frontiers in Neuroscience

ORIGINAL RESEARCH
published: 29 October 2021
doi: 10.3389/fnins.2021.695357

Characterization of Generalizability of Spike Timing Dependent Plasticity Trained Spiking Neural Networks

Biswadeep Chakraborty and Saibal Mukhopadhyay*

Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, United States

A Spiking Neural Network (SNN) is trained with Spike Timing Dependent Plasticity (STDP), which is a neuro-inspired unsupervised learning method for various machine learning applications. This paper studies the generalizability properties of the STDP learning processes using the Hausdorff dimension of the trajectories of the learning algorithm. The paper analyzes the effects of STDP learning models and associated hyper-parameters on the generalizability properties of an SNN. The analysis is used to develop a Bayesian optimization approach to optimize the hyper-parameters for an STDP model for improving the generalizability properties of an SNN.

Keywords: spiking neural networks, leaky integrate and fire, generalization, Hausdorff dimension, logSTDP, addSTDP, multSTDP, Bayesian optimization

OPEN ACCESS

Edited by:
Emre O. Neftci,
University of California, Irvine, United States

Reviewed by:
David Kappel,
Technische Universität Dresden, Germany
Gopalakrishnan Srinivasan,
Apple, United States

****Correspondence:***
Biswadeep Chakraborty
biswadeep@gatech.edu

Specialty section:
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

Received: *14 April 2021*
Accepted: *29 September 2021*
Published: *29 October 2021*

Citation:
Chakraborty B and Mukhopadhyay S (2021) Characterization of Generalizability of Spike Timing Dependent Plasticity Trained Spiking Neural Networks. Front. Neurosci. 15:695357. doi: 10.3389/fnins.2021.695357

1. INTRODUCTION

A Spiking Neural Network (SNN) (Maass, 1997; Gerstner and Kistler, 2002b; Pfeiffer and Pfeil, 2018) is a neuro-inspired machine learning (ML) model that mimics the spike-based operation of the human brain (Bi and Poo, 1998). The Spike Timing Dependent Plasticity (STDP) is a policy for unsupervised learning in SNNs (Bell et al., 1997; Magee and Johnston, 1997; Gerstner and Kistler, 2002a). The STDP relates the expected change in synaptic weights to the timing difference between post-synaptic spikes and pre-synaptic spikes (Feldman, 2012). Recent works using STDP trained SNNs have demonstrated promising results as an unsupervised learning paradigm for various tasks such as object classification and recognition (Masquelier et al., 2009; Diehl and Cook, 2015; Kheradpisheh et al., 2018; Lee et al., 2018; Mozafari et al., 2019; She et al., 2021).

Generalizability is a measure of how well an ML model performs on test data that lies outside of the distribution of the training samples (Kawaguchi et al., 2017; Neyshabur et al., 2017). The generalization properties of Stochastic Gradient Descent (SGD) based training for deep neural network (DNN) have received significant attention in recent years (Allen-Zhu et al., 2018; Allen-Zhu and Li, 2019; Poggio et al., 2019). The dynamics of SGD have been studied via models of stochastic gradient Langevin dynamics with an assumption that gradient noise is Gaussian (Simsekli et al., 2020b). Here SGD is considered to be driven by a Brownian motion. Chen et al. showed that SGD dynamics commonly exhibits rich, complex dynamics when navigating through the loss landscape (Chen et al., 2020). Recently Gurbuzbalaban et al. (2020) and Hodgkinson and Mahoney (2021) have simultaneously shown that the law of the SGD iterates can converge to a heavy-tailed stationary distribution with infinite variance when the step-size η is large and/or the batch-size B is small. These results form a theoretical basis for the origins of the observed heavy-tailed behavior of SGD in practice. The authors proved generalization bounds for SGD under the assumption that its trajectories can be well-approximated by the Feller Process (Capocelli and Ricciardi, 1976), a Markov-based stochastic process. Modeling the trajectories of

SGD using a stochastic differential equation (SDE) under heavy-tailed gradient noise has shed light on several interesting characteristics of SGD.

In contrast, the generalizability analysis of STDP trained SNNs, although important, has received much less attention. Few studies have shown that, in general, the biological learning process in the human brain has significantly good generalization properties (Sinz et al., 2019; Zador, 2019). However, none of them have characterized the generalization of an STDP-trained SNN using a mathematical model. There is little understanding of how hyperparameters of the STDP process impact the generalizability of the trained SNN model. Moreover, the generalization of STDP cannot be characterized by directly adopting similar studies for SGD. For example, SGD has been modeled as a Feller process for studying generalizability.

Rossum et al. showed that random variations arise due to the variability in the amount of potentiation (depression) between the pre-and post-synaptic events at fixed relative timing (Van Rossum et al., 2000). At the neuron level, fluctuations in relative timing between pre-and post-synaptic events also contribute to random variations (Roberts and Bell, 2000). For many STDP learning rules reported in the literature, the dynamics instantiate a Markov process (Bell et al., 1997; Markram et al., 1997; Bi and Poo, 1998; Han et al., 2000; Van Rossum et al., 2000); changes in the synaptic weight depend on the learning rule only on the current weight and a set of random variables that determine the transition probabilities. However, recent literature has shown that weight update using STDP is better modeled as an Ornstein-Uhlenbeck process (Câteau and Fukai, 2003; Legenstein et al., 2008; Aceituno et al., 2020).

As described by Camuto et al. (2021), fractals are complex patterns, and the level of this complexity is typically measured by the Hausdorff dimension (HD) of the fractal, which is a notion of dimension. Recently, assuming that SGD trajectories can be well-approximated by a Feller Process, it is shown that the generalization error,which is the difference between the training and testing accuracy, can be controlled by the Hausdorff dimension of the trajectories of the SDE Simsekli et al. (2020a). That is, the ambient dimension that appears in classical learning theory bounds is replaced with the Hausdorff dimension. The fractal geometric approach presented by Simsekli et al. can capture the low dimensional structure of fractal sets and provides an alternative perspective to the compression-based approaches that aim to understand why over parametrized networks do not overfit.

This paper presents a model to characterize the generalizability of the STDP process and develops a methodology to optimize hyperparameters to improve the generalizability of an STDP-trained SNN. We use the fact that the sample paths of a Markov process exhibit a fractal-like structure (Xiao, 2003). The generalization error over the sample paths is related to the roughness of the random fractal generated by the driving Markov process which is measured by the Hausdorff dimension (Simsekli et al., 2020a) which is in turn dependent on the tail behavior of the driving process. The objective of the paper is to get a model which is more generalizable in the sense that the performance of the network on unknown datasets should not differ much from its performance in the training dataset. It is to be noted that in this paper we are using the generalization error as the metric of generalizability of the network. Generalization error is not a measure of absolute accuracy, but rather the difference between training and test accuracy.

Normally, the validation loss of a model on a testing set is used to characterize the accuracy of that model. However, the validation loss is dependent on the choice of the test set, and does not necessarily give a good measure of the generalization of the learning process. Therefore, generalization error in a model is generally measured normally measured by comparing the difference between training and testing accuracy - a more generalizable model has less difference between training and testing accuracy (Goodfellow et al., 2016). However, such a measure of generalization error requires computing the validation loss (i.e., testing accuracy) for a given test dataset. To optimize the generalizability of the model, we need an objective function that measures the generalizability of the learning process. If the validation loss is used as a measure, then for each iteration of the optimization, we need to compute this loss by running the model over the entire dataset, which will significantly increase the computation time. We use Hausdorff Dimension as a measure of generalizability of the STDP process to address the preceding challenges. First, the Hausdorff measure characterizes the fractal nature of the sample paths of the learning process itself and does not depend on the testing dataset. Hence, HD provides a better (i.e., test set independent) measure of the generalization of the learning process. Second, HD can be computed during the training process itself and does not require running inference on the test data set. This reduces computation time per iteration of the optimization process.

We model the STDP learning as an Ornstein-Uhlenbeck process which is a Markov process and show that the generalization error is dependent on the Hausdorff dimension of the trajectories of the STDP process. We use the SDE representation of synaptic plasticity and model STDP learning as a stochastic process that solves the SDE.

Using the Hausdorff dimension we study the generalization properties of an STDP trained SNN for image classification. We compare three different STDP processes, namely, log-STDP, add-STDP, and mult-STDP, and show that the log-STDP improves generalizability. We show that modulating hyperparameters of the STDP learning rule and learning rate changes the generalizability of the trained model. Moreover, using log-STDP as an example, we show the hyperparameter choices that reduce generalization error increases the convergence time, and training loss, showing a trade-off between generalizability and the learning ability of a model. Motivated by the preceding observations, we develop a Bayesian optimization technique for determining the optimal set of hyperparameters which gives an STDP model with the least generalization error. We consider an SNN model with 6,400 learning neurons trained using the log-STDP process. Optimizing the hyperparameters of the learning process using Bayesian Optimization shows a testing accuracy of 90.65% and a generalization error of 3.17 on the MNIST dataset. This shows a mean increase of almost 40% in generalization performance for a mean drop of about 1% in testing accuracy

in comparison to randomly initialized training hyperparameters. In order to further evaluate the learning methodologies, we also evaluated them on the more complex Fashion MNIST dataset and observed a similar trend.

2. MATERIALS AND METHODS

2.1. Background

2.1.1. Spiking Neural Networks

We chose the leaky integrate-and-fire model of a neuron where the membrane voltage X is described by

$$\tau \frac{dX}{dt} = (E_{\text{rest}} - X) + g_e (E_{\text{exc}} - X) + g_i (E_{\text{inh}} - X)$$

where E_{rest} is the resting membrane potential; E_{exc} and E_{inh} are the equilibrium potentials of excitatory and inhibitory synapses, respectively; and g_e and g_i are the conductances of excitatory and inhibitory synapses, respectively. The time constant τ, is longer for excitatory neurons than for inhibitory neurons. When the neuron's membrane potential crosses its membrane threshold, the neuron fires, and its membrane potential is reset. Hence, the neuron enters its refractory period and cannot spike again for the duration of the refractory period.

Synapses are modeled by conductance changes, i.e., synapses increase their conductance instantaneously by the synaptic weight w when a pre-synaptic spike arrives at the synapse, otherwise, the conductance decays exponentially. Thus, the dynamics of the conductance g can be written as:

$$\tau_g \frac{dg}{dt} = -g \tag{1}$$

If the pre-synaptic neuron is excitatory, the dynamics of the conductance is $g = g_e$ with the time constant of the excitatory post-synaptic potential being $\tau_g = \tau_{g_e}$. On the other hand, if the pre-synaptic neuron is inhibitory, it's synaptic conductance is given as $g = g_i$ and the time constant of the inhibitory post-synaptic potential as $\tau_g = \tau_{g_i}$.

2.1.2. STDP Based Learning Methods

Spike-timing-dependent plasticity is a biologically plausible learning model representing the time evolution of the synaptic weights as a function of the past spiking activity of adjacent neurons.

In a STDP model, the change in synaptic weight induced by the pre-and post-synaptic spikes at times t_{pre}, t_{post} are defined by:

$$\Delta W = \eta(1 + \zeta)H\left(W; t_{\text{pre}} - t_{\text{post}}\right) \tag{2}$$

where the learning rate η determines the speed of learning. The Gaussian white noise ζ with zero mean and variance σ^2 describes the variability observed in physiology. The function $H(W; u)$ describes the long term potentiation (LTP) and depression (LTD) based on the relative timing of the spike pair within a learning window $u = t_{\text{pre}} - t_{\text{post}}$, and is defined by:

$$H(W; u) = \begin{cases} a_+(W) \exp\left(-\frac{|u|}{\tau_+}\right) & \text{for u} < 0 \\ -a_-(W) \exp\left(-\frac{|u|}{\tau_-}\right) & \text{for u} > 0 \end{cases} \tag{3}$$

The shape of the weight distribution produced by STDP can be adjusted via the scaling functions $a_\pm(W)$ in (3) that determine the weight dependence. We study three different types of STDP processes, namely, add-STDP, mult-STDP, and log-STDP. All STDP models follow the Equations (2) and (3), however, they have different scaling functions ($a_\pm$) in (3) as discussed below. The weight distributions of these three different STDP processes at the end of the last training iteration are shown in **Figure 1**. At the beginning of the training iterations, the distribution is uniform for all three reflecting on the weight initialization conditions. Additive STDP gives rise to strong competition among synapses, but due to the absence of weight dependence, it requires hard boundaries to secure the stability of weight dynamics. To reach a stability point for the add-STDP, we followed the analysis done by Gilson and Fukai (2011) and Burkitt et al. (2004) and chose the fixed point $W_0 = 0.006$. **Figure 1** approximates the probability density function based on the weight distributions of the different STDP models. We are using a Gaussian KDE to get this pdf from the empirical weight distribution obtained. Given N independent realizations $\mathcal{X}_N \equiv \{X_1, \ldots, X_N\}$ from an unknown continuous probability density function (p.d.f.) f on $\mathcal{X}$, the Gaussian kernel density estimator is defined as

$$\hat{f}(x; t) = \frac{1}{N} \sum_{i=1}^{N} \phi\,(x, X_i; t)\,, \quad x \in \mathbb{R} \tag{4}$$

where

$$\phi\,(x, X_i; t) = \frac{1}{\sqrt{2\pi t}} e^{-(x-X_i)^2/(2t)} \tag{5}$$

is a Gaussian p.d.f. (kernel) with location X_i and scale $\sqrt{t}$. The scale is usually referred to as the bandwidth. Much research has been focused on the optimal choice of t, because the performance of $\hat{f}$ as an estimator of f depends crucially on its value (Sheather and Jones, 1991; Jones et al., 1992).

Logarithmic STDP (log-STDP) (Gilson and Fukai, 2011). The scaling functions of log-STDP is defined by:

$$a_+(W) = c_+ \exp\left(-W/W_0\gamma\right) \tag{6}$$

$$a_-(W) = \begin{cases} c_- W/W_0 & \text{for W} \leq W_0 \\ c_- \left[1 + \frac{\ln\left[1+\mathcal{S}\left(\frac{W}{W_0}-1\right)\right]}{\mathcal{S}}\right] & \text{for W} > W_0 \end{cases} \tag{7}$$

In this study, W_0 is chosen such that LTP and LTD in log-STDP balance each other for uncorrelated inputs, namely $A\,(W_0) = \tau_+ a_+\,(W_0) - \tau_- a_-\,(W_0) \simeq 0$. Therefore, W_0 will also be referred to as the "fixed point" of the dynamics. Since depression increases sublinearly (blue solid curve for a_- in **Figure 1**), noise in log-STDP is weaker than that for mult-STDP for which depression increases linearly (orange dashed curve for a_- in **Figure 1**).

The weight dependence for LTD in logSTDP is similar to mult-STDP for $W \leq W_0$, i.e., it increases linearly with W. However, the LTD curve a_- becomes sublinear for $W \geq W_0$, and $\mathcal{S}$ determines the degree of the log-like saturation. For larger

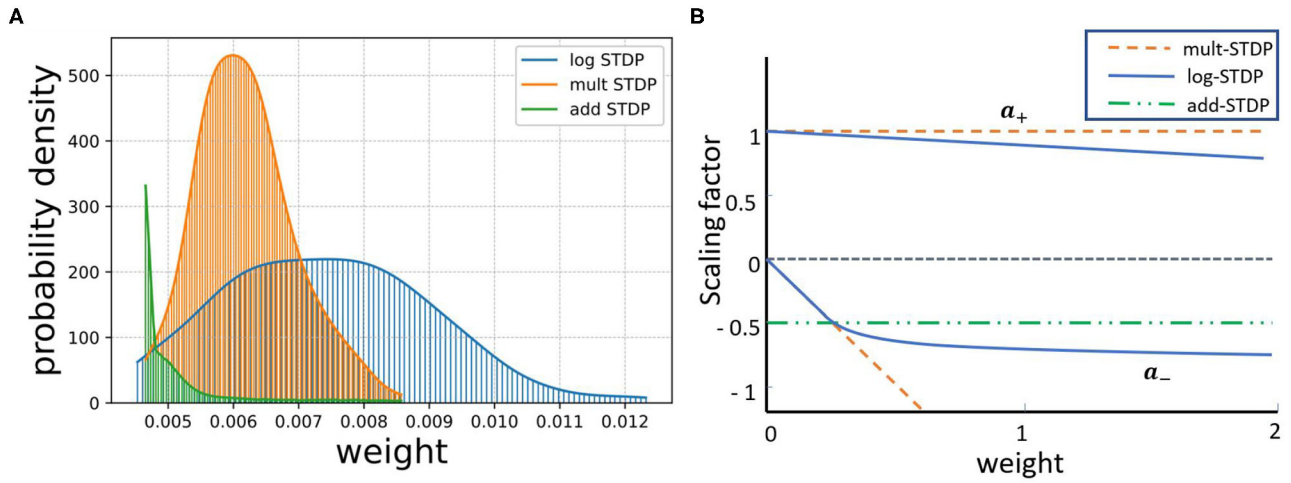

FIGURE 1 | (A) Resulting weight distribution for log-STDP (Gilson and Fukai, 2011);multSTDP (Van Rossum et al., 2000) and add-STDP (Song et al., 2000). **(B)** Plot for Functions a_+ for LTP and $-f_-$ for LTD in log-STDP (blue solid curve); mult-STDP (orange dashed line); and add-STDP model green dashed-dotted curve for depression and orange dashed curve for potentiation.

$\mathcal{S}$, LTD has a more pronounced saturating log-like profile and the tail of the synaptic weight distribution extends further.

We choose the function a_+ for LTP to be roughly constant around W_0, such that the exponential decay controlled by γ only shows for $W >> W_0$. Thus, the scaling functions a_+, a_-, $\mathcal{S}$, and γ are the hyperparameters that can be tuned to model an efficient log-STDP learning model. tAs discussed by Gilson and Fukai (2011), we choose the activation function a_+ to be roughly constant around the threshold fixed-point weight W_0. For $W \geq W_0$, the exponential decay factor γ of log-STDP coincides with mlt-STDP when $\mathcal{S} \to 0$ and $\gamma \to \infty$. Log-STDP tends toward add-STDP when $\mathcal{S} \to \infty$ and $\gamma \to \infty$. High values of $\mathcal{S}$ leads to stronger saturation of LTD and large values of γ leads to a stronger potentiation and keeps LTP almost constant which leads to favoring a winner-takes-all behavior.

Additive STDP (add-STDP) (Song et al., 2000). It is weight independent and is defined by the following scaling functions:

$$\begin{aligned} a_+(W) &= c_+ \\ a_-(W) &= c_- \end{aligned} \tag{8}$$

with $c_+\tau_+ < c_-\tau_-$ such that LTD overpowers LTP. The drift due to random spiking activity thus causes the weights to be depressed toward zero, which provides some stability for the output firing rate (Gilson and Fukai, 2011). For the simulations, we are using a fast learning rate that is synonymous to a high level of noise, and more stability. This requires a stronger depression. Thus, we use $c_+ = 1$ and $c_- = 0.6$.

Multiplicative STDP (mult-STDP) (Van Rossum et al., 2000). The multiplicative STDP has a linear weight dependence for LTD and constant LTP:

$$a_+(W) = c_+ \tag{9}$$

$$a_-(W) = c_- W \tag{10}$$

The equilibrium mean weight is then given by $W^*_{\text{av}} = c_+\tau_+ / (c_-\tau_-)$. For the simulations we use $c_+ = 1$ and $c_- = 0.5/W_0 = 2$. This calibration corresponds to a similar neuronal output firing rate to that for log-STDP in the case of uncorrelated inputs.

2.1.3. Generalization - Hausdorff Dimension and Tail Index Analysis

Recent works have discussed the generalizability of SGD based on the trajectories of the learning algorithm. Simsekli et al. (2020a) identified the complexity of the fractals generated by a Feller process that approximates SGD. The intrinsic complexity of a fractal is typically characterized by a notion called the Hausdorff dimension (Le Guével, 2019; Lőrinczi and Yang, 2019), which extends the usual notion of dimension to fractional orders. Due to their recursive nature, Markov processes often generate random fractals (Xiao, 2003). Significant research has been performed in modern probability theory to study the structure of such fractals (Khoshnevisan, 2009; Bishop and Peres, 2017; Khoshnevisan and Xiao, 2017; Yang, 2018). Thus, the STDP learning method follows an Ornstein-Uhlenbeck (O-U) process which is a special type of Lévy process. Again, the Hausdorff Dimension measures the roughness of the fractal patterns of the sample paths generated by the stochastic process which is measured using the tail properties of the Lévy measure of the O-U process. Lévy measure is a Borel measure on $\mathbb{R}^d \backslash \{0\}$ satisfying $\int_{\mathbb{R}^d} \|x\|^2 / \left(1 + \|x\|^2\right) \nu(\mathrm{d}x) < \infty$. The Ornstein-Uhlenbeck process which is a Lévy process can thus be characterized by the triplet (b, Σ, ν) where $b \in \mathbb{R}^d$ denotes a constant drift, $\Sigma \in \mathbb{R}^{d\times d}$ is a positive semi-definite matrix and ν is the Lévy measure as defined above. Thus, taking Lévy processes as stochastic objects, their sample path behavior can be characterized by the Hausdorff dimension which is in turn measured using the BG-indices. Thus, the generalization properties of the STDP process can be modeled using the Hausdorff dimension of the sample paths of the O-U process. We formally define the Hausdorff dimension for

the Ornstein-Uhlenbeck process modeling the STDP learning process in section 3.2.

2.2. STDP as a Stochastic Process

In this paper, we evaluate the generalization properties of an STDP model using the concept of the Hausdorff dimension. In this section, we discuss the learning methodology of STDP and how the plasticity change can be modeled using a stochastic differential equation. The state of a neuron is usually represented by its membrane potential X which is a key parameter to describe the cell activity. Due to external input signals, the membrane potential of a neuron may rise until it reaches some threshold after which a spike is emitted and transferred to the synapses of neighboring cells. To take into account the important fluctuations within cells, due to the spiking activity and thermal noise, in particular, a random component in the cell dynamics has to be included in mathematical models describing the membrane potential evolution of both the pre-and post-synaptic neurons similar to the analysis shown by Robert and Vignoud (2020). Several models take into account this random component using an independent additive diffusion component, like Brownian motion, of the membrane potential X. In our model of synaptic plasticity, the stochasticity arises at the level of the generation of spikes. When the value of the membrane potential of the output neuron is at $X = x$, a spike occurs at rate $\beta(x)$ where β is the activation function (Chichilnisky, 2001). In particular, we consider the instants when the output neuron spikes are represented by an inhomogeneous Poisson process as used by Robert and Vignoud (2020). Thus, in summary, (1) The pre-synaptic spikes are modeled using a Poisson process and hence, there is a random variable added to membrane potential. (2) The post-synaptic spikes are generated using a stochastic process based on the activation function. Hence, STDP, which depends on the pre-and post-synaptic spike times can be modeled using a *stochastic differential equation (SDE)*. Hence, we formulate the STDP as a SDE. The SDE of a learning algorithm shares similar convergence behavior of the algorithm and can be analyzed more easily than directly analyzing the algorithm.

2.2.1. Mathematical Setup

We consider the STDP as an iterative learning algorithm $\mathcal{A}$ which is dependent on the dataset $\mathcal{D}$ and the algorithmic stochasticity $\mathcal{U}$. The learning process $\mathcal{A}(\mathcal{D},\mathcal{U})$ returns the entire evolution of the parameters of the network in the time frame $[0, T]$ where $[\mathcal{A}(\mathcal{D},\mathcal{U})]_t = W_t$ being the parameter value returned by the STDP learning algorithm at time t. So, for a given training set $\mathcal{D}$, the learning algorithm $\mathcal{A}$ will output a random process $w_{t\in[0,T]}$ indexed by time which is a trajectory of iterates.

In the remainder of the paper, we will consider the case where the STDP learning process $\mathcal{A}$ is chosen to be the trajectories produced by the Ornstein-Uhlenbeck (O-U) process $\mathrm{W}^{(\mathcal{D})}$, whose symbol depends on the training set $\mathcal{D}$. More precisely, given $\mathcal{D} \in \mathcal{Z}^n$, the output of the training algorithm $\mathcal{A}(\mathcal{D},\cdot)$ will be the random mapping $t \mapsto \mathrm{W}_t^{(\mathcal{D})}$, where the symbol of $\mathrm{W}^{(\mathcal{D})}$ is determined by the drift $b_{\mathcal{D}}(w)$, diffusion matrix $\Sigma_{\mathcal{D}}(w)$, and the Lévy measure $\nu_{\mathcal{D}}(w,\cdot)$, which all depend on $\mathcal{U}$. In this context, the random variable $\mathcal{U}$ represents the randomness that is incurred by the O-U process. Finally, we also define the collection of the parameters given in a trajectory, as the image of $\mathcal{A}(\mathcal{D})$, i.e., $\mathcal{W}_{\mathcal{D}} := \left\{w \in \mathbb{R}^d : \exists t \in [0,1], w = [\mathcal{A}(\mathcal{D})]_t\right\}$ and the collection of all possible parameters as the union $\mathcal{W} := \bigcup_{n\geq 1}\bigcup_{\mathcal{D}\in\mathcal{Z}^n}\mathcal{W}_S$. Note that $\mathcal{W}$ is still random due to its dependence on $\mathcal{U}$.

We consider the dynamics of synaptic plasticity as a function of the membrane potential $X(t)$ and the synaptic weight $W(t)$.

2.2.2. Time Evolution of Synaptic Weights and Plasticity Kernels

As described by Robert and Vignoud (2020), the time evolution of the weight distribution $W(t)$ depends on the past activity of the input and output neurons. It may be represented using the following differential equation:

$$\frac{dW(t)}{dt} = M(\Omega_p(t), \Omega_d(t), W(t)) \tag{11}$$

where $\Omega_p(t), \Omega_d(t)$ are two non-negative processes where the first one is associated with potentiation i.e., increase in W and the latter is related to the depression i.e., decrease in W. The function M needs to be chosen such that the synaptic weight W stays at all-time in its definition interval K_W. The function M can thus be modified depending on the type of implementation of STDP that is needed. Further details regarding the choice of M for different types of STDP is discussed by Robert and Vignoud (2020).

When the synaptic weight of a connection between a pre-synaptic neuron and a post-synaptic neuron is fixed and equal to W, the time evolution of the post-synaptic membrane potential $X(t)$ is represented by the following stochastic differential equation (SDE) (Robert and Vignoud, 2020):

$$\mathrm{d}X(t) = -\frac{1}{\tau}X(t)\mathrm{d}t + W\mathcal{N}_\lambda(\mathrm{d}t) - g(X(t-))\mathcal{N}_{\beta,X}(\,\mathrm{d}t) \tag{12}$$

where $X(t-)$ is the left limit of X at $t > 0$, and τ is the exponential decay time constant of the membrane potential associated with the leaking mechanism. The sequence of firing instants of the pre-synaptic neuron is assumed to be a Poisson point process $\mathcal{N}_\lambda$ on $\mathbb{R}_+$ with the rate λ. At each pre-synaptic spike, the membrane potential X is increased by the amount W. If $W > 0$ the synapse is said to be excitatory, whereas for $W < 0$ the synapse is inhibitory. The sequence of firing instants of the post-synaptic neuron is an inhomogeneous Poisson point process $\mathcal{N}_{\beta,X}$ on $\mathbb{R}_+$ whose intensity function is $t \mapsto \beta(X(t-))$. The drop of potential due to a post-synaptic spike is represented by the function g. When the post-synaptic neuron fires in-state $X(t-) = x$, its state $X(t)$ just after the spike is $x - g(x)$.

2.2.3. Uniform Hausdorff Dimension

The Hausdorff dimension for the training algorithm $\mathcal{A}$ is a notion of complexity based on the trajectories generated by $\mathcal{A}$. Recent literature has shown that the synaptic weight update using an STDP rule can be approximated using a type of stochastic process called the Ornstein-Uhlenbeck process which is a type of Markov process (Câteau and Fukai, 2003; Legenstein et al., 2008; Aceituno et al., 2020). Hence, we can infer that the STDP process will also have a uniform Hausdorff dimension for the trajectories.

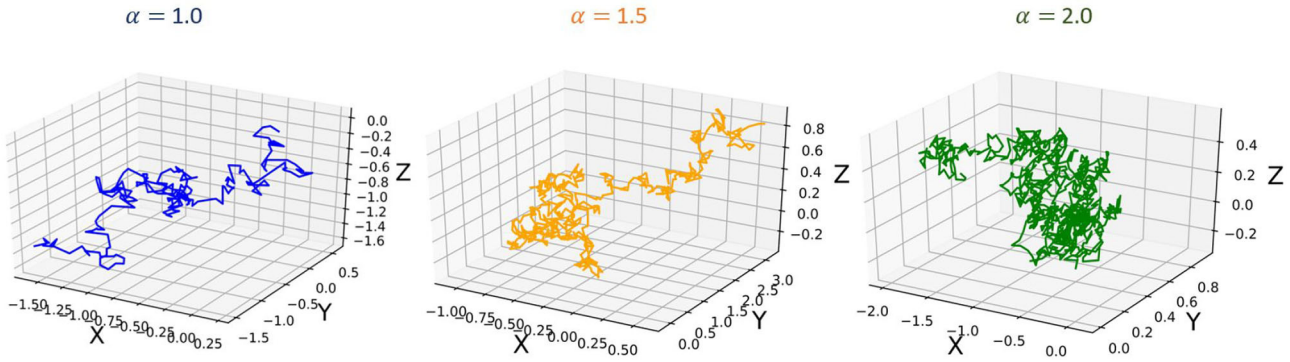

FIGURE 2 | Plot showing the trajectories of the $\alpha-$stable Lévy process L_t^α for varying values of α.

We use the Hausdorff Dimension of the sample paths of the STDP based learning algorithm which has not been investigated in the literature.

Let Φ be the class of functions $\varphi:(0,\delta) \rightarrow (0,\infty)$ which are right continuous, monotone increasing with $\varphi(0+)=0$ and such that there exists a finite constant $K>0$ such that

$$\frac{\varphi(2s)}{\varphi(s)} \leq K, \quad \text{for } 0<s<\frac{\delta}{2} \tag{13}$$

A function φ in Φ is often called a measure function. For $\varphi \in \Phi$, the φ-Hausdorff measure of $E \subseteq \mathbb{R}^d$ is defined by

$$\varphi_m(E) = \lim_{\varepsilon \to 0} \inf \left\{ \sum_i \varphi\left(2r_i\right) : E \subseteq \bigcup_{i=1}^{\infty} B\left(x_i, r_i\right), r_i < \varepsilon \right\} \tag{14}$$

where $B(x,r)$ denotes the open ball of radius r centered at x. The sequence of balls satisfying Equation (14) is called an ε -covering of E. We know that φ_m is a metric outer measure and every Borel set in $\mathbb{R}^d$ is φ_m measurable. Thus, the function $\varphi \in \Phi$ is called the Hausdorff measure function for E if $0<\varphi_m(E)<\infty$.

It is to be noted here that in Equation (14), we only use coverings of E by balls, hence φ_m is usually called a spherical Hausdorff measure in the literature. Under Equation (13), φ_m is equivalent to the Hausdorff measure defined by using coverings by arbitrary sets. The Hausdorff dimension of E is defined by

$$\dim_{\mathrm{H}} E = \inf\left\{\alpha > 0 : s^\alpha - m(E) = 0\right\}. \tag{15}$$

The STDP learning process is modeled using the SDEs for the temporal evolution of the synaptic weights and the membrane potential given in Equations (11), (12). Considering the empirical observation that STDP exhibits a diffusive behavior around a local minimum (Baity-Jesi et al., 2018), we take $w_{\mathcal{D}}$ to be the local minimum found by STDP and assume that the conditions of Proposition hold around that point. This perspective indicates that the generalization error can be controlled by the BG index $\beta_{\mathcal{D}}$ of the Lévy process defined by $\psi_S(\xi)$; the sub-symbol of the process (11) around $w_{\mathcal{D}}$. The choice of the SDE (11) imposes some structure on $\psi_{\mathcal{D}}$, which lets us express $\beta_{\mathcal{D}}$ in a simpler form. This helps us in estimating the BG index for a general Lévy process. As shown by Simsekli et al., there is a layer-wise variation of the tail-index of the gradient noise in a DNN-based multi-layer neural network (Simsekli et al., 2019). Thus, for our STDP model we assume that around the local minimum $w_{\mathcal{D}}$, the dynamics of STDP will be similar to the Lévy motion with frozen coefficients: $\Sigma_2\left(w_S\right) \mathrm{L}^{\alpha\left(w_{\mathcal{D}}\right)}$. Also assuming the coordinates corresponding to the same layer l have the same tail-index α_l around $w_{\mathcal{D}}$, the BG index can be analytically computed as $\beta_{\mathcal{D}} = \max_l \alpha_l \in (0,2]$ (Meerschaert and Xiao, 2005). We note here that $\dim_H \mathcal{W}_{\mathcal{D}}$ and thus, $\beta_{\mathcal{D}}$ determines the order of the generalization error. Using this simplification, we can easily compute $\beta_{\mathcal{D}}$, by first estimating each α_l by using the estimator proposed by Mohammadi et al. (2015), that can efficiently estimate α_l by using multiple iterations of the STDP learning process.

As explained by Simsekli et al. (2020a), due to the decomposability property of each dataset $\mathcal{D}$, the stochastic process for the synaptic weights given by $W^{(\mathcal{D})}(t)$ behaves like a Lévy motion around a local point w_0. Because of this locally regular behavior, the Hausdorff dimension can be bounded by the Blumenthal-Getoor (BG) index (Blumenthal and Getoor, 1960), which in turn depends on the tail behavior of the Lévy process. Thus, in summary, we can use the BG-index as a bound for the Hausdorff dimension of the trajectories from the STDP learning process. Now, as the Hausdorff dimension is a measure of the generalization error and is also controlled by the tail behavior of the process, heavier-tails imply less generalization error (Simsekli et al., 2020a,b).

To further demonstrate the heavy-tailed behavior of the Ornstein-Uhlenbeck process (a type of $\alpha-$stable Lévy process L_t^α) that characterizes the STDP learning mechanism, we plot its trajectories and their corresponding pdf distributions. We plot these for varying values of the stability factor of the Lévy process L_t^α, α. We hence also plotted the probability density function of the Lévy processes to show the heavy-tailed nature of the Lévy trajectories as the tail index α decreases. **Figure 2** shows a continuous-time random walk performed using the O-U random process in 3D space. In the figure, X, Y, Z are random variables for the α-stable distributions generated using the O-U process. **Figure 3**, shows the corresponding probability density function

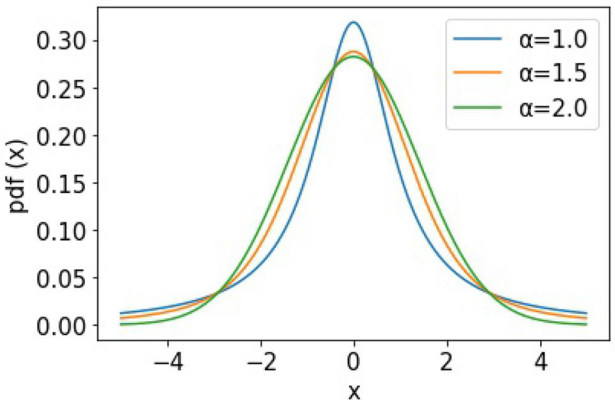

FIGURE 3 | Figure showing the probability density functions of the $\alpha-$stable Lévy process L_t^{α} for varying values of α.

of the O-U process for varying values of α corresponding to the different trajectories shown in **Figure 2**. From **Figures 2**, **3**, that as the O-U process becomes heavier-tailed (i.e., α decreases), and the Hausdorff dimension $\mathtt{dim}_{\mathrm{H}}$ gets smaller.

2.3. Optimal Hyperparameter Selection

Using the Hausdorff Dimension as a metric for the generalizability of the learning process, we formulate an optimization process that selects the hyperparameters of the STDP process to improve the generalizability of the models. The Hausdorff dimension is a function of the hyperparameters of the STDP learning process. Thus, we formulate an optimization problem to select the optimal hyperparameters of the STDP using the Hausdorff dimension of the STDP learning process as the optimization function. Now, since the BG-index is the upper bound of the Hausdorff dimension, as discussed earlier, we in turn aim to optimize the BG-index of the STDP stochastic process. The optimization problem aims to get the optimal set of hyperparameters of the STDP process that can give a more generalizable model without looking at the test set data. Now, given an STDP process, we aim to tune its hyperparameters to search for a more generalizable model. Let us define $\boldsymbol{\lambda} := \{\lambda_1, \ldots, \lambda_N\}$ where $\boldsymbol{\lambda}$ is the set of N hyperparameters of the STDP process, $\lambda_1, \ldots, \lambda_N$. Let Λ_i denote the domain of the hyperparameter λ_i. The hyperparameter space of the algorithm is thus defined as $\boldsymbol{\Lambda} = \Lambda_1 \times \ldots \times \Lambda_N$. Now, we aim to design the optimization problem to minimize the Hausdorff Dimension of the learning trajectory for the STDP process. This is calculated over the last training iteration of the model, assuming that it reaches near the local minima. When trained with $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$ on the training data $\mathcal{D}_{\mathbf{train}}$, we denote the algorithm's Hausdorff dimension as $\dim_{\mathrm{H}} G(\boldsymbol{\lambda}; \mathcal{D}_{\mathrm{train}})$. Thus, using K-fold cross validation, the hyperparameter optimization problem for a given dataset $\mathcal{D}$ is to given as follows:

$$\boldsymbol{\lambda_s} = \arg\min_{\boldsymbol{\lambda} \in \boldsymbol{\Lambda}} \frac{1}{K} \sum_{i=1}^{K} \dim_{\mathrm{H}} G(\boldsymbol{\lambda}; \mathcal{D}_{\mathrm{train}}) \tag{16}$$

We choose the Sequential Model-based Bayesian Optimization (SMBO) technique for this problem (Feurer et al., 2015). SMBO constructs a probabilistic model $\mathcal{M}$ of $f = \dim_{\mathrm{H}} G$ based on point evaluations of f and any available prior information. It then uses that model to select subsequent configurations $\boldsymbol{\lambda}$ to evaluate. Given a set of hyperparameters $\boldsymbol{\lambda}$ for an STDP learning process G, we define the point functional evaluation as the calculation of the BG index of G with the hyperparameters $\boldsymbol{\lambda}$. The BG index gives an upper bound on the Hausdorff dimension of the learning process. In order to select its next hyperparameter configuration $\boldsymbol{\lambda}$ using model $\mathcal{M}$, SMBO uses an acquisition function $a_{\mathcal{M}} : \boldsymbol{\lambda} \rightarrow \mathbb{R}$, which uses the predictive distribution of model $\mathcal{M}$ at arbitrary hyperparameter configurations $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$. This function is then maximized over $\boldsymbol{\Lambda}$ to select the most useful configuration $\boldsymbol{\lambda}$ to evaluate next. There exists a wide range of acquisition functions (Mockus et al., 1978), all of whom aim to trade-off between exploitation and exploration. The acquisition function tries to balance between locally optimizing hyperparameters in regions known to perform well and trying hyperparameters in a relatively unexplored region of the space.

In this paper, for the acquisition function, we use the expected improvement (Mockus et al., 1978) over the best previously-observed function value $f_{\min}$ attainable at a hyperparameter configuration $\boldsymbol{\lambda}$ where expectations are taken over predictions with the current model $\mathcal{M}$:

$$a(\boldsymbol{\lambda}, \mathcal{M}) = \int_{-\infty}^{f_{\min}} \max\left\{f_{\min} - f, 0\right\} \cdot p_{\mathcal{M}}(f \mid \boldsymbol{\lambda}) df \tag{17}$$

3. RESULTS

3.1. Experimental Setup

We empirically study the generalization properties of the STDP process by designing an SNN with 6,400 learning neurons for hand-written digit classification using the MNIST dataset. The MNIST dataset contains 60,000 training examples and 10,000 test examples of 28×28 pixel images of the digits $0 - 9$. It must be noted here that the images from the ten classes in the MNIST dataset are randomized so that there is a reinforcement of the features learned by the network.

3.1.1. Architecture

We use a two-layer SNN architecture as done by the implementation of Diehl and Cook (2015) as shown in **Figure 4**. The first layer is the input layer, containing 28×28 neurons with one neuron per image pixel. The second layer is the processing layer, with an equal number of excitatory and inhibitory neurons. The excitatory neurons of the second layer are connected in a one-to-one fashion to inhibitory neurons such that each spike in an excitatory neuron will trigger a spike in its corresponding inhibitory neuron. Again, each of the inhibitory neurons is connected to all excitatory ones, except for the one from which it receives a connection. This connectivity provides lateral inhibition and leads to competition among excitatory neurons. There is a balance between the ratio of inhibitory and excitatory synaptic conductance to ensure the correct strength of lateral inhibition. For a weak lateral inhibition, the conductance will not

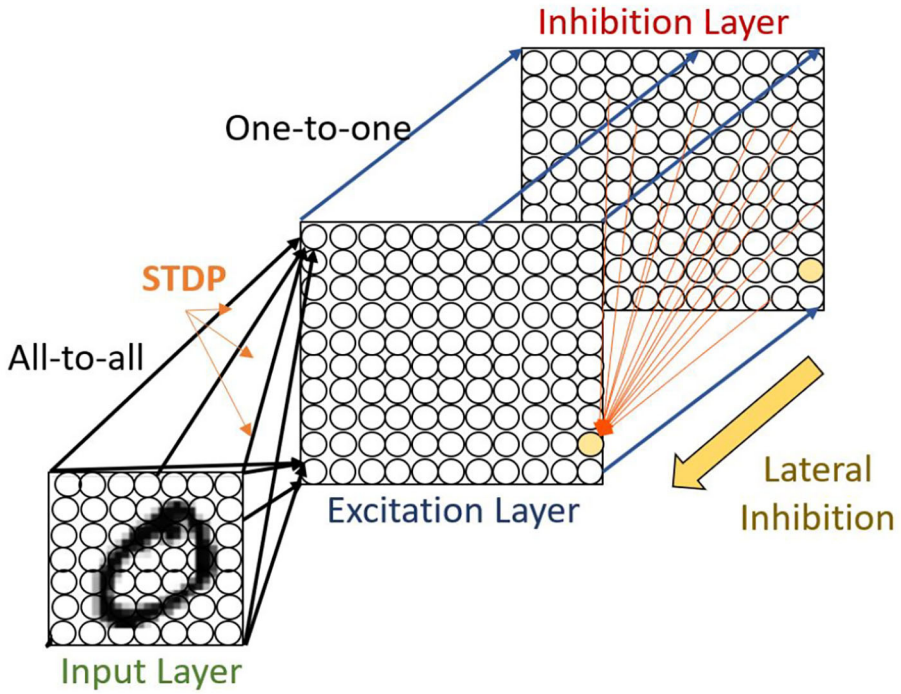

FIGURE 4 | The intensity values of the MNIST image are converted to Poisson-spike trains. The firing rates of the Poisson point process are proportional to the intensity of the corresponding pixel. These spike trains are fed as input in an all-to-all fashion to excitatory neurons. In the figure, the black shaded area from the input to the excitatory layer shows the input connections to one specific excitatory example neuron. The red shaded area denotes all connections from one inhibitory neuron to the excitatory neurons. While the excitatory neurons are connected to inhibitory neurons via one-to-one connection, each of the inhibitory neurons is connected to all excitatory ones, except for the one it receives a connection from.

have any influence while an extremely strong signal would ensue that one dominant neuron suppresses the other ones.

The task for the network is to learn a representation of the dataset on the synapses connecting the input layer neurons to the excitatory layer neurons. The excitatory-inhibitory connectivity pattern creates competition between the excitatory neurons. This allows individual neurons to learn unique filters where the single most spiked neuron in each iteration updates its synapse weights to match the current input digit using the chosen STDP rule. Increasing the number of neurons allows the network to memorize more examples from the training data and recognize similar patterns during the test phase.

3.1.2. Homeostasis

The inhomogeneity of the input leads to different firing rates of the excitatory neurons, and lateral inhibition further increases this difference. However, all neurons should have approximately equal firing rates to prevent single neurons from dominating the response pattern and to ensure that the receptive fields of the neurons differentiate. To achieve this, we employ an adaptive membrane threshold resembling intrinsic plasticity (Zhang and Linden, 2003). Specifically, each excitatory neuron's membrane threshold is not only determined by v_{thresh} but by the sum $v_{\text{thresh}} + \theta$, where θ is increased every time the neuron fires and is exponentially decaying (Querlioz et al., 2013). Therefore, the more a neuron fires, the higher will be its membrane threshold and in turn, the neuron requires more input to a spike soon. Using this mechanism, the firing rate of the neurons is limited because the conductance-based synapse model limits the maximum membrane potential to the excitatory reversal potential E_{exc}, i.e., once the neuron membrane threshold is close to E_{exc} (or higher) it will fire less often (or even stop firing completely) until θ decreases sufficiently.

3.1.3. Input Encoding

The input image is converted to a Poisson spike train with firing rates proportional to the intensity of the corresponding pixel. This spike train is then presented to the network in an all-to-all fashion for 350 ms as shown in **Figure 4**. The maximum pixel intensity of 255 is divided by 4, resulting in input firing rates between 0 and 63.75 Hz. Additionally, if the excitatory neurons in the second layer fire less than five spikes within 350 ms, the maximum input firing rate is increased by 32 Hz and the example is presented again for 350 ms. This process is repeated until at least five spikes have been fired during the entire time the particular example was presented.

3.1.4. Training and STDP Dynamics Analysis

To train the network, we present digits from the MNIST training set to the network. It is to be noted that, before presenting a new image, no input to any variable of any neuron is given

TABLE 1 | Table showing the set of hyperparameters for various STDP processes.

Hyperparameter	logSTDP	addSTDP	multSTDP
Synaptic Delay	0.75 ms	0.75 ms	0.75 ms
Synaptic epsp τ_A	1 ms	1 ms	1 ms
Synaptic epsp τ_B	5 ms	5 ms	5 ms
Number of correlated pools	4	4	4
Number of neurons per pool	50	50	50
Spiking rate of inputs	10 Hz	10 Hz	10 Hz
Learning rate (η)	0.0002	0.0002	0.0002
STDP Apre (LTP) time constant	17 ms	17 ms	17 ms
STDP Apre (LTD) time constant	34 ms	34 ms	34 ms
Increase in A_{pre} (LTP), on pre-spikes A_{pre0}	1.0	1.0	1.0
Increase in A_{post} (LTD), on post-spikes A_{post0}	0.5	0.55	100
LTD curvature factor t($\mathcal{S}$)	5	N/A	N/A
Exponential LTP decay factor t(γ)	50	N/A	N/A
Threshold fixed-point weight (W_0)	0.006	N/A	N/A

for a time interval of 150 ms. This is done to decay to their resting values. All the synaptic weights from input neurons to excitatory neurons are learned using the STDP learning process as described in section 2.1.2. To improve simulation speed, the weight dynamics are computed using synaptic traces such that every time a pre-synaptic spike x_{pre} arrives at the synapse, the trace is increased by 1 (Morrison et al., 2007). Otherwise, x_{pre} decays exponentially. When a post-synaptic spike arrives at the synapse the weight change Δw is calculated based on the pre-synaptic trace as described in section 2.1.2. To evaluate the model, we divide the training set into 100 divisions of 600 images each and check the model performance after each such batch is trained using the STDP learning model. In the remainder of the paper, we call this evaluation of the model after 600 images as one iteration.

3.1.5. Inference

After the training process is done, the learning rate is set to zero and each neuron's spiking threshold is fixed. A class is assigned to each neuron based on its highest response to the ten classes of digits over one presentation of the training set. This is the first time labels are used in the learning algorithm, which makes it an unsupervised learning method. The response of the class-assigned neurons is used to predict the digit. It is determined by taking the mean of each neuron response for every class and selecting the class with the maximum average firing rate. These predictions are then used to measure the classification accuracy of the network on the MNIST test set.

3.1.6. Computation of Generalization Error and Hausdorff Dimension

We empirically study the generalization behavior of STDP trained SNNs. We vary the hyperparameters of the STDP learning process which controls the LTP/LTD dynamics of the STDP learning algorithm. **Table 1** shows the hyperparameters for various STDP processes. We trained all the models for 100 training iterations. In this paper, we consider the synaptic weight update to follow a Lévy process, i.e., continuous with discrete jumps similar to the formulation of Stein (1965) and Richardson and Swarbrick (2010). As discussed in section 2.2, the generalizability can be measured using the Hausdorff dimension which is bounded by BG-index.

Therefore, the BG-index is computed in the last iteration when all the neurons have learned the input representations. We also compute the generalization error as the difference between the training and test accuracy. we study the relations between BG-index, generalization error, testing accuracy, and convergence behavior of the networks.

3.2. Analysis of Generalizability of STDP Processes

3.2.1. Impact of Scaling Functions

Kubota et al. showed that the scaling functions play a vital role in controlling the LTP/LTD dynamic of the STDP learning method (Kubota et al., 2009). In this subsection, we evaluate the impact of scaling functions (i.e., $a_{\pm}$ in the Equation 3) on the generalizability properties of the STDP methods. We define the ratio of the post-synaptic scaling function to the pre-synaptic one (i.e., c_+/c_- in add-, mult-, and log- STDP equations), hereafter referred to as the scaling function ratio (SFR), as our variable. Kubota et al. has shown that the learning behavior is best when this SFR lies between the range of 0.9 to 1.5. Hence, we also modulate the SFR within this set interval. **Table 2** shows the impact of scaling function on Hausdorff dimension (measured using BG-index), generalization error, and testing accuracy. We observe that a smaller SFR leads to a lower Hausdorff dimension and a lower generalization error, while a higher ratio infers a less generalizable model. However, a higher SFR marginally increases the testing accuracy. The analysis shows confirms that a higher Hausdorff dimension (i.e., a higher BG-index) corresponds to a higher generalization error, as discussed in section 2.2, justifying the use of BG-index as a measure of the generalization error. We also plot the learned digit representations for different SFRs for better visualization of the distinction in generalization behavior. The plots are shown in **Figure 5**.

3.2.2. Impact of the Learning Rate

One of the major parameters that control the weight dynamics of the STDP processes is the learning rate i.e., the variable η in Equation (2). In this subsection, we evaluate the effect of the learning rate on the generalizability of the STDP process. We have summarized the results in **Table 3**. We observe that a larger learning rate converges to a lesser generalizable solution. This can be attributed to the fact that a higher learning rate inhibits convergence to sharper minimas; rather facilitates convergence to a flatter one resulting in a more generalizable solution. We also observe the monotonic relation between the BG-index and the generalization error.

3.2.3. Impact of STDP models on Generalizability

In this section, we compare the three different STDP models, namely, add-STDP, mult-STDP, and log-STDP to its generalization abilities with changing SFR (scaling function ratio) and learning rate. The results are summarized in **Tables 2**, **3**. In all the above cases we see that the log-STDP process has a

TABLE 2 | Impact of the post-synaptic to pre-synaptic scaling functions ratio on generalization.

$\frac{c+}{c-}$	log-STDP			add-STDP			mult-STDP		
	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy
2.1	1.352	6.8	89.92	1.969	9.7	88.17	1.824	8.1	89.26
1.7	1.294	6.2	89.98	1.911	9.3	88.12	1.797	7.6	89.15
1.2	1.209	5.9	89.79	1.875	8.9	88.09	1.702	7.0	88.99
0.9	1.174	5.7	89.26	1.799	8.6	88.10	1.633	6.5	88.87

The values noted as generalization error in the table is computed as: (|Training Accuracy–Test Accuracy|).

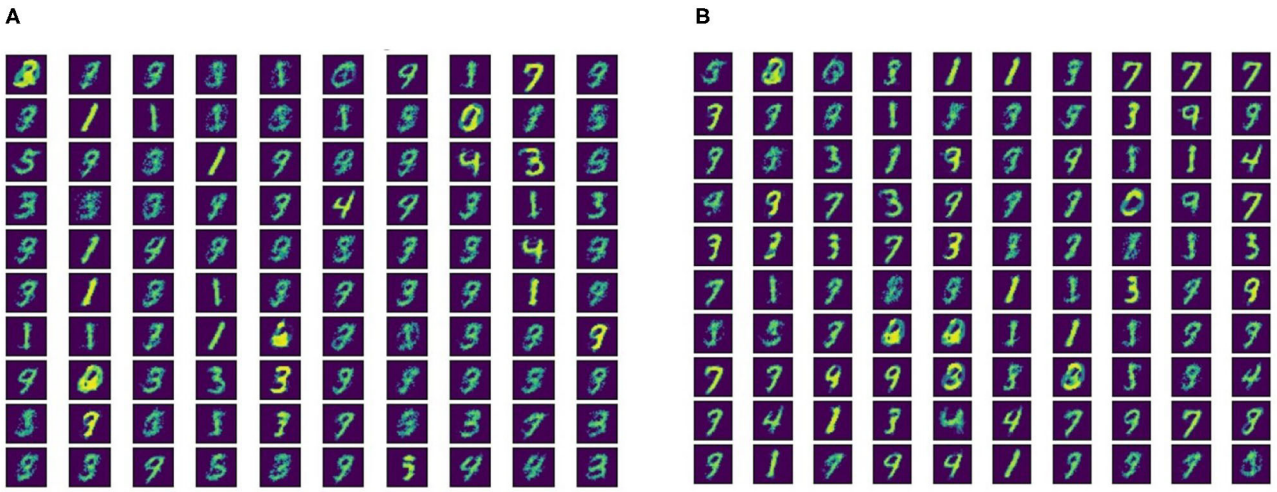

FIGURE 5 | Neuron connection weights for learned digit representations for **(A)** SFR = 0.9 and **(B)** SFR = 2.1.

TABLE 3 | The impact of learning rate on the generalization error.

η	log-STDP			add-STDP			mult-STDP		
	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy
0.2	1.312	6.6	89.12	1.844	9.1	87.69	1.769	7.9	88.38
0.15	1.255	6.1	89.03	1.783	8.9	87.53	1.648	7.4	88.19
0.1	1.112	5.3	88.35	1.698	8.5	87.11	1.596	6.8	87.95
0.05	1.068	5.0	88.01	1.632	8.2	87.02	1.512	6.3	87.82

The values noted as Generalization Error in the table is computed as: (|Training Accuracy–Test Accuracy|).

TABLE 4 | Table showing comparison of the STDP models on the fashion-MNIST dataset.

	log-STDP			add-STDP			mult-STDP		
	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy	BG index	Generalization error (| Train Acc. - Test Acc.|)	Testing accuracy
c+/c_ = 0.9	1.322	10.02	82.43	1.788	13.87	79.16	1.573	11.38	81.92
$\eta = 0.05$	1.204	9.93	81.75	1.692	11.73	79.76	1.489	10.11	80.35

The values noted as generalization error in the table is computed as: (|Training Accuracy–Test Accuracy|).

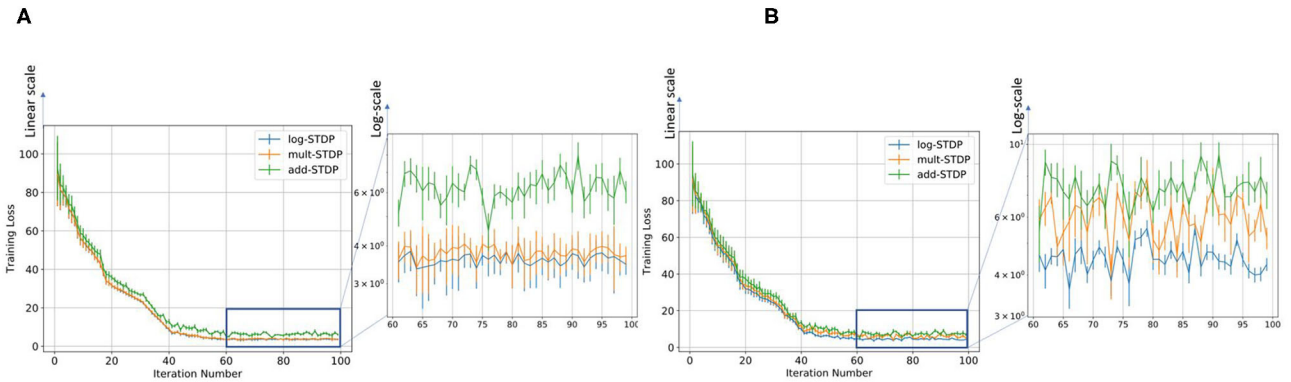

FIGURE 6 | Figure showing the variation in the training loss with increasing iterations for different types of STDP models keeping **(A)** SFR= 0.9 and **(B)** $\eta = 0.05$.

significantly lower generalization error compared to the other two STDP methods. The difference between the generalizability of various STDP models comes from the nature of the stochastic distribution of weights generated by different models.

Gilson and Fukai (2011) has discussed that add-STDP (Gütig et al., 2003) can rapidly and efficiently select synaptic pathways by splitting synaptic weights into a bimodal distribution of weak and strong synapses. However, the stability of the weight distribution requires hard bounds due to the resulting unstable weight dynamics. In contrast in mult-STDP (Rubin et al., 2001), weight-dependent update rules can generate stable unimodal distributions. However, mult-STDP weakens the competition among synapses leading to only weakly skewed weight distributions. The probability distributions of the three different STDP models are shown in **Figure 1**. On the other hand, log-STDP proposed by Gilson and Fukai (2011) bypass these problems by using a weight-dependent update rule while does not make the other synapses weak as in mult-STDP. The log-STDP results in a log-normal solution of the synaptic weight distribution as discussed by Gilson and Fukai (2011). A log-normal solution has a heavier tail and thus a smaller Hausdorff dimension leading to a lower generalization error. A detailed comparison of the weight distributions of the three types of STDP processes can be found in the paper by Gilson and Fukai (2011). We further evaluated the training loss for iterations for the different STDP models. The results are plotted in **Figure 6**. From the figures, we see that the log-STDP outperforms the add-STDP and the mult-STDP in terms of training loss for either case.

3.2.4. Impact on Different Datasets

To demonstrate the generalizability of the STDP models, we also tested its performance on Fashion-MNIST (Xiao et al., 2017) which is an MNIST-like fashion product dataset with 10 classes. Fashion-MNIST shares the same image size and structure of the training and testing splits as MNIST but is considered more realistic as its images are generated from front look thumbnail images of fashion products on Zalando's website via a series of conversions. Therefore, Fashion-MNIST poses a more challenging classification task than MNIST. We preprocessed the data by normalizing the sum of a single sample gray value because of the high variance among examples. The results for SFR $c_+/c_- = 0.9$ and learning rate $\eta = 0.05$ is shown in **Table 4**. We see that as seen in MNIST datasets, for the Fashion-MNIST also, the log-STDP method has a lower generalization error corresponding to a lower BG Index.

3.3. Generalizability vs. Trainability Tradeoff

In this section, we study the relations between the generalizability and trainability of a learning model. For the sake of brevity, we only focus on the log-STDP process as it has shown better generalizability compared to add-STDP and mult-STDP.

We plot the training loss as a function of the time evolution of the synaptic weights trained with the STDP learning method. Since STDP is an online unsupervised learning algorithm, there is no formal concept of training loss. So, to evaluate the performance of the model, we define the training loss as follows: We divide the MNIST dataset into 100 divisions, with each division consisting of 600 images. We evaluate the model after training the model on each subset of the full training dataset and this is considered as one training iteration. We train the SNN model using STDP with this limited training dataset. After the training is done, we set the learning rate to zero and fix each neuron's spiking threshold. Then, the image of each class is given as input to the network, and the total spike count of all the neurons that have learned that class is calculated. Hence, the spike counts are normalized by dividing the number of spikes by the maximum number of spike counts and the cross-entropy loss was calculated across all the classes. This is defined as the training loss. To show the confidence interval of each training iteration, we evaluated each of the partially trained models on the test dataset 5 times, randomizing the order of images for each of the test runs. We see from **Figures 7**, **8**, that initially, as the model was trained with fewer images, the variance in training loss was high demonstrating low confidence. However, as the model is trained on a larger training set, the variance decreases as expected.

Table 2 and **Figure 4** show the training loss vs. the number of iterations for the log-STDP process for various SFR.

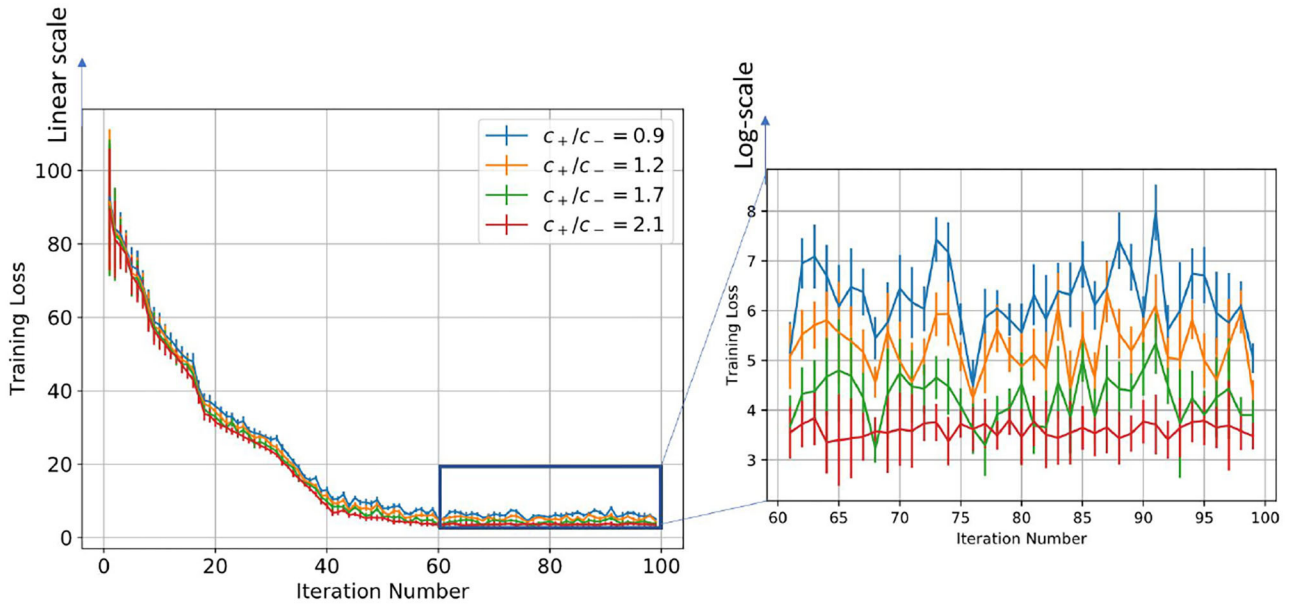

FIGURE 7 | Figure showing the change in training loss with iterations for varying scaling function ratios for the log-STDP learning process.

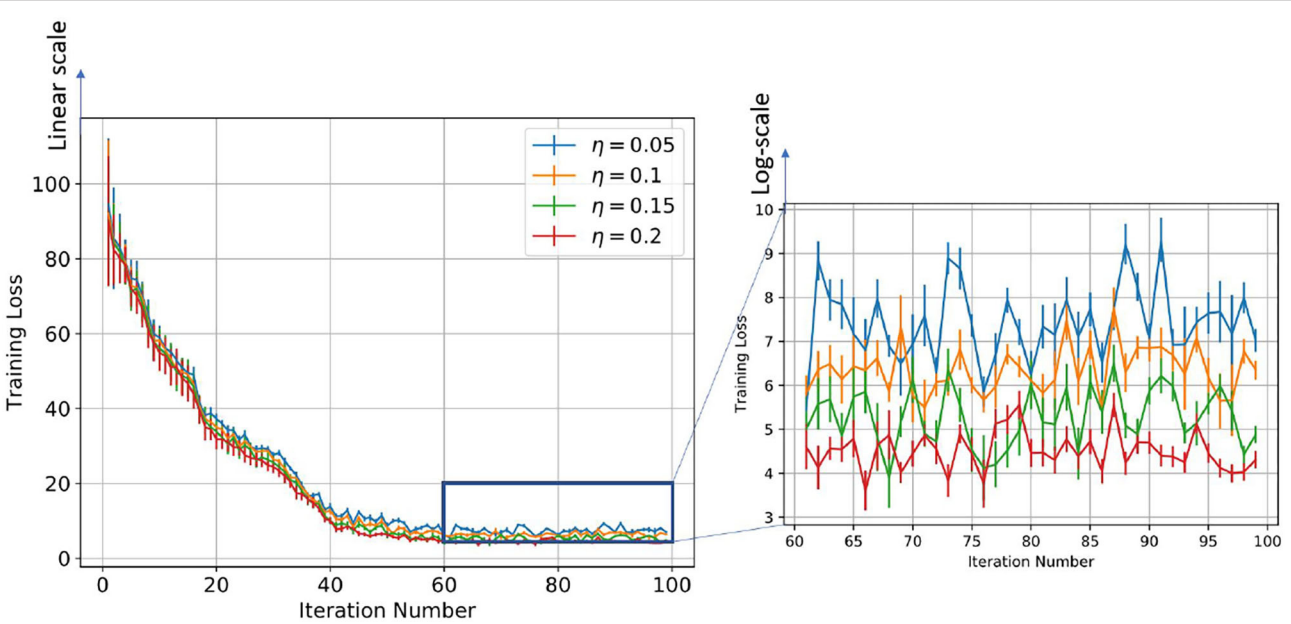

FIGURE 8 | Figure showing the change in training loss with iterations for varying learning rates for the log-STDP learning process.

We see that the $SFR = c_+/c_- = 0.9$ shows a lower generalization error and almost similar testing accuracy, compared to the other SFRs. The results show that increasing the SFR increases the generalization error. If the pre-synaptic scaling function is stronger than the post-synaptic scaling function (i.e., c_+/c_- is lower), it implies that the synaptic weights of the neurons gradually decay. Since we have the images in the MNIST dataset randomized over the ten classes, the more important features which help in the generalization ability of the network over unknown data are reinforced so that the network does not forget these filters as shown by Panda et al. (2017). Thus, the network only forgets the less important features and preserves the more important ones, hence making it more generalizable. Since the neuron forgets some features which would help to fit better into the current dataset, it affects its training/testing

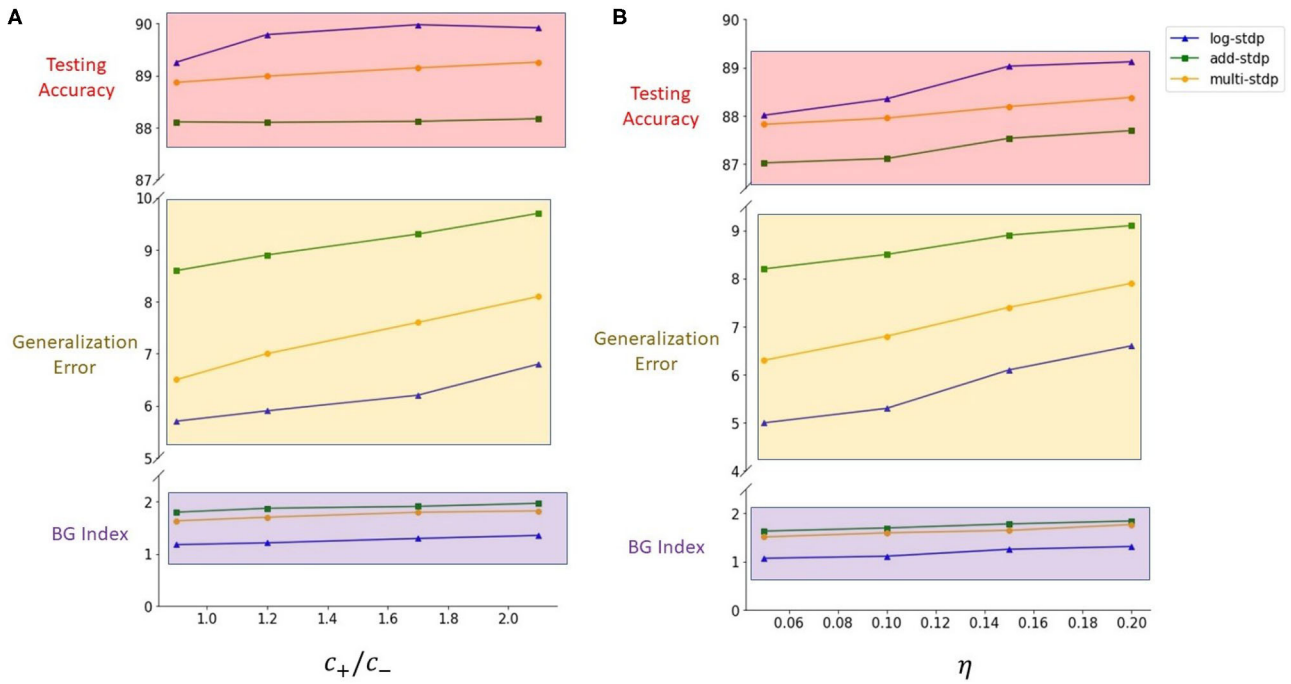

FIGURE 9 | (A) Plots for the impact of the scaling function ratios on generalization (results shown in **Table 2**). **(B)** Plots for the impact of the learning rates on generalization (results shown in **Table 3**).

TABLE 5 | Table showing the set of hyperparameters used for the Bayesian optimization problem for the log-STDP process.

Hyperparameter	Domain	logSTDP		add-STDP	
		Before BO	After BO	Before BO	After BO
Learning rate (η)	[0.05, 0.2]	0.1	0.063	0.1	0.017
Variance of Noise ζ (σ)	[0.1, 1]	0.5	0.581	0.5	0.632
Degree of log-like saturation ($\mathcal{S}$)	$\mathbb{Z} \in [1, 10]$	3	5	N/A	N/A
Exponential Decay factor (γ)	$\mathbb{Z} \in [10, 100]$	45	57	N/A	N/A
Threshold Fixed-point weight (W_0)	[0, 1]	0.5	0.244	N/A	N/A
Scaling functions (c_+, c_-)	(0,1]×(0,1]	0.5, 0.45 ($\frac{c_+}{c_-} = 1.11$)	0.752, 0.788 ($\frac{c_+}{c_-} = 0.954$)	0.5, 0.45 ($\frac{c_+}{c_-} = 1.11$)	0.894, 0.652 ($\frac{c_+}{c_-} = 1.371$)
Time Constants (ms) (τ_+, τ_-)	[10,20]×[20, 40]	15,30	17, 36	15,30	18,22
Testing accuracy		91.41	90.65	88.68	89.77
Generalization error		5.79	3.17	8.29	4.61

accuracy as can be seen in **Tables 2**, **3**. Thus, the model learns the more important features and is essentially more generalizable. The results for testing accuracy, generalization error and BG index for varying SFR and learning rates are shown in **Figure 9**.

Note here that the training loss for the STDP processes all reach their convergence around iteration 60 - i.e., images after that added little information for the training of the model. The models here are not optimized and hence optimizing the hyperparameters can also help in reducing the number of images required for extracting enough information from the training dataset. Thus, if SFR is too high, training gets messed up since a neuron starts spiking for multiple patterns, in which case there is no learning. As the SFR value increases from 1, the SNN tends

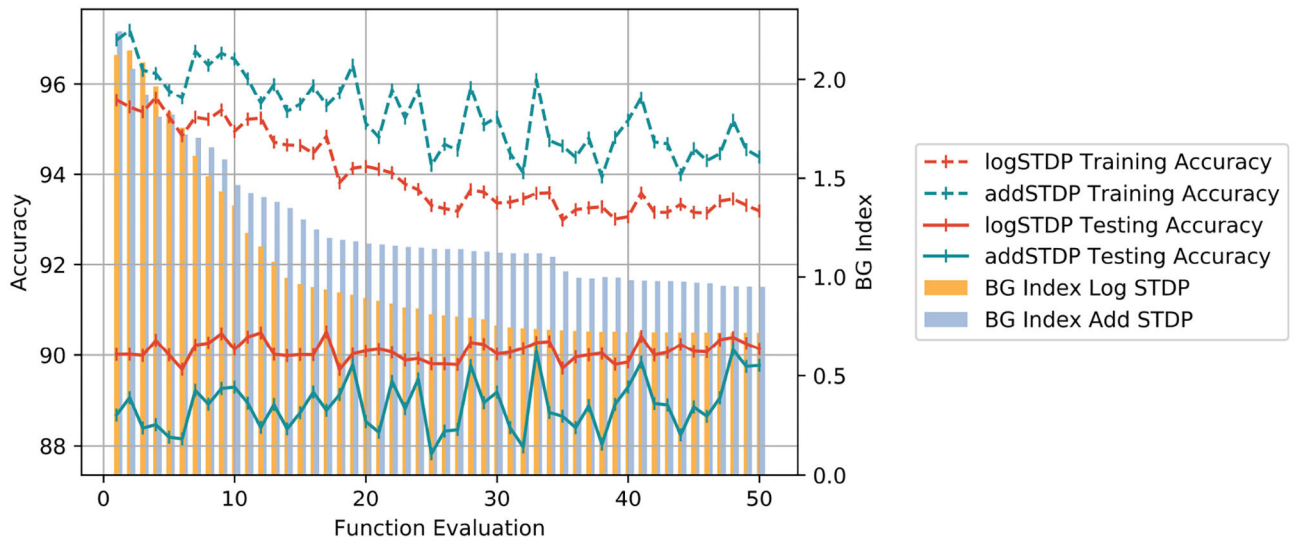

FIGURE 10 | Plot showing the change of BG Index, Training and Testing Accuracy between the add-STDP and log-STDP over functional evaluations during Bayesian Optimization.

to memorize the training pattern and hence the generalization performance is poor. On the other hand, if when SFR is less than 1 but is close to 1, it is hard to memorize the training patterns as the STDP process tends to forget the patterns which are non-repeating, leading to better generalization.

On the other hand, if the post-synaptic scaling function is stronger than the pre-synaptic one (i.e., c_+/c_- is higher), then the neurons tend to learn more than one pattern and respond to all of them. Similar results can be verified from **Figure 7** where the learning rate was varied instead of the SFR. In this study as well we observe that a higher learning rate, although leads to faster convergence and lower training loss, leads to a less generalizable model. Hence, we empirically observe that hyperparameters of STDP models that lead to better generalizability can also make an SNN harder to train.

3.4. Results of Hyperparameter Optimization

In section 3.2, we empirically showed that the Hausdorff dimension is a good measure of the generalizability of the model and it can be very efficiently controlled using the hyperparameters of the STDP learning process. In this section, we show the application of our Bayesian optimization strategy to search for the optimal hyperparameters to increase the generalizability of an STDP-trained SNN model. For the sake of brevity, we demonstrate the application of Bayesian optimization on the log-STDP process. **Table 5** shows the set of hyperparameters that are optimized and their optimal values obtained by our approach. It should be clarified here that the hyper-parameters are not necessarily the absolute global optimum but a likely local optimum found in the optimization algorithm. The optimized log-STDP model results in a training accuracy of 93.75%, testing accuracy of 90.49%, and a BG Index of 0.718 for the MNIST dataset.

We study the behavior of Bayesian optimization. Each iteration in the Bayesian optimization process corresponds to a different set of hyperparameters for the log-STDP model. Each such iteration is called a functional evaluation. For each functional evaluation, the Bayesian Optimization trains the SNN with the corresponding set of hyperparameters of the log-STDP model and measures the BG-index of the weight dynamics of the trained-SNN. **Figure 10** shows the change in the BG-Index as a function of a number of the function evaluations of the search process. It is to be noted here that at each functional evaluation, we train the network with the STDP learning rule with the chosen hyperparameters and estimate the Hausdorff dimension from the trained network. We see that for the optimal set of hyperparameters, the BG Index converges to 0.7. **Figure 10** also shows the corresponding training accuracy of the model with the change in iteration number. We see that the log-STDP configurations during Bayesian optimization that have a higher BG Index (i.e., a higher generalization error) also have has a higher training accuracy. These results further validate our observations on the generalizability vs. trainability tradeoff for a log-STDP trained SNN.

3.4.1. Comparison With Add-STDP

In order to compare the performance of the log-STDP, we performed a similar analysis using the add-STDP model. The results of the Bayesian Optimization for the add-STDP and the log-STDP are plotted in **Figure 10**. We see that the log-STDP process outperforms the add-STDP model in terms of both training/testing accuracy and the generalization error thus showing the robustness of the log-STDP process. We see that though the add-STDP has a higher training accuracy, and a comparable test accuracy, its generalization error is higher compared to the log-STDP method.

4. DISCUSSION

In this paper, we presented the generalization properties of the spike timing-dependent plasticity (STDP) models. A learning process is said to be more generalizable if it can extract features that can be transferred easily to unknown testing sets thus decreasing the performance gap between the training and testing sets. We provide a theoretical background for the motivation of the work treating the STDP learning process as a stochastic process (an Ornstein-Uhlenbeck process) and modeling it using a stochastic differential equation. We control the hyperparameters of the learning method and empirically study their generalizability properties using the Hausdorff dimension as a measure. From **Tables 2**, **3** and corresponding **Figure 9**, we observed that the Hausdorff Dimension is a good measure for the estimation of the generalization error of an STDP-trained SNN. We compared the generalization error and testing error for the log-STDP, add-STDP, and mult-STDP models, as shown in **Tables 2**, **3**. We observed that the lognormal weight distribution obtained from the log-STDP learning process leads to a more generalizable STDP-trained SNN with a minimal decrease in testing accuracy. In this paper, when we refer to a model as more generalizable, we mean there is a smaller difference between the training and testing performance, i.e., the generalization error. The objective of the paper was to get a model which is more generalizable in the sense that the performance of the network on unknown datasets should not differ much from its performance in the training dataset. It is to be noted that in this paper we are using the generalization error as the metric of generalizability of the network. Generalization error is not a measure of absolute accuracy, but rather the difference between training and testing datasets. As such, we see that models which have lower generalization error extract lesser and more important features compared to less generalizable models. However, we see that with this reduced set of features, the model has almost no drop in the testing accuracy, showing the generalizability of the model at comparable accuracy. Thus, we get a model which is more generalizable in the sense that the performance of the network on unknown datasets does not differ much from its performance in the training dataset. As such, these "more generalizable" models, extract lesser and more important features compared to less generalizable models. However, we see that with this reduced set of features, the model has almost no drop in the testing accuracy, showing the generalizability of the model as we can see from **Tables 2**, **3**. This phenomenon can be explained using the observations of Panda et al. (2017) on how the ability to forgets boosts the performance of spiking neuron models. The authors showed that the final weight value toward the end of the recovery phase is greater for the frequent input. The prominent weights will essentially encode the features that are common across different classes of old and new inputs as the pre-neurons across those common feature regions in the input image will have frequent firing activity. This eventually helps the network to learn features that are more common with generic representations across different input patterns. This extraction of more generalizable features can be interpreted as a sort of regularization wherein the network tries to generalize over the input rather than overfitting such that the overall accuracy of the network improves. However, due to this regularization, we see that the training performance of the network decreases. However, since the model is more generalizable, the testing performance remains almost constant as seen in **Figure 10**. We further observe that the log-STDP models which have a lower Hausdorff dimension and hence have lower generalization error, have a worse trainability i.e., takes a long time to converge during training and also converges to a higher training loss. The observations show that an STDP model can have a trade-off between generalizability and trainability. Finally, we present a Bayesian optimization problem that minimizes the Hausdorff dimension by controlling the hyperparameter of a log-STDP model leading to a more generalizable STDP-trained SNN.

Future work on this topic will consider other models of STDP. In particular, the stochastic STDP rule where the probability of synaptic weight update is proportional to the time difference of the arrival of the pre and post-synaptic spikes has shown improved accuracy over deterministic STDP studied in this paper. The trajectories of such a stochastic STDP model will lead to a Feller process as shown by Kuhn (Helson, 2017). Hence, in the future, we will perform a similar Hausdorff dimension-based analysis for generalization for the stochastic STDP model. Moreover, in this work, we have only considered the hyperparameters of the STDP model to improve the generalizability of the SNN. An important extension is to consider the properties of the neuron dynamics, which also controls the generation of the spikes and hence, weight distribution. The choice of the network architecture will also play an important role in the weight distribution of the SNN. Therefore, a more comprehensive optimization process that couples hyperparameters of the STDP dynamics, neuron dynamics, and network architecture like convolutional SNN (Kheradpisheh et al., 2018) and heterogeneous SNN (She et al., 2021) will be interesting future work.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data can be found here: https://deepai.org/dataset/mnist.

AUTHOR CONTRIBUTIONS

BC developed the main concepts, performed simulation, and wrote the paper under the guidance of SM. Both authors assisted in developing the concept and writing the paper.

FUNDING

This material was based on work sponsored by the Army Research Office and was accomplished under Grant Number W911NF-19-1-0447. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government.

REFERENCES

Aceituno, P. V., Ehsani, M., and Jost, J. (2020). Spiking time-dependent plasticity leads to efficient coding of predictions. *Biol. Cybernet*. 114, 43–61. doi: 10.1007/s00422-019-00813-w

Allen-Zhu, Z., and Li, Y. (2019). Can SGD learn recurrent neural networks with provable generalization? *arXiv preprint arXiv:1902.01028*.

Allen-Zhu, Z., Li, Y., and Liang, Y. (2018). Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*.

Baity-Jesi, M., Sagun, L., Geiger, M., Spigler, S., Arous, G. B., Cammarota, C., et al. (2018). "Comparing dynamics: deep neural networks versus glassy systems," in *International Conference on Machine Learning* (PMLR) (Stockholm), 314–323. doi: 10.1088/1742-5468/ab3281

Bell, C. C., Han, V. Z., Sugawara, Y., and Grant, K. (1997). Synaptic plasticity in a cerebellum-like structure depends on temporal order. *Nature* 387, 278–281. doi: 10.1038/387278a0

Bi, G.-Q., and Poo, M.-M. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *J. Neurosci.* 18, 10464–10472. doi: 10.1523/JNEUROSCI.18-24-10464.1998

Bishop, C. J., and Peres, Y. (2017). *Fractals in Probability and Analysis*, Vol. 162. Cambridge University Press. doi: 10.1017/9781316460238

Blumenthal, R. M., and Getoor, R. K. (1960). Some theorems on stable processes. *Trans. Am. Math. Soc*. 95, 263–273. doi: 10.1090/S0002-9947-1960-0119247-6

Burkitt, A. N., Meffin, H., and Grayden, D. B. (2004). Spike-timing-dependent plasticity: the relationship to rate-based learning for models with weight dynamics determined by a stable fixed point. *Neural Comput*. 16, 885–940. doi: 10.1162/089976604773135041

Camuto, A., Deligiannidis, G., Erdogdu, M. A., Gürbüzbalaban, M., Şimşekli, U., and Zhu, L. (2021). Fractal structure and generalization properties of stochastic optimization algorithms. *arXiv preprint arXiv:2106.04881*.

Capocelli, R., and Ricciardi, L. (1976). On the transformation of diffusion processes into the feller process. *Math. Biosci*. 29, 219–234. doi: 10.1016/0025-5564(76)90104-8

Câteau, H., and Fukai, T. (2003). A stochastic method to predict the consequence of arbitrary forms of spike-timing-dependent plasticity. *Neural Comput*. 15, 597–620. doi: 10.1162/089976603321192095

Chen, G., Qu, C. K., and Gong, P. (2020). Anomalous diffusion dynamics of learning in deep neural networks. *arXiv preprint arXiv:2009.10588*.

Chichilnisky, E. (2001). A simple white noise analysis of neuronal light responses. *Netw. Comput. Neural Syst*. 12, 199–213. doi: 10.1080/713663221

Diehl, P. U., and Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci*. 9:99. doi: 10.3389/fncom.2015.00099

Feldman, D. E. (2012). The spike-timing dependence of plasticity. *Neuron* 75, 556–571. doi: 10.1016/j.neuron.2012.08.001

Feurer, M., Springenberg, J., and Hutter, F. (2015). "Initializing bayesian hyperparameter optimization via meta-learning," in *Proceedings of the AAAI Conference on Artificial Intelligence* (Austin, TX).

Gerstner, W., and Kistler, W. M. (2002a). Mathematical formulations of hebbian learning. *Biol. Cybernet*. 87, 404–415. doi: 10.1007/s00422-002-0353-y

Gerstner, W., and Kistler, W. M. (2002b). *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press. doi: 10.1017/CBO9780511815706

Gilson, M., and Fukai, T. (2011). Stability versus neuronal specialization for STDP: long-tail weight distributions solve the dilemma. *PLoS ONE* 6:e25339. doi: 10.1371/journal.pone.0025339

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Gurbuzbalaban, M., Simsekli, U., and Zhu, L. (2020). The heavy-tail phenomenon in SGD. *arXiv preprint arXiv:2006.04740*.

Gütig, R., Aharonov, R., Rotter, S., and Sompolinsky, H. (2003). Learning input correlations through nonlinear temporally asymmetric Hebbian plasticity. *J. Neurosci*. 23, 3697–3714. doi: 10.1523/JNEUROSCI.23-09-03697.2003

Han, V. Z., Grant, K., and Bell, C. C. (2000). Reversible associative depression and nonassociative potentiation at a parallel fiber synapse. *Neuron* 27, 611–622. doi: 10.1016/S0896-6273(00)00070-2

Helson, P. (2017). A new stochastic stdp rule in a neural network model. *arXiv preprint arXiv:1706.00364*.

Hodgkinson, L., and Mahoney, M. (2021). "Multiplicative noise and heavy tails in stochastic optimization," in *International Conference on Machine Learning* (PMLR), 4262–4274.

Jones, M. C., Marron, J. S., and Sheather, S. J. (1992). *Progress in Data-Based Bandwidth Selection for Kernel Density Estimation*. Technical report. North Carolina State University. Dept. of Statistics, Raleigh, NC, United States.

Kawaguchi, K., Kaelbling, L. P., and Bengio, Y. (2017). Generalization in deep learning. *arXiv preprint arXiv:1710.05468*.

Kheradpisheh, S. R., Ganjtabesh, M., Thorpe, S. J., and Masquelier, T. (2018). STDP-based spiking deep convolutional neural networks for object recognition. *Neural Netw*. 99, 56–67. doi: 10.1016/j.neunet.2017.12.005

Khoshnevisan, D. (2009). "From fractals and probability to Lévy processes and stochastic PDES," in *Fractal Geometry and Stochastics IV*, eds C. Bandt, M. Zähle, and P. Mörters (Basel: Springer), 111–141. doi: 10.1007/978-3-0346-0030-9_4

Khoshnevisan, D., and Xiao, Y. (2017). "On the macroscopic fractal geometry of some random sets," in *Stochastic Analysis and Related Topics,* eds F. Baudoin and J. Peterson (Cham: Springer), 179–206. doi: 10.1007/978-3-319-59671-6_9

Kubota, S., Rubin, J., and Kitajima, T. (2009). Modulation of LTP/LTD balance in STDP by an activity-dependent feedback mechanism. *Neural Netw*. 22, 527–535. doi: 10.1016/j.neunet.2009.06.012

Lőrinczi, J., and Yang, X. (2019). Multifractal properties of sample paths of ground state-transformed jump processes. *Chaos Solitons Fractals* 120, 83–94. doi: 10.1016/j.chaos.2019.01.008

Le Guével, R. (2019). The hausdorff dimension of the range of the Lévy multistable processes. *J. Theoret. Probabil*. 32, 765–780. doi: 10.1007/s10959-018-0847-8

Lee, C., Srinivasan, G., Panda, P., and Roy, K. (2018). Deep spiking convolutional neural network trained with unsupervised spike-timing-dependent plasticity. *IEEE Trans. Cogn. Dev. Syst*. 11, 384–394. doi: 10.1109/TCDS.2018.2833071

Legenstein, R., Pecevski, D., and Maass, W. (2008). A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Comput. Biol*. 4:e1000180. doi: 10.1371/journal.pcbi.1000180

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural Netw*. 10, 1659–1671. doi: 10.1016/S0893-6080(97)00011-7

Magee, J. C., and Johnston, D. (1997). A synaptically controlled, associative signal for Hebbian plasticity in hippocampal neurons. *Science* 275, 209–213. doi: 10.1126/science.275.5297.209

Markram, H., Lübke, J., Frotscher, M., and Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APS and EPSPS. *Science* 275, 213–215. doi: 10.1126/science.275.5297.213

Masquelier, T., Guyonneau, R., and Thorpe, S. J. (2009). Competitive STDP-based spike pattern learning. *Neural Comput*. 21, 1259–1276. doi: 10.1162/neco.2008.06-08-804

Meerschaert, M. M., and Xiao, Y. (2005). Dimension results for sample paths of operator stable Lévy processes. *Stochast. Process. Appl*. 115, 55–75. doi: 10.1016/j.spa.2004.08.004

Mockus, J., Tiesis, V., and Zilinskas, A. (1978). The application of bayesian methods for seeking the extremum. *Towards global optimization* 2:2.

Mohammadi, M., Mohammadpour, A., and Ogata, H. (2015). On estimating the tail index and the spectral measure of multivariate α-stable distributions. *Metrika* 78, 549–561. doi: 10.1007/s00184-014-0515-7

Morrison, A., Aertsen, A., and Diesmann, M. (2007). Spike-timing-dependent plasticity in balanced random networks. *Neural Comput*. 19, 1437–1467. doi: 10.1162/neco.2007.19.6.1437

Mozafari, M., Ganjtabesh, M., Nowzari-Dalini, A., Thorpe, S. J., and Masquelier, T. (2019). Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks. *Pattern Recogn*. 94, 87–95. doi: 10.1016/j.patcog.2019.05.015

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. (2017). Exploring generalization in deep learning. *arXiv preprint arXiv:1706.08947*.

Panda, P., Allred, J. M., Ramanathan, S., and Roy, K. (2017). ASP: learning to forget with adaptive synaptic plasticity in spiking neural networks. *IEEE J. Emerg. Select. Top. Circ. Syst*. 8, 51–64. doi: 10.1109/JETCAS.2017.2769684

Pfeiffer, M., and Pfeil, T. (2018). Deep learning with spiking neurons: opportunities and challenges. *Front. Neurosci*. 12:774. doi: 10.3389/fnins.2018.00774

Poggio, T., Banburski, A., and Liao, Q. (2019). Theoretical issues in deep networks: approximation, optimization and generalization. *arXiv preprint arXiv:1908.09375*.

Querlioz, D., Bichler, O., Dollfus, P., and Gamrat, C. (2013). Immunity to device variations in a spiking neural network with memristive nanodevices. *IEEE Trans. Nanotechnol.* 12, 288–295. doi: 10.1109/TNANO.2013.2250995

Richardson, M. J., and Swarbrick, R. (2010). Firing-rate response of a neuron receiving excitatory and inhibitory synaptic shot noise. *Phys. Rev. Lett.* 105:178102. doi: 10.1103/PhysRevLett.105.178102

Robert, P., and Vignoud, G. (2020). Stochastic models of neural synaptic plasticity. *arXiv preprint arXiv:2010.08195*.

Roberts, P. D., and Bell, C. C. (2000). Computational consequences of temporally asymmetric learning rules: II. Sensory image cancellation. *J. Comput. Neurosci.* 9, 67–83. doi: 10.1023/A:1008938428112

Rubin, J., Lee, D. D., and Sompolinsky, H. (2001). Equilibrium properties of temporally asymmetric Hebbian plasticity. *Phys. Rev. Lett.* 86:364. doi: 10.1103/PhysRevLett.86.364

She, X., Dash, S., Kim, D., and Mukhopadhyay, S. (2021). A heterogeneous spiking neural network for unsupervised learning of spatiotemporal patterns. *Front. Neurosci.* 14:1406. doi: 10.3389/fnins.2020.615756

Sheather, S. J., and Jones, M. C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *J. R. Stat. Soc. Ser. B* 53, 683–690. doi: 10.1111/j.2517-6161.1991.tb01857.x

Simsekli, U., Sagun, L., and Gurbuzbalaban, M. (2019). A tail-index analysis of stochastic gradient noise in deep neural networks. *arXiv preprint arXiv:1901.06053*.

Simsekli, U., Sener, O., Deligiannidis, G., and Erdogdu, M. A. (2020a). "Hausdorff dimension, heavy tails, and generalization in neural networks," in *Advances in Neural Information Processing Systems 33*.

Simsekli, U., Zhu, L., Teh, Y. W., and Gurbuzbalaban, M. (2020b). "Fractional underdamped langevin dynamics: retargeting SGD with momentum under heavy-tailed gradient noise," in *International Conference on Machine Learning* (PMLR) (Vienna), 8970–8980.

Sinz, F. H., Pitkow, X., Reimer, J., Bethge, M., and Tolias, A. S. (2019). Engineering a less artificial intelligence. *Neuron* 103, 967–979. doi: 10.1016/j.neuron.2019.08.034

Song, S., Miller, K. D., and Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nat. Neurosci.* 3, 919–926. doi: 10.1038/78829

Stein, R. B. (1965). A theoretical analysis of neuronal variability. *Biophys. J.* 5, 173–194. doi: 10.1016/S0006-3495(65)86709-1

Van Rossum, M. C., Bi, G. Q., and Turrigiano, G. G. (2000). Stable Hebbian learning from spike timing-dependent plasticity. *J. Neurosci.* 20, 8812–8821. doi: 10.1523/JNEUROSCI.20-23-08812.2000

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xiao, Y. (2003). Random fractals and markov processes. *Math. Preprint Arch.* 2003, 830–907.

Yang, X. (2018). Multifractality of jump diffusion processes. *Ann. Inst. H. Probab. Stat.* 54, 2042–2074. doi: 10.1214/17-AIHP864

Zador, A. M. (2019). A critique of pure learning and what artificial neural networks can learn from animal brains. *Nat. Commun.* 10, 1–7. doi: 10.1038/s41467-019-11786-6

Zhang, W., and Linden, D. J. (2003). The other side of the engram: experience-driven changes in neuronal intrinsic excitability. *Nat. Rev. Neurosci.* 4, 885–900. doi: 10.1038/nrn1248

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.