748.

Arash Samadi, Timothy P. Lillicrap, and Douglas B. Tweed. Deep Learning with Dynamic Spiking Neurons and Fixed Feedback Weights. *Neural Computation*, 29(3):578–602, January 2017. ISSN 0899-7667. doi: 10.1162/NECO_a_00929. URL https://doi.org/10.1162/NECO_a_00929.

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

Yuwen Xiong, Mengye Ren, and Raquel Urtasun. LoCo: Local contrastive representation learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 11142–11153. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/7fa215c9efebb3811a7ef58409907899-Abstract.html.

Gongpei Zhao, Tao Wang, Yidong Li, Yi Jin, Congyan Lang, and Haibin Ling. The cascaded forward algorithm for neural network training. *arXiv preprint arXiv:2303.09728*, 2023.