Empirical Study on Biologically Plausible Initialization: ZerO Initialization With SoftHebb

Minh Le, Sarah Xu, Kai Zhuang
CSC413 Neural Networks & Deep Learning Project

Abstract

Traditional neural networks often employ random weight initialization, yet recent works have demonstrated that deterministic schemes, such as initializing weights with zeros and ones, can also yield robust outcomes. Our work explores the application of the ZerO initialization method within the context of biologically plausible Hebbian-based learning networks, specifically the multilayer SoftHebb algorithm, which operates without targets or error signals. ZerO initialization, which sets initial weights to either 0 or 1, not only avoids training degeneracy and accuracy loss, but also has implicit biological plausibility: human brain neurons fire completely (1) or not at all (0). By integrating the ZerO scheme into SoftHebb, we investigate whether it could maintain the learning efficacy of SoftHebb while enhancing its biological plausibility. We conducted a series of experiments across different datasets including MNIST and CIFAR-10 modifying various hyperparameters to reach an accuracy of 96.07% for MNIST.

Our code is available at: `https://github.com/Lemiex/ZerOSoftHebb`

1 Problem Definition

The dominant training algorithm in today's artificial intelligence landscape is Backpropagation, but it lacks biological plausibility and significantly falls short in terms of the brain's efficiency and robustness Patterson et al. [2021]. While backpropagation relies heavily on labeled data to calculate error signals, most data in the world is unlabled, which biological intelligence takes as input and employs in meaningful ways. Considering the profound impact of neurobiology on the development of AI, we believe biologily-inspired alternatives holds the potential for crucial advancements in deep learning.

One biologically plausible self-organizing deep learning algorithm we explore is multilayer SoftHebb, an algorithm proposed by Moraitis et al. [2022] and thoroughly experimented by Journé et al. [2023]. The SoftHebb network is distinctive in that it does not rely on the traditional backpropagation mechanism, thereby circumventing the biological implausibility and computational inefficiencies associated with the weight transport problem and non-local plasticity. Instead, it utilizes a local learning rule grounded in the Hebbian theory, which posits that the efficacy of synaptic transmission is increased when both the presynaptic and postsynaptic neurons are active simultaneously Oja [1982].

A critical factor in training any neural network is the initialization of weights, which influences learning dynamics, convergence rates, and performance of the trained model. Traditional approaches commonly rely on random weight initialization, which, while effective, requires careful management of the weight variance to avoid issues such as exploding or vanishing gradients as the network depth increases. This practice, although standard, introduces complexities, especially in the scalability and reproducibility of training deep neural networks. At the very least, this dominant weight initialization method also lacks biological plausibility.

Recent advances have challenged the necessity of random initialization. Notably, the ZerO initialization method, as outlined by Zhao et al. [2022], proposes a deterministic approach using only

zeros and ones to initialize weights. ZerO's methodology maintains the network's expressivity and simplifies the initialization process by eliminating the need to tune initial variances. This method has demonstrated state-of-the-art performance in architectures like ResNet, suggesting that deterministic initialization could be a viable alternative to random weight settings.

Our work seeks to bridge these two innovative approaches by applying ZerO initialization to the Hebbian learning context of the SoftHebb framework. We explore whether ZerO can maintain the learning efficacy of SoftHebb while enhancing its biological plausibility, through conducting a series of empirical evaluations where ZerO initialization is applied to SoftHebb networks trained on datasets including MNIST, CIFAR-10, and STL-10.

2 Related Work

2.1 Multilayer SoftHebb

Moraitis et al. [2022] put forward a novel approach with their multilayer SoftHebb algorithm, which trains deep neural networks without targets or error signals. This method eliminates limitations such as weight transport, non-local plasticity, and time-locking of layer updates, which are required in backpropagation. The proposed model introduces a soft Winner-Take-All (soft WTA) mechanism that facilitates competition-based neuron activation and learning through distributed processing, reflecting the probabilistic inference and efficiency seen in biological neural networks Journé et al. [2023].

Currently, SoftHebb achieves 99.4% and 80.3% accuracy in MNIST and CIFAR-10, respectively, and is competitive in some computer vision tasks.

2.1.1 Competition via Soft WTA

A key component of SoftHebb is the enforcement of competition between neurons of the same layer. In a layer consisting of K post-synaptic neurons, SoftHebb implements a refined WTA dynamic using the softmax function, which is modulated by a temperature parameter τ. Here, u_k represents the aggregate weighted input of the k-th neuron, while y_k denotes its output, which reflects competitive interactions among the l neurons.

$$u_k = \sum_{i=1}^{P} w_{ki} x_i, \text{ with } P \text{ pre-synaptic neurons.}$$

$$y_k = \frac{e^{\frac{u_k}{\tau}}}{\sum_{l=1}^{K} e^{\frac{u_l}{\tau}}}$$

Soft WTA allows multiple leading neurons with higher activations to contribute to their subsequent layer. As the temperature τ increases, the activation of more leading neurons influence the post-synaptic neuron. For example, out of K neurons, the leading r neurons with the highest u_k will have a non-zero y_k activation, where r depends on τ. This accomplishes a competition between neurons of the same layer, training neurons to become selective to specific, distinct input patterns (such as a stroke of a MNIST digit) as opposed to learning a general pattern Moraitis et al. [2022].

2.1.2 Local Plasticity

The second key component of the SoftHebb algorithm is its local adaptation of synaptic weights. The model's synaptic plasticity is described by the weight adjustment equation for a synaptic connection w_{ik} from a presynaptic neuron i with activation x_i to the k-th postsynaptic neuron and learning rate η, given by

$$\Delta w_{ik}^{(\text{SoftHebb})} = \eta \cdot y_k \cdot (x_i - u_k \cdot w_{ik})$$

This weight update is a rule inspired by Hebbian principles that enhances the soft WTA's general model. This proposed rule not only mirrors Oja's rule (Oja [1982]), $dw = y(x - yw)$, by taking into account the linear weighted sum of the inputs u_k but also factors in the nonlinear output y_k of the soft WTA.

Oja's rule refines the classic Hebbian Rule that serves as a model for the adaptive changes in synaptic connectivity, reflecting learning processes in both biological brains and artificial neural networks. It

encapsulates the principle often summarized as "neurons that fire together, wire together," highlighting the strengthening of connections between simultaneously active pre- and post-synaptic neurons.

It is important to highlight that the only indices in the formula are i and k, which locate neurons in the presynaptic and post synaptic layer, respectively. Every variable in this equation is localized to the synapse both temporally and spatially. Unlike traditional weight-update methods, local synaptic plasticity does not rely on error signals that need to propagate through and back the network, which is expensive in both time and memory.

Soft WTA and local plasticity are biologically plausible, and while learning occurs independently of errors with feed-forward communication, a critical feature of Hebbian networks is their shallow architecture, which restricts them to solving mostly simple tasks.

2.2 ZerO Initialization

ZerO Initialization is an unconventional weight initialization method for neural networks. Traditional methods rely on random weight initialization, which can be sensitive to initial variance. Effects of this include vanishing or exploding gradients, especially as the network increases in depth. ZerO Initialization deterministically initializes weights using only zeros and ones, using identity and Hadamard transforms.

Undergone theoretical and empirical studies, ZerO is shown to maintain a stable signal propagation during training, avoids training degeneracy or accuracy loss, and achieve state-of-the-art performance on various datasets, such as ImageNet and CIFAR-10, without relying on batch normalization. Other identified benefits of ZerO include enabling the training of ultra deep networks, a low rank learning trajectory, leading to sparse and low-rank solutions, and enhancing training reproducibility due to being fully deterministic. Extensive experimenting revealed that ZerO Initialization does not deter performance, indicating that randomness can be eliminated in weight initialization.

While these benefits of ZerO initialization may be intriguing for traditional neural networks, they are not as enticing for the shallow, error-independent SoftHebb network. One overlooked quality is it's biological plausibility: neurons fire completely (1) or not at all (0). As biological plausibility is rigorously upheld for Hebbian networks, ZerO could further enhance the integrity of SoftHebb.

3 Implementation: ZerO Initialization with SoftHebb

3.1 Applying ZerO Algorithm 2

Our implementation of ZerO with SoftHebb focuses on executing *Algorithm 2: ZerO Initialization on Convolution* detailed by in Table 2. Specifically, each dimensional-increasing convolutional layer is initialized with Hadamard matrices. The Hadamard matrices are rescaled with a normalization factor of $2^{\frac{-(m-1)}{2}}$, producing an orthonormal Hadamard transform. Remaining layers are initialized with a partial identity matrices.

Algorithm 2 *ZerO Initialization on Convolution, Page 7 of Zhao et al. [2022].*

Input: number of input channels c_{in}, number of output channels c_{out}, odd kernel size k.
Return: a $c_{out} \times c_{in} \times k \times k$ convolutional kernel .
Let $n \leftarrow \lfloor k/2 \rfloor$
If $c_{out} = c_{in}$**:** $[:, :, n, n] \leftarrow I$
If $c_{out} < c_{in}$**:** $[:, :, n, n] \leftarrow I^*$, the partial identity matrix
If $c_{out} > c_{in}$**:** $[:, :, n, n] \leftarrow c\, I^* H_m I^*,$ where $m = \lceil \log_2(P_l) \rceil$ and $c = 2^{-(m-1)/2}$

4 Preliminary Trial

4.1 Results

Our initial experimentation tested ZerO with the SoftHebb's preset model 4SoftHebbCnnCIFAR, which has 3 Hebbian-CNN layers and 1 MLP layer. While this final MLP layer is trained with

backpropagation, it is used as a simple linear classifier with a sole purpose for obtaining accuracy and loss metrics, given features outputted by the final Hebbian-CNN layer. We chose the 4SoftHebbCnnCIFAR model because SoftHebb carried out extensive hyperparameter experimentation with this model, achieving state-of-the-art performance on MNIST, CIFAR10, and STL10, allowing for a thorough comparison across different datasets and hyperparameters. To study the direct effects of ZerO initialization on 4SoftHebbCnnCIFAR, we directly employed the preset model without tuning.

	SoftHebb + Random Init	SoftHebb + ZerO Untuned
CIFAR10	80.3	36.97
FashionMNIST	89.55	68.99
MNIST	99.4	71.4
STL10	72.725	39.05

Table 1: Test accuracies of ZerO initialized SoftHebb on various datasets without hyperparameter tuning

Our preliminary tests show sub-par accuracies across datasets and hyperparameters. With no architectural changes, 4SoftHebbCnnCIFAR with ZerO achieves test accuracies of 71.4% on MNIST and 36.97% on CIFAR10. While this is significantly underperforming compared to 4SoftHebbCnnCIFAR with random initialization, we considered the difficulty of classifying CIFAR10, and the untunned nature of the model. Before tunning the model, we attempt to investigate potential reasoning behind the results, which will aid overall understanding and the tuning process.

4.2 Investigation

4.2.1 Importance of non-zero activations for soft WTA

SoftHebb is a shallow network that enforces competitive learning between neurons through soft WTA. SoftHebb's random initialization scheme gave all weights a value between 0 and 1, providing all neurons a chance to activate: $0 < w_{ik} < 1 \Rightarrow |w_{ik}x_i| > 0$. On the other hand, as detailed in Figure 2, ZerO initializes convoltional kernel $\boldsymbol{K}$ with the rescaled orthonormal Hadamard transform matrix when $c_{out} > c_{in}$, consisting of 0s and 1s. Note that in all SoftHebb preset models, it is the case that channel sizes increase by layer ($c_{out} > c_{in}$).

To further investigating the impact of weights being 0s and 1s, we counted the number of non zero-valued entries in the initialized matrices, detailed in Table 2. A clear distinction is observed, where ZerO-initialized 4SoftHebbCnnCIFAR has around 10 times less non-zero entries compared to SoftHebb's random initialization scheme.

	Random	ZerO	Random ZerO
CNN Layer 1	2400 = 100%	$96 \approx 4\%$	$1173 \approx 50\%$
CNN Layer 2	331776 = 100%	$36864 \approx 11\%$	$165784 \approx 50\%$
CNN Layer 3	5308416 = 100%	$589824 \approx 11\%$	$2653880 \approx 50\%$
MNIST Acc	98.9	70	98.87

Table 2: Number of non-zero entries in 4SoftHebbCnnCIFAR weight matrices for different initialization methods

We believe this potentially explains the sub-par performance of ZerO-initialized SoftHebb. A small number of non-zero weights implies a small number of activating neurons. The majority of neurons with weights initialized as 0 fail to contribute their activation x_i to their post-synaptic layer: $w_{ik} = 0 \Rightarrow |w_{ik}x_i| = 0$. In the worst case, if neuron x_i with $w_{ik} = 0$ frequently fires simultaneously with the k-th postsynpatic neuron – an indicator that the learning of a specific input pattern is taking place – their hebbian connection will be lost because x_i's activation is ignored. As a result, the learning of that specific input's pattern is lost. The large number of zero-initialized weights break bonds between neurons that wish to strengthen their connection, leading to a failure to learn features of the input.

To validate this hypothesis, we implemented a random ZerO initialization method, which initializes weight matrices with a random number of 0s and 1s in random placements. While maintaining the usage of strictly 0s and 1s, their distribution is random, and as a result, approximately 50% of entries are 1s. As detailed in the final column of Table 2, the performance is just as competitive as original SoftHebb on MNIST, achieving 98.87%. This is a potential indicator that the extremity of weights being 0 or 1 is not the cause for the loss in accuracy, but the lack of non-zero weights.

4.2.2 Ablation Study

To further investigate the impact of ZerO on SoftHebb, we conducted an ablation study, applying ZerO Initialization to selected layers and maintaining SoftHebb's original initialization for other layers. Results are detailed in Figure 1. By only applying ZerO initialization to individual and grouped layers, we noticed that accuracy only significantly decreased when ZerO was applied to all layers.

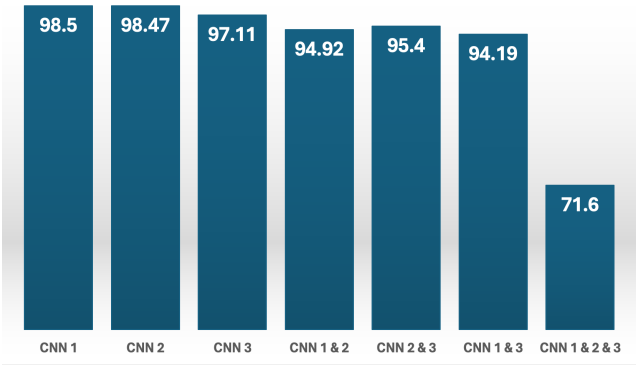


Figure 1: Test accuracies of 3 CNN-layer SoftHebb on MNIST, where the horizontal axis details which CNN layer(s) are ZerO initialized.

An observation requires particular emphasis: within a group of models with the same number of layers ZerO initialized, the specific index of the ZerO initialized layer(s) does not significantly impact performance.

This suggests that diminishing the strength of connections between pre- and postsynaptic neurons (at a ZerO-initialized layer) does not completely obstruct the feed forward communication of learned features to subsequent layers. Notably, when solely the first layer (CNN 1) undergoes ZerO initialization, the model still reaches a competitive accuracy of 98.5%, indicating that a ZerO-initializaed layer has the capacity to learn meaningful features that is passed onto subsequent layers to achieve an accurate classification.

Another notable case is when solely the last layer (CNN 3) undergoes ZerO initialization, the model retains a high accuracy level of 97.11%. This indicates that ZerO initialization is adept at conveying the final feature vector to the MLP layer, thereby facilitating an accurate classification. The insights from this observation affirm that ZerO initialization holds promise for effective integration with the SoftHebb network, through support of further adaptation and optimization.

5 Hyperparameter Search & Adaptation

To optimize ZerO initialized SoftHebb on MNIST, we focused on the learning rate parameter (η), the non-linearity parameter (q), the convolution kernel size, and the inverse temperature ($1/\tau$). Additionally, we tested different pooling types and sizes as well as activation functions with varying powers. We based our search ranges on those explored by Moraitis et al. [2022], but adjusted our ranges in some cases after preliminary experiments.

Table 3 and Table 4 presents our hyperparameter search results for both 2-layer and 4-layer configurations of the SoftHebb + ZerO model, respectively.

layer	operation	hyperparameters	searched range	found optimum
1	conv	η	[0.001-0.1]	0.07
		q	[0.3, 0.4, 0.5]	0.4
		kernel_size	[1, 2, 3, 4, 5, 6]	3
		$1/\tau$	[0.1-10]	1
	pooling	type	[AvgPooling, MaxPooling]	MaxPooling
		kernel_size	[2, 3, 4]	4
	activation	function	[Softmax, RePU, Triangle]	Triangle
		power	[0.1-10]	0.7

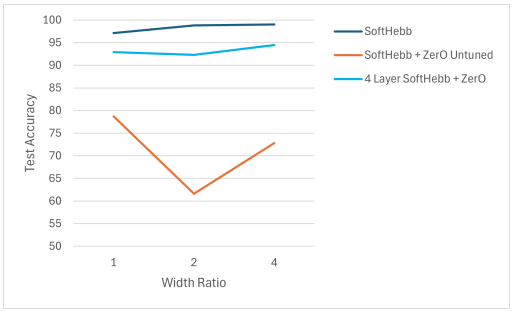
Table 3: Hyperparameter search range and optimum results on MNIST for 2-layer ZerO-initialized SoftHebb

layer	operation	hyperparameters	searched range	found optimum
1	conv	η	[0.001-0.1]	0.08
		q	[0.3, 0.4, 0.5]	0.5
		kernel_size	[1, 2, 3, 4, 5, 6]	3
		$1/\tau$	[0.1-10]	1
	pooling	type	[AvgPooling, MaxPooling]	MaxPooling
		kernel_size	[2, 3, 4]	4
	activation	function	[Softmax, RePU, Triangle]	Triangle
		power	[0.1-10]	0.7
2	conv	η	[0.001-0.1]	0.003
		q	[0.3, 0.4, 0.5]	0.3
		kernel_size	[1, 2, 3, 4, 5, 6]	4
		$1/\tau$	[0.1-10]	0.65
	pooling	type	[AvgPooling, MaxPooling]	MaxPooling
		kernel_size	[2, 3, 4]	4
	activation	function	[Softmax, RePU, Triangle]	Triangle
		power	[0.1-10]	1.4
3	conv	η	[0.001-0.1]	0.001
		q	[0.3, 0.4, 0.5]	0.1
		kernel_size	[1, 2, 3, 4, 5, 6]	1
		$1/\tau$	[0.1-10]	0.25
	pooling	type	[AvgPooling, MaxPooling]	AvgPooling
		kernel_size	[2, 3, 4]	2
	activation	function	[Softmax, RePU, Triangle]	Triangle
		power	[0.1-10]	1

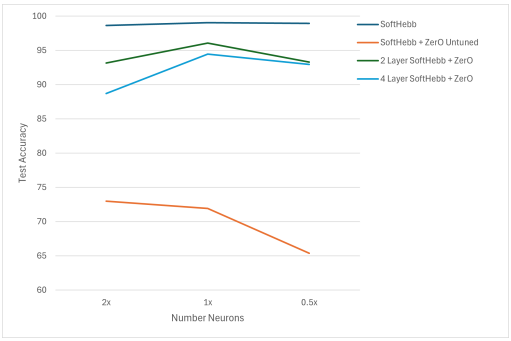
Table 4: Hyper parameter search and best results on MNIST for 4 layered ZerO initialized SoftHebb

Table 5 details experiments with our optimized models revealed a training accuracy of 91.382% and a test accuracy of 96.07% on MNIST with our 2-layer model, while the 4-layer model reached a training accuracy of 92.491% and a test accuracy of 94.45%. While both models obtain high accuracies, the simpler 2-layer model exceeds, reinforcing the shallow property of Hebbian networks, which compared to traditional neural networks, is significantly less complex while maintaining a competitive performance.

Based on our preliminary tests with SoftHebb + ZerO before tuning, we had hypothesized that increasing the number of neurons could have a positive effect on the model, if it meant that more neurons would have a non-zero activation. However, performing the same experiments on our optimally tuned SoftHeb + ZerO model actually worsened the performance as seen in Figure 2. In hindsight, this could be explained that while the total number of neurons increased, the overall proportion of those with non-zero activations remains the same. This performance also affirms knowledge we know about the human brain. According to a 2019 study on the human brain by Raman et al. [2019], adding neurons and connections can help learning up to a certain point, but after that an increase in size can actually impair learning. As SoftHebb is a biologically plausible



(a) Varying width ratio



(b) Varying number of neurons

Figure 2: Comparison of Test Accuracies for Tuned SoftHebb + ZerO Model on MNIST vs. Untuned SoftHebb

network and our implementation with ZerO aims to further enforce its plausibility, it follows that its performance would reflect characteristics of the human brain.

We also performed preliminary hyperparameter search for a CIFAR10 SoftHebb model, but ultimately, CIFAR10 proved to be a much more difficult task for SoftHebb + ZerO. Experiments were costly in terms of time and compute resources and often yielded little to no improvement from the baseline.

	2 Layer SoftHebb + ZerO	4 Layer SoftHebb + ZerO
Training Loss	0.0110281	0.00840335
Training Accuracy	91.382	92.491
Test Loss	0.000173179	0.000201055
Test Accuracy	96.07	94.45

Table 5: SoftHebb + ZerO performance on MNIST with optimal hyper parameters

6 Discussion and Conclusion

Through experimentation, evaluation, and adaptation with the ZerO initialization method for the SoftHebb model, potential of combining biologically plausible initialization methods with biologically plausible models is demonstrated.

An intriguing finding from our experiments is the potential of random binary (0 and 1) initialization, which yielded competitive accuracies similar to traditional random weight initialization. This suggests that while the variance factor of initialization is lost with random binary initialization, it's value-wise determinism and biological plausibility is worth exploring across various types of neural network architectures and tasks.

While we demonstrated adequate accuracy on simpler datasets like MNIST, the performance substantially declined on more complex datasets such as CIFAR-10. The structured initialization with zeros and ones, while simplifying the initialization process and reducing the reliance on random weight assignments, also introduces constraints that may not align well with the complex feature learning required for higher-dimensional data. We encourage future research to explore biologically plausible initialization and networks that are suitable for more challenging tasks. This exploration should include extensive hyperparameter optimization via grid search and learning rate scheduling, techniques employed by Zhao et al. [2022] that were unavailable for this project.

As artificial intelligence becomes increasingly integrated into daily life, traditional models trained with backpropagation are proving costly in terms of time, memory, and energy Patterson et al. [2021]. Developing time-efficient, space-efficient, self-organized models that do can meaningfully employ unlabeled data could not only benefit the environment but also bring us closer to understanding human-like intelligence. Major industry players like Huawei (Zahid et al. [2023]) and Intel (Davies et al. [2018]) are already constructing neuromorphic computers, highlighting the significance of this

research area. This field has the potential to significantly influence both the field of machine learning and broader human society.

7 Individual Contributions

Sarah Yi Xu: I came up with the high-level idea of the project, stemming from my personal interest in the topic. While all members delve into understanding the SoftHebb and ZerO algorithm, I focused on grasping the theoretical aspects to better understand and explain our experimental findings relation to SoftHebb as discussed in section 4.2 and 2.1. In terms of practical application, I conducted initial experiments to assess the performance of SoftHebb with ZerO on MNIST, FashionMNIST, and CIFAR10, exploring adjustments in kernel size and the number of layers. I also participated in fine-tuning the hyperparameters for the 2 and 4-layer SoftHebb models on MNIST, as well as a 4-layer model on CIFAR10, though these results did not make it into our final report due to their poor results. For paper writing, I contributed to and edited all sections.

Minh Le: I helped debug the provided codebase and update libraries to successfully recreate the SoftHebb paper's results. I also set up and ran experiments for preliminary trials on STL-10 and ImageNette, although the latter did not make it into the final paper due to limited computational resoruce. I also helped search for optimal hyperparameters, particularly on learning rate power (q). In paper writing, I am mainly responsible for Abstract, Introduction, and Discussion & Conclusion, as well as proof-reading the entire paper.

Kai Zhuang: I implemented ZerO initialized SoftHebb, ran both preliminary experiments and post hyperparameter search experiments with an emphasis on exploring the effects of width ratio and number of neuron changes. I also contributed to aggregating the data to generate graphs and tables for our paper. Additionally, I contributed to writing in several sections, primarily section 5 on hyperparameter search.

References

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. doi: 10.1109/MM.2018.112130359.

Adrien Journé, Hector Garcia Rodriguez, Qinghai Guo, and Timoleon Moraitis. Hebbian deep learning without feedback. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8gd4M-_Rj1.

Timoleon Moraitis, Dmitry Toichkin, Adrien Journé, Yansong Chua, and Qinghai Guo. Softhebb: Bayesian inference in unsupervised hebbian soft winner-take-all networks. *Neuromorphic Computing and Engineering*, 2(4):044017, 2022.

Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15:267–273, 1982.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training, 2021.

Dhruva Venkita Raman, Adriana Perez Rotondo, and Timothy O'Leary. Fundamental bounds on learning performance in neural circuits. *Proceedings of the National Academy of Sciences*, 116 (21):10537–10546, 2019. doi: 10.1073/pnas.1813416116. URL https://www.pnas.org/doi/abs/10.1073/pnas.1813416116.

Umais Zahid, Qinghai Guo, and Zafeirios Fountas. Predictive coding as a neuromorphic alternative to backpropagation: A critical evaluation. *Neural Computation*, 35(12):1881–1909, 2023.

Jiawei Zhao, Florian Tobias Schaefer, and Anima Anandkumar. Zero initialization: Initializing neural networks with only zeros and ones. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=1AxQpKmiTc.