A correspondence between normalization strategies in artificial and biological neural networks

Yang Shen[1], Julia Wang[1], and Saket Navlakha*[1]

[1]Cold Spring Harbor Laboratory, Simons Center for Quantitative Biology, Cold Spring Harbor, NY 11724

Abstract

A fundamental challenge at the interface of machine learning and neuroscience is to uncover computational principles that are shared between artificial and biological neural networks. In deep learning, normalization methods, such as batch normalization, weight normalization, and their many variants, help to stabilize hidden unit activity and accelerate network training, and these methods have been called one of the most important recent innovations for optimizing deep networks. In the brain, homeostatic plasticity represents a set of mechanisms that also stabilize and normalize network activity to lie within certain ranges, and these mechanisms are critical for maintaining normal brain function. Here, we propose a functional equivalence between normalization methods in deep learning and homeostatic plasticity mechanisms in the brain. First, we discuss parallels between artificial and biological normalization methods at four spatial scales: normalization of a single neuron's activity, normalization of synaptic weights of a neuron, normalization of a layer of neurons, and normalization of a network of neurons. Second, we show empirically that normalization methods in deep learning push activation patterns of hidden units towards a homeostatic state, where all neurons are equally used — a process we call "load balancing". Third, we develop a neural normalization algorithm, inspired by a phenomena called *synaptic scaling*, and show that this algorithm performs competitively against existing normalization methods. Overall, we hope this connection will enable neuroscientists to propose new hypotheses for why normalization works so well in practice and new normalization algorithms based on established neurobiological principles. In return, machine learners can help quantify the trade-offs of different homeostatic plasticity mechanisms in the brain and offer insights about how stability may promote plasticity.

*Corresponding author: navlakha@cshl.edu

Introduction

Since the dawn of machine learning, normalization methods have been used to pre-process input data to lie on a common scale. For example, min-max normalization, unit vector normalization, z-scoring, or the like, are all well known to improve model fitting, especially when different input features have different ranges (e.g., age vs. salary). In deep learning, normalizing the input layer has also proved beneficial; for example, "whitening" input features so that they are decorrelated and have zero mean and unit variance leads to faster training and convergence [1, 2]. More recently, normalization has been extended to hidden layers of deep networks, whose activity can be viewed as inputs to a subsequent layer. This type of normalization modifies the activity of hidden units to lie within a certain range or to have a certain distribution, independent of input statistics or network parameters [3]. While the theoretical basis for why these methods improve performance has been subject to much debate — e.g., reducing co-variate shift [3], smoothening the objective landscape [4], de-coupling the length and direction of weight vectors [5], acting as a regularizer [6–8] — normalization is now a standard component of state-of-the-art architectures and has been called one of the most important recent innovations for optimizing deep networks [5].

In the brain, normalization has long been regarded as a canonical computation [9, 10] and occurs in many sensory areas, including in the auditory cortex to varying sound intensities [11]; in the olfactory system to varying odor concentrations [12]; and in the retina to varying levels of illumination and contrast [13–15]. Normalization is believed to help generate intensity-invariant representations for input stimuli, which improve discrimination and decoding that occurs downstream [9].

But beyond the sensory (input) level, there is an additional type of normalization found ubiquitously in the brain, which goes by the name of *homeostatic plasticity* [16]. Homeostasis refers to the general ability of a system to recover to some set point after being changed or perturbed [17]. A canonical example is a thermostat used to maintain an average temperature in a house. In the brain, the set point can take on different forms at different spatial scales, such as a target firing rate for an individual neuron, or a distribution of firing rates over a population of neurons. This set point is typically approached over a relatively long period of time (hours to days). The changes or perturbations occur due to other plasticity mechanisms, such as LTP or LTD, that modify synaptic weights and firing rates at much faster time scales (seconds to minutes). Thus, the challenge of homeostasis is to ensure that set points are maintained on average without "erasing" the effects of learning. This gives rise to a basic stability versus plasticity dilemma. Disruption of homeostasis mechanisms has been implicated in numerous neurological disorders [18–23], indicating their importance for normal brain function.

In this perspective, we highlight parallels between normalization algorithms used in deep learning and homeostatic plasticity mechanisms in the brain. Identifying these parallels can serve two purposes. First, machine learners have extensive experience analyzing normalization methods and have developed a sense of how they work, why they work, and when using certain methods may be preferred over others. This experience can translate to quantitative insights about outstanding challenges in neuroscience, including the stability versus plasticity trade-off, the roles of different homeostasis mechanisms used across space and time, and whether there are parameters critical for maintaining homeostatic function that have been missed experimentally. Second, there are many normalization techniques used in the brain that have not, to our knowledge, been deeply explored in machine learning. This represents an opportunity for neuroscientists to propose new normalization algorithms from observed phenomena or established principles [24] or to provide new perspectives on why existing normalization schemes used in deep networks work so well in practice.

The benefits of load balancing (homeostasis)

In computer science, the term "load balancing" means to distribute a data processing load over a set of computing units [25]. Typically, the goal is to distribute this load evenly to maximize efficiency and reduce the amount of time that units are idle (e.g., for servers handling traffic on the Internet). For neural networks, we define load balancing based on how frequently a set of neurons are activated, and how similar their mean activation levels are, on average. Why might load balancing in neural networks be attractive computationally? Three reasons come to mind:

First, load balancing increases the coding capacity of the network; i.e., the number of unique stimuli that can be represented using a fixed number of resources (neurons). Suppose that under standard training, a certain fraction (say, 50%) of the hidden units are just not used; that is, they are never, or rarely ever, activated. This wasted capacity would reduce the number of possible patterns the network could represent and would introduce unnecessary parameters that can prolong training. Load balancing of neurons could avoid these problems by pressing more hidden units into service. In the brain, equal utilization of neurons also promotes distributed representations, in which each stimuli is represented by many neurons, and each neuron participates in the representation of many stimuli (often called a combinatorial code [26, 27]). This property is particularly attractive when such representations are formed independent of input statistics or structure.

Second, load balancing can improve fine-grained discrimination. Suppose there are two hidden units that are similarly activated for the same input stimuli (e.g., images of dogs). The training process could just choose one of them and turn off the other. But if both units are used, then the door remains open for future fine-grained discrimination; e.g., discriminating between subclasses of dogs, such as chihuahuas and labradoodles. In general, if more nodes are used to represent a stimulus, then the nodes may better preserve finer details of the pattern, which can serve later as the basis for discrimination, if necessary. Relatedly, if a neuron has a sigmoidal activation function, normalization keeps the neuron in its non-saturated regime. This is believed to help the neuron be maximally informative and discriminative [28–32].

Third, load balancing can serve as a regularizer, which is commonly used in deep networks to constrain the magnitude of weights or the activity levels of units. Regularizers typically improve generalization and reduce over-fitting [33], and can be specified explicitly or implicitly [34]. There are many forms of regularization used in deep learning; for example, Dropout [35], in which a random fraction of the neurons is set inactive during training; or weight regularization, in which ℓ_1 or ℓ_2 penalties are applied to the loss function to limit how large weight vectors become [36, 37]. Although regularization is a powerful tool to build robust models, regularization alone is not guaranteed to generate homeostatic representations.

Normalization methods across four spatial scales

We begin by describing artificial and neural normalization strategies that occur across four spatial scales (Figure 1, Table 1): normalization of a single neuron's activity via intrinsic neural properties; normalization of synaptic weights of a neuron; normalization of a layer of neurons; and normalization of an entire network of neurons.

Normalization of a single neuron's activity

Here, we focus on normalization methods that directly modify the activity level of a neuron via intrinsic mechanisms.

In deep learning, the current most popular form of single neuron normalization is called *batch normalization* [3]. It has long been known that z-scoring the input layer — i.e., shifting and scaling the inputs to have zero mean and unit variance — speeds up network training [1]. Batch normalizaton essentially applies this idea to each hidden layer by ensuring that, for every batch of training examples, the activation of a hidden unit over the batch has zero mean and unit variance.

Mathematically, let $\{z_1, z_2, \ldots, z_B\}$ be the activations of hidden unit z for each of the $i = 1 : B$ inputs in a training batch. Let μ_B and σ_B^2 be the mean and variance of all z_i's, respectively. Then, the batch-normalized activation of z for the i^{th} input is:

$$\hat{z}_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}},$$

where ϵ is a small constant.

In practice, the effect of this simple transformation is profound: it leads to significantly faster convergence (larger learning rates) and improved stability (less sensitivity to parameter initialization and learning rate) [3, 4, 38, 39]. Numerous extensions of this method have since been proposed with various tweaks and perks on a similar underlying idea (Table 1).

In the brain, normalizing the activity of a neuron has long been appreciated as an important stabilizing mechanism [40]. For example, if neuron u drives neuron v to fire, the synapse between them may get strengthened by Hebbian plasticity. Then, the next time u fires, it is even more likely that v fires, and this positive feedback loop can lead to excessive activity. Similarly, if the synapse undergoes depression, then it is less likely for v to fire in the future, and this negative feedback can lead to insufficient activity. The job of homeostasis is to prevent neurons from being both over-utilized (hyperactive) and under-utilized (hypoactive) [41].

Modifying a neuron's excitability — e.g., its firing threshold or bias — represents one intrinsic neural mechanism used to achieve homeostasis [42–44]. The idea is simple (Figure 1A); each neuron has an approximate target firing rate at which it prefers to fire. A neuron with sustained activity above its target will increase its firing threshold, such that it becomes harder to fire, and likewise, a neuron with depressed activity below its target will decrease its firing threshold, thus becoming more sensitive to future inputs. The net effect of these modifications is that the neuron hovers around its target firing rate, on average over time. Several parameters are involved in this process, such as the rate at which thresholds are adjusted, which affects how quickly homeostasis is approached, and the value of the target itself, which may be cell-type specific. Other intrinsic mechanisms, such as modifying ion channel density, can also be used to intrinsically regulate firing rates (Figure 1).

Both of these methods are unsupervised; they adjust the activity of a neuron to lie within a preferred, narrow range with respect to recently observed data.

Normalization of synaptic weights

Here, we focus on normalization methods that indirectly modify the activity of a neuron by changing its weights.

In deep learning, one popular way to normalize the inputs to a unit (post-synaptically) is called *weight normalization* [45]. The main idea is to re-parameterize the conventional weight vector $\mathbf{w}$ of a unit into two components:

$$\mathbf{w} = \frac{c}{\|\mathbf{v}\|}\mathbf{v},$$

where c is a scalar and $\mathbf{v}$ is a parameter vector, both of which are learned. This transformation fixes the length (Euclidean norm) of the weight vector, such that $\|\mathbf{w}\| = c$, for any $\mathbf{v}$. Backpropagation is then applied to c and $\mathbf{v}$, instead of to $\mathbf{w}$. Thus, the length of the weight vector (c) is de-coupled from the direction of the weight vector ($\mathbf{v}/\|\mathbf{v}\|$). Such "length-direction" decoupling leads to faster learning and exponential convergence in some cases [5].

In the brain, the most well-studied type of weight normalization is called *synaptic scaling* [41] (Figure 1B, Left). If a neuron is on average firing above its target firing rate, then all of its incoming excitatory synapses are downscaled (i.e., multiplied by some factor, $0 < \alpha < 1$), to reduce its future activity. Similarly, if a neuron is firing far below its target, then all its excitatory synapses are upscaled ($\alpha > 1$); in other words, prolonged inactivity leads to an *increase* in synaptic size [46]. These rules may seem counter-intuitive, but remember that these changes are happening over longer time scales than the changes caused by standard plasticity mechanisms. Indeed, it is hypothesized that one way to resolve the plasticity versus stability dilemma is to temporally segregate Hebbian and homeostatic plasticity so that they do not interfere [47]. This could be done, for example, by activating synaptic scaling during sleep [48, 49].

Interestingly, synapse sizes are scaled on a per neuron basis using a multiplicative update rule (Figure 1B, Left). For example, if a neuron has four incoming synapses with weights 1.0, 0.8, 0.6, and 0.2, and if the neuron is firing above its target rate, then the new weights would be downscaled to 0.5, 0.4, 0.3, and 0.1, assuming a multiplicative factor of $\alpha = 1/2$. Critically, multiplicative updates ensure that the relative strengths of synapses are preserved, which is believed to help maintain synapse-specificity of the neuron's response caused by learning. The value of the multiplicative factor need not be constant, and could depend, for example, on how far away the neuron is from reaching its target rate. Thus, synaptic scaling keeps the firing rate of a neuron within a range while preserving the relative strength between synapses.

Another form of weight normalization in the brain is called *dendritic normalization* [50–52], which occurs locally on individual branches of a neuron's dendritic arbor (Figure 1B, Right). The idea is that if one synapse gets strengthened, then its neighboring synapses on the arbor compensate by weakening. This process is homeostatic because the total strength of all synapses along a local part of the arbor remains approximately constant. This process could be mediated by a shared resource, for example, a fixed number of post-synaptic neurotransmitter receptors available amongst neighboring synapses [53]. Computationally, this process creates sharper boundaries between spatially adjacent synapses receiving similar inputs, which could enhance discrimination and contrast.

Normalization of a layer of neurons

Here, we focus on normalization schemes that modify the activity of an entire layer of neurons, as opposed to just a single neuron's activity.

In deep learning, *layer normalization* [54] was recently proposed to overcome several drawbacks of batch normalization. In batch normalization, the mean and variance statistics of each neuron's activity is computed across a batch of training examples, and then each neuron is normalized with respect to its own statistics over the batch. In layer normalization, the mean and variance is instead computed over an entire layer of neurons for each training example, and then each neuron in the layer is normalized by the same mean and variance. Thus, layer normalization can be used online (i.e., batch size of one), which makes it more amenable to training recurrent neural networks [54].

In the brain, layer-wise normalization has most prominently been observed in sensory systems (Figure 1C, Left). For example, in the fruit fly olfactory system, the first layer of (receptor) neurons encode odors via a combinatorial code, in which, for any individual odor, most neurons respond at a low rate, and very few neurons respond at a high rate [26]. Specifically, the distribution of firing rates over all receptor neurons is exponential with a mean that depends on the concentration of the odor (higher concentration → higher mean). In the second layer of the circuit, projection neurons receive odor excitation from receptor neurons, as well as inhibition from lateral inhibitory neurons [12]. The result is that the concentration-dependence is largely removed; i.e., the distribution of firing rates for projection neurons follows an exponential distribution with approximately the same mean, for all odors and all odor concentrations [55] (Figure 1C, Right). Thus, while an individual neuron's firing rate can change depending on the odor, the distribution of firing rates over all neuron's remains nearly the same for any odor. This process is dubbed *divisive normalization* and is believed to help fruit flies identify odors independent of the odor's concentration. Divisive normalization has also been studied in the visual system, for example, light adaptation in the retina, or contrast adjustment in the visual cortex [9].

Overall, layer normalization helps maintain some invariant response property of a layer of neurons by dividing the responses of individual neurons by a factor that relates to the summed activity of all the neurons in the layer. These normalizations can be considered "homeostatic" because they preserve, for any input, properties of a distribution of firing rates (e.g., the mean or variance). In the brain, other non-linear transformations are also used alongside these transformations, for example, to adjust saturation rates of individual neurons and to amplify signals prior to normalization [9].

Normalization of a network of neurons

In the brain, recent work has challenged the conventional view that homeostasis applies at the level of a single neuron or a strict layer of neurons, and have instead attributed homeostasis properties to a broader network of neurons. In one experiment, the firing rates of individual neurons in a hippocampal network were monitored for two days after applying baclofen, a chemical agent that suppresses neural activity. After two days, the distribution of firing rates over the population was compared to the distribution of firing rates for a control group of neurons that received no baclofen. Strikingly, both were approximated by the same log-normal distribution. Moreover, the firing rates of many individual neurons, in both conditions, significantly changed from day 0 to day 2 [56, 57] (Figure 1D), suggesting that homeostasis may not strictly apply at the level of an individual neuron but is rather maintained at the population level. Similar observations have been made in the stomatogastric ganglion of crabs and lobsters, where rhythmic bursting is robustly maintained despite many perturbations to the circuit [58]. This remains a beautiful yet mysterious property of network stability implemented by neural circuits, and the mechanisms driving this level of network regulation remain poorly understood [59].

In deep learning, we are not aware of a normalization strategy that is applied across an entire network of units, or even across a population of units beyond a single layer. Network homeostasis could in principle be an emergent property from local homeostasis rules implemented by individual units, or could be a global constraint intrinsically enforced by some unknown mechanism. Either way, we hypothesize that network homeostasis may be attractive in deep networks because it allows for more flexible local representations while still providing stability at the network level.

An empirical comparison of normalization algorithms

The empirical results in this section serve two purposes. The first is to show that two popular normalization methods generate homeostatic representations, thus offering a new perspective on the benefits of normalization in deep learning. The second is to show that a method inspired by synaptic scaling also generates homeostatic effects and performs competitively against existing normalization methods. These results are not meant to represent a full-fledged comparison between normalization methods across multiple architectures, datasets, or hyper-parameter settings. Rather, these results are simply meant to demonstrate a proof-of-concept of the bi-directional perspective argued here.

We define homeostasis based on two properties: 1) The probability or frequency that each unit is activated (i.e., outputs a value > 0) over all inputs in a batch; and 2) The average activation value of each unit when activated. At homeostasis, each unit should be activated with a similar probability (i.e., no units are over- or under- utilized), and the average response magnitudes of units should lie within a narrow range.

Experimental setup. For our basic architecture, we used the original LeNet5 [60] with two convolutional layers and three fully connected layers with ReLU activation functions. Like neurons, ReLU units include a firing threshold; only nodes with a value greater than a threshold can fire.

We experimented with two datasets. The first is CIFAR-10, a standard benchmark for classification tasks, which contains 60,000 color images, each of size 32×32, and each belonging to one of 10 classes (airplanes, cats, trucks, etc.). The second dataset is SVHN (Street House View Numbers), which contains 73,257 color images, each of size 32×32, and each belonging to one of 10 classes (digits from 0–9). SVHN is analogous to MNIST but is more difficult to classify because it includes house numbers in natural scene images taken from a street view.

Each normalization method is applied to every layer, except the input and output layers, with all affine parameters trainable. For each dataset, all methods used Adam optimization using PyTorch with default parameters. Additional hyper-parameters were fixed for each dataset: CIFAR-10 (batch size of 32, learning rate of 0.003, train for about 45,000 iterations), SVHN (batch size of 256, learning rate of 0.01, train for about 8,000 iterations). Batch statistics are calculated using training data during training and testing data during testing.

Table 2 provides the equations for each normalization algorithm.

A synaptic-scaling-inspired normalization algorithm. Of the many normalization methods discussed above, we choose to model synaptic scaling because it is one of the most well studied and widely-observed mechanisms across brain regions and species.

We propose a simplified model of synaptic scaling that captures two keys aspects of the underlying biology: multiplicative scaling of synaptic weights, and constraining a node to be activated around a target activation probability on average. In the first step, the incoming weight vector $\boldsymbol{w}$ for a hidden unit is multiplied by a factor α, i.e., $\boldsymbol{w} = \alpha\boldsymbol{w}$. Each hidden unit has its own α value, which is made learnable during training. The α values are initialized to 1. In the second step, for each hidden unit, we subtract its mean activation (over a batch) from its actual activation for each input in the batch. This process ensures that each unit has a mean activation (before ReLU) of 0 and hence, a probability of activation (output value > 0) of around 50%, and thus resembles the biological observation that no neuron is over- or under- utilized. This step is also the same as mean-only batch normalization [45]. One advantage of this synaptic scaling model compared to batch normalization is that it removes the division by the variance term, which can lead to exploding gradients when the variance is close to zero.

An "ideal" model of synaptic scaling might only multiplicatively scale the weights of a hidden unit such that a given target activation probability is achieved on average. Instead, we first scale the weights by a learnable parameter (α), which allows the network to learn the optimal range of activation values for the unit, and we then constrain the unit to hit its target in step two. Similarly, batch normalization does not simply use z-scored activation values for each hidden unit (Table 2), but rather includes two learnable parameters (γ, β) per unit to shift and scale its normalized activation. In both cases, this flexibility likely increases the representation power of the network [3].

Mathematically, for each hidden unit, the forward-pass operations for synaptic scaling are:

$$\begin{aligned}
\boldsymbol{w} &= \alpha\boldsymbol{w} \\
z_i &= \boldsymbol{w}\boldsymbol{x}_i + b \\
y_i &= \mathrm{ReLU}(z_i - \mu_B),
\end{aligned}$$

where the subscript i indicates the i^{th} example in a batch of size B ($i = 1 : B$); $\boldsymbol{w}$, $\boldsymbol{x_i}$, b, z_i, y_i are the incoming weights to the hidden unit, the inputs for the i^{th} example from the previous layer, the bias of the unit, the value of the unit before activation, and the output of the unit, respectively; μ_B is the average of all z_i's over a batch.

To explore how the two steps independently affect classification performance, we tested each of them without the other. We call these models "Mean-only" and "Scale-only", respectively (Table 2).

Existing normalization methods generate homeostatic representations

First, we confirmed that two state-of-the-art normalization methods — batch normalization (BatchNorm) and weight normalization (WeightNorm) — improve classification accuracy on CIFAR-10: from $59.3 \pm 1.4\%$ for the original version of LeNet5 without normalization (Vanilla) to $63.8 \pm 0.9\%$ (WeightNorm) and $65.8 \pm 0.5\%$ (BatchNorm) (Figure 2A). Normalized networks also learned faster; i.e., they required fewer training iterations to achieve high accuracy.

Second, we show that BatchNorm and WeightNorm demonstrate load balancing effects. For the first property of homeostasis, Figure 2B shows that hidden units in normalized networks had more similar activation probabilities than in Vanilla: the coefficients of variation of activation probabilities across hidden units were 0.20 (BatchNorm) and 1.38 (WeightNorm) compared to 1.65 (Vanilla). BatchNorm and WeightNorm also used more units in the network; for example, in the first fully-connected layer, BatchNorm and WeightNorm activated $41.3 \pm 2.6\%$ and $18.7 \pm 2.4\%$ of units per iteration, respectively, compared to Vanilla ($10.3 \pm 2.2\%$) (Figure 2C). For the second property, Figure 2D shows that when active, the activation values of hidden units were more similar when normalized compared to than without; the coefficients of variation of activation values across hidden units were 0.16 (BatchNorm) and 0.30 (WeightNorm) compared to 0.55 (Vanilla). The average output value for hidden units was also significantly reduced in BatchNorm (0.89 ± 0.14) and WeightNorm (0.76 ± 0.23) compared to Vanilla (13.36 ± 7.36).

Together, hidden units in BatchNorm- and WeightNorm- trained networks are activated with a more similar probability and have activation values constrained to a narrower ranger, compared to hidden units in Vanilla networks.

Synaptic scaling performs load balancing and obtains competitive performance

We next tested the Synaptic Scaling method and found that its classification accuracy ($66.0 \pm 0.7\%$) was very similar to BatchNorm ($65.8 \pm 0.5\%$) on CIFAR-10 (Figure 2A). In contrast, Mean-only and Scale-only performed worse than Synaptic Scaling, suggesting that both steps — multiplicative scaling of synapses and setting target activation probabilities — are better when combined.

Synaptic Scaling also generates homeostatic representations that are on par or slightly better than those of BatchNorm. For the first property, Figure 2B shows that each hidden unit had a very similar probability of being activated; coefficient of variation of 0.11 for Synaptic Scaling and 0.20 for BatchNorm, compared to 1.65 for Vanilla. Synaptic Scaling activated $50.7 \pm 0.3\%$ of the hidden units total in each iteration, which is slightly higher than BatchNorm ($41.3 \pm 2.6\%$) and much higher than Vanilla ($10.3 \pm 2.2\%$) (Figure 2C). For the second property, Figure 2D shows that the activation values across hidden units were similar after normalization; coefficient of variation of 0.17 for Synaptic Scaling and 0.16 for BatchNorm, compared to 0.55 for Vanilla. The average output value for hidden units was also reduced in Synaptic Scaling and BatchNorm (0.63 ± 0.11 versus 0.89 ± 0.14, respectively) compared to Vanilla (13.36 ± 7.36).

Interestingly, the learned α parameters for Synaptic Scaling are all positive, meaning no weights flipped sign during training, and all the $\alpha < 1$, meaning the weights are all scaled down (Figure 2E). We did not set any upper or lower bounds on α, and the fact that the learned values stay within $[0, 1]$ indicates that down-scaling of weights, which in turn reduce activation values, may generally be beneficial for this classification task.

Validation on a second dataset. To ensure these results were not specific to one dataset, we ran all methods on a second dataset (SVHN) and found similar trends (Figure 3). To summarize, Synaptic Scaling and BatchNorm improve classification accuracy compared to all other methods (Figure 3A), and generate more homeostatic representations (Figure 3B–D) than Vanilla.

Discussion

We showed that widely used normalization methods in deep learning are functionally equivalent to homeostatic plasticity mechanisms in the brain. While the implementation details vary, both ensure that the activity of a neuron is centered around some fixed value or lies within some fixed distribution, and both are temporally local in the sense that changes only depend on recent behavior (recent firing rate or recent data observed). In summary, both attempt to stabilize and bound neural activity in an unsupervised manner, and both are critical for efficient learning.

We showed that two state-of-the-art normalization methods (BatchNorm and WeightNorm), as well as a new normalization algorithm inspired by scaling scaling, generate homeostatic representations in artificial neural networks and improve classification accuracy on two datasets. Interestingly, WeightNorm achieves lower accuracy and generates representations that are less homeostatic, compared to both BatchNorm and Synaptic Scaling (Figures 2 and 3). This suggests that learning algorithms are more efficient when coupled with homeostatic load balancing, and either without the other degrades performance. This perspective contributes to the growing list of explanations for why normalization is so useful in deep networks [3–6, 8], and a natural next step is to develop a theoretical understanding for why stability (i.e., creating homeostatic representations) may actually promote plasticity (i.e., improving classification accuracy and learning efficiency), as opposed to being in conflict.

Moving forward, there are several challenges that remain in bridging the gap between normalization in the artificial and biological neural networks. First, the implementation details of both types of networks are well-acknowledged to be different [61]. For example, unlike most artificial networks, the brain has a strict division of excitatory and inhibitory neurons, which means different homeostasis rules can be applied to excitatory and inhibitory synapses [62]. Second, our model of synaptic scaling assumed that each hidden unit had the same target "fixed point", whereas in reality, adjustable fixed points might further improve performance. Indeed, batch normalization allows the fixed points to be learned through the affine parameter, β. In artificial networks, fixed points could vary based on the dataset, network architecture, or other hyper-parameters. In the brain, different cell types may use different fixed points, or fixed points of a single cell may change during different phases of training. Third, it is unclear how the time-scales of homeostasis in the brain map to time-scales of learning in artificial networks. Normalization is typically applied per input or per batch in deep learning, but other time scales remain unexplored [48, 49, 63]. Similarly, normalization that operates simultaneously across different spatial scales (e.g., combining batch normalization and layer normalization) have also not been analyzed. Fourth, there are different constraints between what a hidden unit can store and compute and what a neuron can (likely) store and compute. For example, it seems plausible for a neuron to track its own mean firing rate over a given time window, but tracking its own variance seems trickier.

There are also several challenges in understanding the neuroscience of homeostasis that remain outstanding. For example, network-wide homeostasis, which goes beyond fixed points for individual neurons, has been observed in the brain, but the circuit mechanisms that give rise to these effects remain elusive. Further, it remains unclear what the advantages and disadvantages of different homeostatic mechanisms are, and when to use which. For example, many homeostatic plasticity mechanisms reset a neuron's firing rate to a target firing rate on average; but when would it be appropriate to achieve this goal by modifying intrinsic excitability versus modifying pre- or post-synaptic weights? Indeed, there may be multiple means towards the same end, and it remains unclear what the trade-offs are among these different paths.

We hope these insights provide an avenue for building future collaborations, where computer scientists can use quantitative frameworks to evaluate how different plasticity mechanisms affect neural function; indeed, understanding the advantages and disadvantages of homeostatic plasticity from a machine learning perspective could shed new light on the biological mechanisms that enable homeostasis and could help identify parameters important for maintaining homeostatic function that have not been measured experimentally. In return, neuroscientists can provide new perspectives on the benefits of normalization in neural networks and inspiration for designing new normalization algorithms based on neurobiological principles.

Acknowledgements. The authors thank Sanjoy Dasgupta, Alexei Koulakov, Vishaal Krishnan, Ankit Patel, and Shyam Srinivasan for helpful comments on the manuscript.

Table 1: **Correspondences between normalization mechanisms in artificial and biological neural networks across four spatial scales.** See also reviews: [47, 64–66].

Scale	**Deep learning**	**Refs.**	**Neural circuits**	**Refs.**
Single neuron	Batch normalization	[3]	Firing thresholds	[42–44]
			Ion channel density	[67, 68]
			Cell-type specificity	[16]
	Group normalization	[69]	—	—
	Instance normalization	[70]	—	—
	Self-normalization	[71]	—	—
	Normalization propagation	[72]	—	—
Synaptic weights	Weight normalization	[45]	(Post) synaptic scaling	[16, 41]
	—	—	(Pre) release probability	[16, 46]
	—	—	(Branch) dendritic normalization	[50–52]
Layer of neurons	Whitening	[1, 2, 73]	Decorrelation / whitening	[74–78]
	Layer normalization	[54]	Divisive normalization	[9]
Network of neurons	—	—	Network homeostasis	[56, 79]

Table 2: **Normalization algorithms.** All equations show the forward-pass update equations for a single hidden unit. For Weight normalization, backpropagation is performed on c and $\boldsymbol{v}$, instead of $\boldsymbol{w}$.

Algorithm	**Equations**	**Notation**
Batch normalization	$z_i = \boldsymbol{w}\boldsymbol{x_i} + b$ $\hat{z}_i = \dfrac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$ $y_i = \text{ReLU}(\gamma \hat{z}_i + \beta)$	i : i^{th} example in a batch of size B z_i : value of the unit (before activation) for i y_i : value of the unit (after activation) for i
Weight normalization	$y_i = \text{ReLU}(\boldsymbol{w}\boldsymbol{x}_i + b)$ $\boldsymbol{w} = \dfrac{c}{\lvert\lvert\boldsymbol{v}\rvert\rvert}\boldsymbol{v}$	$\boldsymbol{w}$: incoming weights to the unit $\boldsymbol{x_i}$: inputs to the unit for i
Synaptic scaling	$\boldsymbol{w} = \alpha\boldsymbol{w}$ $z_i = \boldsymbol{w}\boldsymbol{x}_i + b$ $y_i = \text{ReLU}(z_i - \mu_B)$	b : bias of the unit μ_B : average of z_i's over the batch
Mean-only	$z_i = \boldsymbol{w}\boldsymbol{x}_i + b$ $y_i = \text{ReLU}(z_i - \mu_B)$	σ_B^2 : variance of z_i's over the batch γ, β : trainable parameters (BatchNorm)
Scale-only	$\boldsymbol{w} = \alpha\boldsymbol{w}$ $z_i = \boldsymbol{w}\boldsymbol{x}_i + b$ $y_i = \text{ReLU}(z_i)$	$c, \boldsymbol{v}$: training parameters (WeightNorm) α : trainable parameter (Synaptic Scaling) ϵ : a small constant

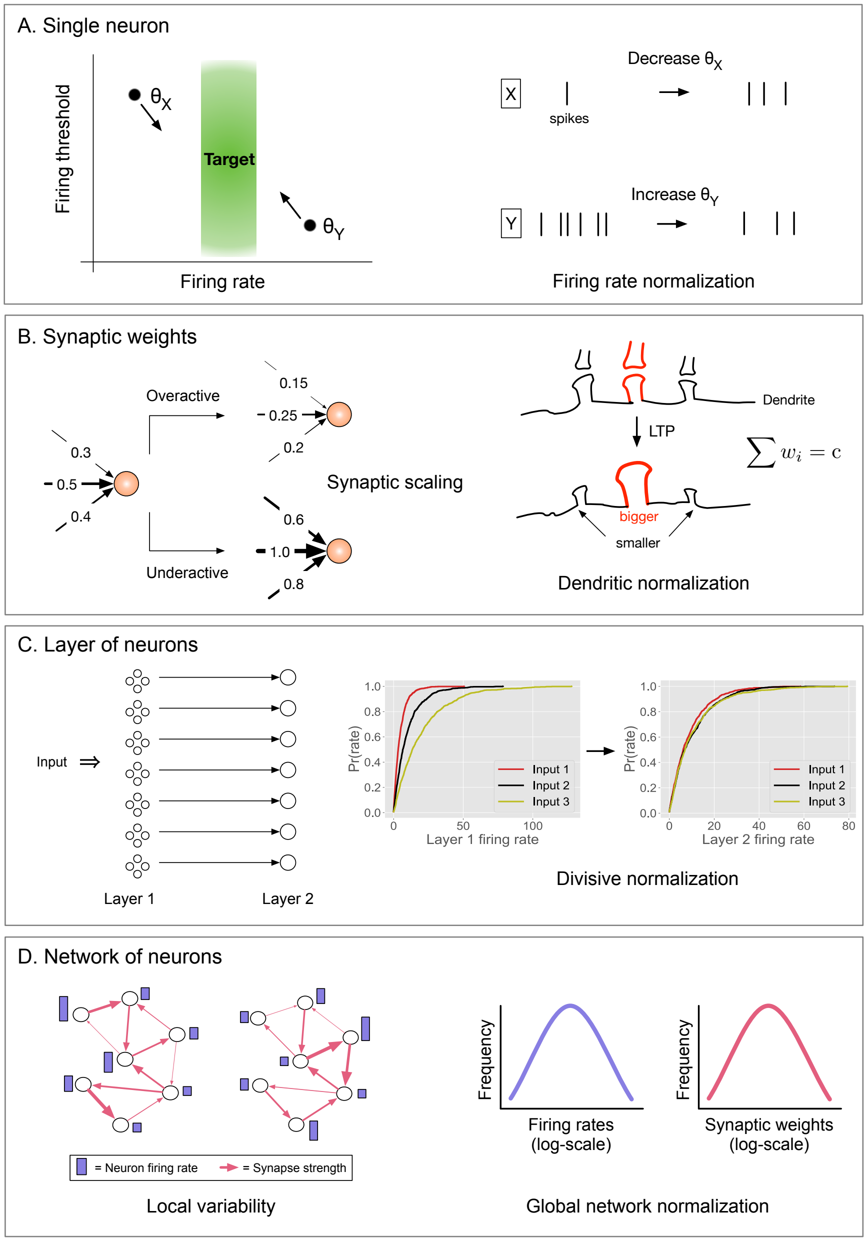

Figure 1: **Neural homeostatic plasticity mechanisms across four spatial scales.** A) Normalization of a single neuron's activity. Left: Neuron X has a relatively low firing rate and a high firing threshold, θ_X, and vice-versa for neuron Y. Right: Both neurons can be brought closer to their target firing rate by decreasing θ_X and increasing θ_Y. B) Normalization of synaptic weights. Left (synaptic scaling): If a neuron is firing above its target rate, its synapses are multiplicatively decreased, and vice-versa if the neuron is firing below its target rate. Right (dendritic normalization): If a synapse size increases due to strong LTP, its neighboring synapses decrease their size. C) Normalization of a layer of neurons. Left: Two layers of neurons with feed-forward connections, and other feed-back inhibitory connections (not shown). Right: The cumulative distribution of firing rates for neurons in the first layer is exponential with a different mean for different inputs. The activity of neurons in the second layer are normalized such that the means of the three exponentials are approximately the same. D) Left: Example of a neural circuit with the same units and connections, but different activity levels for neurons (purple bars) and different weights (pink arrow thickness) under two different conditions. Right: Despite local variability, the global distributions of firing rates and synaptic weights for the network remains stable (log-normally distributed) under both conditions.

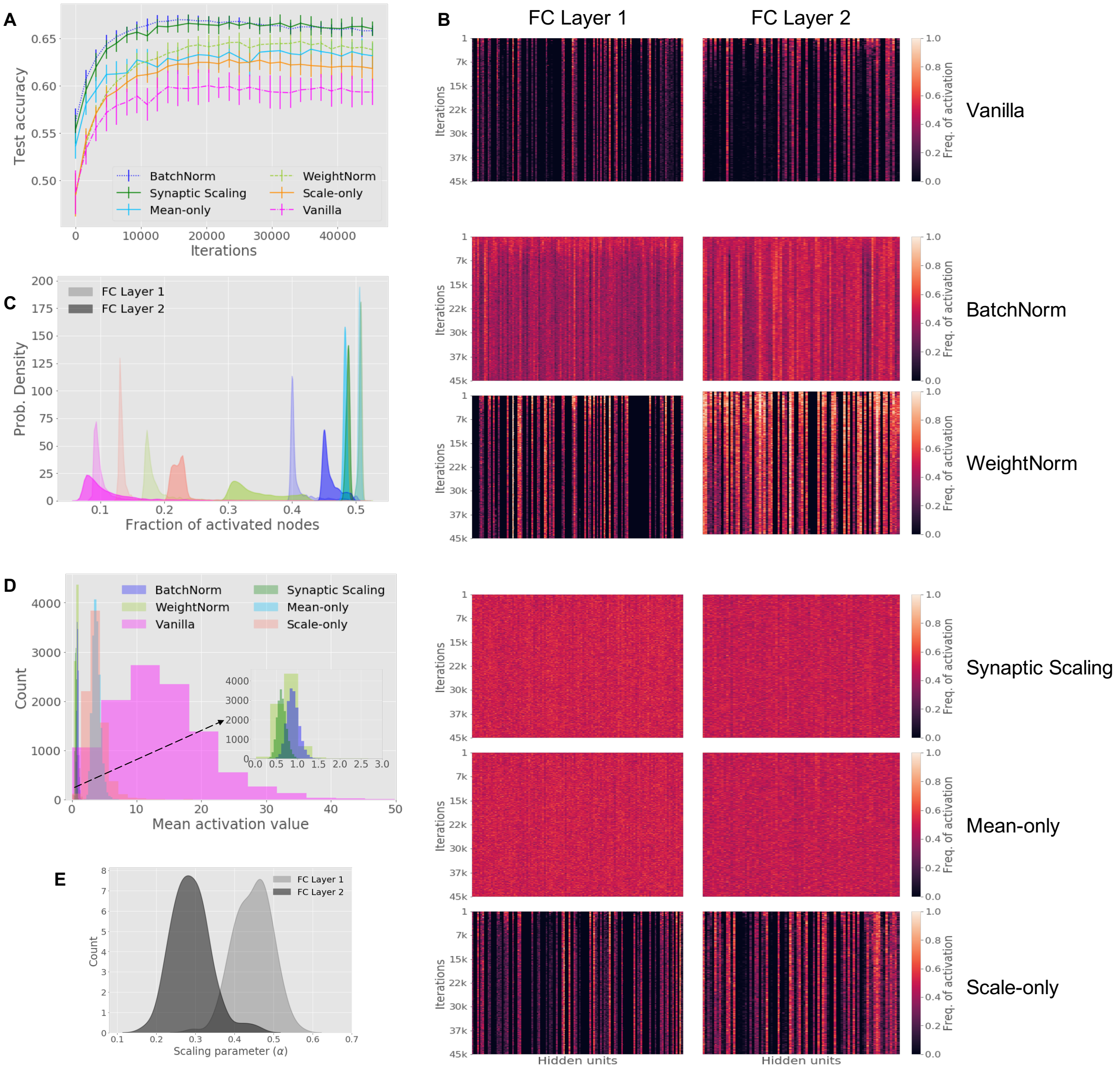

Figure 2: **[CIFAR-10] Normalization increases performance and drives neural networks towards a "homeostatic" state.** A) Test accuracy (y-axis) versus training iteration (x-axis). Error bars show standard deviation over 10 random initializations. BatchNorm and Synaptic Scaling achieve higher accuracy at the beginning and at the end of training compared to all other methods, including Vanilla. B) The frequency of each hidden unit (columns) being activated over all inputs in a batch, computed on every 100th training iteration (rows). Heatmaps are shown for hidden units in both fully-connected (FC) layers. C) Histogram of the fraction of activated hidden nodes, averaged over each batch. Lighter and darker colored histograms show the first and second FC layers, respectively. D) Histogram of the mean activation values for hidden units in the first FC layer, calculated using the test dataset. E) Distribution of the trained α parameters for Synaptic Scaling, for each FC layer.

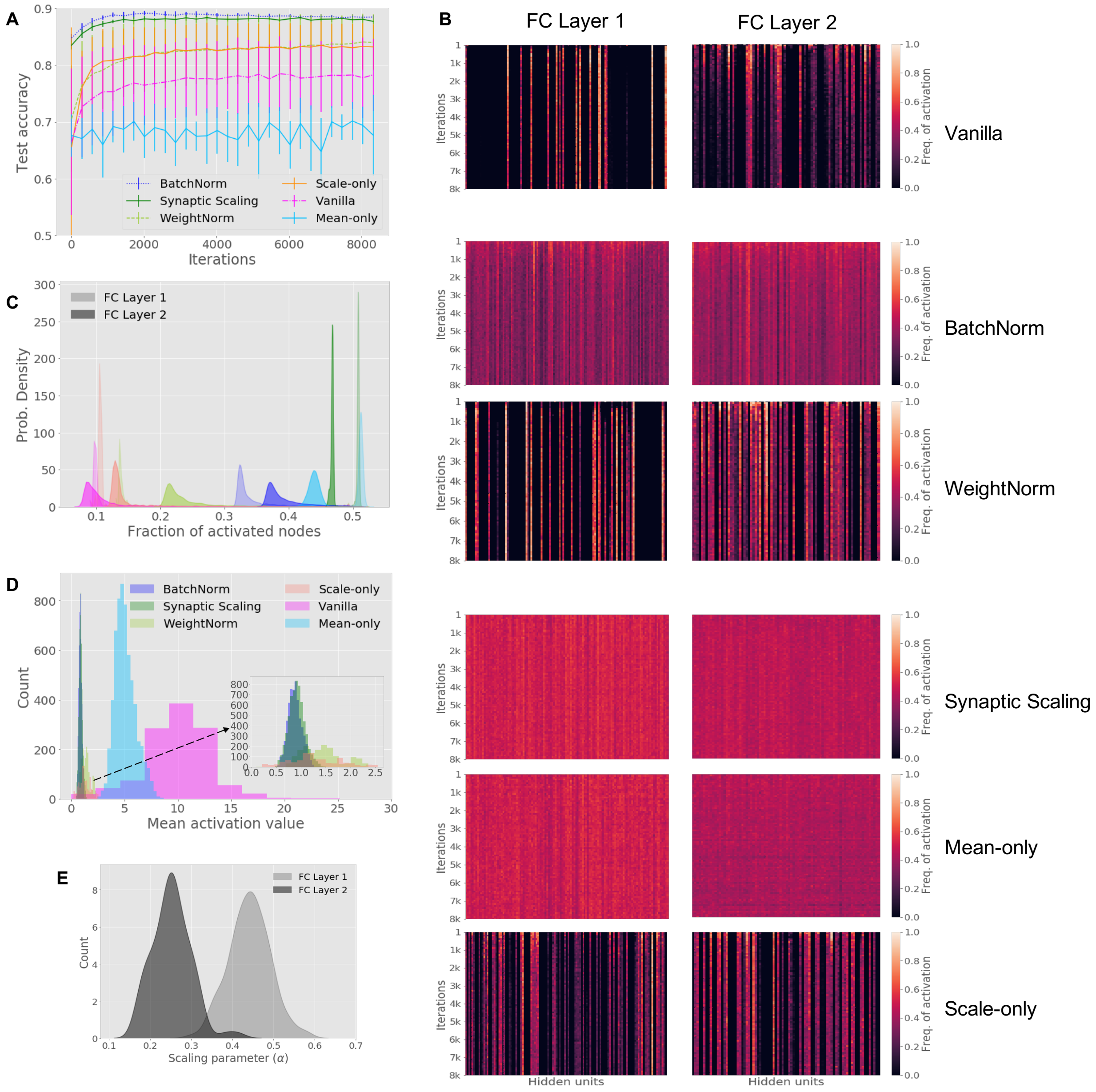

Figure 3: **[SVHN] Similar benefits of normalization on a second dataset.** Synaptic Scaling and BatchNorm have the highest classification accuracy (A), and generate homeostatic representations (B,C: fraction of activated hidden units; D: mean activation values for hidden units). See Figure 2 caption for detailed panel descriptions.

References

[1] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, page 9–50, Berlin, Heidelberg, 1998. Springer-Verlag.

[2] Guillaume Desjardins, Karen Simonyan, Razvan Pascanu, and Koray Kavukcuoglu. Natural neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2071–2079, Cambridge, MA, USA, 2015. MIT Press.

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.

[4] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2483–2493. Curran Associates, Inc., 2018.

[5] Jonas Kohler, Hadi Daneshmand, Aurelien Lucchi, Thomas Hofmann, Ming Zhou, and Klaus Neymeyr. Exponential convergence rates for batch normalization: The power of length-direction decoupling in non-convex optimization. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 806–815. PMLR, 16–18 Apr 2019.

[6] Xiaoxia Wu, Edgar Dobriban, Tongzheng Ren, Shanshan Wu, Zhiyuan Li, Suriya Gunasekar, Rachel Ward, and Qiang Liu. Implicit regularization of normalization methods, 2019.

[7] Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. Towards understanding regularization in batch normalization. In *arXiv preprint arXiv:1809.00846*, page 17, 2018.

[8] T. Poggio, Q. Liao, and A. Banburski. Complexity control by gradient descent in deep networks. *Nat Commun*, 11(1):1027, Feb 2020.

[9] M. Carandini and D. J. Heeger. Normalization as a canonical neural computation. *Nat. Rev. Neurosci.*, 13(1):51–62, Nov 2011.

[10] A. I. Weber, K. Krishnamurthy, and A. L. Fairhall. Coding Principles in Adaptation. *Annu Rev Vis Sci*, 5:427–449, Sep 2019.

[11] N. C. Rabinowitz, B. D. Willmore, J. W. Schnupp, and A. J. King. Contrast gain control in auditory cortex. *Neuron*, 70(6):1178–1191, Jun 2011.

[12] S. R. Olsen, V. Bhandawat, and R. I. Wilson. Divisive normalization in olfactory population codes. *Neuron*, 66(2):287–299, Apr 2010.

[13] R. Shapley. Retinal physiology: adapting to the changing scene. *Curr. Biol.*, 7(7):R421–423, Jul 1997.

[14] R.W. Rodieck. *The First Steps in Seeing*. Sinauer, 1998.

[15] V. Mante, R. A. Frazor, V. Bonin, W. S. Geisler, and M. Carandini. Independence of luminance and contrast in natural scenes and in the early visual system. *Nat. Neurosci.*, 8(12):1690–1697, Dec 2005.

[16] G. Turrigiano. Homeostatic synaptic plasticity: local and global mechanisms for stabilizing neuronal function. *Cold Spring Harb Perspect Biol*, 4(1):a005736, January 2012.

[17] W.B. Cannon. *The wisdom of the body.* W.W. Norton & Company, inc., 1932.

[18] S. B. Laughlin and T. J. Sejnowski. Communication in neuronal networks. *Science*, 301(5641):1870–1874, September 2003.

[19] G. G. Turrigiano and S. B. Nelson. Homeostatic plasticity in the developing nervous system. *Nat. Rev. Neurosci.*, 5(2):97–107, February 2004.

[20] A. R. Houweling, M. Bazhenov, I. Timofeev, M. Steriade, and T. J. Sejnowski. Homeostatic synaptic plasticity can explain post-traumatic epileptogenesis in chronically isolated neocortex. *Cereb. Cortex*, 15(6):834–845, June 2005.

[21] H. Yu, D. Sternad, D. M. Corcos, and D. E. Vaillancourt. Role of hyperactive cerebellum and motor cortex in Parkinson's disease. *Neuroimage*, 35(1):222–233, March 2007.

[22] A. Bakker, G. L. Krauss, M. S. Albert, C. L. Speck, L. R. Jones, C. E. Stark, M. A. Yassa, S. S. Bassett, A. L. Shelton, and M. Gallagher. Reduction of hippocampal hyperactivity improves cognition in amnestic mild cognitive impairment. *Neuron*, 74(3):467–474, May 2012.

[23] J. Wondolowski and D. Dickman. Emerging links between homeostatic synaptic plasticity and neurological disease. *Front Cell Neurosci*, 7:223, 2013.

[24] D. Hassabis, D. Kumaran, C. Summerfield, and M. Botvinick. Neuroscience-Inspired Artificial Intelligence. *Neuron*, 95(2):245–258, Jul 2017.

[25] N. A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[26] C. F. Stevens. What the fly's nose tells the fly's brain. *Proc. Natl. Acad. Sci. U.S.A.*, 112(30):9460–9465, Jul 2015.

[27] Bettina Malnic, Junzo Hirono, Takaaki Sato, and Linda B Buck. Combinatorial receptor codes for odors. *Cell*, 96(5):713–723, 1999.

[28] Z. Wang, A. A. Stocker, and D. D. Lee. Efficient Neural Codes That Minimize Lp Reconstruction Error. *Neural Comput*, 28(12):2656–2686, 12 2016.

[29] D. Ganguli and E. P. Simoncelli. Implicit encoding of prior probabilities in optimal neural populations. *Adv Neural Inf Process Syst*, 2010:658–666, 2010.

[30] D. Ganguli and E. P. Simoncelli. Efficient sensory encoding and Bayesian inference with heterogeneous neural populations. *Neural Comput*, 26(10):2103–2134, Oct 2014.

[31] Zhuo Wang, Alan A Stocker, and Daniel D Lee. Optimal neural tuning curves for arbitrary stimulus distributions: Discrimax, infomax and minimum l_p loss. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2168–2176. Curran Associates, Inc., 2012.

[32] Simon Laughlin. A simple coding procedure enhances a neuron's information capacity. *Zeitschrift für Naturforschung c*, 36(9-10):910–912, 1981.

[33] Jan Kukacka, Vladimir Golkov, and Daniel Cremers. Regularization for deep learning: A taxonomy. *CoRR*, abs/1710.10686, 2017.

[34] Behnam Neyshabur, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro. Geometry of optimization and implicit regularization in deep learning, 2017.

[35] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014.

[36] Kevin J. Lang and Geoffrey E. Hinton. *Dimensionality Reduction and Prior Knowledge in E-Set Recognition*, page 178–185. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[37] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[38] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7694–7705. Curran Associates, Inc., 2018.

[39] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[40] K. Pozo and Y. Goda. Unraveling mechanisms of homeostatic synaptic plasticity. *Neuron*, 66(3):337–351, May 2010.

[41] G. G. Turrigiano. The self-tuning neuron: synaptic scaling of excitatory synapses. *Cell*, 135(3):422–435, October 2008.

[42] E. L. Bienenstock, L. N. Cooper, and P. W. Munro. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *J. Neurosci.*, 2(1):32–48, Jan 1982.

[43] W. Zhang and D. J. Linden. The other side of the engram: experience-driven changes in neuronal intrinsic excitability. *Nat. Rev. Neurosci.*, 4(11):885–900, Nov 2003.

[44] G. Turrigiano. Too many cooks? Intrinsic and synaptic homeostatic mechanisms in cortical circuit refinement. *Annu. Rev. Neurosci.*, 34:89–103, 2011.

[45] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 901–909. Curran Associates, Inc., 2016.

[46] V. N. Murthy, T. Schikorski, C. F. Stevens, and Y. Zhu. Inactivity produces increases in neurotransmitter release and synapse size. *Neuron*, 32(4):673–682, November 2001.

[47] G. G. Turrigiano. The dialectic of Hebb and homeostasis. *Philos. Trans. R. Soc. Lond., B, Biol. Sci.*, 372(1715), 03 2017.

[48] G. Tononi and C. Cirelli. Time to be SHY? Some comments on sleep and synaptic homeostasis. *Neural Plast.*, 2012:415250, 2012.

[49] Giri P Krishnan, Timothy Tadros, Ramyaa Ramyaa, and Maxim Bazhenov. Biologically inspired sleep algorithm for artificial neural networks, 2019.

[50] S. Royer and D. Pare. Conservation of total synaptic weight through balanced synaptic depression and potentiation. *Nature*, 422(6931):518–522, Apr 2003.

[51] I. Rabinowitch and I. Segev. Two opposing plasticity mechanisms pulling a single synapse. *Trends Neurosci.*, 31(8):377–383, Aug 2008.

[52] M. Chistiakova, N. M. Bannon, J. Y. Chen, M. Bazhenov, and M. Volgushev. Homeostatic role of heterosynaptic plasticity: models and experiments. *Front Comput Neurosci*, 9:89, 2015.

[53] H. Li, Y. Li, Z. Lei, K. Wang, and A. Guo. Transformation of odor selectivity from projection neurons to single mushroom body neurons mapped with dual-color calcium imaging. *Proc. Natl. Acad. Sci. U.S.A.*, 110(29):12084–12089, Jul 2013.

[54] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv e-prints*, page arXiv:1607.06450, Jul 2016.

[55] C. F. Stevens. A statistical property of fly odor responses is conserved across odors. *Proc. Natl. Acad. Sci. U.S.A.*, 113(24):6737–6742, 06 2016.

[56] E. Slomowitz, B. Styr, I. Vertkin, H. Milshtein-Parush, I. Nelken, M. Slutsky, and I. Slutsky. Interplay between population firing stability and single neuron dynamics in hippocampal networks. *Elife*, 4, 2015.

[57] Y. Ziv, L. D. Burns, E. D. Cocker, E. O. Hamel, K. K. Ghosh, L. J. Kitch, A. El Gamal, and M. J. Schnitzer. Long-term dynamics of CA1 hippocampal place codes. *Nat. Neurosci.*, 16(3):264–266, March 2013.

[58] A. A. Prinz, D. Bucher, and E. Marder. Similar network activity from disparate circuit parameters. *Nat. Neurosci.*, 7(12):1345–1352, Dec 2004.

[59] G. Buzsaki and K. Mizuseki. The log-dynamic brain: how skewed distributions affect network operations. *Nat. Rev. Neurosci.*, 15(4):264–278, Apr 2014.

[60] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[61] Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, pages 1–12, 2020.

[62] Annelise Joseph and Gina G Turrigiano. All for one but not one for all: excitatory synaptic scaling and intrinsic excitability are coregulated by camkiv, whereas inhibitory synaptic scaling is under independent control. *Journal of Neuroscience*, 37(28):6778–6785, 2017.

[63] F. Zenke and W. Gerstner. Hebbian plasticity requires compensatory processes on multiple timescales. *Philos. Trans. R. Soc. Lond., B, Biol. Sci.*, 372(1715), 03 2017.

[64] T. Keck, T. Toyoizumi, L. Chen, B. Doiron, D. E. Feldman, K. Fox, W. Gerstner, P. G. Haydon, M. H?bener, H. K. Lee, J. E. Lisman, T. Rose, F. Sengpiel, D. Stellwagen, M. P. Stryker, G. G. Turrigiano, and M. C. van Rossum. Integrating Hebbian and homeostatic plasticity: the current state of the field and future research directions. *Philos. Trans. R. Soc. Lond., B, Biol. Sci.*, 372(1715), 03 2017.

[65] K. Fox and M. Stryker. Integrating Hebbian and homeostatic plasticity: introduction. *Philos. Trans. R. Soc. Lond., B, Biol. Sci.*, 372(1715), 03 2017.

[66] G. W. Davis. Homeostatic control of neural activity: from phenomenology to molecular design. *Annu. Rev. Neurosci.*, 29:307–323, 2006.

[67] E. Marder and J. M. Goaillard. Variability, compensation and homeostasis in neuron and network function. *Nat. Rev. Neurosci.*, 7(7):563–574, Jul 2006.

[68] G. W. Davis. Homeostatic signaling and the stabilization of neural function. *Neuron*, 80(3):718–728, Oct 2013.

[69] Yuxin Wu and Kaiming He. Group normalization. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[70] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv e-prints*, page arXiv:1607.08022, Jul 2016.

[71] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 971–980. Curran Associates, Inc., 2017.

[72] Devansh Arpit, Yingbo Zhou, Bhargava U. Kota, and Venu Govindaraju. Normalization propagation: A parametric technique for removing internal covariate shift in deep networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1168–1176. JMLR.org, 2016.

[73] Tapani Raiko, Harri Valpola, and Yann Lecun. Deep learning made easier by linear transformations in perceptrons. In Neil D. Lawrence and Mark Girolami, editors, *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 924–932, La Palma, Canary Islands, 21–23 Apr 2012. PMLR.

[74] X. Pitkow and M. Meister. Decorrelation and efficient coding by retinal ganglion cells. *Nat. Neurosci.*, 15(4):628–635, Mar 2012.

[75] D. J. Graham, D. M. Chandler, and D. J. Field. Can the theory of ”whitening” explain the center-surround properties of retinal ganglion cell receptive fields? *Vision Res.*, 46(18):2901–2913, Sep 2006.

[76] C. Pozzorini, R. Naud, S. Mensi, and W. Gerstner. Temporal whitening by power-law adaptation in neocortical neurons. *Nat. Neurosci.*, 16(7):942–948, Jul 2013.

[77] C. G. Huang, Z. D. Zhang, and M. J. Chacron. Temporal decorrelation by SK channels enables efficient neural coding and perception of natural stimuli. *Nat Commun*, 7:11353, Apr 2016.

[78] A. A. Wanner and R. W. Friedrich. Whitening of odor representations by the wiring diagram of the olfactory bulb. *Nat. Neurosci.*, 23(3):433–442, 03 2020.

[79] A. Maffei and A. Fontanini. Network homeostasis: a matter of coordination. *Curr. Opin. Neurobiol.*, 19(2):168–173, Apr 2009.