

Article

A Particle Swarm Optimization-Based Interpretable Spiking Neural Classifier with Time-Varying Weights

Mohammed Thousif ^{1,*} , Shirin Dora ²  and Suresh Sundaram ¹ ¹ Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560012, India; vssuresh@iisc.ac.in² Department of Computer Science, Loughborough University, Loughborough LE11 3TU, UK; s.dora@lboro.ac.uk

* Correspondence: pagalat@iisc.ac.in

Abstract: This paper presents an interpretable, spiking neural classifier (IpT-SNC) with time-varying weights. IpT-SNC uses a two-layered spiking neural network (SNN) architecture in which weights of synapses are modeled using amplitude-modulated, time-varying Gaussian functions. Self-regulated particle swarm optimization (SRPSO) is used to update the amplitude, width, and centers of the Gaussian functions and thresholds of neurons in the output layer. IpT-SNC has been developed to improve the interpretability of spiking neural networks. The time-varying weights in IpT-SNC allow us to describe the rationale behind predictions in terms of specific input spikes. The performance of IpT-SNC is evaluated on ten benchmark datasets in the UCI machine learning repository and compared with the performance of other learning algorithms. According to the performance results, IpT-SNC enhances classification performance on testing datasets from a minimum of 0.5% to a maximum of 7.7%. The significance level of IpT-SNC with other learning algorithms is evaluated using statistical tests like the Friedman test and the paired *t*-test. Furthermore, on the challenging real-world BCI (Brain Computer Interface) competition IV dataset, IpT-SNC outperforms current classifiers by about 8% in terms of classification accuracy. The results indicate that IpT-SNC has better generalization performance than other algorithms.

Keywords: self-regulated particle swarm optimization; spiking neural network; classifier; interpretable**MSC:** 68T20

Citation: Thousif, M.; Dora, S.; Sundaram, S. A Particle Swarm Optimization-Based Interpretable Spiking Neural Classifier with Time-Varying Weights. *Mathematics* **2024**, *12*, 2846. <https://doi.org/10.3390/math12182846>

Academic Editor: Antonio Bolufé-Röhler

Received: 17 August 2024

Revised: 7 September 2024

Accepted: 10 September 2024

Published: 13 September 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Biological neurons process and transmit information based on discrete temporal events, which are termed spikes. Artificial neurons that use spikes for signal processing are known as spiking neurons, and neural networks composed of these neurons are termed spiking neural networks (SNNs). In [1], it is shown that SNNs are more computationally powerful than neural networks made up of sigmoidal neurons. This enabled the development of SNNs for various tasks, such as classification [2], recognition [3], object detection [4], etc.; the Gradient-based backpropagation learning rule cannot be directly implemented in SNNs because of the discontinuous nature of spikes. Hence, in SpikeProp [5], to evaluate the gradient, linearity is assumed at the time of the spike in the potential developed using a spiking neuron. Similarly, many studies are available in the literature, such as [6,7], which evaluate surrogate gradients to update weights in SNNs.

Other than gradient-based approaches, several bio-inspired learning algorithms for SNNs are also available in the literature. Spike timing-dependent plasticity (STDP) [8] is one such biological phenomenon that served as the foundation for SNN learning algorithms in [2,3]. In STDP, if the neuron connected at the input side of the synapse (presynaptic neuron) fires earlier than the neuron connected to the output side (postsynaptic neuron), then the weight of the synapse is increased, and vice versa. Other biologically inspired

phenomena, like rank-order learning [9] and astrocyte-inspired learning [10], have also been utilized as a basis for developing learning techniques for SNNs. All of the learning techniques previously stated assume that the weight of synaptic connections between two neurons in an SNN does not change over time after updation. This long-lasting phenomenon in the weights of the synapses is defined as long-term plasticity (LTP) [11] in the brain.

Contrary to this, short-term plasticity (STP) refers to changes exhibited by biological synapses that last up to a short duration [12,13]. The synapses that exhibit STP are also termed dynamic synapses, and their update functions depend on the input spikes. This ability of dynamic synapses increases the computational power of the SNN, making them suitable for the development of compact neural network architectures [14]. SNNs with dynamic synapses have been utilized for speech recognition [14–16], pattern classification [17], neuromorphic computing [18], etc. Even though dynamic synapses are updated using a dynamic function, their strengths are represented using a static function. In [19], a synaptic efficacy function-based leaky integrate and fire neuron (SEFRON) is proposed in which a synapse model whose strength and updating are both formulated using time-varying functions. Each synapse in SEFRON is modeled using amplitude-modulated, time-varying Gaussian functions. The update for this time-varying weight of a synapse is also distributed over time using Gaussian functions. The time-varying weight synapse model proposed in SEFRON has shown better classification performance when compared with the dynamic synapse model. This alleviates the need to develop SNNs with synapses that are more dynamic in nature for better performance results. The classifier developed in SEFRON is suitable only for binary classification problems. In [20], SEFRON work is further extended for multi-classification problems with the addition of a meta-neuron to the network. Hence, it is termed a meta-neuron spiking classifier with a time-varying weight model (MeST). MeST utilizes the concept of a meta-neuron for online learning in SNNs [10]. Both SEFRON and MeST employ dynamic synapses, but it is not possible to interpret the reasoning underlying their predictions. In [21], a method is developed to process the time-varying weights in SEFRON for better interpretability. For each input feature, a shape function is estimated from the time-varying weights in [21] to understand the significance of features for classification performance. The approximation used to estimate the shape functions may result in a loss of information and lead to inaccurate interpretations.

In [22], an evolving SNN architecture is proposed that updates the weights based on the normalization of the potential of neurons. The weights in [22] are updated until the difference between the spike times of neurons reaches a certain threshold; hence, it is termed a threshold margin maximization learning algorithm for an evolving spiking neural network (TMM-SNN). Researchers have also trained SNNs by incorporating fuzzy logic techniques as in contribution degree-based SNN (CD-SNN) [23] and triangular spike response function-SNN (T-SNN) [24]. In CD-SNN, three membership functions are evaluated using the potential developed via spiking. These membership functions are further used to train an SNN. In T-SNN, a fuzzy C-means clustering algorithm is used to encode real-valued data to spikes. However, fuzzy techniques have limitations related to dimensionality, scalability, etc. In addition to fuzzy techniques and conventional learning, the search-based particle swarm optimization (PSO) algorithm is also used to train artificial neural networks in recommendation systems [25], to protect wireless sensor networks [26], etc. Optimization methods based on evolutionary computation provide an effective alternative approach to developing SNNs without relying on approximations for learning. Quantum-inspired particle swarm optimization (QiPSO) for feature selection and optimizing the parameters of an evolving SNN was proposed in [27,28]. QiPSO utilizes the probabilistic properties of quantum bits to optimize the parameters of SNNs with stochastic responses. In [29], a cooperative PSO (CPSO) was developed to optimize the parameters of SNNs. It was demonstrated in [29] that, by improving the weights and delays of the SNN, CPSO outperforms SpikeProp. It can be observed that existing optimization methods for SNNs do not focus on the interpretability of solutions.

In this paper, we develop an interpretable, time-varying, weight-spiking neural classifier (IpT-SNC) in which weights are modeled using multiple amplitude-modulated Gaussian functions. The centers of the Gaussian functions are located at different time instants over the simulation interval. Thus, the weight of a given presynaptic spike at a particular synapse is determined according to the temporal separation between the time of that spike and the center of all Gaussian functions associated with that particular synapse. A search-based algorithm based on self-regulated particle swarm optimization (SRPSO) [30] is implemented to update the amplitude, width, and centers of all Gaussian functions, along with the potential thresholds of the neurons in the output layer of IpT-SNC. SRPSO is a variant of PSO [31] that employs self-regulation to overcome premature convergence. Further works are carried out to improve the performance of SRPSO, such as in [30,32]. As a preliminary step in this work, SRPSO is used to update the parameters of IpT-SNC in order to overcome the computational complexities in the learning algorithms. We show that the output of IpT-SNC can be interpreted in terms of the times of the spikes generated via the input neurons. The interpretability of IpT-SNC is demonstrated using a synthetic classification problem by identifying a region using the simulation interval (with respect to input spikes) that influences the predictions.

Ten benchmark datasets from the UCI machine learning repository [33] are used to evaluate the performance of IpT-SNC. The performance of IpT-SNC is then compared with Spikeprop [5], TMM-SNN [22], DC-SNN [2], SEFRON [19], MeST [20], CD-SNN [23], and T-SNN [24]. Hypothesis testing is used to statistically validate the significance of IpT-SNC over other learning algorithms. These statistical findings demonstrate that IpT-SNC outperforms other learning algorithms utilized for comparison in terms of generalization performance. Further, the performance of IpT-SNC is evaluated on the Brain-Computer Interface (BCI) competition IV-2a dataset to classify left-hand vs. right-hand movements.

The main contributions of this paper are as follows:

- To the best of our knowledge, this is the first time in the literature that a heuristic-based SRPSO algorithm has been used to update the parameters of a spiking neural classifier.
- Inherent interpretations of the proposed IpT-SNC's performance are drawn in relation to the input information using trained weights.
- Performance results indicate that the proposed IpT-SNC has 8% higher accuracy with 3.5% less of a standard deviation than existing classifiers while classifying complex EEG-based BCI datasets, resulting in better generalization.
- Performance results show that, in classifying complex EEG-based BCI datasets, the proposed IpT-SNC achieves a 3.5% lower standard deviation and 8% higher accuracy than existing classifiers, leading to better generalization.

The paper is organized as follows. The architecture of the IpT-SNC and the SRPSO-based parameter update are described in Section 2. The performance evaluation and comparison with different learning algorithms for SNNs results are presented in Section 3. The conclusions are summarized in Section 4.

2. Interpretable Spiking Neural Classifier IpT-SNC

The network architecture of the IpT-SNC and the modeling of the time-varying weights are described in this section. Self-regulated particle swarm optimization (SRPSO) for updating the parameters of IpT-SNC is also described in this section.

2.1. IpT-SNC Architecture

A two-layered SNN architecture is used in IpT-SNC. The input layer consists of m neurons, which are used to feed the network with input samples $\{(x_1, c_1), \dots, (x_n, c_n), \dots\}$. The spike pattern of the n^{th} input sample is represented as $x_n (= [x_n^1, \dots, x_n^i, \dots, x_n^m])$, which is m -dimensional. If C represents the total classes in the dataset, then the class label of input sample x_n is represented as $c_n, (\in \{1, \dots, C\})$. The input spike pattern with F_i spikes is represented as $x_n^i (= [t_i^1, \dots, t_i^f, \dots, t_i^{F_i}])$, where t_i^f denotes the f^{th} spike produced via the

i^{th} input neuron. The output layer consists of C leaky integrated and fire-spiking neuron models, as in [5]. The neurons in the output layer are associated with a class in the dataset and designated as $\{y_1, \dots, y_j, \dots, y_c\}$. The spike patterns of these output neurons are used to predict the class for a given input sample. Figure 1a shows the network architecture of IpT-SNC.

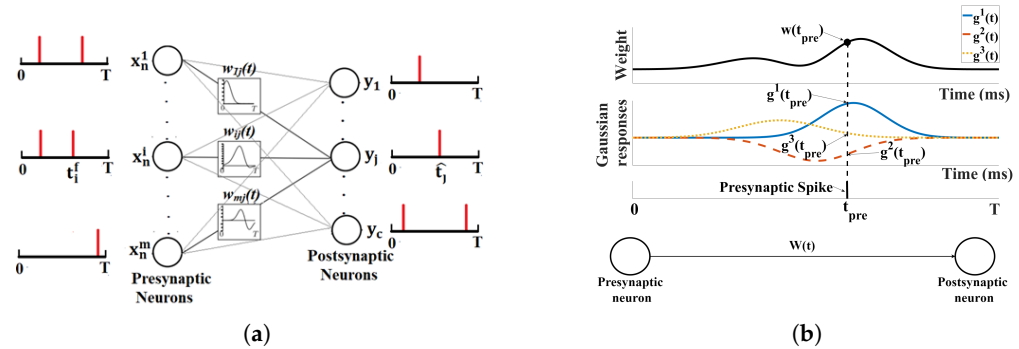


Figure 1. IpT-SNC architecture and time-varying weight model of a synapse. (a) The architecture of the time-varying weight spiking neural network. (b) Time-varying weight $w(t)$ of a synapse connecting a presynaptic neuron and postsynaptic neuron. $g^1(t)$, $g^2(t)$, and $g^3(t)$ represent the Gaussian functions constituting $w(t)$.

$w_{ij}(t)$ represents the time-varying weight of the synapse between the i^{th} input neuron and the j^{th} output neuron. Hence, the input neurons are termed presynaptic neurons, and the output neurons are termed postsynaptic neurons. The potential developed in the j^{th} postsynaptic neuron due to the incoming spikes is given as

$$v_j(t) = \sum_i \sum_f w_{ij}(t_i^f) \cdot \epsilon(t - t_i^f) \quad (1)$$

where ϵ is the unweighted potential known as the spike response function [5] given by Equation (2).

$$\epsilon(s) = \frac{s}{\tau} \exp\left(1 - \frac{s}{\tau}\right) \quad (2)$$

where τ denotes the neuron's time constant. A spike is produced when the potential of the j^{th} postsynaptic neuron rises over a potential threshold (θ_j). The class predicted via the network for a given input spike pattern, x_n , is identified using spikes produced through postsynaptic neurons, as shown below.

$$\hat{c}_n = \arg \min_{j \in \{1, \dots, C\}} \hat{t}_j \quad (3)$$

where $\hat{t}_j$ represents the time of the first spike generated via the j^{th} output neuron. From Equation (1), note that the potential contributed via a particular presynaptic spike depends on the weight, $w_{ij}(t)$, of that connection at the time of the spike. The weight of each synapse, $w_{ij}(t)$, in IpT-SNC is modeled as a mixture of multiple amplitude-modulated Gaussian functions. If K_{ij} Gaussian functions are present in the synapse connecting the i^{th} presynaptic neuron and the j^{th} postsynaptic neuron, weight $w_{ij}(t)$ of the synapse is evaluated as

$$w_{ij}(t) = \sum_{k=1}^{K_{ij}} g_{ij}^k(t) \quad (4)$$

where $g_{ij}^k(t)$ represents the output of the k^{th} Gaussian function at time t , given by

$$g_{ij}^k(t) = \alpha_{ij}^k \exp\left(-\frac{(t - \mu_{ij}^k)^2}{2\sigma_{ij}^{k2}}\right) \quad (5)$$

where, for the connection between the i^{th} input neuron and the j^{th} output neuron, α_{ij}^k , μ_{ij}^k , and σ_{ij}^k represent the amplitude, center, and width of the k^{th} Gaussian function present in that connection, respectively. It is assumed that the IpT-SNC architecture number of Gaussians in all synapses is equal. Therefore, K_{ij} can be replaced with K , which represents the number of Gaussians per synapse in the IpT-SNC architecture. Figure 1b shows the time-varying weight for a synapse between a presynaptic neuron and a postsynaptic neuron. Below the synapse is a presynaptic spike, t_{pre} , between 0 to T , where T represents the duration of the simulation, g^1 , g^2 , and g^3 represent the response of the Gaussians on this synapse at each time step during the simulation. $w(t_{pre})$ represents the weight at the presynaptic spike, t_{pre} .

The amplitudes, centers, and widths of all synapses and the potential thresholds of all postsynaptic neurons in the IpT-SNC network are updated using SRPSO. This update is carried out in such a way that the postsynaptic neuron associated with the same class as that of the input sample always fires first. The SRPSO-based parameter-update strategy is presented in the next section.

2.2. IpT-SNC Parameter Update Strategy Using SRPSO

SRPSO is a variant of particle swarm optimization (PSO) that adds features such as self-cognition and social cognition to PSO for better optimization. In this work, SRPSO is used to search for optimal parameter settings of IpT-SNC in order to obtain better classification performance. At first, SRPSO is used to initialize a population of solutions (termed swarm size, P), each of which represents a single possible setting for the parameters being optimized. The individual solutions in the population are also termed particles, and the associated parameter values represent the position of a given particle. The position of particles is then adjusted in each iteration to estimate parameter values that yield the best value for the considered fitness function. In this paper, SRPSO is used to estimate the values of the amplitude, center, and width of all the Gaussian functions and the threshold of all output neurons in IpT-SNC. It uses classification accuracy in evaluating the fitness function.

Without losing generality, let us assume that h iterations have been completed and that the l^{th} particle's current position is indicated by ρ_h^l . ρ_h^l consists of the updated values of parameters of all Gaussian functions in the IpT-SNC architecture, along with the potential thresholds of postsynaptic neurons after h iterations.

$$\rho_h^l = \left\{ \alpha_{ij}^k, \mu_{ij}^k, \sigma_{ij}^k (\forall i \in [1, m], j \in [1, C], k \in [1, K]) \right\}^h \cup \left\{ \theta_j (j \in [1, C]) \right\}^h \quad (6)$$

For ease of understanding, the particle ρ_h^l can be expressed as below:

$$\rho_h^l = \{\rho_h^{l_1}, \rho_h^{l_2}, \rho_h^{l_3}, \dots, \rho_h^{l_d}, \dots, \rho_h^{l_D}\} \quad (7)$$

where $\rho_h^{l_d}$ denotes the ρ_h^l position in d^{th} dimension. D denotes the number of dimensions in the ρ_h^l particle. It is equal to the total number of parameters that are being updated using SRPSO, and it is evaluated as

$$D = (m \cdot C \cdot K \cdot 3) + C \quad (8)$$

The position of the $\rho_h^{l_d}$ in $h + 1^{th}$ iteration is obtained using the velocity function $v_{h+1}^{l_d}$, as shown below

$$\begin{aligned} \rho_{h+1}^{l_d} &\leftarrow \rho_h^{l_d} + v_{h+1}^{l_d} \\ v_{h+1}^{l_d} &= \omega v_h^{l_d} + z_1 r_1 \gamma_1 (Pbest_h^{l_d} - \rho_h^{l_d}) + z_2 r_2 \gamma_2 (Gbest_h^{l_d} - \rho_h^{l_d}) \end{aligned} \quad (9)$$

where $Pbest_h^{l_d}$ denotes the personal best position of $p_h^{l_d}$ in the d^{th} dimension, and $Gbest_h^{l_d}$ denotes the personal best position in the d^{th} dimension until h iterations. These best positions are the ones that give the least values for the fitness function F . The fitness function F is evaluated such that decreasing it improves the overall accuracy, η_o , as shown in Equation (12). Mathematically the fitness function, F , is given as

$$F = 100 - \eta_o \quad (10)$$

The acceleration coefficients z_1 and z_2 in Equation (9) are selected such that the particle reaches its best position as quickly as possible. According to [30], these z_1 and z_2 coefficients are set to 1.49. r_1 and r_2 in Equation (9) are set so that the solution obtained using SRPSO does not get stuck in the local optimum. The uniform distribution over the range $[0, 1]$ is used to generate the random values for r_1 and r_2 [30].

The inertia weight, ω , in Equation (9) is used to balance the processes of exploration and exploitation while searching in SRPSO. The best particle ω is linearly incremented to ensure that it maintains the current rate of acceleration in the search space. For particles other than the best particle, the ω decreases linearly, as the direction in which these particles are heading is not the best direction [30].

γ_1 and γ_2 in Equation (9) are factors for self-cognition and social cognition, respectively. The self-cognition factor γ_1 enables particles to accelerate toward their personal best-known position. Due to the social cognition factor γ_2 , particles are made to advance more quickly toward the global best-known position. For the best particle, γ_1 and γ_2 are set to zero. This is because the best particle disregards both self- and social cognitions because it thinks its orientation is the best. For other particles, γ_1 is set to one, thus enabling them to search in the direction of their personal best-known positions. γ_2 is set to either zero or one. When γ_2 is set to one, the solution will be biased towards the global best-known position in the current iteration, which may not be the overall global optimum. When γ_2 is set to zero, the solution may never converge to the global optimum. If u is any uniform random number, then, to achieve balanced social cognition, the value of γ_2 is chosen as described in [30].

$$\gamma_2 = \begin{cases} 1, & \text{if } u > 0.5 \\ 0, & \text{else} \end{cases} \quad (11)$$

The SRPSO search process is performed for all particles for 500 iterations. The pseudocode for searching the gaussian parameters and threshold of output neurons of IpT-SNC is provided in Algorithm 1. The final performance accuracies are evaluated for a given dataset. The demonstration of interpretability and a performance comparison of IpT-SNC are described in the next section.

Algorithm 1 Pseudocode for the updating of the IpT-SNC parameters using SRPSO

```

1: for each spike pattern in training do
2:   Compute output spike pattern
3:   Compute accuracy,  $\eta_o$ , using Equation (12)
4:   Compute fitness function,  $F$ , using Equation (10)
5:   while  $F \neq 0$  Or  $h \leq \text{max iterations}$  do
6:     Search for Gaussian parameters ( $\alpha, \mu, \sigma$ ) & potential threshold using SRPSO
7:     Compute accuracy,  $\eta_o$ , using Equation (12)
8:     Compute updated fitness function  $F$  using Equation (10)
9:      $h = h + 1$ 
10:  end while
11: end for

```

3. Interpretability Demonstration and Performance Evaluation of IpT-SNC

The performance of the IpT-SNC is evaluated in this section using ten benchmark datasets from the UCI machine learning repository [33]. The problems were chosen to

include binary and multiclass problems with different levels of difficulty. The results obtained are compared with existing SNN classifiers such as Spikeprop, TMM-SNN, DC-SNN SEFRON, MeST, CD-SNN, and T-SNN. When the performance is compared in multiclass problems, SEFRON is not considered, as it was developed only for binary classification problems. The overall accuracy of IpT-SNC on a given dataset is evaluated using Equation (12).

$$\eta_o = \frac{100}{\sum_{a,b} q_{ab}} \sum_a q_{aa} \quad (12)$$

where Q represents the confusion matrix, and q_{ab} stands for the element in the a^{th} row and b^{th} column in Q . MATLAB 2021a and a Windows computer with 12 logical cores, 16 GB of memory, and a speed of 3.2 GHz are used for the performance evaluation of IpT-SNC.

Population coding: Real-value data-input features are encoded to spike patterns using the population-coding approach [5]. In this approach, each real-valued feature (v) is encoded into spikes using R Gaussian receptive field neurons. Each receptive field neuron emits a single spike in the interval $[0, T_e]$. If t_r denotes the spike emitted via the receptive field ϕ_r for v , then

$$t_r = T_e(1 - \phi_r) \quad (13)$$

where ϕ_r represents the output of the r^{th} Gaussian receptive field neuron. The upper limit of the encoding interval T_e used for population coding is set to 3 ms. If μ_{pop}^r and σ_{pop} are the center and spread of the receptive field ϕ_r , then

$$\phi_r = \exp\left(-\frac{(v - \mu_{pop}^r)^2}{2\sigma_{pop}^2}\right) \quad (14)$$

$$\mu_{pop}^r = \frac{2r - 3}{2(R - 2)} \quad (15)$$

$$\sigma_{pop} = \frac{1}{0.7(R - 2)} \quad (16)$$

The values for μ_{pop}^r and σ_{pop} are set based on the method presented in [5].

3.1. Discussion on IpT-SNC Parameters

To obtain the best performance of IpT-SNC, a suitable search range for parameters that are optimized using SRPSO should be selected. The range $[0, T]$ ms is used as the search interval for the centers of Gaussian functions, which is the same as the simulation interval (where T is the limit of the simulation interval). T is set to 4 ms. The search interval for widths of Gaussian functions is set to $[0.01, 0.6]$ ms. The search interval for widths is chosen to ensure that Gaussians have a non-zero output over the entire encoding interval even if the center is set at its maximum permitted value in the search range (i.e., 4 ms). The threshold (θ) of each neuron is another parameter that is optimized using SRPSO. The search range for threshold (θ) is set to $[0.1, 2]$. From Equation (3), as only the first spikes of the postsynaptic spiking neurons are used for decision-making, the neuron time constant τ is set to T_e .

Table 1 shows the search for the parameters such as the magnitude of Gaussian amplitude (A_R) (where $[-A_R, +A_R]$ is the search range for amplitudes of Gaussians in each synapse), the number of Gaussians per synapse (K), and the swarm size (P) that have a significant effect on the performance of IpT-SNC. Figure 2 shows the impact of variations in these parameters on the performance of IpT-SNC in the IRIS classification problem (a description of IRIS presented in Table 2). As shown in Figure 2, the training and testing performance accuracies of IpT-SNC on IRIS are used for this study. Next, the impact of variation in each parameter on IpT-SNC performance is discussed.

- *Number of Gaussians per synapse (K):* A high value for the number of Gaussians per synapse (K) results in a large number of parameters that need to be updated using SRPSO. This, in turn, affects the complexity of the model. From Figure 2a, it

can be observed that the testing performance of IpT-SNC increases from $K = 3$ and is the maximum for $K = 6$. For values of K greater than 8, it can be observed that the training performance of IpT-SNC starts decreasing. Due to the trade-off between performance and computational complexity, the suitable values for K are chosen in the interval $[5, 8]$, as shown in Table 1.

- *Magnitude of Gaussian amplitude (A_R)*: From Figure 2b, it can be observed that the impact of variation in the magnitude of Gaussian amplitude (A_R) is high on the performance of IpT-SNC. However, the testing performance of IpT-SNC reaches its peak multiple times when A_R is in the range of (1–3). Therefore, the satisfactory values for A_R are chosen in the interval $[1, 3]$.
- *Swarm size (P)*: The testing performance of IpT-SNC in Figure 2c is higher when the value of swarm size (P) increases beyond 90. Hence, for the rest of the classification problems, the suitable initialization interval for swarm size (P) is set to $[90, 100]$.

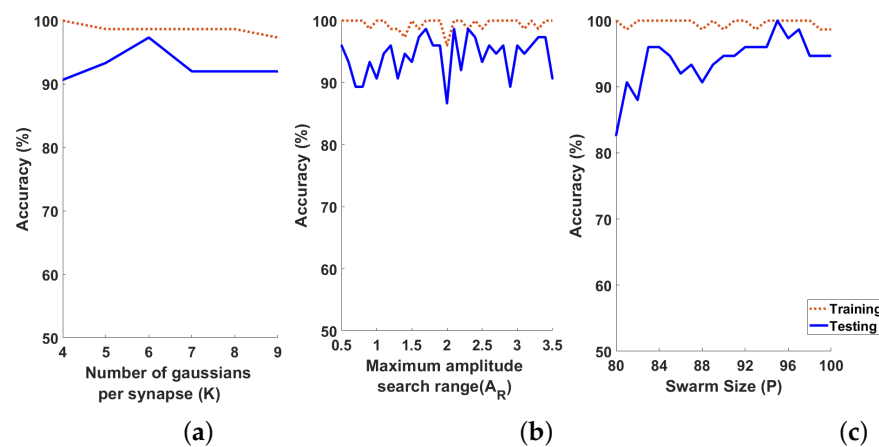


Figure 2. The variation in the training and testing classification accuracies of IpT-SNC with respect to variations in (a) the number of Gaussians per synapse- K , (b) the magnitude of Gaussian amplitude (A_R) (where $[-A_R, +A_R]$ is the search range for amplitudes of Gaussians in each synapse), and (c) swarm size- P .

Table 1. Hyperparameters.

Parameter	Value
K	$[5, 8]$
A_R	$[1, 3]$
P	$[90, 100]$
R	6
T_e	3 ms
T	4 ms

Table 2. UCI repository datasets used for performance evaluation.

Datasets	# Features	# Classes	# Training Samples	# Testing Samples
Liver	6	2	170	175
ECG	10	2	66	65
MMG	9	2	80	11
HEPATITIS	19	2	78	77
PIMA	9	2	384	384
BC	9	2	350	333
ION	34	2	175	176
Iris	4	3	75	75
WINE	13	3	60	118
AE	5	4	62	137

3.2. Computational Complexity

The computational complexity of IpT-SNC is estimated separately for feedforward propagation through the network and optimization using SRPSO. In the feedforward stage, the computations are carried out until the IpT-SNC predicts a certain class for a given input sample (Equation (3)). The computational complexity for the feedforward stage in O-notation is given by

$$\kappa_1 = O(m \cdot C^2 \cdot K) \quad (17)$$

If P and D denote the swarm size and dimension of each particle, then the computational complexity of the SRPSO is computed as mentioned in [30], which is given by

$$\kappa_2 = O(P \cdot D) \quad (18)$$

3.3. Interpretability Demonstration of IpT-SNC Using a Synthetic Classification Problem

To study interpretability, IpT-SNC is used to classify a two-class synthetic dataset. Each sample in the synthetic dataset consists of two features, X and Y , as shown below

$$class - 1 = \begin{cases} 0 \leq X \leq 0.4 \\ 0 \leq Y \leq 0.4 \end{cases} \quad class - 2 = \begin{cases} 0.6 \leq X \leq 1 \\ 0.6 \leq Y \leq 1 \end{cases} \quad (19)$$

If the value of both features is lower than 0.4, then the sample belongs to class 1. In class 2 samples, both features are incorporated at greater than 0.6. Each class contains 50 randomly generated samples, which are divided into training and testing sets. The two classes in the dataset are represented by two neurons in the output layer. Each synapse is modeled using three Gaussian functions. SRPSO is used to update the parameters associated with all Gaussian functions in the network, as described in Section 2.2.

The relation between the presynaptic spike times for a randomly chosen sample from each class and the Gaussian functions present in synapses of the IpT-SNC architecture is depicted in Figure 3. All synapses connected to the output neuron associated with class 1 are shown in the top two rows of Figure 3 (i.e., from Figure 3a–h). Similarly, synapses connected to the output neuron associated with class 2 are shown in the bottom two rows of Figure 3 (i.e., from Figure 3i–p). A random class 1 sample is selected, and the response of all Gaussian functions in the network for the spikes generated via this sample is represented as black dots in Figure 3. Similarly, black diamonds represent the same for spikes generated via a random class 2 input sample. Three lines (dashed red, solid blue, and dotted orange) in each figure represent the Gaussian functions associated with that synapse over the duration of the simulation. The parameters for these Gaussian functions were estimated using SRPSO. From Figure 3a, it can be observed that the sum of the outputs of Gaussian functions is higher at the time of class 1 presynaptic spikes when compared with class 2 presynaptic spikes. Similarly, it can be reported from Figure 3i that the sum of the outputs of Gaussian functions is higher for class 2 presynaptic spikes when compared with class 1 presynaptic spikes. This property enables the correct class output neuron in IpT-SNC to fire first.

Figure 4 shows the weights of the synapses whose Gaussians are represented in Figure 3. These synapses transmit spikes generated via the first presynaptic neuron to the class 1 and class 2 postsynaptic neurons, respectively. It can be observed that the amplitudes of weights connected to the class 1 output neuron are larger before 1.1 ms. This indicates that input spikes before 1.1 ms strongly determine the response of the output neurons associated with class 1. Similarly, input spikes after 1.4 ms strongly determine the response of the output neuron that belongs to class 2. This kind of similar interpretation can be drawn from the weights of other synapses in the IpT-SNC network.

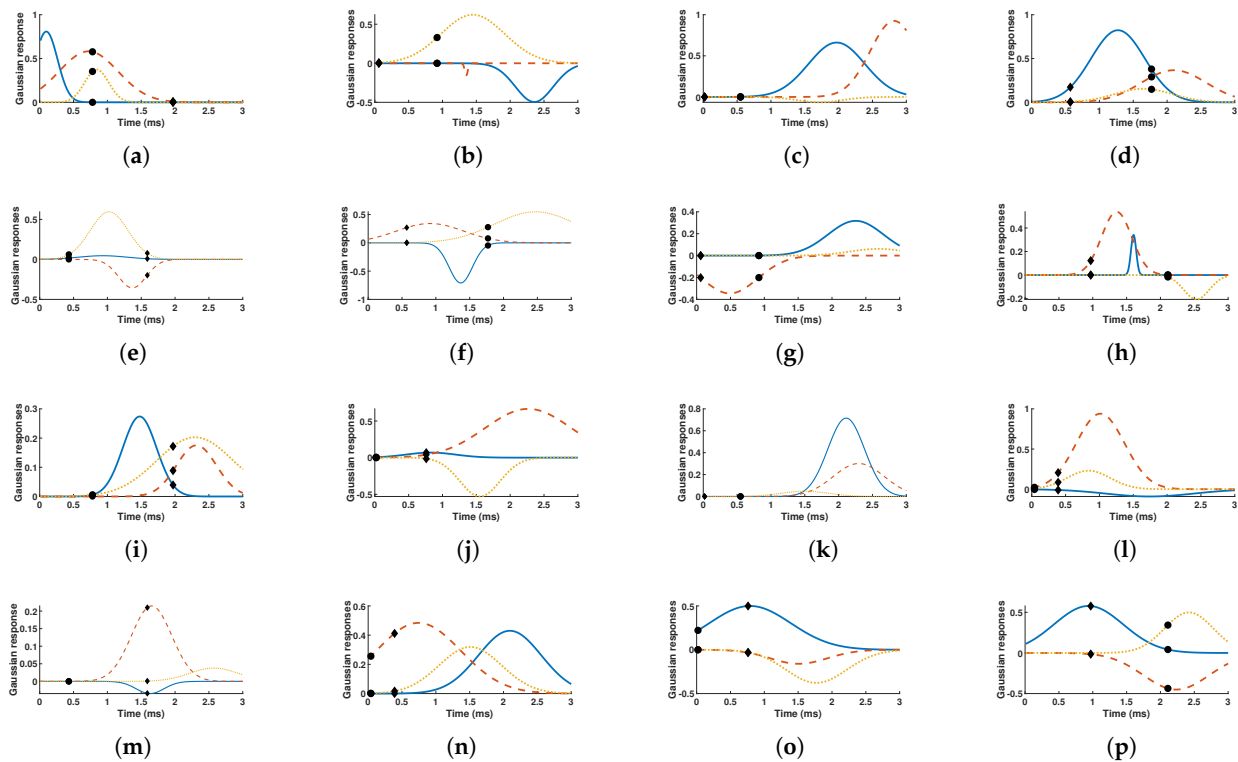


Figure 3. Gaussian functions present in synapses connecting input and output neurons. The first two rows of the figure show the gaussians present in synapses connected to output neuron 1 and the next two rows show the gaussians present in synapses connected to output neuron 2. The dot represents the response of Gaussian functions for spikes generated via a class 1 input sample, and the diamond represents the same for the class 2 input sample. Gaussian functions in synapses connecting (a) input neuron 1 to output neuron 1, (b) input neuron 2 to output neuron 1, (c) input neuron 3 to output neuron 1, (d) input neuron 4 to output neuron 1, (e) input neuron 5 to output neuron 1, (f) input neuron 6 to output neuron 1, (g) input neuron 7 to output neuron 1, (h) input neuron 8 to output neuron 1, (i) input neuron 1 to output neuron 2, (j) input neuron 2 to output neuron 2, (k) input neuron 3 to output neuron 2, (l) input neuron 4 to output neuron 2, (m) input neuron 5 to output neuron 2, (n) input neuron 6 to output neuron 2, (o) input neuron 7 to output neuron 2, and (p) input neuron 8 to output neuron 2.

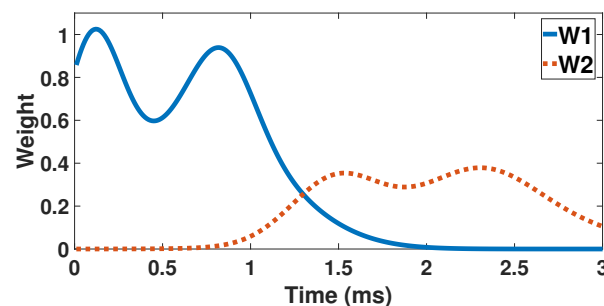


Figure 4. The blue line denotes the synapse's weight connected in between input neuron 1 and output neuron 1. The red line denotes the synapse's weight connected between input neuron 1 and output neuron 2.

The performance of IpT-SNC is evaluated using ten benchmark datasets from the UCI [33] machine learning repository. Table 2 lists the details of these datasets, such as the number of features in each sample of the dataset, the number of classes in a dataset, and the number of training and testing samples in the dataset. Table 3 compares the performance of

the IpT-SNC with those of the Spikeprop, TMM-SNN, DC-SNN, SEFRON, MeST, CD-SNN, and T-SNN spiking neural classifiers for binary-class datasets, whereas Table 4 shows the performance comparison for multi-class datasets. The performance accuracies mentioned in Tables 3 and 4 are the mean values obtained across 10 random trials, as in [21]. In each trial, the whole dataset is randomly split into training and testing sets. Among the learning algorithms in Table 3, Spikeprop, TMM-SNN, and DC-SNN utilize constant weights, whereas SEFRON and MeST employ time-varying weights. CD-SNN and T-SNN incorporate fuzzy logic into the SNNs. Tables 3 and 4 display $N_i : N_h : N_j$ as the architecture of an SNN employed with the learning algorithms. The total number of input neurons, hidden neurons, and output neurons are denoted, respectively, as N_i , N_h , and N_j . For two-layer networks, the architecture is presented as $N_i : N_j$. IpT-SNC, SEFRON, and MeST utilize single-layer networks, while other algorithms employ multi-layer networks.

Table 3. Performance comparison of IpT-SNC on binary-class classification datasets with SpikeProp, TMM-SNN, DC-SNN, SEFRON, MeST, CD-SNN, and T-SNN.

Dataset	Learning Algorithm	Architecture	Training Accuracy (%)	Testing Accuracy (%)	# Epochs
Liver	SpikeProp	37-15-2	71.5(5.2)	65.1(4.7)	3000
	TMM-SNN	36-(5-8)-2	74.2(3.5)	70.4(2.0)	442
	DC-SNN	36-8-2	68.8(4.4)	70.0(2.0)	1000
	SEFRON	37-1	91.5(5.4)	67.7(1.3)	100
	MeST	36-2	74.8(3.9)	71.1(3.2)	100
	CD-SNN	37-10-2	73.9(1.4)	72.9(1.7)	74
	T-SNN	36-12-2	79.1(2.1)	73.0(1.2)	35
	IpT-SNC	36-2	85.4(0.6)	80.7(2.8)	500
ECG	SpikeProp	61-10-2	86.6(2.5)	84.5(3.0)	1000
	TMM-SNN	60-(2-3)-2	86.5(2.1)	85.4(1.7)	177
	DC-SNN	60-8-2	88.2(4.1)	86.5(2.1)	1000
	SEFRON	61-1	88.6(3.9)	86.5(3.3)	100
	MeST	60-2	86.5(3.2)	85.5(1.5)	100
	CD-SNN	61-8-2	87.0(2.0)	87.0 (2.2)	45
	T-SNN	60-8-2	81.5(2.8)	78.5(7.2)	81
	IpT-SNC	60-2	92.8(0.7)	92.7(2.3)	500
MMG	SpikeProp	55-10-2	82.8(4.7)	81.8(6.1)	1000
	TMM-SNN	54-(5-7)-2	87.2(4.4)	84.9(8.6)	176
	DC-SNN	54-8-2	80.4(6.2)	94.5(8.3)	1000
	SEFRON	55-1	92.8(5.0)	82.7(10.0)	100
	MeST	54-2	98.2(0.8)	98.0(0.4)	100
	CD-SNN	55-6-2	85.3 (7.2)	84.4 (9.1)	53
	T-SNN	54-8-2	85.7(2.1)	85.6(6.7)	57
	IpT-SNC	54-2	93.7(0.7)	95.6(4.2)	500
HEPATITIS	SpikeProp	115-15-2	87.8(5.0)	83.5(2.5)	1000
	TMM-SNN	114-(3-9)-2	91.2(2.5)	86.6(2.2)	192
	DC-SNN	114-8-2	87.7(3.0)	88.3(1.4)	1000
	SEFRON	115-1	94.6(3.5)	82.7(3.3)	100
	MeST	114-2	90.9(5.7)	88.7(2.0)	100
	CD-SNN	115-8-2	95.0(3.2)	95.0(2.6)	69
	T-SNN	114-8-2	88.5(1.7)	86.4(4.2)	75
	IpT-SNC	114-2	95.3(0.6)	94.1(1.6)	500

Table 3. Cont.

Dataset	Learning Algorithm	Architecture	Training Accuracy (%)	Testing Accuracy (%)	# Epochs
PIMA	SpikeProp	55-20-2	78.6(2.5)	76.2(1.8)	3000
	TMM-SNN	54-(5-14)-2	79.7(2.3)	78.1(1.7)	160
	DC-SNN	49-12-2	78.6(1.9)	77.8(1.2)	1000
	SEFRON	49-1	84.1(1.5)	74.0(1.2)	100
	MeST	48-2	78.4(2.5)	77.3(1.3)	100
	CD-SNN	49-13-2	79.4(1.2)	78.4(2.4)	48
	T-SNN	24-18-2	79.6(1.0)	79.0(0.8)	48
	IpT-SNC	48-2	82.5(0.6)	82.5(1.2)	500
BC	SpikeProp	55-15-2	97.3(0.6)	97.2(0.6)	1000
	TMM-SNN	54-(2-8)-2	97.4(0.3)	97.2(0.5)	70
	DC-SNN	54-8-2	97.4(0.7)	97.8(0.5)	1000
	SEFRON	55-1	98.3(0.8)	96.4(0.7)	100
	MeST	54-2	98.2(0.8)	98.0(0.4)	100
	CD-SNN	55-8-2	97.8(1.0)	97.3(1.1)	55
	T-SNN	24-15-2	97.8(0.3)	97.4(0.2)	36
	IpT-SNC	54-2	98.4(0.1)	98.8(1.5)	500
ION	SpikeProp	205-25-2	89.0(7.9)	86.5(7.2)	3000
	TMM-SNN	204-(23-34)-2	98.7(0.4)	92.4(1.8)	246
	DC-SNN	199-20-2	97.1(1.2)	92.7(1.5)	1000
	SEFRON	199-1	97.0(2.5)	88.9(1.7)	100
	MeST	198-2	98.3(2.6)	93.2(1.8)	100
	CD-SNN	205-12-2	93.0(2.6)	91.2(2.1)	38
	T-SNN	204-15-2	97.8(1.2)	94.5(0.7)	35
	IpT-SNC	198-2	99.2(0.4)	95.4(1.1)	500

Table 4. Performance comparison of IpT-SNC on multi-class classification datasets with SpikeProp, TMM-SNN, DC-SNN, SEFRON, MeST, CD-SNN, and T-SNN.

Dataset	Learning Algorithm	Architecture	Training Accuracy (%)	Testing Accuracy (%)	# Epochs
IRIS	SpikeProp	25-10-3	97.2(1.9)	96.7(1.6)	1000
	TMM-SNN	24-(4-7)-3	97.5(0.8)	97.2(1)	246
	DC-SNN	24-11-3	96.1(2.9)	97.7(1.4)	1000
	MeST	24-3	98.1(1.5)	98.0(1.4)	100
	CD-SNN	25-8-3	98.3(0.3)	97.9(1.7)	51
	T-SNN	24-10-3	97.5(0.5)	97.3(0.4)	48
	IpT-SNC	24-3	98.7(0)	98.9(0.8)	500
WINE	SpikeProp	79-10-3	99.2(1.2)	96.8(1.6)	1000
	TMM-SNN	78-3-3	100(0)	97.5(0.8)	80
	DC-SNN	78-10-3	98.2(2.0)	96.3(1.0)	1000
	MeST	78-3	100(0)	98.0(1.2)	100
	CD-SNN	79-8-3	100(0)	98.8(1.1)	33
	T-SNN	52-8-3	99.8(0.2)	98.9(0.4)	38
	IpT-SNC	78-3	100(0)	98.8(0.8)	500
AE	SpikeProp	31-10-4	98.5(1.7)	97.2(3.5)	1000
	TMM-SNN	30-(4-7)-4	97.6(1.3)	97.5(0.7)	12
	DC-SNN	30-20-4	96.6(2.2)	96.5(1.5)	1000
	MeST	30-4	99.2(0.9)	99.1(0.3)	100
	CD-SNN	31-8-4	99.4(0.2)	99.0(1.0)	58
	T-SNN	25-10-4	99.4(0.1)	99.1(0.2)	35
	IpT-SNC	30-4	98.7(0.6)	98.9(0.4)	500

3.4. Performance Comparison of IpT-SNC on UCI Repository Datasets

Liver and PIMA are low-dimensional, high-interclass-overlap binary classification tasks. The performance of IpT-SNC on a training dataset of the liver is better than that of the other learning algorithms except for SEFRON. The performance of IpT-SNC on a testing dataset of the liver is 7.7–19% higher than that of all the other algorithms. For the PIMA dataset, SEFRON has the highest training accuracy, which is 2% higher than that of IpT-SNC. But IpT-SNC exhibits 3.5–10% higher performance on a testing dataset than all the other algorithms used for comparison. A mammogram (MMG) is another problem similar to that of the liver and PIMA. The performance of IpT-SNC on the testing dataset of MMG is better than the 1–14% of the other learning algorithms except for MeST. In this case, MeST outperforms IpT-SNC by 2.5%.

Echocardiogram (ECG) and breast cancer (BC) are simple binary classification problems mentioned in Table 3. All learning algorithms perform almost similarly on the training dataset of ECG, while, on the testing dataset of ECG, IpT-SNC has the highest performance accuracy compared to that of the other algorithms. An increment of approximately 1% can be seen in the performance of IpT-SNC on the testing dataset of BC. The multi-class classification problems mentioned in Table 4 are IRIS, wine, and acoustic emission (AE). IRIS and wine are three-class problems, while AE is a four-class problem. From Table 4, it is observed that almost all algorithms perform similarly on these simple multi-class classification datasets.

For high-dimensional problems like ionosphere (ION) and hepatitis, IpT-SNC achieves the best performance on training and testing datasets of ionosphere (ION). IpT-SNC exhibits an improvement of 0.5–13% on the training dataset and 0.9–9% on the testing dataset of ION, respectively, when compared with the other learning algorithms, while an increment of 1–8% and 6–11% is observed in the performance of IpT-SNC on the training and testing datasets of hepatitis, respectively, when compared with other learning algorithms except CD-SNN. CD-SNN performs 0.9% better than IpT-SNC in the case of a testing dataset of hepatitis. These findings indicate that IpT-SNC exhibits a significant performance increment on complex problems with high interclass overlap and high dimensional problems when compared with other learning algorithms, while there is a moderate increment in the performance of IpT-SNC on simple classification problems when compared with other learning algorithms.

Statistical analysis of the performance comparison: Statistical tests such as the Friedmann test and paired *t*-tests are used to evaluate the significance of the performance of IpT-SNC when compared with that of SpikeProp, TMM-SNN, DC-SNN, SEFRON, MeST, CD-SNN, and T-SNN. The testing accuracies of the algorithms are used for this statistical analysis. At first, with the help of the Friedmann test, the significance of all the algorithms as a group is evaluated. That the testing accuracies of all the algorithms used for comparison do not differ significantly from one another is the null hypothesis considered for the Friedmann test. Using the Friedmann test, a *p*-value of 0.000009 was achieved, which indicates a 95% confidence level in rejecting the null hypothesis. This suggests that at least one algorithm operates differently from the others in terms of performance. The pair-wise significance between IpT-SNC and each learning algorithm in Table 3 is evaluated using a paired *t*-test. The null hypothesis of the two algorithms does not differ significantly, and it is considered for the paired *t*-tests. The *p*-values of 0.0017, 0.0018, 0.007, 0.0009, 0.0613, 0.0165, and 0.0153 are obtained using the paired *t*-test when IpT-SNC is compared against SpikeProp, TMM-SNN, DC-SNN, SEFRON, MeST, CD-SNN, and T-SNN respectively. From these *p*-values, it can be inferred that the null hypothesis can be rejected with a 95% degree of confidence when IpT-SNC is compared with SpikeProp, TMM-SNN, DC-SNN, SEFRON, CD-SNN, and T-SNN. The *p*-value of 0.0613 obtained when IpT-SNC is compared with MeST indicates that the null hypothesis is rejected with a 90% confidence level.

3.5. Performance Comparison of IpT-SNC on BCI Competition IV Dataset

In this section, the performance of IpT-SNC in classifying right-hand versus left-hand movements from the Brain-Computer Interface (BCI) competition IV dataset is discussed [34]. The BCI competition IV dataset is a real-world dataset obtained from the recordings of electroencephalograph (EEG) signals from the brain. These EEG recordings are obtained from various participants while they are imagining movements of body parts without actually moving them. The BCI competition IV dataset has several such recordings, which are termed motor-imagery tasks. The 2a dataset in BCI-IV consists of data when participants are imagining the movement of the right hand, the left hand, both feet, and the tongue. These recordings are obtained using 22 electrodes placed on different regions of the scalp of the participant. A total of nine participants are considered, and recordings are carried out in two different sessions, each with 144 trials. In this paper, the problem of right-hand versus left-hand movement classification is considered.

The problem of hand movement classification for each participant is evaluated separately using IpT-SNC. Out of the total 144 trials 72 trials from each session are considered for training, and the remaining 72 are considered for testing. For each, six features are extracted using the Robust Common Spatial Pattern technique [35]. The testing performance of IpT-SNC on this motor imagery task is compared with the Common Spatial Pattern-Linear Discriminant Analysis (CSP-LDA) [36] classifier, the Self Regulated Interval Type-2 Neuro-Fuzzy Inference System (SRIT2NFIS) [37], and DC-SNN. Table 5 presents this comparison results. The nine participants in this task are indicated as A1–A9 subjects in Table 5. The mean and deviation indicated in the comparison clearly show that IpT-SNC outperforms the other learning algorithms by a margin of 8–13%. The lower standard deviation indicates that IpT-SNC has better generalization performance compared to the other algorithms.

Table 5. Performance comparison of IpT-SNC with CSP-LDA, SRIT2NFIS, and DC-SNN on BCI competition IV 2a dataset.

Subjects	CSP-LDA	SRIT2NFIS	DC-SNN	IpT-SNC
A1	88.89	93.06	92.36	97.22
A2	51.39	68.75	66.67	79.86
A3	96.53	97.22	96.53	100
A4	70.14	75.00	77.78	86.81
A5	54.86	65.97	63.89	78.47
A6	71.53	72.22	73.61	86.80
A7	81.25	86.11	82.64	93.75
A8	93.75	97.22	97.92	100
A9	93.75	93.75	93.75	97.92
Mean	78.01	83.26	82.79	91.2
Standard deviation	17.01	12.76	12.27	8.45

4. Conclusions

This paper has presented an IpT-SNC for classification problems and a search-based method for IpT-SNC parameter updation. The weight of each synapse in the IpT-SNC was modeled using multiple amplitude-modulated Gaussian functions with centers located at various points throughout the simulation interval. The time-varying weight modeling of a synapse is used in the IpT-SNC architecture to exploit the advantages of its dynamic nature. A search-based self-regulated particle swarm optimization (SRPSO) technique is used to update the values of the parameters in the IpT-SNC in order to achieve better performance. In the interpretability study, it was shown that the rationale behind the predictions was described with the help of time-varying weights in IpT-SNC. The performance of IpT-SNC was compared with that of other learning algorithms with the help of ten benchmark datasets from the UCI machine learning repository. The IpT-SNC outperformed the other

existing classifiers by a maximum of 1–7.7% while classifying highly overlapped datasets, whereas, for datasets with higher dimensions, IpT-SNC outperformed other classifiers by 0.9–6%. The statistical analysis of performance comparison showed that IpT-SNC performed better in terms of generalization while still providing the benefit of interpretable outcomes. Further, the performance of IpT-SNC was evaluated in real-world motor imagery classification tasks. An average improvement of 8% in the performance of IpT-SNC was seen in the performance of right-hand versus left-hand movement classification when compared with the other algorithms.

Author Contributions: Conceptualization, M.T.; Methodology, M.T.; Software, M.T.; Validation, M.T.; Writing—Original draft, M.T.; Writing—review & editing, S.D. and S.S.; Supervision, S.D. and S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The datasets used in the study are openly available in [UCI repository at Available online: <https://archive.ics.uci.edu/> (accessed on 1 August 2024)] and [BCI competition IV dataset at Available online: <https://www.bbc.de/competition/iv/> (accessed on 1 August 2024)].

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Maass, W. Noisy Spiking Neurons with Temporal Coding have more Computational Power than Sigmoidal Neurons. *Adv. Neural Inf. Process. Syst.* **1996**, *9*, 211–217.
2. Machingal, P.; Thousif, M.; Dora, S.; Sundaram, S. Self-regulated Learning Algorithm for Distributed Coding Based Spiking Neural Classifier. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–7. [\[CrossRef\]](#)
3. Kheradpisheh, S.R.; Ganjtabesh, M.; Thorpe, S.J.; Masquelier, T. STDP-based spiking deep convolutional neural networks for object recognition. *Neural Netw.* **2018**, *99*, 56–67. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Kim, S.; Park, S.; Na, B.; Yoon, S. Spiking-yolo: Spiking neural network for energy-efficient object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 4, pp. 11270–11277.
5. Bohte, S.M.; Kok, J.N.; La Poutré, H. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing* **2002**, *48*, 17–37. [\[CrossRef\]](#)
6. Perez-Nieves, N.; Goodman, D. Sparse spiking gradient descent. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 11795–11808.
7. Neftci, E.O.; Mostafa, H.; Zenke, F. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Process. Mag.* **2019**, *36*, 51–63. [\[CrossRef\]](#)
8. Markram, H.; Lübke, J.; Frotscher, M.; Sakmann, B. Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science* **1997**, *275*, 213–215. [\[CrossRef\]](#)
9. Kasabov, N.; Capecci, E. Spiking neural network methodology for modelling, classification and understanding of EEG spatio-temporal data measuring cognitive processes. *Inf. Sci.* **2015**, *294*, 565–575. [\[CrossRef\]](#)
10. Dora, S.; Suresh, S.; Sundararajan, N. Online Meta-neuron based Learning Algorithm for a spiking neural classifier. *Inf. Sci.* **2017**, *414*, 19–32. [\[CrossRef\]](#)
11. Yang, Y.; Calakos, N. Presynaptic long-term plasticity. *Front. Synaptic Neurosci.* **2013**, *5*, 8. [\[CrossRef\]](#)
12. Stevens, C.F.; Wang, Y. Facilitation and depression at single central synapses. *Neuron* **1995**, *14*, 795–802. [\[CrossRef\]](#)
13. Abbott, L.F.; Regehr, W.G. Synaptic computation. *Nature* **2004**, *431*, 796–803. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Liaw, J.S.; Berger, T.W. Dynamic synapse: A new concept of neural representation and computation. *Hippocampus* **1996**, *6*, 591–600. [\[CrossRef\]](#)
15. Liaw, J.S.; Berger, T. Robust speech recognition with dynamic synapses. In Proceedings of the 1998 IEEE International Joint Conference on Neural Networks Proceedings, Anchorage, AK, USA, 4–9 May 1998; IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227); Volume 3, pp. 2175–2179. [\[CrossRef\]](#)
16. Namarvar, H.; Liaw, J.S.; Berger, T. A new dynamic synapse neural network for speech recognition. In Proceedings of the IJCNN'01, International Joint Conference on Neural Networks, Washington, DC, USA, 15–19 July 2001; Proceedings (Cat. No. 01CH37222); Volume 4, pp. 2985–2990. [\[CrossRef\]](#)
17. Wade, J.J.; McDaid, L.J.; Santos, J.A.; Sayers, H.M. SWAT: A Spiking Neural Network Training Algorithm for Classification Problems. *IEEE Trans. Neural Netw.* **2010**, *21*, 1817–1830. [\[CrossRef\]](#) [\[PubMed\]](#)
18. Hu, M.; Chen, Y.; Yang, J.J.; Wang, Y.; Li, H.H. A Compact Memristor-Based Dynamic Synapse for Spiking Neural Networks. *IEEE Trans.-Comput.-Aided Des. Integr. Circuits Syst.* **2017**, *36*, 1353–1366. [\[CrossRef\]](#)
19. Jeyasothy, A.; Sundaram, S.; Sundararajan, N. SEFRON: A new spiking neuron model with time-varying synaptic efficacy function for pattern classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *30*, 1231–1240. [\[CrossRef\]](#)

20. Jeyasothy, A.; Ramasamy, S.; Sundaram, S. Meta-neuron learning based spiking neural classifier with time-varying weight model for credit scoring problem. *Expert Syst. Appl.* **2021**, *178*, 114985. [CrossRef]
21. Jeyasothy, A.; Suresh, S.; Ramasamy, S.; Sundararajan, N. Development of a Novel Transformation of Spiking Neural Classifier to an Interpretable Classifier. *IEEE Trans. Cybern.* **2024**, *54*, 3–12. [CrossRef]
22. Dora, S.; Sundaram, S.; Sundararajan, N. An Interclass Margin Maximization Learning Algorithm for Evolving Spiking Neural Network. *IEEE Trans. Cybern.* **2019**, *49*, 989–999. [CrossRef]
23. Liu, F.; Yang, J.; Pedrycz, W.; Wu, W. A new fuzzy spiking neural network based on neuronal contribution degree. *IEEE Trans. Fuzzy Syst.* **2021**, *30*, 2665–2677. [CrossRef]
24. Liu, F.; Pedrycz, W.; Zhang, C.; Yang, J.; Wu, W. Coding method based on fuzzy C-Means clustering for spiking neural network with triangular spike response function. *IEEE Trans. Fuzzy Syst.* **2023**, *31*, 1–14. [CrossRef]
25. Hamada, M.; Hassan, M. Artificial neural networks and particle swarm optimization algorithms for preference prediction in multi-criteria recommender systems. *Informatics* **2018**, *5*, 25. [CrossRef]
26. Narayanan, S.L.; Kasiselvanathan, M.; Gurumoorthy, K.; Kiruthika, V. Particle swarm optimization based artificial neural network (PSO-ANN) model for effective k-barrier count intrusion detection system in WSN. *Meas. Sen.* **2023**, *29*, 100875. [CrossRef]
27. Hamed, H.N.A.; Kasabov, N.; Shamsuddin, S.M. Integrated feature selection and parameter optimization for evolving spiking neural networks using quantum inspired particle swarm optimization. In Proceedings of the 2009 International Conference of Soft Computing and Pattern Recognition, Malacca, Malaysia, 4–7 December 2009; pp. 695–698.
28. Hamed, H.N.A.; Kasabov, N.; Shamsuddin, S. Probabilistic evolving spiking neural network optimization using dynamic quantum-inspired particle swarm optimization. *Aust. J. Intell. Inf. Process. Syst.* **2010**, *11*, 23–28.
29. Hong, S.; Ning, L.; Xiaoping, L.; Qian, W. A cooperative method for supervised learning in spiking neural networks. In Proceedings of the the 14th International Conference on Computer Supported Cooperative Work in Design, Shanghai, China, 14–16 April 2010; pp. 22–26.
30. Tanweer, M.; Suresh, S.; Sundararajan, N. Self Regulating Particle Swarm Optimization Algorithm. *Inf. Sci.* **2015**, *294*, 182–202. [CrossRef]
31. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
32. Tanweer, M.R.; Auditya, R.; Suresh, S.; Sundararajan, N.; Srikanth, N. Directionally driven self-regulating particle swarm optimization algorithm. *Swarm Evol. Comput.* **2016**, *28*, 98–116. [CrossRef]
33. Dua, D.; Graff, C. UCI Machine Learning Repository. 2017. Available online: <https://archive.ics.uci.edu> (accessed on 1 August 2024).
34. Naeem, M.; Brunner, C.; Leeb, R.; Graimann, B.; Pfurtscheller, G. Seperability of four-class motor imagery data using independent components analysis. *J. Neural Eng.* **2006**, *3*, 208. [CrossRef]
35. Wang, J.; Feng, Z.; Lu, N. Feature extraction by common spatial pattern in frequency domain for motor imagery tasks classification. In Proceedings of the 29th Chinese Control and Decision Conference (CCDC), Chongqing, China, 28–30 May 2017; pp. 5883–5888. [CrossRef]
36. Wu, S.L.; Wu, C.W.; Pal, N.R.; Chen, C.Y.; Chen, S.A.; Lin, C.T. Common spatial pattern and linear discriminant analysis for motor imagery classification. In Proceedings of the 2013 IEEE Symposium on Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), Singapore, 16–19 April 2013; pp. 146–151. [CrossRef]
37. Das, A.K.; Sundaram, S.; Sundararajan, N. A self-regulated interval type-2 neuro-fuzzy inference system for handling nonstationarities in EEG signals for BCI. *IEEE Trans. Fuzzy Syst.* **2016**, *24*, 1565–1577. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.