Surrogate Gradient Design for LIF networks

Anonymous authors
Paper under double-blind review

Abstract

Spiking Neuromorphic Computing uses binary activity to improve Artificial Intelligence energy efficiency. However, the non-smoothness of binary activity requires approximate gradients, known as Surrogate Gradients (SG), to close the performance gap with Deep Learning. Several SG have been proposed in the literature, but it remains unclear how to determine the best SG for a given task and network. Good performance can be achieved with most SG shapes, after an extensive search of hyper-parameters that can be costly. Thus, we aim at experimentally and theoretically define the best SG across different stress tests, to reduce future need of grid search. Here we first show that the derivative of the fast sigmoid outperforms other SG across tasks and networks, for a wide range of learning rates. Secondly, we focus on the Leaky Integrate and Fire (LIF) spiking neural model, and we show that a SG with low dampening, high sharpness, and low tail fatness, systematically leads to higher accuracy. Thirdly, we observe that the Orthogonal initialization leads the LIF to higher accuracy with most SG. Fourthly, we note that high initial firing rates, combined with a sparsity encouraging loss term, can lead to better generalization, depending on the SG shape. Finally, we provide a theoretical solution, inspired by Glorot and He initializations, to reduce the need of extensive grid-search, to find an SG and initialization that experimentally result in improved accuracy.

1 Introduction

In any learning algorithm we desire to reduce the time spent finding good hyper-parameters. Using Spiking Neuromorphic Computing to achieve higher energy efficiency (Henderson et al., 2020; Blouw et al., 2019; Davies et al., 2021), requires an additional hyper-parameter to choose: what is the best gradient approximation. This is so because we use binary signals (Lapique, 1907; Izhikevich, 2003), with zero gradient almost everywhere. Thus, to train successfully we need to turn the true derivative, a delta function, into a smoother shape, the *Surrogate Gradient* (SG) (Esser et al., 2016; Zenke and Ganguli, 2018; Bellec et al., 2018). However, its fine details depend on e.g. the network, the task, and the initialization scheme. A method for SG design could reduce the time spent in trial and error.

The standard approach is to pick one SG for all the experiments (Bohte, 2011; Hubara et al., 2016a; Bellec et al., 2018; Zenke and Ganguli, 2018; Zenke and Vogels, 2021; Yin et al., 2021), possibly exploring the effect of changing a width factor (sharpness) (Zenke and Ganguli, 2018) or a height factor (dampening) (Bellec et al., 2018). Only in the past five years, the possibility of choosing an optimal SG has been considered (Neftci et al., 2019; Zenke and Vogels, 2021). Moreover, even if many SG can achieve good performance (Zenke and Vogels, 2021), some shapes have more chances to fail training or achieve lower accuracy.

To form a better view on the SG training variability, we test the effect of the SG on training more complex neural models and tasks, and we measure the increase in sensitivity with complexity. We then focus on arguably the simplest spiking neural model, the Leaky and Integrate and Fire (LIF) neuron (Lapique, 1907; Gerstner et al., 2014). We use it to assess the effect of fine details of the SG, and we closely look at (1) its dampening, (2) its sharpness, and (3) how fast it decays to zero, i.e. tail-fatness. We also asses the effect of the initialization

scheme on each SG. Moreover, we observe that low initial sparsity, can help generalization, depending on the SG. Finally, in the spirit of Glorot and He initializations (Glorot and Bengio, 2010; He et al., 2015), we look for conditions that can keep the representations and gradients constant with time. We show that these conditions provide hyper-parameters that result in improved performance without additional hyper-parameter grid-search.

This paper further clarifies the role of SG shape in training variability of spiking networks. Furthermore, it proposes a direction for SG design, based on theories of gradient stability (Glorot and Bengio, 2010; He et al., 2015). Our contribution is five-fold:

- We observe that the derivative of the fast-sigmoid outperforms other SG across tasks and networks;
- Low dampening, high sharpness and low tail-fatness, lead to higher accuracy on the LIF network;
- Orthogonal initialization leads to higher accuracy on the LIF network;
- High initial firing rate can promote generalization with low final firing rate;
- We provide a theoretical method for SG choice based on bounding representations that improves experimental performance.

2 Preliminaries

2.1 Initialization Schemes

The initial network parameters have a strong impact on training speed (Hanin and Rolnick, 2018) and peak performance (Glorot and Bengio, 2010; He et al., 2015). The theory focuses on feed-forward networks (FFN), given their mathematical tractability. FFNs are defined as $\boldsymbol{y}_l = \boldsymbol{b}_l + W_l\sigma(\boldsymbol{y_{l-1}})$, where $\boldsymbol{y}_0$ is the data, $\boldsymbol{y}_L$ the network output for depth L, $\sigma(\cdot)$ an activation and $\boldsymbol{b}_l \in \mathbb{R}^{n_l}, W_l \in \mathbb{R}^{n_l \times n_{l-1}}$ are the biases and weights of each layer, where n_l is the size of layer l. Typically, biases are sampled as zero and weights such that $Mean[W_l] = 0$ and $Var[W_l] = c_l$. The theory recommends a $1/c_l \propto n_l$ to avoid exploding representations (Glorot and Bengio, 2010; He et al., 2015). (Glorot and Bengio, 2010) finds $Var[W_l] = 2/(n_{l-1}+n_l)$ as optimal for linear networks, $\sigma(y) = y$, known as *Glorot* initialization, while (He et al., 2015) finds $Var[W_l] = 2/n_{l-1}$ for $ReLU$ networks, $\sigma(y) = max(0, y)$, known as *He* initilization. Instead, (Saxe et al., 2014) finds a column orthogonal W_l as optimal for mean squared error linear networks, known as *Orthogonal* initialization. Usually W_l elements are drawn from a uniform or a normal distribution. We propose the BiGamma distribution, that we define as $w_{ij} \sim \text{Gamma}(w; \alpha, \beta)/2 + \text{Gamma}(-w; \alpha, \beta)/2$, Fig. 1. The BiGamma keeps the optimal variance and orthogonality without ever sampling a zero.

Theoretical justification for recurrent networks initialization has been proposed for the LSTM (Mehdipour Ghazi et al., 2019), where constraints on the weights are found. On spiking networks, an initialization theory is missing, since arguments such as the Echo State Network (Jaeger et al., 2007) do not apply to non convex activations, or activations without a slope one regime. In practice, (Zenke and Vogels, 2021) samples a $Var[W_l] = 1/3n_{l-1}$ Uniform, while (Bellec et al., 2018) samples a $Var[W_l] = 1/n_{l-1}$ Normal distribution.

2.2 Neural Models

Arguably the simplest neuromorphic model is the LIF (Lapique, 1907; Gerstner et al., 2014; Woźniak et al., 2020). When upgraded with a dynamical threshold to maintain longer memories, it is known as the Adaptive LIF (ALIF) (Gerstner et al., 2014; Bellec et al., 2018). To understand SG dependence with network complexity, we compare LIF to ALIF and we propose the spiking LSTM (sLSTM), App. G. We define the latter by changing the LSTM (Hochreiter and Schmidhuber, 1997) activations for neuromorphic equivalents.

We focus on the discrete time LIF, defined as $\boldsymbol{y}_t = \boldsymbol{\alpha}_{decay}\boldsymbol{y}_{t-1}(1 - \boldsymbol{x}_{t-1}) + \boldsymbol{i}_t$ where $\boldsymbol{i}_t = W_{rec}\boldsymbol{x}_{t-1} + W_{in}\boldsymbol{z}_t + \boldsymbol{b}$, and y_t is the neuron membrane voltage, to keep (Glorot and Bengio, 2010; He et al., 2015) notation. We define $x_t = \sigma(y_t) = \tilde{H}(y_t - \vartheta) = \tilde{H}(v_t)$ as the spiking activity, where ϑ is the spiking threshold, $v_t = y_t - \vartheta$ the centered voltage, and $\tilde{H}(v_t)$ a Heaviside function with SG. The term $(1 - \boldsymbol{x}_{t-1})$ represents a hard reset, that takes

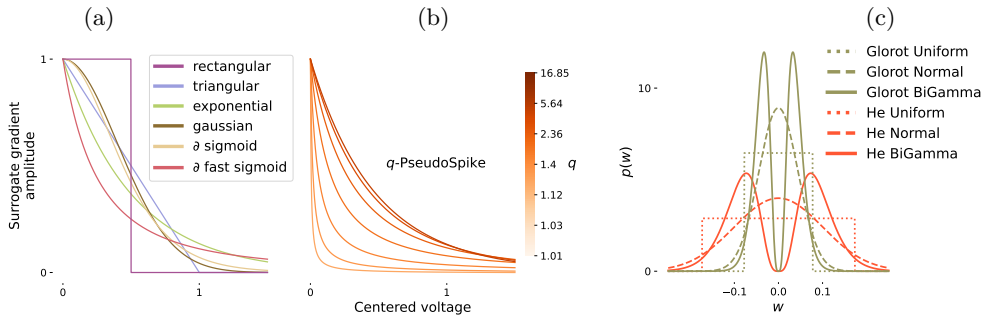

Figure 1: **Surrogate Gradient shapes and initialization distributions.** Panel (a) shows the SG used in this work, and (b) the tail dependence of the q-PseudoSpike SG for $q \in [1.01, 16.85]$. The SG considered are symmetrical around $v_t = y_t - \vartheta = 0$, so we only plot half the curve (centered voltage $v_t > 0$). Panel (c) shows the weight sampling distributions used, Gaussian (dashed), Uniform (dotted) and BiGamma (solid) distributions, with *He*, dark orange. and *Glorot* initialization, olive green, for a weight shape of $(n_0, n_1) = (200, 300)$.

the voltage to zero after firing. The input z_t can represent the data, or a layer below. It is common to write $\alpha_{decay} = 1 - \frac{dt}{\tau_m}$, where dt is the computation time, τ_m the membrane potential time constant, and to multiply the other terms by biologically meaningful constants. Here we compress them for cleanliness. Each neuron can have its own speed α_{decay}, intrinsic current b and ϑ. Here, all of them are learnable.

We denote vectors as $\boldsymbol{a}$, matrices as A, and their elements as a. The matrix $W_{rec} \in \mathbb{R}^{n_l \times n_l}$ connects neurons in the same layer, with zero diagonal, and $W_{in} \in \mathbb{R}^{n_l \times n_{l-1}}$ connects the layer with the layer below, or the data if $l = 0$. We use curved brackets $A(\cdot)$ for functions, and square brackets $A[\cdot]$ for functionals that depend on a probability distribution. We use interchangeably $\overline{x} = Mean[x]$, $\hat{x} = Max[x]$, and $\check{x} = Min[x]$. We use θ for any parameter. In a stack of layers, we add an index l to each parameter and variable. Since the equation only depends on the previous timestep and layer, the probability distribution is a Markov chain in time and depth. Therefore the statistics we discuss are computed element-wise with respect to the distribution $p(y_{t,l}|t,l) = p(\boldsymbol{y}_{t-1,l}, \boldsymbol{z}_{t,l-1}, W_{rec,l}, W_{in,l}, b_l, \alpha_{decay,l}, \vartheta_l|t,l)$.

2.3 Surrogate Gradients

The Heaviside $\tilde{H}(v)$, is zero for $v < 0$ and one for $v \geq 0$. The tilde means that a SG is used for training, defined as $\tilde{H}'(v) = \gamma f(\beta \cdot v)$, where β is the sharpness, γ the dampening, f is the shape of choice and $\cdot$ the product. Unless explicitly stated, dampening and sharpness are set to one. A high sharpness, mostly passes the gradient for v close to zero, while low sharpness also passes the gradient for a wider range of voltages. Therefore, the gradient can pass when the neuron has not fired. The SG shapes f we investigated are (1) rectangular (Hubara et al., 2016b), (2) triangular (Esser et al., 2016; Bellec et al., 2018), (3) exponential (Shrestha and Orchard, 2018), (4) gaussian (Yin et al., 2020), (5) the derivative of a sigmoid (Zenke and Vogels, 2021), and (6) the derivative of a fast-sigmoid, also SuperSpike (Zenke and Ganguli, 2018). We propose a generalization of the derivative of the fast-sigmoid, that we call q-PseudoSpike SG. Its tail fatness is controlled by a hyper-parameter q and we use it to study tail dependence in section 3.2. More in Fig. 1 and App. D.

2.4 Datasets

More details on the datasets can be found in App. A.

Spike Latency MNIST (sl-MNIST): the MNIST digits (LeCun et al., 1998) pixels (10 classes) are rescaled between zero and one, presented as a flat vector, and each vector value x is transformed into a spike timing using the transformation $T(x) = \tau_{eff} \log(\frac{x}{x-\vartheta})$ for $x > \vartheta$ and $T(x) = \infty$ otherwise, with $\vartheta = 0.2, \tau_{eff} = 50$ms (Zenke and Vogels, 2021). The network input is a sequence of 50ms, 784 channels (28×28), with one spike per row.

Spiking Heidelberg Digits (SHD): is a dataset developed to benchmark spiking neural networks (Cramer et al., 2020). It is based on the Heidelberg Digits (HD) audio dataset which comprises 20 classes of spoken digits, from zero to nine, in English and German, spoken by 12 individuals. These audio signals are encoded into spikes through an artificial model of the inner ear and parts of the ascending auditory pathway.

PennTreeBank (PTB): is a language modelling task. The PennTreeBank dataset (Marcus et al., 1993), is a large corpus of American English texts. We perform next time-step prediction at the word level. The vocabulary consists of 10K words, which we consider as 10K classes. The one hot encoding of words can be seen as a spiking representation, even if it is the standard representation in the non neuromorphic literature.

2.5 Training Details

Our networks are composed of two recurrent layers. The output of the first is the input to the second, and the output of the second is fed to a linear readout. Our LIF network has 128 neurons per layer on the sl-MNIST task, 256 on SHD, but one layer of 1700 and another with 300 on PTB, in line with (Woźniak et al., 2020). On the SHD task, the ALIF has 256 neurons and the sLSTM 85, to keep a comparable number of 350K parameters. We use crossentropy loss. The optimizer had a strong effect, where Stochastic Gradient Descent (Robbins and Monro, 1951; Kiefer and Wolfowitz, 1952) was often not able to learn, and AdaM (Kingma and Ba, 2015) would perform consistently worse than AdaBelief (Zhuang et al., 2020). AdaBelief hyper-parameters are set to the default, as in (Radford et al., 2018; Zenke and Vogels, 2021). The remaining hyper-parameters are reported in App. A. Unless explicitly stated, we use Glorot Uniform initialization. Each experiment is run 4 times and the mean and standard deviation are reported. Experiments are run in single Tesla V100 NVIDIA GPUs. We call our metric the mode accuracy: the network predicts the target at every timestep, and the chosen class is the one that fired the most for the longest.

3 Results

3.1 The derivative of the fast-sigmoid outperforms other SG across tasks and networks

3.1.1 Methodology: Sensitivity to Complexity

We show the SG training sensitivity dependence with task and network complexity in Fig. 2, g-h). We estimate the task complexity by the number of classes. Thus, if $C_T(\cdot)$ measures task complexity, $C_T(sl\text{-}MNIST) < C_T(SHD) < C_T(PTB)$. We quantify neural complexity as in (Yin et al., 2021), Tab. 1. In essence, if $C_M(\cdot)$ measures model complexity, then $C_M(LIF) < C_M(ALIF) < C_M(sLSTM)$. To have comparable losses across tasks and networks, we normalize their validation values between 0 and 1. We remove the lowest loss achieved by a network in a task for any seed and learning rate, and divide by the distance between the highest and lowest loss. We call the result the *post-training normalized loss*. We call *sensitivity* the standard deviation of the *post-training normalized perplexity* across SG, for each learning rate. We report mean and standard deviation across learning rates.

3.1.2 Experiments

Each SG success depends on the architecture (LIF, ALIF and spiking LSTM) and task (sl-MNIST, SHD, PTB) complexity, as shown in Fig. 2. We run a grid search over learning rates and SG shapes. The sensitivity to the task is shown in the upper panels, for the LIF network. We see that different SG agree on the optimal learning rate. We also see that the ∂ fast-sigmoid performs well for a wider range of learning rates. The rectangular SG is competitive on some tasks, but fails to learn with most learning rates on PTB. Then we focus on network sensitivity, fixing the SHD task, lower panels. The triangular SG performs similarly to the exponential on the LIF network, while it underperforms on ALIF, and fails on sLSTM. The exponential SG matches the best SG on both the LIF and the sLSTM, but not on the ALIF. Surprisingly, the sLSTM lags behind the LIF and ALIF, with a comparable

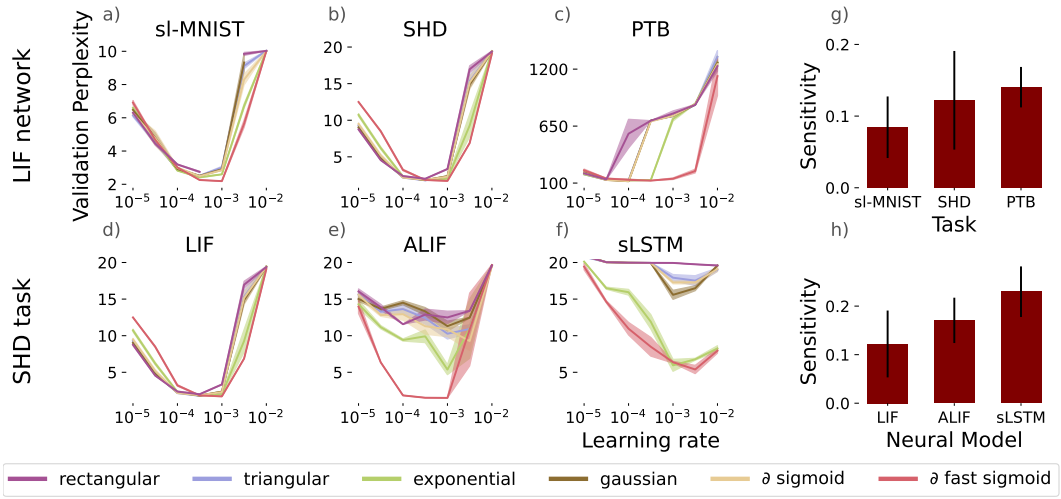

Figure 2: **The derivative of the fast-sigmoid outperforms other SG across tasks and networks**. Grid search over SG shapes, learning rates, tasks and networks. Perplexity is a loss, so, the lower the better. We report lowest validation perplexity after converged training. Panels a-f) show perplexity (y-axis), against learning rate (x-axis). In a-c) we fix the LIF network and change task, while in d-f) we fix the SHD task and change network. Panels g-h) show SG sensitivity (y-axis) against task and neural model (x-axis). Plots b) and d) are repeated for clarity. a-f) Different SG roughly agree on the best learning rate. The best loss is achieved by the ∂ fast-sigmoid, also the most resilient to changes in the learning rate, as shown in (Zenke and Vogels, 2021). g-h) The more complex the task or the network, the more variance in the performance we see across SG choices and learning rates.

number of parameters. The gating mechanism devised to keep the LSTM representations from exploding exponentially, are not relevant anymore for a Heaviside that cannot explode exponentially, and might have become a computational burden. Fig. 2, g-h), confirm that there is a correlation between task and network complexity, and SG sensitivity.

3.2 Low dampening, high sharpness and low tail fatness, lead to higher accuracy on the LIF network

Fig. 3 shows the accuracy of the LIF network on the sl-MNIST task, for different dampening, sharpness and tail-fatness values. When we vary the dampening, we set the sharpness to 1, and viceversa, which are the default in the SG literature. Each SG has its own tail decay: inverse quadratic for the ∂ fast-sigmoid, no tail for the triangular and rectangular SG, and exponential decays for the remaining. Low dampening is preferred by all SG, with an optimum at 0.8, while high sharpness are preferred. Interestingly, the accuracy of the ∂ fast-sigmoid degrades less with suboptimal dampening and sharpness. The vertical dashed lines are predicted by our theoretical method, condition (III) for the dampening and (IV) for the sharpness of an exponential SG. We observe that they find a dampening and a sharpness with high experimental accuracies. This supports the claim that to reduce hyper-parameter search of dampening and sharpness is possible. We use our q-PseudoSpike SG to study the dependence with the tail-fatness for the LIF network on the sl-MNIST task, panel (c) in Fig. 3. All tail-fatness values perform reasonably well, but there is a maximum at $q = 1.56$, smaller than the $q = 2$ of the ∂ fast-sigmoid.

3.3 Orthogonal initialization leads to higher accuracy

Fig. 6 App. C, shows the initialization scheme effect on SG training, for the LIF network on the SHD task. We compare the *Glorot* ($2/Var[W_l] = n_{l-1} + n_l$, (Glorot and Bengio, 2010)) with the *He* ($2/Var[W_l] = n_{l-1}$, (He et al., 2015)) and the *Orthogonal* initialization (W_{rec} and W_{in} as column orthogonal matrices, (Saxe et al., 2014)). We use three sampling distri-

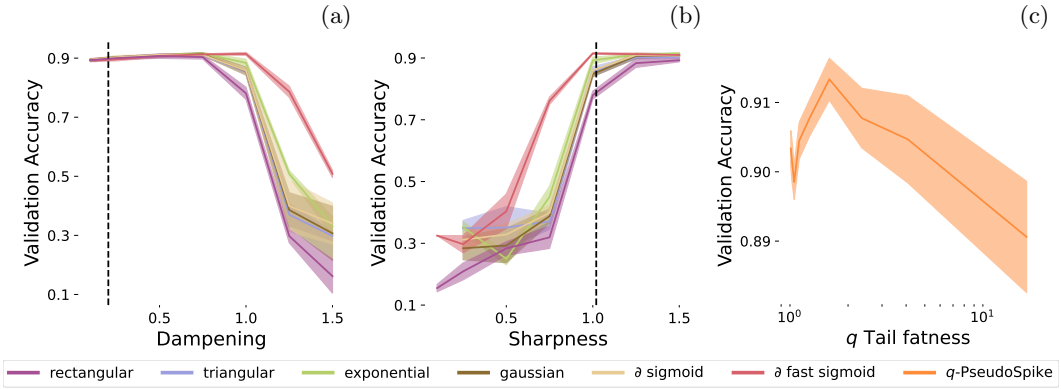

Figure 3: **Low dampening, high sharpness and low tail-fatness, lead to higher accuracy on the LIF network.** Analysis done on the LIF network over the sl-MNIST task. Panel (a) shows the sensitivity to dampening, sharpness set to 1, and viceversa on (b), both with the given SG tail-fatness. Dampenings higher than 1 worsen performance while the pattern is the opposite for sharpness. Dashed vertical lines are our theoretical prediction for the exponential SG, (III) for the dampening ($\gamma = 0.20 \pm 0.02$) and (IV) for the sharpness ($\beta = 1.02 \pm 0.17$), which agree empirically as some of the best the dampening and sharpness. Panel (c) shows tail-fatness sensitivity of the q-PseudoSpike SG, for $\beta = \gamma = 1$. The dependence with the tail fatness is more complex, with an optimal fatness of $q = 1.56$.

butions: Uniform, Gaussian and BiGamma, Fig. 1. Orthogonal Uniform is not considered since after orthogonalization, the distribution was not Uniform anymore. We use the same initialization for W_{rec} and W_{in}. The best SG with *Glorot* Uniform is the ∂ fast-sigmoid, while with *Orthogonal* Normal is the exponential. *He* gives the best outliers, and *Orthogonal* Normal gives the best mean accuracy. The BiGamma reduces the result variance.

3.4 High initial firing rate can promote generalization with low final firing rate

3.4.1 Methodology: Sparsity and Binarity Roles in Generalization

In the neuromorphic literature training success is judged by (1) training performance and (2) activity sparsity. We show in Sec. 3.4.2, that high initial firing rate can induce better generalization in synergy with a sparsity encouraging loss term (SELT). However, the energy gain of spiking networks also comes from their binary activity. A matrix-vector multiplication, with a $\mathbb{R}^{m \times n}$ matrix, has an energy cost of mnE_{MAC} for a real valued vector, and of $mnpE_{AC}$ for a binary vector, where p is the firing rate, or the firing probability, and E_{AC}, E_{MAC} are the energies of an accumulate and a multiply-accumulate operation (Yin et al., 2021; Hunger, 2005). Since MAC are more costly than AC, 31-times on a 45nm complementary metal–oxide–semiconductor (Yin et al., 2021; Horowitz, 2014), we have energy savings with any p. In particular, it holds when all neurons fire ($p = 1$) and when they fire half of the time ($p = 1/2$). This gain does not depend on the simulation speed, since it is about comparing a spiking and an analogue computation, at the same computation speed.

We quantify the sparsity as $1-p$. Notice that we do not implement forced refractory periods that would prevent the neuron from firing too fast, as sometimes done in the neuromorphic literature. We do it to reduce the non differentiable steps in the system. Thus, $p = 1$ is possible if the inputs are strong and frequent enough. We measure the Pearson correlation of initial and final firing rate p_i, p_f with test loss after training. We do it in two settings, with and without a SELT. The SELT is a mean squared error between a target firing rate $p_t = 0.01$ and the actual firing rate of each layer. To achieve different p_i, we pre-train $\boldsymbol{b}_l$ on the dataset of interest, holding the other parameters untrained, using only the SELT without the classification loss. The coefficient to multiply the loss term is chosen to make all losses comparable only when the task is learned, to let the network focus first on the task and then on the sparsity. We therefore chose as the multiplicative factor the minimal

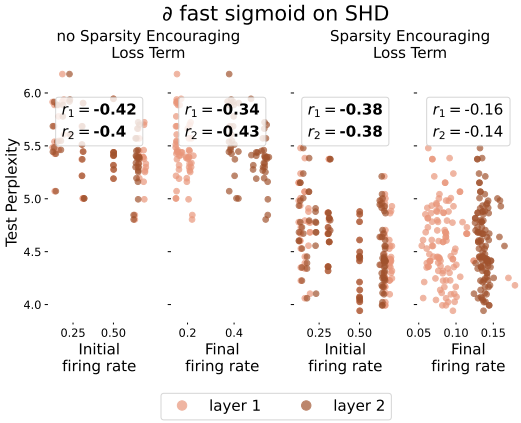

Figure 4: **High initial firing rate can promote generalization with low final firing rate.** We use SHD task and the ∂ fast-sigmoid SG. Bold correlation means p-value ≤ 0.05. On the two left panels, learning starts from different p_i without a SELT, while on the two right panels a target sparsity is encouraged. In both cases, the initial firing rate correlates with final performance, and a low p_f is achieved successfully using a SELT. Notice as well that the combination of high initial firing rate and sparsity encouragement resulted in better test accuracy than on the two panels on the left, suggesting that both factors acted synergistically as a regularization mechanism.

training crossentropy loss achieved without SELT, since the SELT takes values between zero and one. We switch on the SELT gradually during training. The switch starts as zero, and moves linearly to one between 1/5 and 3/5 of training. We focus on the ∂ fast-sigmoid and the SHD task in the main text, but we show different SG and tasks in App. H.

3.4.2 Experiments

The lower layer always reaches higher sparsity, across seeds (Fig. 4), SG shapes (Fig. 7) and tasks (Fig. 8), for reasons to elucidate. We show correlations in bold when the p-value ≤ 0.05. We can see that with and without a SELT, higher p_i correlates with performance. Notice that SELT achieved worse final train loss (not shown). However, the high p_i combined with SELT resulted in better test loss, thus, better generalization. However, this observation is not consistent across SG shapes, Fig. 7, but is consistent across tasks, Fig. 8 App. H. In fact, the triangular SG prefers low p_i and the exponential SG does not show a clear trend.

3.5 Our theoretical method for SG choice improves experimental performance

We propose a method for SG design in spiking recurrent networks inspired by FFN initializations (Glorot and Bengio, 2010; He et al., 2015). We use their variance stability method to set a SG that keeps stable gradients with time. We propose four conditions, as four hypothesis to test, that give a SG that depends on the network and the task. We present what we use as the mathematical equivalent in their subsection:

- I Each neuron has to fire half of the timesteps.
- II Recurrent and input variances should match.
- III Gradients must have equal maxima across time.
- IV Gradients must have equal variance across time.

3.5.1 Recurrent matrix mean sets the firing rate (I)

SG curves have non zero gradient even when the neuron has not fired, but their strongest value is when the neuron fires, Fig. 1. Thus, if the voltage stays close to firing, the gradient is stronger, which is always so if $Median[v] = 0$ and $Var[v] = 0$. However, $Var[v] = 0$ turns off all higher moments. So, we only assume $Median[v] = 0$ as the mathematical equivalent of our desiderata. We show in App. E.1 that, when (I) is applied to a LIF network, the mean of the recurrent weight matrix fixes the firing rate, further assuming $\overline{w}_{in} = 0$, $\boldsymbol{b} = 0$, the approximation $Mean[v] \approx Median[v]$, and constant $\bar{i}_t$ over time, we find

$$\overline{w}_{rec} = \frac{1}{n_l - 1}\Big(2 - \alpha_{decay}\Big)\vartheta \tag{I}$$

To justify $Mean[v] \approx Median[v]$, if v is sampled from a unimodal distribution with the first two moments defined, then $|Mean[v] - Median[v]| \leq \sqrt{0.6Var[v]}$ is true (Basu and

DasGupta, 1997). Experimentally, we observe always unimodal distributions, that verify $|Mean[v] - Median[v]| \leq \sqrt{c\,\mathrm{Var}[v]}$, with $c = 10^{-4}$ for the SHD task, $c = 3 \times 10^{-2}$ for sl-MNIST, and $c = 10^{-3}$ for PTB, with and without (I).

3.5.2 Recurrent matrix variance can make recurrent and input contribution to voltage comparable (II)

We want the neuron to be as sensitive to the network history as it is to new input. We describe it mathematically as $Var[W_{rec}x_{t-1}] = Var[W_{in}z_t]$. We show in App. E.2 that, in a LIF network, it sets the variance of the recurrent matrix that makes both contributions equal, further assuming $\mathbb{E}\,x = 1/2$, and $\overline{w}_{in} = 0$, we obtain

$$Var[w_{rec}] = 2(Var[z_t] + \overline{z}_t^2)\frac{n_{l-1}}{n_l - 1}Var[w_{in}] - \frac{1}{2}\overline{w}_{rec}^2 \tag{II}$$

3.5.3 Dampening and sharpness set gradient maximum and variance (III, IV)

We want the gradient to be stable with time. We describe mathematically (III) as $Max[\frac{\partial}{\partial\theta}y_t] = Max[\frac{\partial}{\partial\theta}y_{t-1}]$ and (IV) as $Var[\frac{\partial}{\partial\theta}y_t] = Var[\frac{\partial}{\partial\theta}y_{t-1}]$. We show in App. E.3, E.4 that, on a LIF network, they set the dampening and the non centered second moment of the SG that keep the maximum and variance of the gradient stable with time. Assuming σ' and $\frac{\partial}{\partial\theta}y_{t-1}$ as independent, and zero mean gradients at initialization, we find:

$$\gamma = \frac{1}{(n_l - 1)\hat{w}_{rec}}\Big(1 - \alpha_{decay}\Big) \tag{III}$$

$$\overline{\sigma'^2} = \frac{1 - \frac{1}{2}\alpha_{decay}^2}{(n_l - 1)\overline{w^2}_{rec}} \tag{IV}$$

We show in App. E.5 how sharpness and the non centered second moment of the SG relate.

3.5.4 Summary

(II), (III) and (IV) are not restricted to spiking neurons. (I) was introduced to find the SG maxima, but all deep learning activations, have their regime of interest around zero. So, (I) is also universal. Even if not stated, (Glorot and Bengio, 2010; He et al., 2015) use (I) in the form $Mean[v_l] = 0$, to determine that $Mean[W_l] = Mean[b_l] = 0$ at initialization. Our theory applies to convolutional W_{in} and W_{rec}. Take $Var[W_{in}] = n_k Var[w_{in}]$ instead of $Var[W_{in}] = n_{l-1}Var[w_{in}]$, where n_k is the number of presynaptic neurons for each postsynaptic neuron. For 2D convolutions, $n_k = k^2 n_f$, where k is the squared kernel size, and n_f the number of filters, and for 1D convolutions, $n_k = kn_f$. Our conditions also apply to any neuron definition e.g. to different reset methods, App. F. $Var[z_t]$ and $\overline{z}_t$ are computed on the train set before training. Notice that the conditions fix γ and β, which turns them into tools for SG design. We provide a theoretical understanding of the different roles played by dampening and sharpness. The dampening γ keeps the maximal gradients stable through time, while the sharpness β keeps the gradient variances stable through time.

3.5.5 Experiments

Fig. 5 shows training results with our conditions for the LIF network on the SHD task, with exponential SG, against the unconditioned baseline. (II) improves accuracy the most when applied on its own, but the best performance is achieved with all conditions. When all conditions are applied, a LIF network achieves a 92.7 ± 1.5 validation and 75.8 ± 3.1 test accuracy, compared to 87.3 ± 1.4 validation and 69.0 ± 5.8 test accuracy without conditions.

4 Discussion and Conclusions

Surrogate Gradients have reduced the gap between Spiking Neuromorphic Computing and Deep Learning, with the consequent energy efficiency gains. Different SG can achieve similar performance, at the expense of potentially costly hyper-parameter search. Our goal was to reduce the need of such search in the future, with experimental and theoretical insights.

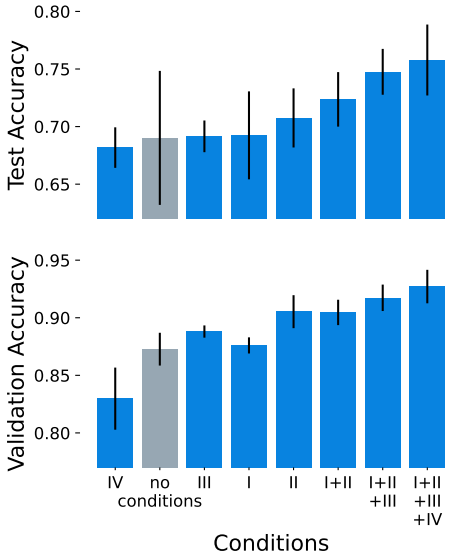

Figure 5: **Our theoretical method for SG choice improves experimental performance.** Simulations are run on the SHD task for a LIF network. We propose 4 conditions to select a SG for a LIF networks. (I) requires spiking half the timesteps, (II) encourages comparable presynaptic and input contribution to voltage, while (III) and (IV) encourage constant gradients maxima and variance with time. The baseline has *no conditions* applied, in gray, with an exponential SG and Glorot Uniform initialization. Lower and upper panels show validation and test accuracies. (II) has the highest impact on its own, but all conditions combined achieve the best result, showing that a theory of SG design can reduce the need of extensive hyper-parameter search.

We saw that best SG across networks and tasks was the derivative of the fast-sigmoid, or SuperSpike, and the SG sensitivity increase with task and network complexity. Incidentally, we reached spiking state-of-the-art on the PTB task with the triangular SG. Best average over 12 seeds had 122.8 ± 10.7 validation and 114.2 ± 9.2 test perplexity, and best seed had 117.2 validation and 109.5 test perplexity. Previous spiking SOTA on PTB was 137.7 test perplexity (Woźniak et al., 2020). We saw that the LIF network on the sl-MNIST task preferred low dampening and high sharpness. We saw that our theoretical dampening and sharpness were among the best empirically, Fig. 3. Thus, it is possible to reduce hyper-parameter search. Our q-PseudoSpike showed that a low tail decay of $q = 1.56$ improves accuracy. Passing gradients for voltages far from zero, could allow the network learn from outliers. Then, we saw the *Orthogonal* Normal as the best initialization across SG, on the LIF network and the SHD task, and Our BiGamma weight distribution reduced final variance. We saw that for some SG, a high initial firing rate, combined with a sparsity encouraging loss term, can improve generalization.

The literature on optimal SG is growing in activity (Neftci et al., 2019; Zenke and Vogels, 2021; Yin et al., 2021). However, a theoretically framework is still needed. We provide a principled method for dampening and sharpness choice, that depends on other hyper-parameters, in line with what we found numerically. All conditions apply to any architecture, but we derived the implications on the LIF. We showed how the conditions improved training. This method can thus help reduce the costly hyper-parameter grid search. In summary, this work is in response to the call made by (Zenke and Vogels, 2021) for a theory of SG choice, and it is a first step in that direction.

References

S. Basu and A. DasGupta. The mean, median, and mode of unimodal distributions: a characterization. *Theory of Probability & Its Applications*, 41(2):210–223, 1997.

G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *NeurIPS*, 2018.

P. Blouw, X. Choo, E. Hunsberger, and C. Eliasmith. Benchmarking keyword spotting efficiency on neuromorphic hardware. In *Proceedings of the 7th Annual Neuro-inspired Computational Elements Workshop*, pages 1–8, 2019.

S. M. Bohte. Error-backpropagation in networks of fractionally predictive spiking neurons. In *ICANN*, pages 60–68. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-21735-7.

B. Cramer, Y. Stradmann, J. Schemmel, and F. Zenke. The heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

M. Davies, A. Wild, G. Orchard, Y. Sandamirskaya, G. A. F. Guerra, P. Joshi, P. Plank, and S. R. Risbud. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5):911–934, 2021. doi: 10.1109/JPROC.2021.3067593.

S. K. Esser, P. A. Merolla, J. V. Arthur, A. S. Cassidy, R. Appuswamy, A. Andreopoulos, D. J. Berg, J. L. McKinstry, T. Melano, D. R. Barch, et al. Convolutional networks for fast, energy-efficient neuromorphic computing. *Proceedings of the national academy of sciences*, 113(41):11441–11446, 2016.

W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

B. Hanin and D. Rolnick. How to start training: The effect of initialization and architecture. In *NeurIPS*, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, pages 1026–1034, 2015.

P. Henderson, J. Hu, J. Romoff, E. Brunskill, D. Jurafsky, and J. Pineau. Towards the systematic reporting of the energy and carbon footprints of machine learning. *JMLR*, 21 (248):1–43, 2020.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

M. Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pages 10–14. IEEE, 2014.

I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *NeurIPS*, volume 29. Curran Associates, Inc., 2016a. URL https://proceedings.neurips.cc/paper/2016/file/d8330f857a17c53d217014ee776bfd50-Paper.pdf.

I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. *NeurIPS*, 29, 2016b.

R. Hunger. *Floating point operations in matrix-vector calculus*, volume 2019. Munich University of Technology, Inst. for Circuit Theory and Signal . . . , 2005.

E. M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003.

P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. Wilson. Averaging weights leads to wider optima and better generalization. 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018, 2018.

H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural networks*, 20(3):335–352, 2007.

J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, pages 462–466, 1952.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

L. Lapique. Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization. *Journal of Physiology and Pathololgy*, 9:620–635, 1907.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

M. Mehdipour Ghazi, M. Nielsen, A. Pai, M. Modat, M. J. Cardoso, S. Ourselin, and L. Sørensen. On the initialization of long short-term memory networks. In *ICONIP*, pages 275–286. Springer, 2019.

E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019. doi: 10.1109/MSP.2019.2931595.

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018.

H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR*, 2014.

S. Shrestha and G. Orchard. Slayer: Spike layer error reassignment in time. In *NeurIPS*, 2018.

S. Woźniak, A. Pantazi, T. Bohnstingl, and E. Eleftheriou. Deep learning incorporating biologically inspired neural dynamics and in-memory computing. *Nature Machine Intelligence*, 2(6):325–336, 2020.

B. Yin, F. Corradi, and S. M. Bohté. Effective and efficient computation with multiple-timescale spiking recurrent neural networks. In *ICONS*, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450388511. doi: 10.1145/3407197.3407225. URL https://doi.org/10.1145/3407197.3407225.

B. Yin, F. Corradi, and S. M. Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10):905–913, 2021.

F. Zenke and S. Ganguli. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation*, 30(6):1514–1541, 2018.

F. Zenke and T. P. Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural Computation*, 33(4):899–925, 2021.

J. Zhuang, T. Tang, Y. Ding, S. C. Tatikonda, N. Dvornek, X. Papademetris, and J. Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *NeurIPS*, 33, 2020.