Improving Deep Learning Speed and Performance Through Synaptic Neural Balance

Antonios Alexos*, Ian Domingo*, Pierre Baldi

University of California, Irvine
aalexos@uci.edu, idomingo@uci.edu, pfbaldi@uci.edu

Abstract

We present theory of synaptic neural balance and we show experimentally that synaptic neural balance can improve deep learning speed, and accuracy, even in data-scarce environments. Given an additive cost function (regularizer) of the synaptic weights, a neuron is said to be in balance if the total cost of its incoming weights is equal to the total cost of its outgoing weights. For large classes of networks, activation functions, and regularizers, neurons can be balanced fully or partially using scaling operations that do not change their functionality. Furthermore, these balancing operations are associated with a strictly convex optimization problem with a single optimum and can be carried out in any order. In our simulations, we systematically observe that: (1) Fully balancing before training results in better performance as compared to several other training approaches; (2) Interleaving partial (layer-wise) balancing and stochastic gradient descent steps during training results in faster learning convergence and better overall accuracy (with L_1 balancing converging faster than L_2 balancing); and (3) When given limited training data, neural balanced models outperform plain or regularized models; and this is observed in both feedforward and recurrent networks. In short, the evidence supports that neural balancing operations could be added to the arsenal of methods used to regularize and train neural networks. Furthermore, balancing operations are entirely local and can be carried out asynchronously, making them plausible for biological or neuromorphic systems. Our code is publicly available on github.

Code — https://github.com/antonyalexos/Neural-Balance

Introduction

Broadly speaking, neural balance refers to the idea of achieving or keeping a certain equilibrium in a neural network during training or after training, whereby such equilibrium may facilitate better information flow, or lower energy expenditure (Shwartz-Ziv 2022). As such, there are different notions of neural balance including, for example, the notion of balance between excitation and inhibition in biological neural networks (Froemke 2015; Field et al. 2020; Howes and Shatalina 2022; Kim and Lee 2022; Shirani and Choi 2023). Here we develop the concept of synaptic neural balance which refers to any systematic relationship between the input and output synaptic weights of individual neurons, or layers of neurons. Specifically, we consider the case where the cost of the input weights is equal to the cost of the output weights, where the cost is defined by some regularizer. One of the most basic examples of such a relationship, described below, is when the sum of the squares of the input weights of a neuron is equal to the sum of the squares of its output weights. In this work, we briefly describe the theory of synaptic neural balance and demonstrate its applications to deep learning regularization. We now describe the base case of synaptic neural balance.

Base Case: Consider a neuron with a ReLU activation function inside a network trained to minimize a regularized error function $\mathcal{E} = E + R$, where E is the data-dependent error (typically the negative log-likelihood of the data) and R is the regularizer (typically L_2 regularizer). If we multiply the incoming weights of the neuron by some $\lambda > 0$ (including the bias) and divide the outgoing weights of the neuron by the same λ, it is easy to see that this scaling operation does not affect in any way the contribution of the neuron to the rest of the network. Thus, the error E which depends only on the input-output function of the network is unchanged. However, the value of the L_2 regularizer changes continuously with λ, and the corresponding contribution is given by:

$$\sum_{i \in IN} (\lambda w_i)^2 + \sum_{i \in OUT} (w_i/\lambda)^2 = \lambda^2 A + \frac{1}{\lambda^2} B \qquad (1)$$

where IN and OUT denote the set of incoming and outgoing weights respectively, $A = \sum_{i \in IN} w_i^2$, and $B = \sum_{i \in OUT} w_i^2$. When λ moves away from 1, the contribution increases in one direction and decreases in the other. In the direction where it decreases, we can solve for the value λ^* associated with the minimal cost. Without taking derivatives, we note that the product of the two terms on the right-hand side of Equation 1 is equal to AB and does not depend on λ. Thus, the minimum is achieved when these two terms are equal, which yields: $(\lambda^*)^4 = B/A$ for the optimal λ^*. The corresponding new set of weights, $v_i = \lambda^* w_i$ for the input weights and $v_i = w_i/\lambda^*$ for the outgoing weights, must be balanced: $\sum_{i \in IN} v_i^2 = \sum_{i \in OUT} v_i^2$. This is because the optimal scaling factor for the optimal synaptic weights can only

*These authors contributed equally.

be $\lambda^* = 1$. Thus, we can define two operations that can be applied to the incoming and outgoing weights of a neuron: scaling and balancing. In between, we can also consider favorable scaling, or partial balancing, where λ is chosen to reduce the cost without necessarily minimizing it.

There have been isolated previous studies of this kind of synaptic balance (Du, Hu, and Lee 2018; Stock et al. 2022) under special conditions. For instance, in (Du, Hu, and Lee 2018), it is shown that if a deep network is initialized in a balanced state with respect to the sum of squares metric, and if training progresses with an infinitesimal learning rate, then balance is preserved throughout training. However, using an infinitesimal learning rate is not practical. Furthermore, there are many intriguing questions that can be raised. For instance: Why does balance occur? Does it occur only with ReLU neurons? Does it occur only with L_2 regularizers? Does it occur only in fully connected feedforward architectures? Does it occur only at the end of training? What happens if we iteratively balance neurons at random in a large network? And can partial or full balancing, before or during learning, be used as an effective regularization technique? All these questions, but the last one, are addressed by the theory of synaptic neural balance that we have developed and briefly describe in the next section. The last question, on using balancing as a learning regularizer, is the main topic of this paper and is addressed by the experiments presented in the following sections. Unless otherwise specified, throughout the paper, terms like "balancing" or "neural balancing" refer to "synaptic neural balancing".

The Theory of Synaptic Neural Balance

We present a brief summary of the main point of the theory. The complete theory is described in the Appendix with the detailed proofs of all the theorems. The first key point is that the base case described in the Introduction, can be extended in three main directions in terms of the activation functions, the regularizers, and the network architectures.

Activation Functions: The key property of the activation function f used in the base case is homogeneity. An activation function f is said to be homogeneous if and only if for any $\lambda > 0$ we have: $f(\lambda x) = \lambda f(x)$. This is what enables the overall network to keep the exact same functionality when the input and output synaptic weights are rescaled by λ and $1/\lambda$ respectively. The class of homogeneous activation functions contains many important and commonly used activation functions, such as the linear, ReLU, and leaky ReLU activation functions. More broadly, it can be shown that the class of homogeneous activation functions is exactly equal to the BiLU (bi-linear units) class of activation functions, whose graph consists of two half lines connected at the origin. In fact, even greater generality can be sought by studying activation functions that satisfy the equation: $f(g(\lambda)x) = h(\lambda)f(x)$. In this case, if we multiply the incoming synapses by $g(\lambda)$ and divide the outgoing synapses by $h(\lambda)$ the role of the corresponding neuron in the overall network remains unchanged and thus we can again seek the scaling of this form that minimizes the regularizer (see Appendix).

Regularizers: The base case can also be extended to very broad classes of regularizers, including all L_p regularizers ($p > 0$), which are by far the most widely used regularizers, particularly the L_2 and L_1 regularizers. It is easy to see that for any L_p regularizer Equation 1 still holds by replacing λ^2 with λ^p everywhere in the right hand side, and defining: $A = \sum_{IN} |w|^p$ and $B = \sum_{OUT} |w|^p$. In this case, the same balance reasoning can be applied yielding $(\lambda^*)^{2p} = B/A$. This is the optimal value of λ that we use in all our experiments. The following theorem provides further details and generality.

Theorem: (Balance and Regularizer Minimization) *Consider a neural network with BiLU activation functions in all the hidden units and overall error function of the form:*

$$\mathcal{E} = E(W) + R(W) \quad \text{with} \quad R(W) = \sum_w g_w(w) \tag{2}$$

where each function $g_w(w)$ is continuously differentiable, depends on the magnitude $|w|$ alone, and grows monotonically from $g_w(0) = 0$ to $g_w(+\infty) = +\infty$. For any setting of the weights W and any hidden unit i in the network and any $\lambda > 0$ we can multiply the incoming weights of i by λ and the outgoing weights of i by $1/\lambda$ without changing the overall error E. Then, for any neuron, there exists at least one optimal value λ^* that minimizes $R(W)$. Any optimal value must be a solution of the consistency equation:

$$\lambda^2 \sum_{w \in IN(i)} w g'_w(\lambda w) = \sum_{w \in OUT(i)} w g'_w(w/\lambda) \tag{3}$$

Once the weights are rebalanced accordingly, the new weights must satisfy the generalized balance equation:

$$\sum_{w \in IN(i)} w g'(w) = \sum_{w \in OUT(i)} w g'(w) \tag{4}$$

In particular, if $g_w(w) = |w|^p$ for all the incoming and outgoing weights of neuron i, then the optimal value λ^* is unique and equal to:

$$\lambda^* = \Big(\frac{\sum_{w \in OUT(i)} |w|^p}{\sum_{w \in IN(i)} |w|^p} \Big)^{1/2p} = \Big(\frac{||OUT(i)||_p}{||IN(i)||_p} \Big)^{1/2} \tag{5}$$

The decrease $\Delta R \geq 0$ in the value of the L_p regularizer $R = \sum_w |w|^p$ is given by:

$$\Delta R = \left(\big(\sum_{w \in IN(i)} |w|^p \big)^{1/2} - \big(\sum_{w \in OUT(i)} |w|^p \big)^{1/2} \right)^2 \tag{6}$$

After balancing neuron i, its new weights satisfy the generalized L_p balance equation:

$$\sum_{w \in IN(i)} |w|^p = \sum_{w \in OUT(i)} |w|^p \tag{7}$$

Proof: The proof is given in the Appendix. We use the value λ^* in eq. (5) for our experiments in the next Section.

Network Architectures: Finally, it is easy to see that the reasoning behind the base case can be applied to any BiLU

neurons inside any architecture, such a fully connected feedforward, locally connected feedforward, or recurrent. Again this is because scaling does not change the effect of the neuron on the rest of the network and therefore we can always scale the neuron in a way that minimizes a particular cost function or regularizer. It is even possible to train a network with a certain regularizer and balance it with respect to a different regularizer.

This brings us to the main result of the theory which is related to balancing algorithms. Imagine that we have a neural network containing BiLU (e.g. ReLU) neurons (it could also contain other kinds of neurons), with a fixed set of weights W. These could be the weights before learning has started, during learning (i.e. at a particular epoch), or after learning has finished. Imagine that we start balancing the BiLU neurons one after the other, in some regular order or, more generally, even in a stochastic order. Balancing the weights of a neuron may break the balance of another neuron. So while the value of the regularizer always decreases after each balancing operation, it is not clear what happens to the weights of the network, whether they converge to a stable value, and if so whether this value is unique. The main theorem of the theory is the proof that indeed not only the regularizer converges, but the weights themselves must converge and, most interestingly, they must converge to a unique point, which depends only on the initial set of weights W. The limit does *not* depend on the order in which the balancing operations are applied. This is due to the following theorem.

Main Theorem: *Given a network with weights W containing BiLU neurons. If the BILU neurons are iteratively balanced in any order with respect to a regularizer L_p ($p > 0$), the weights of the network will converge to a fixed point that depends only on W and corresponds to the solution of a strictly convex optimization problem defined over a linear manifold of constraints.*

Proof: Here we can only sketch the proof which is given in full in the Appendix. Consider a neuron j connected to a neuron i by a weight w_{ij}. Suppose that at time t after some iterations of the stochastic balancing algorithm, neuron j has been scaled by $\lambda_1(j) \dots \lambda_{n_j(t)}(j)$ and, similarly, neuron i has been scaled by $\lambda_1(i) \dots \lambda_{n_i(t)}(i)$. Then on can see that it is as if neuron j has been scaled by $\Lambda_j(t) = \Pi_k \Lambda_k(i) > 0$ and similarly for neuron i with a total scaling factor of $\Lambda_i(t) > 0$. As a result, after this scaling operations, the updated connections between neurons j and i is given by: $w_{ij}(t) = w_{ij}\Lambda_i(t)/\Lambda_j(t)$. The value of the regularizer R at the same time t is given by: $R(t) = \sum_{ij}(\Lambda_i(t)/\Lambda_j(t))^p |w_{ij}|^p$. Thus, the stochastic balancing process is trying to minimize the quantity $R = \sum_{ij}(\Lambda_i/\Lambda_j)^p |w_{ij}|^p$. This is not a strictly convex optimization problem in the variables Λ_i, nor in the variables $M_{ij} = \Lambda_i/\Lambda_j$. However it is strictly convex optimization problem in the variables: $L_{ij=\log M_ij}$. The objective function is given by:

$$\min_{L_{ij}} \sum_{ij} \exp(pL_{ij})|w_{ij}|^p \tag{8}$$

which is strictly convex, as a sum of strictly convex functions (exponentials), in the variables L_{ij} for any $p > 0$. What is left, is to understand what are the constraints on the variables L_{ij} that define the manifold over which the optimization is being carried out. In particular, from the set of Λ_i's it is easy to construct a unique set of L_{ij}. However what about the converse?

One can see that for any path π connecting an input unit to an output unit, one must have $\sum_\pi L_{ij} = 0$. Thus the set of constraints is a set of linear constraints, defining a linear, hence convex, manifold, which completes the sketch of the proof.

Related Work

Neural balance-like methods have been explored in the literature before, but from a different perspective, mostly by proposing to add an extra term in the loss function. This term would be the ratio of output divided by the input neuron weights of all layers, in a form imitating neural balance. More specifically Yang et al. (2022) propose to replace the L_2 regularization term in the loss with the sum of products of l2 (not squared) norms of the input and output weights associated each ReLU activation. They also prove the equivalence between L_2 regularization and the proposed term. The work in Stock et al. (2022) proposes a new local heterosynaptic learning rule by adding a kind of reconstruction loss term in which neurons try to balance themselves. In Du, Hu, and Lee (2018), the authors prove that gradient descent with infinitesimal step size effectively conserves the differences between squared norms of inputs and outputs weights of each layer without explicit regularization. So if the initial weights are balanced, the balance is preserved during learning for all layers. Related results are also described in (Arora et al. 2018). Saul (2023) computes multiplicative rescaling factors—one at each hidden unit— to balance the weights of neural networks and then analyzes the optimization procedure for learning in these neural networks. Other works have also explored symmetry and balance effects on training neural networks. For example, Neyshabur, Salakhutdinov, and Srebro (2015) shows that training with stochastic gradient descent does not work well in highly unbalanced neural networks, so they proposed a rescaling-invariant solution analyzed in (Neyshabur, Tomioka, and Srebro 2015). In the same vein, other authors have proposed that learning in neural networks can be accelerated with rescaling transformations (Zhao et al. 2022; Armenta et al. 2023) without mentioning balancing the weights though. In our case we present both theoretical results on neural synaptic balance, including the existence and uniqueness of a globally balanced state (given an initial set of weights W), and experimental results showing that balancing neurons can expedite learning convergence and improve learning performance. This is also the first work experimenting with neural balance in both feedforward and recursive neural networks.

Experiments and Results

In the following experiments, we train and compare various neural network architectures using full neural balancing, partial balancing, and L_1 or L_2 regularization. The term "plain" is used to refer to training of neural networks without bal-

ancing and without regularizers. Full balance is obtained by iteratively balancing all BiLU neurons in the network until convergence is achieved. Partial balance is implemented by balancing the neurons in a layer-wise fashion, starting from the input layer and moving towards the output layer or vice-versa (no significant differences are observed). Due to the gradual nature of partial balance, the periodicity of the balancing operation is key to its implementation. In partial balance, the balancing operation can be performed up to once per epoch.

Through the use of partial balancing during training, it has been observed that the ratio of the norms of a neuron's output to input weights tends to equalize, irrespective of the periodicity of epochs that we perform partial balancing operations. We have also observed that partial balancing helps the network converge faster and achieve a balanced state as is expected in a fully-trained network, same is in full balancing. The balancing operations for each neuron in each layer take place in parallel so they do not impose a bottleneck during training. Our results suggest that neural balancing is effective in training various types of neural networks with limited data. Furthermore, this approach proves beneficial in reducing overfitting and enhancing generalization in data-scarce environments.

To ensure reproducibility and fairness, experiments comparing training methodologies use the same range of seeds, learning rates, and train/test splits. Every experiment was run with 8 different seeds and the result reported is the average of them. Some Figures also include the std apart from the mean. A more detailed description of our experimental setup can be found in the Appendix. The roadmap of our experiments is organized as follows: first, we present experiments with the full dataset on both FCNs and RNNs. Then, we move onto data-scarce environments, amplifying the complexity of the experiments. For every experiment we deploy FCNs and RNNs ranging from smaller to larger sizes. The term FCN refers to Feedforward-layered networks with full connectivity between the layers.

Toy Experiment on a Circle Toy Dataset

To validate our initial hypothesis, which is that the balancing operation results in the equalization of the norms of the input and output weights for every neuron in a neural network, we observe the ratio between the aforementioned norms during training. We do this through a toy network trained on a simple 2-dimensional dataset for a binary classification task, where the limited number of layers and 'neurons' allow us to measure weights without the computational intensity attributed to accessing values from a large network. We compare the use of full balancing with partial balancing during training. Both methodologies result in the optimal factor λ^* calculated during balancing to converge to 1, confirming that the norms of the input and output weights for each neuron equalize through the use of balancing. fig. 1 contains partial balancing performed every epoch on a 5-neuron toy model trained on a 2-dimensional concentric circle toy dataset showing that the input and output weight norms equalize for each neuron.

To contextualize the rate of convergence of the norms from the partial balancing toy experiment, we measure the input

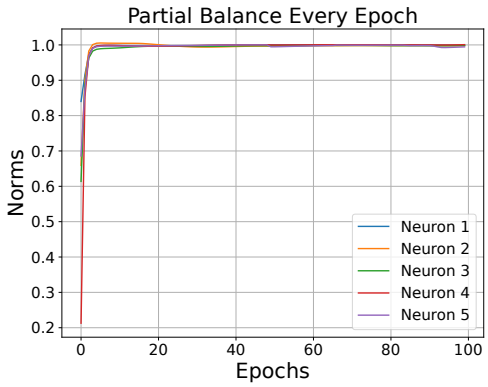

Figure 1: Partial balancing performed every epoch on a 5-neuron toy model trained on a 2-dimensional dataset for a binary classification task showing that the input and output weight norms equalize for each neuron

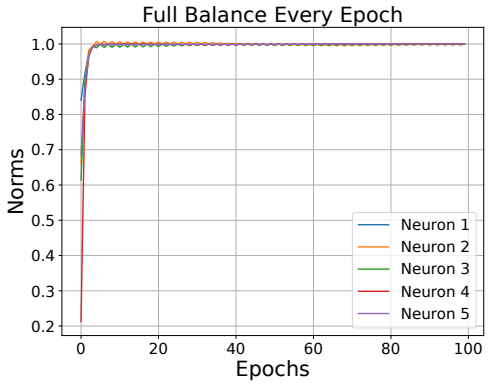

Figure 2: Full balancing performed every epoch on a 5-neuron toy model trained on a 2-dimensional dataset for a binary classification task showing that the input and output weight norms equalize for each neuron

and output norms of each neuron after a full-balance has been performed on the network. While the full-balance guarantees that the input and output norms of each neuron will always be close to each other, since full balancing is performed until that requirement is met, it remains useful as a benchmark for the rate of convergence of partial-balancing. fig. 2 delineates the rate of convergence of the input and output norms, doing so almost immediately, due to the methodology of full balancing. fig. 1 demonstrates the efficacy of partial-balancing, resulting in a rapid, and computationally less expensive method of 'balancing' neurons.

Assessment of Full Balance Before Training

In fig. 3 and table 1, we assess the use of the full balancing operation before the commencement of training. Compared to a standard initialization, the application of full balancing results in faster convergence, and higher overall accuracy when using the same model architecture, hyperparameters, and training methodologies. Partial balancing at every epoch

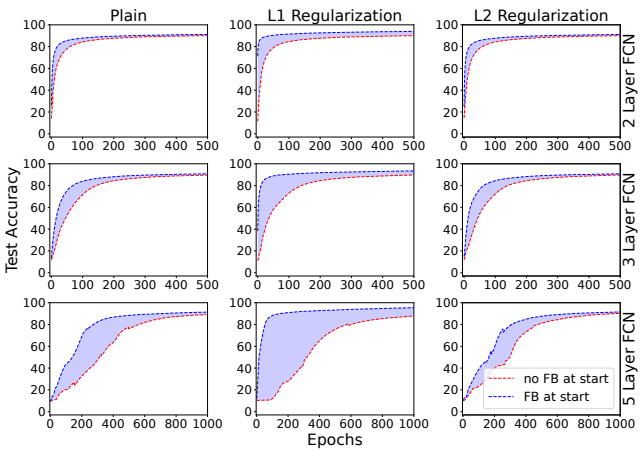

Figure 3: A demonstration of the effect of a full neural balance before the start of training on various sizes of fully connected networks, using various training methodologies. Regardless of L2 Regularization, neural partial balancing, or plain accuracy used in training, a neural full balance results in faster convergence, and a higher overall accuracy.

Type	No FB at Start			FB at Start		
	Plain	L1	L2	Plain	L1	L2
2 Layer FCN	90.1%	90.1%	90.1%	**91.2%**	**94%**	**91.2%**
3 Layer FCN	89.6%	89.7%	89.7%	**90.8%**	**93.5%**	**90.8%**
5 Layer FCN	89.1%	87.9%	90.3%	**91.4%**	**95.5%**	**91.6%**

Table 1: Accompanying fig. 3, Test accuracy during training of Plain, L1 Regularized, and L2 Regularized Fully Connected Networks trained on MNIST, comparing full balancing before training with no full balance before training. As observed in fig. 3, full balancing before training results in faster convergence, as well as universally higher attained test accuracy.

after a full balance results in the least change due to the fundamentally similar nature of the full balancing operation to the partial balancing operation, hence its omission from the plots. Repeated partial balancing results in the same outcome weights when using the same seed, albeit, over time since those weights aren't balanced from the start. Larger model sizes tend to exhibit a stronger correlation between the use of neural balancing, and the model's rate of convergence. These observations are especially exhibited in the normally trained models, where the use of full balancing at the start results in much faster convergence, as well as a higher final accuracy achieved by the model. fig. 3 pictures the comparison between full balancing and no balancing performed on an FCN before training. Each square in the grid represents a combination of a model size and a training methodology, where in every case, neural balancing results in an improvement in the rate of convergence and model accuracy.

Partial Balance on a Fully Connected Network

MNIST We test all forms of neural balancing on the MNIST handwritten digit dataset exclusively through FCNs.

Type	Plain	L2 NB	L1 NB	L2 1e-5	L1 1e-5
2-FCN	91.22%	91.19%	**94.542%**	91.18%	93.96%
3-FCN	90.84%	90.86%	**93.94%**	90.79%	93.47%
5-FCN	91.37%	91.63%	**96.26%**	91.59%	95.48%

Table 2: Test accuracy across training comparisons of partial balancing, L2 Regularization, and Plain Accuracy for FCNs of varying sizes on MNIST. We observe that L1 partial balancing outperforms the other training methodologies on all model sizes

Type	Plain	L1 Regularization	L2 Regularization
No NB at Start	88.26	88.23	88.22
NB at Start	**88.64%**	**88.24%**	**88.57%**

Table 3: Accompanying fig. 5, test accuracy for a Recurrent Neural Network trained on the IMDB sentiment analysis dataset, comparing Plain, L1 Regularized, and L2 Regularized models with and without a full balance at the start of training. As observed in fig. 5, neural balancing universally results in a higher test accuracy during training.

To fully capture the regularization capability of neural balancing, we test on a range of model architectures. From fig. 4 and table 2, we observe that neural partial balancing results in faster convergence and test accuracy across all model sizes.

FashionMNIST: Following the line of inquiry on the performance of neural balancing on FCNs trained on MNIST, we assess its performance on FashionMNIST using the same model architectures. We use FCNs of various sizes, and perform a partial balance on the model at every epoch, identically to the MNIST experiments. We observed similar results on performance and convergence on FashionMNIST. Regardless of the size of the model, or the methodology used to train said model, neural balancing significantly increases the rate of convergence, as well as its overall test accuracy. These experiments are included in the Appendix.

Full Balance on Recurrent Neural Networks

IMDB Sentiment Analysis: We continue our assessment of neural balancing with experiments performed on the RNN architecture. We train a 3-layered RNN on the IMDB sentiment analysis dataset, once again assessing full neural balancing with a 'plain', and regularized models. fig. 5 demonstrates that when full balancing is performed before training, the model has a better final accuracy when compared to equivalent, non-balanced methodlologies.

Neural Balance in Limited-data Environments

In data-scarce environments, models employing neural partial balancing techniques demonstrate accelerated convergence compared to unmodified models, or ones that utilize a traditional regularization approach. These experiments are executed by stratifying samples equally according to their class labels to maintain a balanced distribution of classes within the training data. Specifically, the experiments are conducted using the MNIST dataset, wherein the training dataset is deliberately reduced to 600 samples, constituting

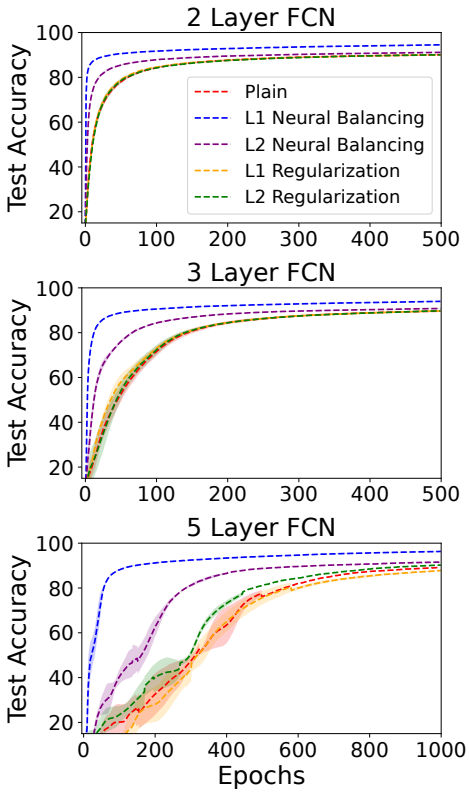

Figure 4: Comparison of neural balance, L1 and L2 Regularization on MNIST. We observe that as the models grow bigger, neural balance helps model converge faster and perform better than the other techniques.

Type	No FB at Start			FB at Start		
	2 Layers	3 Layers	5 Layers	2 Layers	3 Layers	5 Layers
Plain	84.15%	73.49%	88.9%	**91.39%**	**91.42%**	**90.86%**
L1 NB	91.25%	90.57%	92.87%	**93.26%**	**93.3%**	**92.92%**
L2 NB	87.99%	84.53%	91.06%	**91.94%**	**91.37%**	**90.59%**
L1	83.92%	72.03%	11.35%	**85.98%**	**81.33%**	**19.79%**
L2	83.35%	75.21%	86.81%	**91.16%**	**90.86%**	**88.78%**

Table 4: Accompanying fig. 7, comparing test accuracy during training of various methodologies, using 1% of the MNIST dataset to simulate a limited data environment. As shown in fig. 7, the use of full balancing at the start of training not only increases the rate of convergence of every training methodology, but also allows for a higher attainable overall accuracy.

only 1% of the original dataset. The test dataset remains unchanged. fig. 7 pictures partial balancing performed on an FCN trained using 1% of the MNIST dataset. We observe that neural balancing results in higher accuracy and faster convergence, which could be attributed to better performance in data-scarce environments.

Further, we continue this study with an assessment of neural balancing on an RNN trained with a fraction of the original IMDB dataset. fig. 6 contains the comparison between training methodologies using a 3 layer RNN trained on the IMDB

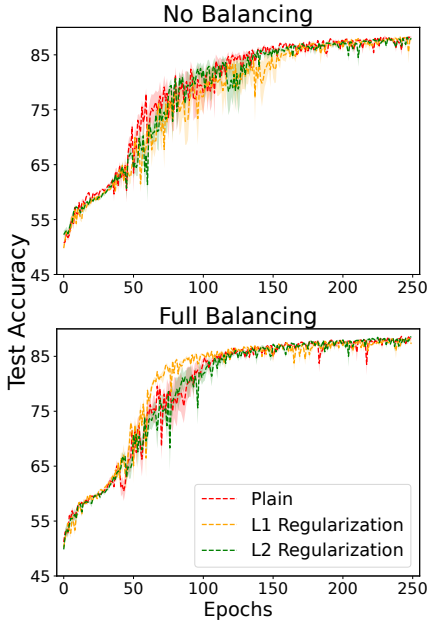

Figure 5: A comparison between partial balancing, L2 Regularization, and Plain Accuracy on a 3 Layer RNN using the IMDB sentiment analysis dataset. We also contrast the standard initialization with a full neural balancing operation performed before the start of training. We observe that neural partial balancing performed every epoch, paired with a full balance before training, results in the best overall accuracy, and convergence speed.

sentiment analysis dataset with 5% of the available training data. Partial balancing has higher accuracy on average, as well as a faster convergence, hinting at a characteristic of being able to generalize without a lot of training data.

Discussion

Summing up our experiments we observe the following quantitative results. In FCNs, Neural Balance yields a notable improvement in model performance and convergence speed. Specifically, this method results in a 3-5% performance increase over plain models, and more than a 1% improvement over optimally L1-regularized models. Additionally, L1 neural balancing facilitates convergence at a rate 1.5 to 10 times faster, contingent on model size. When trained on limited datasets (1% of the full data), L1 neural balancing enhances performance by 3-10% compared to plain models, and by 1-5% relative to models regularized with L1 and L2 techniques. Moreover, it achieves up to a 10-fold increase in convergence speed, depending on model size. In RNNs, L1 neural balancing contributes to a 2-5% increase in convergence speed, with the application of L2 neural balancing leading to a more than 15% acceleration in convergence when training on 5% of the data. These findings underscore the efficacy of L1 neural balancing in optimizing both performance and training efficiency across different model architectures.

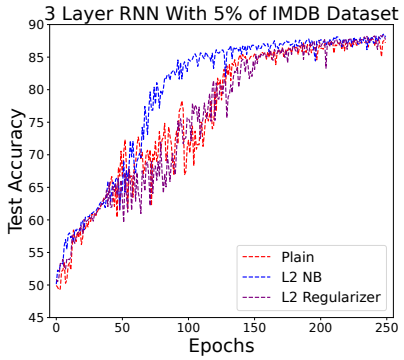

Figure 6: A comparison between partial balancing, Plain Accuracy, and L2 Regularization as performed on a 3 Layer RNN using 5% of the available dataset. Neural balancing reports the best overall performance.

Ablation Study

This is an ablation study that we conducted on Bioplausible Regularization. More specifically we applied neural balance to Direct Feedback Alignment, a backpropagation alternative.

Bioplausible Regularization. We continue our exploration into the domain of biologically plausible machine learning, where we specifically focus on the application of neural balancing to Direct Feedback Alignment (DFA). DFA represents a significant shift from traditional backpropagation techniques by replacing the biologically impossible transposed weight matrices with randomized matrices, through which error gradients are propagated. Building on the foundation of DFA, we introduce neural balancing as a biologically plausible regularizer due to its locality of operations, that serves as an alternative to the conventional LP regularization methods. Our findings indicate that neural balancing not only performs comparably to LP regularizers, but also integrates well with the biologically plausible nature of DFA, offering a framework for biologically plausible training methodologies. When we apply this combination to both a 2-layer and 7-layer FCN and train on the MNIST dataset, neural balancing demonstrates performance metrics that are on par with those achieved through traditional L2 regularization. Though this approach does not surpass conventional regularization in terms of raw performance, it compellingly illustrates that neural balancing can function as a viable, biologically plausible alternative that does not compromise the effectiveness of the training process. We note that DFA and other backpropagation alternatives perform poorly with CNNs on CIFAR10. In the future, we plan to endeavor further into the avenue of bioplausibility.

Conclusions

Synaptic balancing provides a novel approach to regularization that is supported by an underlying theory. Synaptic balancing is very general in the sense that it can be applied with all usual cost functions, including all L_p cost functions.

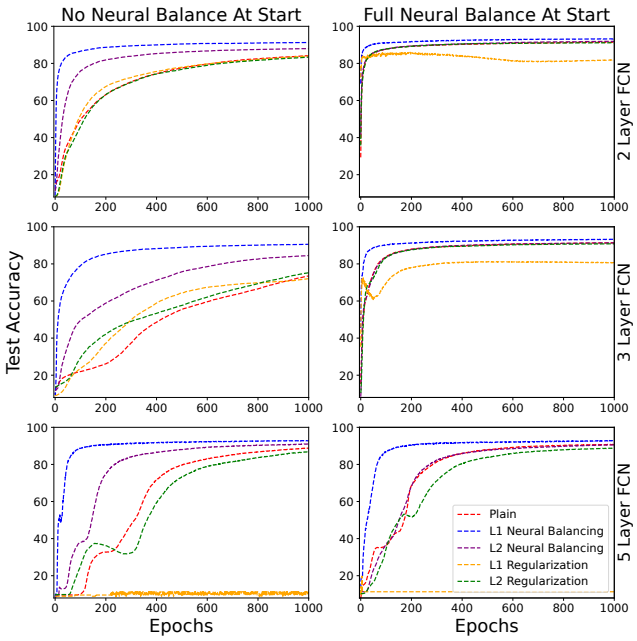

Figure 7: A comparison between partial balance, standard regularization, and Plain Accuracy, with and without a full balance before training, on various Fully Connected Networks trained on 1% of the MNIST dataset. We observe that neural balancing consistently has a positive impact on the rate of convergence and overall accuracy of the model.

Synaptic balancing can be carried in full or in partial manner, due to the convexity connection provided by the main theorem. Full or partial synaptic balancing can be applied effectively to any set of weights, at any time during the learning process: at the start of learning, at the end of learning, or during learning, by alternating balancing steps with stochastic gradient steps. It can be applied in combination with any training algorithm and any other regularizer. For example, one could train a network with L_2 regularization and apply L_1 balancing to the weights after the training is complete. Given, neural balance has some limitations; as mentioned earlier it can be applied only to neurons with specific activation functions (BiLU or slightly more general activation functions as shown in the Appendix). Another limitation is that it cannot be applied to neurons in Convolution layers due to the nature of the convolution operation with the kernels. Simulations show that these approaches can improve learning in terms of speed (fewer epochs), accuracy or generalization abilities. Thus, in short, balancing is a novel effective approach to regularization that can be added to the list of tools available to regularize networks, like dropout, and other regularization tools. Finally, a neuron can balance its weights independently of all other neurons in the network. The knowledge required for balancing is entirely *local* and available at each neuron. In this sense, balancing is a natural algorithm for distributed asynchronous architectures and physical neural systems, and as such it may find applications in neuromorphic chip designs or brain studies.

References

Armenta, M.; Judge, T.; Painchaud, N.; Skandarani, Y.; Lemaire, C.; Gibeau Sanchez, G.; Spino, P.; and Jodoin, P.-M. 2023. Neural teleportation. *Mathematics*, 11(2): 480.

Arora, S.; Cohen, N.; Golowich, N.; and Hu, W. 2018. A convergence analysis of gradient descent for deep linear neural networks. *arXiv preprint arXiv:1810.02281*.

Du, S. S.; Hu, W.; and Lee, J. D. 2018. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in neural information processing systems*, 31.

Field, R. E.; D'amour, J. A.; Tremblay, R.; Miehl, C.; Rudy, B.; Gjorgjieva, J.; and Froemke, R. C. 2020. Heterosynaptic plasticity determines the set point for cortical excitatory-inhibitory balance. *Neuron*, 106(5): 842–854.

Froemke, R. C. 2015. Plasticity of cortical excitatory-inhibitory balance. *Annual review of neuroscience*, 38: 195–219.

Howes, O. D.; and Shatalina, E. 2022. Integrating the neurodevelopmental and dopamine hypotheses of schizophrenia and the role of cortical excitation-inhibition balance. *Biological psychiatry*.

Kim, D.; and Lee, J.-S. 2022. Neurotransmitter-Induced Excitatory and Inhibitory Functions in Artificial Synapses. *Advanced Functional Materials*, 32(21): 2200497.

Neyshabur, B.; Salakhutdinov, R. R.; and Srebro, N. 2015. Path-sgd: Path-normalized optimization in deep neural networks. *Advances in neural information processing systems*, 28.

Neyshabur, B.; Tomioka, R.; and Srebro, N. 2015. Norm-based capacity control in neural networks. In *Conference on learning theory*, 1376–1401. PMLR.

Saul, L. K. 2023. Weight-balancing fixes and flows for deep learning. *Transactions on Machine Learning Research*.

Shirani, F.; and Choi, H. 2023. On the physiological and structural contributors to the dynamic balance of excitation and inhibition in local cortical networks. *bioRxiv*, 2023–01.

Shwartz-Ziv, R. 2022. Information flow in deep neural networks. *arXiv preprint arXiv:2202.06749*.

Stock, C. H.; Harvey, S. E.; Ocko, S. A.; and Ganguli, S. 2022. Synaptic balancing: A biologically plausible local learning rule that provably increases neural network noise robustness without sacrificing task performance. *PLOS Computational Biology*, 18(9): e1010418.

Yang, L.; Zhang, J.; Shenouda, J.; Papailiopoulos, D.; Lee, K.; and Nowak, R. D. 2022. A better way to decay: Proximal gradient training algorithms for neural nets. In *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)*.

Zhao, B.; Dehmamy, N.; Walters, R.; and Yu, R. 2022. Symmetry teleportation for accelerated optimization. *Advances in neural information processing systems*, 35: 16679–16690.