International Journal of Neural Systems, Vol. 0, No. 0 (2005) 1–13

A Federated Learning Protocol for Spiking Neural Membrane Systems

Mihail-Iulian Pleșa
Department of Computer Science, University of Bucharest,
Bucharest, Romania
E-mail: mihail-iulian.plesa@s.unibuc.ro

Marian Gheorghe
School of Electrical Engineering and Computer Science, University of Bradford,
Bradford, UK
E-mail: M.Gheorghe@bradford.ac.uk

Florentin Ipate
Department of Computer Science, University of Bucharest,
Bucharest, Romania
E-mail: florentin.ipate@fmi.unibuc.ro

Gexiang Zhang*
School of Automation, Chengdu University of Information Technology,
Chengdu 610225, China
E-mail: zhgxdylan@126.com

Although deep learning models have shown promising results in solving problems related to image recognition or natural language processing, they do not match how the biological brain works. Some of the differences include the amount of energy consumed, the way neurons communicate, or the way they learn. To close the gap between artificial neural networks and biological ones, researchers proposed the spiking neural network. Layered Spiking Neural P systems (LSN P systems) are networks of spiking neurons used to solve various classification problems. In this paper, we study the LSN P systems in the context of a federated learning client-server architecture over horizontally partitioned data. We analyze the privacy implications of pre-trained LSN P systems through membership inference attacks. We also perform experiments to assess the performance of an LSN P system trained in the federated learning setup. Our findings suggest that LSN P systems demonstrate higher accuracy and faster convergence compared to federated algorithms based on either perceptron or spiking neural networks.

Keywords: Spiking Neural P systems and Privacy-preserving and Cryptography and Layered Spiking Neural P systems

1. Introduction

Classical deep learning algorithms have proven to be effective for various use-cases [30; 34; 29; 28; 27; 22; 14]. However, they do not mimic the way the biological brain works, which leads to high energy consumption. This led to the design of spiking neural networks (SNNs for short) that are much closer to biological foundations [9]. One difference between the two types is energy consumption. The human brain consumes about 20W compared to an ordinary computer, which consumes about 175W [43]. Another difference is how artificial neural networks learn (ANN).

*Corresponding author

The standard way for training an ANN is using the backpropagation algorithm [20]. This algorithm involves updating the weights of an ANN-based on an error signal computed from a loss function. The objective is to discover the set of weights that results in the lowest loss. While such error signals are messages on 32 or 64 bits in an ANN, the biological neurons communicate using spikes, i.e., 1 bit of information, and they seem to learn using a rather unsupervised strategy [5]. Also, the neurons from an ANN are modeled by nonlinear functions such as ReLU, sigmoid, or tanh [20]. This is very unlikely to happen in biological neurons [31].

When implemented on conventional hardware, SNNs do not bring an improvement in terms of the amount of energy consumed. Still, on a special type of hardware, called neuromorphic hardware, their energy consumption improves considerably [42; 19; 49]. In this work, we address a special type of SNN called Layered Spiking Neural P systems (LSN P systems for short) [48]. LSN P systems are a sub-class of membrane computing that investigates computational models inspired by the living cell structure and behavior [33]. SN P systems and SNNs were used to provide solutions to many machine-learning-related problems, e.g., classification, image segmentation, or image classification [8; 7; 40; 39; 11; 41; 35; 48; 21].

This paper proposes a federated learning protocol for LSN P systems over horizontally partitioned data. Our contribution includes a privacy analysis and experimental results showing that our method outperforms the current state-of-the-art federated learning for both perceptron and spiking neural networks. The purpose of the protocol is to enable a central party to train an LSN P system on multiple local datasets while preserving their privacy. Since our approach is based on the idea of sharing the weights of a locally trained LSN P model to the server, it becomes vulnerable to membership inference attacks [26]. To this end, we experimentally assess the success of such an attack on a pre-trained LSN P system and suggest one mitigation strategy based on additive homomorphic encryption. We compare the performances of our protocol with other federated learning approaches for both spiking and artificial neural networks.

The rest of the paper is organized as follows: Section 2 presents related work and LSN P systems; The membership inference attack on pre-trained LSN P systems is given in Section 3; Section 4 introduces the federated learning protocol; Section 5 discusses the experiments and comparison with other approaches while Section 6 is left for conclusions.

2. Background

Significant research has been conducted on private federated learning. A privacy-preserving training algorithm allowing multiple parties to train a deep learning model using gradient descent is proposed in Ref. [4]. Each client updates the gradient on local data and then sends the encrypted gradient update to a remote server. The server computes over encrypted data another model, which is equivalent to one trained over all local data. The protocol is based on encryption schemes homomorphic with respect to addition. In Ref. [10], the authors proposed a protocol based on homomorphic encryption that assumes that a remote server already has a trained deep-learning model. The goal of the protocol is to allow clients to use the model without revealing private data. In Ref. [12], the authors used somewhat homomorphic encryption to enable private training and inference for a deep learning model. Secure multi-party computation was used in Ref. [23] to construct a scalable system for privacy-preserving machine learning. Differential privacy represents another technology that can be used to build privacy-preserving machine learning algorithms [16]. Various protocols for privacy-preserving SNNs were also proposed. In Ref. [18], the authors showed how to transform a trained ANN into an SNN without revealing the weights of the original model. Another approach to construct an SNN that recognizes traffic signs based on private federated learning was presented in Ref. [17]. A privacy-preserving algorithm to train an SNN for time series forecasting on health data was proposed in Ref. [24]. The federated learning protocol described in In Ref. [44] is based on gradients aggregation. To ensure robustness, at each round of the training process, a subset of participants is randomly chosen to update the master model. A federated learning system for neuromorphic hardware is proposed in Ref. [38]. Ref. [13] proposes another approach to federated learning for SNNs. Their idea is to add noise to the local model before sharing it, an approach based on differential privacy. Regarding privacy aspects in SN P systems, in Ref. [32], the authors proposed a privacy-preserving protocol for evaluating linear SN P systems.

Layered Spiking Neural P systems (LSN P systems), introduced in Ref. [48], represent a new type of SN P system aiming to solve classification problems. In the original paper, the authors showed through extensive experiments that the system can provide efficient solutions to classification problems using real-world datasets, e.g., MNIST dataset [3]. The LSN P system has three layers: the input layer, the hidden layer, and the output layer. The system input is encoded as a nonlinear mixture of variables approximated by a Taylor polynomial [48]. Each neuron of the LSN P system has associated a fuzzy truth value, a real number in $[0, 1]$. All operations performed over the fuzzy truth value of a neuron are implemented by fuzzy operators described in Ref. [45]. Two types of neurons are in an LSN P system:

(1) The proposition neurons are denoted by $\sigma^h_{p_i}$, where h is the layer and i is the index of the neuron in that layer. When a proposition neuron receives multiple spikes, a boolean OR operator is applied to its inputs.
(2) The rule neurons are denoted by $\sigma^h_{r_j}$, where h is the layer and j is the index of the neuron in that layer. When multiple spikes enter a rule neuron, the addition operator is applied.

The weights of the synapses linking the input and the hidden layer are real numbers from $[0, 1]$ and are initialized randomly. The weights adjust the potential sent by a firing neuron by applying the multiplication operator between the original potential and the weight. During the training process, the weights are updated by the supervised Widrow-Hoff learning law [46]. The LSN P system structure is described in Figure 1.

3. Membership inference attack on LSN P systems

The goal of a membership inference attack is to determine, given a pre-trained model, i.e., a target model, whether a particular sample was part of the training dataset. The fact that the attacker can determine whether certain data were used in a study causes damage to the holder of that data [36; 25]. For example, identifying a person in a medical dataset reveals information about their health condition. We experimentally prove that pre-trained LSN P systems are vulnerable to membership attacks by showing that the model is more confident in predictions made on the training set than in those made over the test set. This type of attack is possible due to the fact that most models behave differently on the training data than on the test data. In this section, we describe a membership inference attack on pre-trained LSN P systems based on the framework introduced in Ref. [37]. Unlike the original approach, which is based on a black-box model, in our federated learning protocol, the server has access to the entire set of weights, so the model is easier to attack. In the security model, we consider the server to be a third-party honest-but-curious, i.e., it follows the protocol but tries to find information about the underlying data.

Following the scenario described in Ref. [37], we suppose that the attacker has access to the data distribution on which the model was trained. Such statistics can be gained by exploiting the difference between the confidence obtained on the training dataset and the one on the test dataset [37]. We denote this distribution by $\mathcal{D}$. Since the attacker knows $\mathcal{D}$, it can employ this to train multiple LSN P systems using data akin to the training data of the target model.

We define two quantities related to the confidence of a pre-trained LSN P system: confidence values and model confidence. The confidence values of an LSN P system are the potential values of the neurons from the hidden layer. Let $\boldsymbol{\alpha^2_r}$ be a vector of these values. It is reasonable to assume that the attacker has access to these values since it has access to the entire set of weights. The model confidence on a single sample is as the softmax function over $\boldsymbol{\alpha^2_r}$. Each neuron on the hidden layer is assigned to a particular class; thus, by applying the softmax function, we gain the probability of each class. The model confidence over a dataset $\mathbf{X}$ is computed as the average of the model confidence values of each sample.

The main idea of the attack is to construct a dataset composed of confidence values and labels that indicate whether the confidence values were obtained from a sample belonging to the training or testing dataset. The attacker takes the steps below. An overview of the attack is depicted in Figure 2.

(1) Initialize a number S of LSN P systems called shadow models.
(2) Initialize an empty dataset D_{attack}.
(3) For each shadow LSN P system execute:

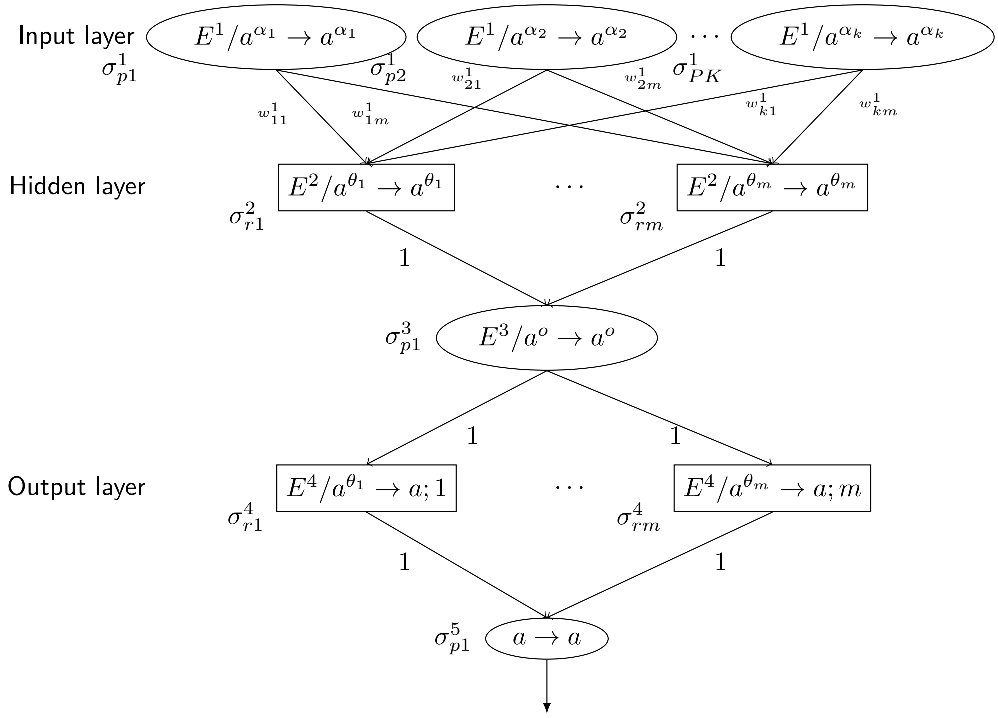

Fig. 1. LSN P system

(a) Given $\mathcal{D}$, generate a training dataset D_{train} and a testing dataset D_{test} such that $D_{train} \cap D_{test} = \emptyset$. Each dataset contains the same number of samples.
(b) Train the LSN P system over D_{train}.
(c) For each sample of D_{train}, compute the confidence values, $\boldsymbol{\alpha_r^2}$. Store the sample $\left(\boldsymbol{\alpha_r^2}, 1\right)$ in the D_{attack}.
(d) Proceed similarly with D_{test}, but assign the label 0 to each vector of confidence values.

(4) Split the dataset D_{attack} into C partitions: D_{attack_y}, $0 \leq y < C$ where C represents the number of possible outputs of the classification algorithm. Each partition D_{attack_y} represents a subset of D_{attack} for which all samples were classified as y by the pre-trained model.
(5) Trains a binary classifier for each partition D_{attack_y} of D_{attack}.
(6) Given an unknown sample $\mathbf{x}$, the attacker first determines its class and $\boldsymbol{\alpha_r^2}$ using the pre-trained LSN P system. To decide whether the sample was part of the original training dataset of the target LSN P system, the attacker classifies the vector of potential values using the binary classifier for that specific class. The attack accuracy for each class is defined as the accuracy of the binary classifier for that particular class.

4. Federated learning protocol

In the federated learning protocol, a central party called the server trains an LSN P system over multiple local datasets owned by the clients. The goal is to train the central model without compromising the privacy of the local datasets. Our protocol is derived from FedMA, an algorithm based on weight averaging instead of gradient averaging [15]. In the case of

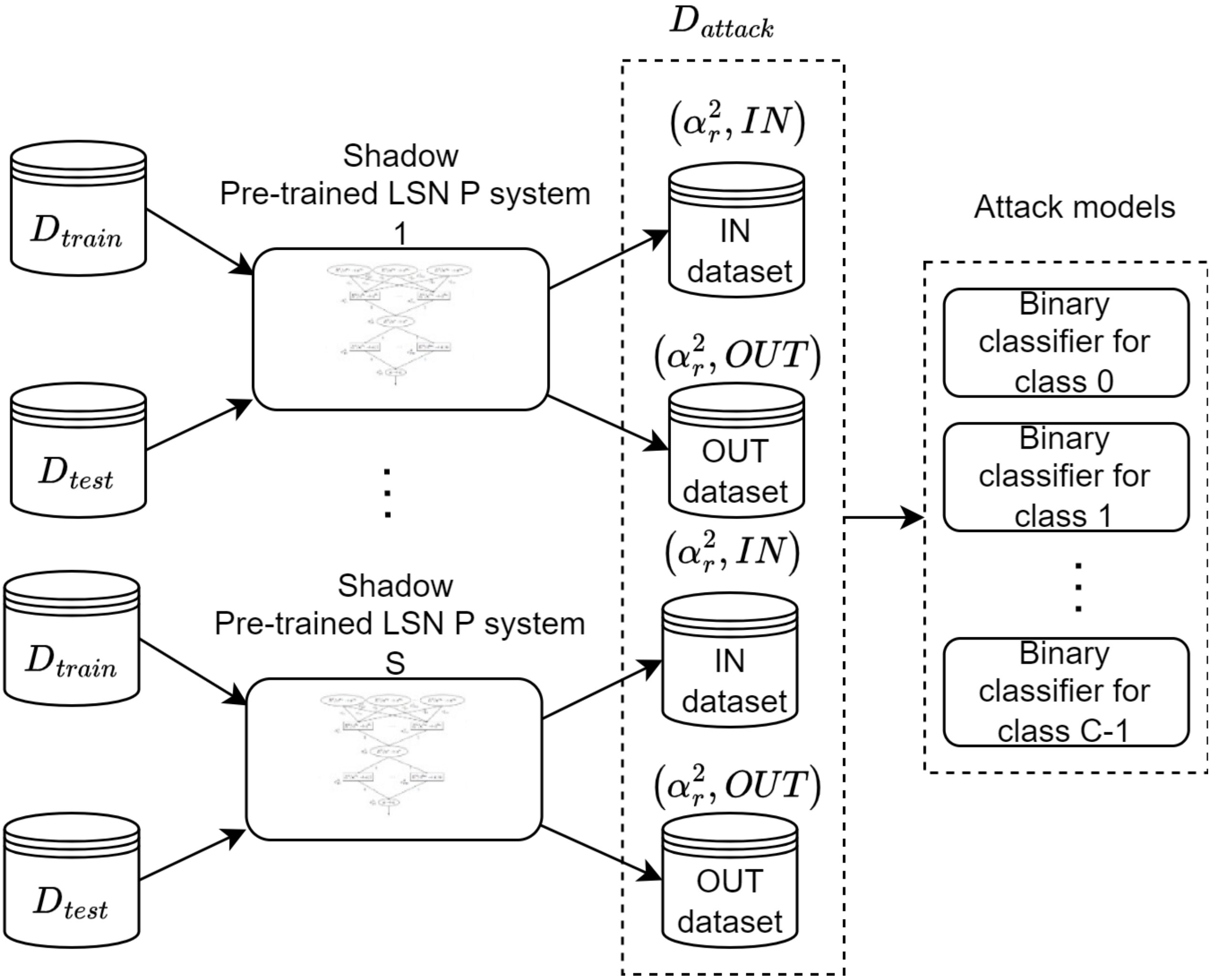

Fig. 2. An overview of the attack strategy

LSN P systems, the weights are between each pair of neurons $\sigma^1_{p_i}$ and $\sigma^2_{r_j}$ with $1 \leq i \leq k$ and $1 \leq j \leq m$.

We make the following notations:

(1) $D_1, D_2, \ldots, D_N$ are the local training datasets. $|D_c|$ represents the number of samples from the local dataset, $1 \leq c \leq N$, where N is the number of clients.
(2) R is the number of rounds of the federated learning protocol.
(3) E - the number of local epochs
(4) $W_1, W_2, \ldots, W_N$ - the weights of the local LSN P systems.
(5) W_S - the weights of the central LSN P system.

At each round of the federated learning protocol, the server sends to each client the weights of the central LSN P system. Each client trains an LSN P system initialized with the weights received from the server. After training, the clients share the weights of their local LSN P systems with the server to be aggregated according to Eq. (1). A complete description is given in Algorithm 1.

$$\mathbf{W}_S = \sum_{i=1}^{N} \left(|D_i| / \sum_{j=1}^{n} |D_j| \right) \mathbf{W}_i \tag{1}$$

Before encoding the input sample $\mathbf{x}$ into potential values of the input neurons, its values are scaled in $[0, 1]$:

$$\mathbf{x} \leftarrow \frac{\mathbf{x} - \mathbf{x_{min}}}{\mathbf{x_{max}} - \mathbf{x_{min}}} \tag{2}$$

The weights of an LSN P system are updated during training according to the Hebbian learning rule:

$$W_c \leftarrow W_c + \eta\,(y - \widetilde{y})\,\mathbf{x} \tag{3}$$

where η is the learning rate, y and $\widetilde{y}$ represent the output of the system, respectively the real label corresponding to the input sample $\mathbf{x}$.

Algorithm 1 Federated learning

Require: $D_1, D_2, \ldots, D_N$, R, E
Ensure: W_S

```
Randomly initialize W_S
for round ← 1 to R do
    for c ← 1 to N do
        W_c ← TrainClientLSNP(D_c, W_M, E)
        W_c ← (1/|D_c|) × W_C
    end for
    W_M ← (Σ_{c=1}^{N} W_c) / (Σ_{c=1}^{N} |D_c|)
end for
return W_M
```

Algorithm 2 Train LSN P system

Require: D_c, W_c, E
Ensure: W_c

```
function TrainClientLSNP(D_c, W_c, E)
    Initialize the LSN P system with the weights W_c
    for x, ỹ in D_c do
        Encode x using Eq. (2)
        Add noise to x
        Initialize the potential values of the input neurons α^1_{pi} with the encoded values.
        for epoch ← 1 to E do
            Compute y, the result of the classification as the spiking time of neuron σ^5_{p1}.
            Update W_c using Eq. (3)
        end for
    end for
end function
```

Our experiments show that a pre-trained LSN P system is vulnerable to membership inference attacks; thus, sending the pre-trained models to the server to be aggregated can reveal information about the datasets of the participants. A solution to this problem is based on additive homomorphic encryption (AHE for short) [1]. An AHE scheme allows one party to compute the ciphertext corresponding to the sum of the plaintexts using only the associated ciphertexts:

$$Enc(m_1) \oplus Enc(m_2) = Enc(m_1 + m_2) \tag{4}$$

The clients choose an AHE scheme and generate the secret key together with the corresponding public key over a secure channel. They encrypt the weights of the locally trained model with the public key and send the ciphertext to the server. Using the homomorphic property described in Eq. (4), the server computes the encryption of the central model weights and sends the result to the clients. The clients decrypt the ciphertext from the server and retrieve the weights of the central model that will be used in the next round of the federated learning protocol. After the last round, the clients will share the model with the server. An overview of the system is depicted in Figure 3.

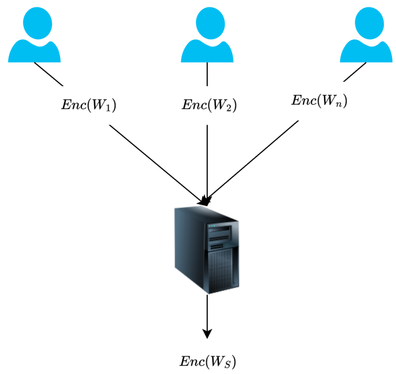

Fig. 3. An overview of the system

5. Experiments

The first set of experiments shows that an LSN P system is prone to membership inference attacks, proving the need to encrypt each model before sending it to the server. The second set of experiments studies various performance matrices of the LSN P system in the federated setup.

The following datasets are used:

(1) Iris plant dataset: this dataset is composed of 150 samples. Each sample contains 4 numeric attributes about plants: the sepal length, the sepal width, the petal length, and the petal width. It is aimed to classify each sample in one of the following classes: iris-setosa, iris-versicolour, and iris-virginica [6].
(2) The handwritten digits dataset: this dataset is composed of 5620 samples. Each sample represents a 8×8 grayscale image of a handwritten digit. The goal is to associate each image digit to the corresponding digit, one of 0 to 9 [3].
(3) Wine recognition dataset: this dataset is composed of 178 samples. Each sample contains 13 numerical attributes that characterize different types of wine, e.g., alcohol, ash, magnesium, flavanoids etc. The goal is to classify each sample into one of the three types of wine encoded as 0, 1, and 2 [2].
(4) The breast cancer dataset: this dataset is composed of 569 samples. Each sample contains 30 numerical attributes related to breast tumors, e.g., radius, texture, perimeter, symmetry, etc. The goal is to classify each sample into benign or malignant [47].

5.1. *Membership inference attacks*

In the first experiment, we show that a pre-trained LSN P system is prone to membership inference attacks. For each dataset, we first train an LSN P system, and then we compute the model confidence over the training dataset and the testing dataset. Each trial of the experiment involves the following steps:

(1) For a dataset D, chose uniformly at random 50% of the samples into the training dataset D_{train}. The testing dataset is $D_{test} = D \setminus D_{train}$.
(2) Train an LSN P system over the set D_{train}.
(3) Output the confidence of the model over the D_{train} and D_{test}.

We run the experiment for 100 trials and average the results. The outcomes are shown in Table 1. For all datasets, the confidence of the model over the training dataset is higher than that over the test dataset. This shows that a pre-trained LSN P system is prone to membership attacks since an attacker can use the mean confidence to determine if a sample is part of the training dataset. We denote by **CTr**, **CTs**, and **D** the confidence over the training dataset, the confidence over the testing dataset, and the difference between the two.

Table 1. The confidence of the pre–trained model

ID	CTr	CTs	D
Iris	0.74	0.70	0.04
Wine	0.72	0.69	0.03
Breast Cancer	0.88	0.84	0.04
Digits	0.99	0.96	0.03

The difference between the training confidence and the test confidence enables the attacker to gain information about the training data distribution.

The second experiment shows how the accuracy of the attack, defined as the accuracy of the binary classifier, varies with respect to the class. We followed the steps of the attack described in Section 3 on the handwritten digits dataset with 50 shadow LSN P systems. The results are presented in Figure 4. This shows that the distribution of the model's outputs is different depending on the true class of the sample. Note the accuracy of the attack on LSN P systems is higher than that on ANN, which is 0.51 on the same dataset [37].

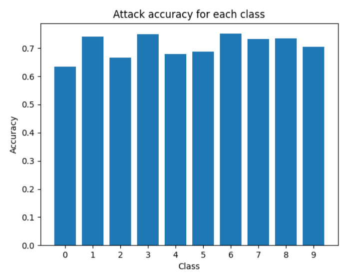

Fig. 4. Attack accuracy with respect to the class

Figure 5 shows how the attack accuracy varies in terms of the number of shadow LSN P systems for each class. It is noted that there is no connection between the number of shadow systems and the accuracy of the attack. The results are similar to the ones presented in Ref. [37].

5.2. *Comparison with other approaches*

For each dataset, 20% of it was kept to evaluate the accuracy of the LSN P model resulting from the federated training protocol. The rest of 80% was split

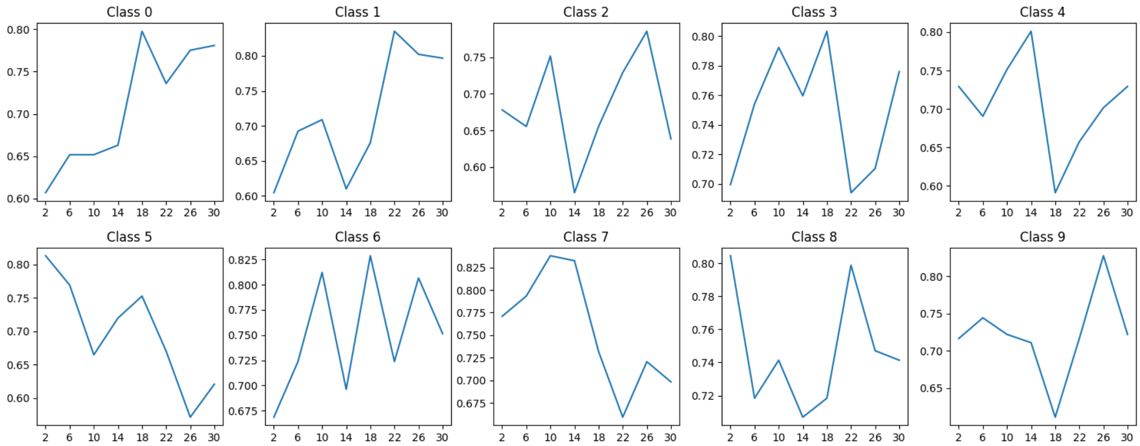

Fig. 5. Attack accuracy depending on the number of shadow models

equally and randomly between each participant. We compare our approach with federated spiking neural networks described in Ref. [44] and with federated artificial neural networks described in Ref. [15]. The ANNs and SNNs used for comparison have the same configuration as LSN P systems, i.e., one hidden layer followed by an output layer.

Figure 6 shows the accuracy of the test data depending on the number of rounds in the federated learning protocol. Although the curve is not smooth, increasing the number of rounds leads to an increase in accuracy. A similar effect is also present in Ref. [44]. The experiment was performed with 5 participants. Figure 7 presents the accuracy with respect to the number of rounds over the handwritten digits dataset for federated LSN P systems, federated SNNs, and federated ANNs. Our protocol archives the best accuracy on every round. The system based on LSN P systems converges more quickly than the one based on ANNs, and both converge faster than the system based on SNNs.

Table 2 presents the accuracy of the protocol on each benchmark dataset using various values of the number of clients. Increasing the number of clients causes a slight decrease in accuracy. However, it remains close to the value obtained by the original LSN P system. Table 3 presents the impact of the number of clients on the accuracy of the central model for the three compared models. The experiment was performed over the handwritten digits dataset. Our protocol obtains the best accuracies for each number of clients. In terms of robustness, the loss in accuracy from one client to 8 clients is, in our case, 0.01. In the case of SNNs, the loss is 0.28, while for ANNs is 0.03.

Table 2. Accuracy comparison on multiple datasets

ID	**1**	**2**	**5**	**8**
Iris	0.98	0.96	0.96	0.93
Wine	0.99	0.98	0.97	0.97
Breast Cancer	0.97	0.95	0.95	0.94
Digits	0.97	0.96	0.96	0.96

Table 3. Accuracy comparison with other approaches

	1	**2**	**5**	**8**
Federated LSN P systems	0.97	0.96	0.96	0.96
Federated SNNs [44]	0.93	0.89	0.80	0.65
Federated ANNs [15]	0.97	0.95	0.95	0.94

We perform experiments with a large number of participants using the handwritten digits dataset. The accuracy of the compared protocols is depicted in Figure 8. Increasing the number of participants decreases the accuracy for all three models, although the most impacted is the one based on SNNs. This effect is also present in Ref. [44]. The reason for this is that, during the experiments, the length of the dataset that is shared between the participants remains constant, which implies that each of them re-

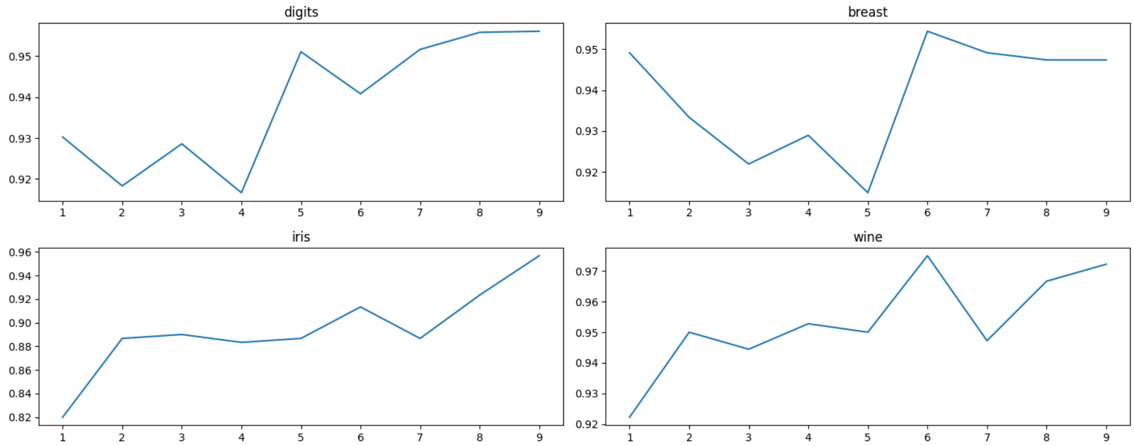

Fig. 6. number of rounds and accuracy - a comparison over multiple datasets

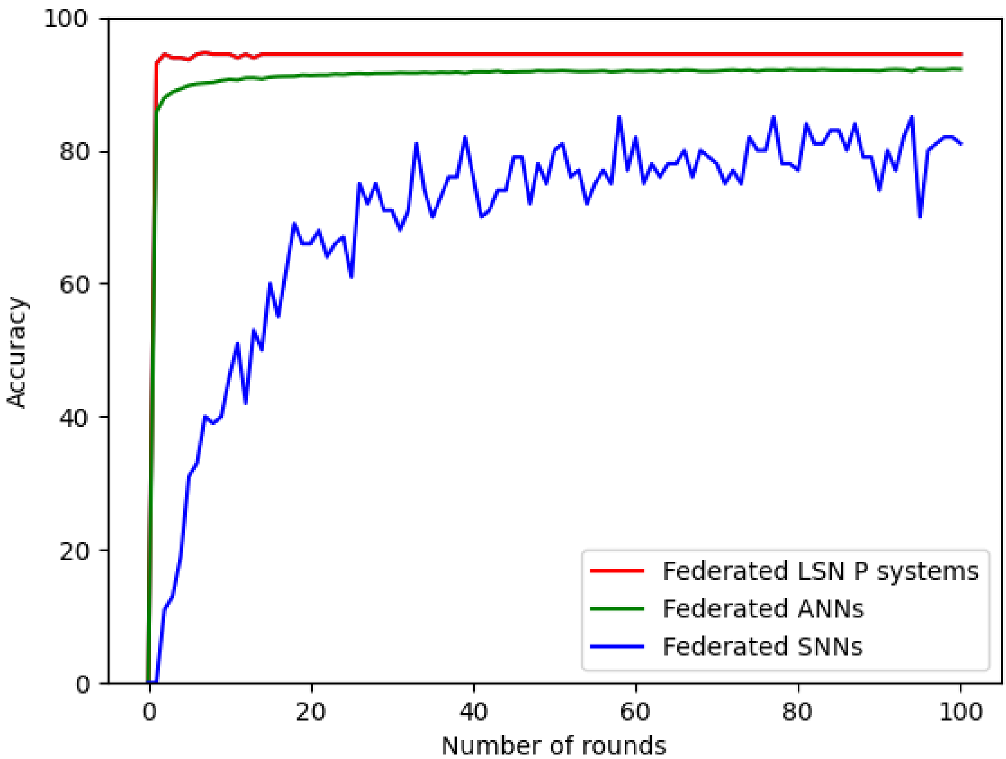

Fig. 7. number of rounds and accuracy - a comparison over multiple datasets

ceives less data as their number increases. The experiment focused on situations where there is a large number of clients who do not have local datasets with a large number of samples. The federated learning protocol based on LSN P systems exceeds the other two. This indicates that the system is scalable.

All the experiments performed so far had the training data randomly and equally divided among

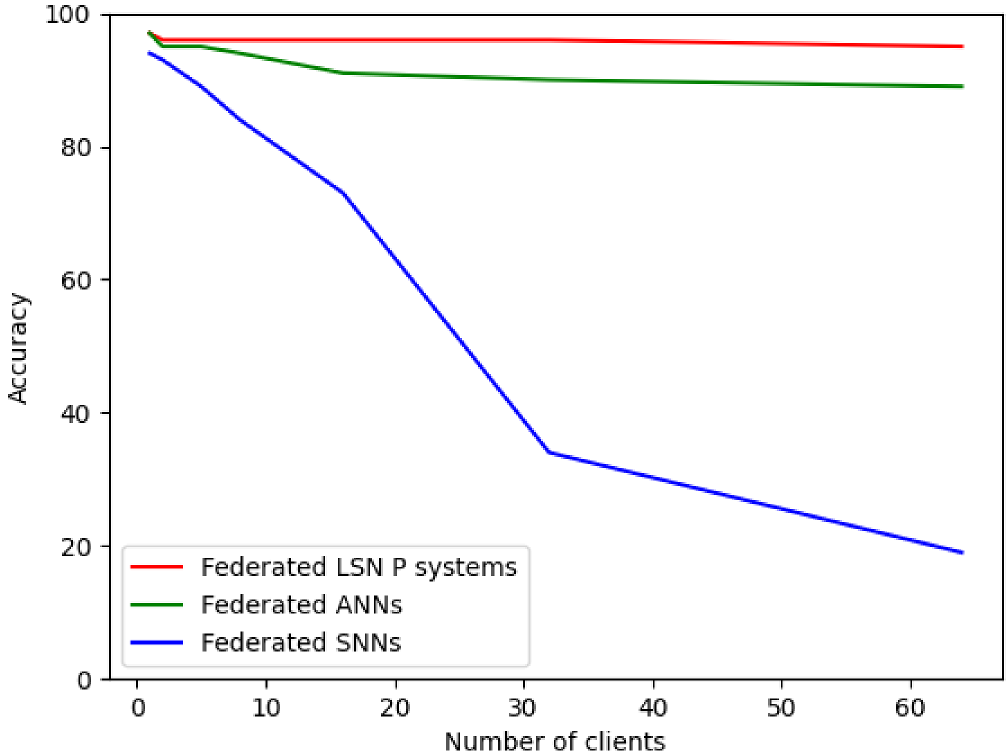

Fig. 8. The effect of the number of participants on accuracy

the participants. In this final experiment, we study the accuracy of our protocol when the training data is partitioned among the participants with respect to the labels. Suppose that we want to solve a classification problem for which we have n training samples. Each sample is assigned one of the c labels from the set $\{y_1, y_2, \ldots, y_c\}$. Let D be the set of proper divisors of c. We perform the following experiment:

(1) For each value $d \in D$ we initialize the protocol with d participants.
(2) The i^{th} participant will receive all training examples that have labels between $\frac{(i-1)\cdot c}{d}$ and $\frac{i \cdot c}{d} - 1$ thus all training examples with a certain label will be assigned to only one participant.
(3) Run the distributed training protocol and compute the accuracy over the test dataset.

The results are given in Table 4.

Table 4. Accuracy of the trained LSN P model with data partitioned with respect to labels

ID	**Num participants**	**Acc**
Iris	3	0.28
Wine	3	0.27
Breast Cancer	2	0.55
Digits	2	0.86
Digits	5	0.40
Digits	10	0.03

For this experiment, we conclude that to train the LSN P model distributed with similar accuracy to centralized training, each participant must train his local LSN P model with data as diverse as possible regarding the labels. From Table 4, we see that when each participant receives a single type of data, i.e., data with a single label (e.g., the first participant received only images with the digit 0, the second participant received only images with the digit 1, etc.), the accuracy is similar to random guessing. This last experiment studied the sensitivity of the LSN P systems in a federated learning setup.

All experiments were performed on an HP EliteBook 650 with 32GB of RAM. The code for the experiments is available at https://github.com/miiip/Federated-LSNP.

6. Conclusions and further developments

In this paper, we proposed a federated learning protocol for LSN P systems. Our approach involves sharing the weights of the locally trained models with the server. We also assessed the impact of a membership inference attack on pre-trained LSN P systems and suggested a solution based on additive homomorphic encryption. We compared our protocol with other federated learning approaches for spiking and artificial neural networks. We proved experimentally that our approach yields better accuracies, converges faster during training, and is more robust for small local datasets.

The first direction for future research is to construct a protocol that obtains usable accuracy even if the data is not randomly and equally distributed. The paper shows that accuracy drops when each participant has data of only one type (one label).

The second direction of research is to investigate the behavior of LSN P systems in a federated learning setup over large datasets.

A third research direction of interest is to investigate other types of architectures for federated learning of LSN P systems. There are cases in which the data is not horizontally partitioned and when clients must manage the updating of local models themselves without a server to direct the process.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (61972324), Sichuan Science and Technology Program (2023NS-FSC1985, 2023YFG0046) and Innovation Research Group of CUIT (KYTD202212).

References

1. Acar, Abbas and Aksu, Hidayet and Uluagac, A Selcuk and Conti, Mauro, A survey on homomorphic encryption schemes: Theory and implementation, *ACM Computing Surveys (Csur)* **51**(4) (2018) 1–35.
2. S. Aeberhard and M. Forina, Wine UCI Machine Learning Repository, (1991), DOI: https://doi.org/10.24432/C5PC7J.
3. E. Alpaydin and C. Kaynak, Optical Recognition of Handwritten Digits UCI Machine Learning Repository, (1998), DOI: https://doi.org/10.24432/C50P49.
4. Aono, Yoshinori and Hayashi, Takuya and Wang, Lihua and Moriai, Shiho and others, Privacy-preserving deep learning via additively homomorphic encryption, *IEEE Transactions on Information Forensics and Security* **13**(5) (2017) 1333–1345.
5. Bi, Guo-qiang and Poo, Mu-ming, Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type, *Journal of Neuroscience* **18**(24) (1998) 10464–10472.
6. Fisher, Ronald A, The use of multiple measurements in taxonomic problems, *Annals of Eugenics* **7**(2) (1936) 179–188.
7. Ghosh-Dastidar, Samanwoy and Adeli, Hojjat, Improved spiking neural networks for EEG classification and epilepsy and seizure detection, *Integrated Computer-Aided Engineering* **14**(3) (2007) 187–212.
8. Ghosh-Dastidar, Samanwoy and Adeli, Hojjat, A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection, *Neural Networks* **22**(10) (2009) 1419–1431.
9. Ghosh-Dastidar, Samanwoy and Adeli, Hojjat, Spiking neural networks, *International Journal of Neural Systems* **19**(04) (2009) 295–308.
10. Gilad-Bachrach, Ran and Dowlin, Nathan and Laine, Kim and Lauter, Kristin and Naehrig, Michael and Wernsing, John, Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy, *International Conference on Machine Learning*, PMLR, (Cambridge MA: JMLR, 2016), pp. 201–210.
11. Gou, Xiantai and Liu, Qifen and Rong, Haina and Hu, Meng and Paul, Pirthwineel and Deng, Fang and Zhang, Xihai and Yu, Zhibin, A Novel Spiking Neural P System for Image Recognition., *International Journal of Unconventional Computing* **16**(2-3) (2021) 121–139.
12. Graepel, Thore and Lauter, Kristin and Naehrig, Michael, ML confidential: Machine learning on encrypted data, *International Conference on Information Security and Cryptology*, Springer, (Springer Cham, 2012), pp. 1–21.
13. Han, Bing and Fu, Qiang and Zhang, Xinliang, Towards Privacy-Preserving Federated Neuromorphic Learning via Spiking Neuron Models, *Electronics* **12**(18) (2023) p. 3984.
14. Hassanpour, Ahmad and Moradikia, Majid and Adeli, Hojjat and Khayami, Seyed Raouf and Shamsinejadbabaki, Pirooz, A novel end-to-end deep learning scheme for classifying multi-class motor im-

agery electroencephalography signals, *Expert Systems* **36**(6) (2019) p. e12494.
15. W. Hongyi, Y. Mikhail, S. Yuekai, P. Dimitris and K. Yasaman, Federated learning with matched averaging, *International Conference on Learning Representations*, 2020.
16. Ji, Zhanglong and Zachary C., Lipton and Elkan, Charles, Differential privacy and machine learning: a survey and review, *arXiv preprint arXiv:1412.7584* (2014).
17. Kan Xie and Zhe Zhang and Bo Li and Jiawen Kang and Dusit Tao Niyato and Shengli Xie and Yi Wu, Efficient Federated Learning With Spike Neural Networks for Traffic Sign Recognition, *IEEE Transactions on Vehicular Technology* **71** (2022) 9980–9992.
18. Kim, Youngeun and Venkatesha, Yeshwanth and Panda, Priyadarshini, PrivateSNN: Privacy-Preserving Spiking Neural Networks, *Proceedings of the AAAI Conference on Artificial Intelligence*, **36**(1)2022, pp. 1192–1200.
19. Koravuna, Shamini and Rückert, Ulrich and Jungeblut, Thorsten and others, Evaluation of Spiking Neural Nets-Based Image Classification Using the Runtime Simulator RAVSim., *International Journal of Neural Systems* (2023) 2350044–2350044.
20. LeCun, Yann and Bengio, Yoshua and Hinton, Geoffrey, Deep learning, *Nature* **521**(7553) (2015) 436–444.
21. Liu, Qian and Huang, Yanping and Yang, Qian and Peng, Hong and Wang, Jun, An Attention-Aware Long Short-Term Memory-Like Spiking Neural Model for Sentiment Analysis., *International Journal of Neural Systems* (2023) 2350037–2350037.
22. Martins, Guilherme Brandão and Papa, João Paulo and Adeli, Hojjat, Deep learning techniques for recommender systems based on collaborative filtering, *Expert Systems* **37**(6) (2020) p. e12647.
23. Mohassel, Payman and Zhang, Yupeng, Secureml: A system for scalable privacy-preserving machine learning, *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, (IEEE, 2017), pp. 19–38.
24. Muhammad Arsalan and Davide Di Matteo and Sana Imtiaz and Zainab Abbas and Vladimir Vlassov and Vadim Issakov, Energy-Efficient Privacy-Preserving Time-Series Forecasting on User Health Data Streams, *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)* (2022) 541–546.
25. Narayanan, Arvind and Shmatikov, Vitaly, Robust de-anonymization of large sparse datasets, *2008 IEEE Symposium on Security and Privacy (SP 2008)*, IEEE, (IEEE, 2008), pp. 111–125.
26. Nasr, Milad and Shokri, Reza and Houmansadr, Amir, Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning, *2019 IEEE symposium on security and privacy (SP)*, IEEE2019, pp. 739–753.
27. Nogay, Hidir Selcuk and Adeli, Hojjat, Machine learning (ML) for the diagnosis of autism spectrum disorder (ASD) using brain imaging, *Reviews in the Neurosciences* **31**(8) (2020) 825–841.
28. Nogay, Hidir Selcuk and Adeli, Hojjat, Detection of epileptic seizure using pretrained deep convolutional neural network and transfer learning, *European Neurology* **83**(6) (2021) 602–614.
29. Nogay, Hidir Selcuk and Adeli, Hojjat, Diagnostic of autism spectrum disorder based on structural brain MRI images using, grid search optimization, and convolutional neural networks, *Biomedical Signal Processing and Control* **79** (2023) p. 104234.
30. Nogay, Hidir Selcuk and Adeli, Hojjat, Multiple Classification of Brain MRI Autism Spectrum Disorder by Age and Gender Using Deep Learning, *Journal of Medical Systems* **48**(1) (2024) p. 15.
31. O'Reilly, Randall C and Munakata, Yuko, *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain* (MIT press, 2000).
32. Pleşa, Mihail-Iulian and Gheorghe, Marian and Ipate, Florentin, Privacy-preserving Linear Computations in Spiking Neural P Systems, *arXiv preprint arXiv:2309.13803* (2023).
33. Păun, Gheorghe, Membrane computing, *Scholarpedia* **5**(1) (2010) p. 9259.
34. M. H. Rafiei, L. V. Gauthier, H. Adeli and D. Takabi, Self-supervised learning for electroencephalography, *IEEE Transactions on Neural Networks and Learning Systems* **35**(2) (2024) 1457–1471.
35. Rong, Haina and Wu, Tingfang and Pan, Linqiang and Zhang, Gexiang, Spiking neural P systems: Theoretical results and applications, *Enjoying natural computing: Essays dedicated to Mario de Jesús Pérez-Jiménez on the occasion of his 70th birthday* (2018) 256–268.
36. Shabtai, Asaf and Elovici, Yuval and Rokach, Lior, *A survey of data leakage detection and prevention solutions* (Springer Science & Business Media, 2012).
37. Shokri, Reza and Stronati, Marco and Song, Congzheng and Shmatikov, Vitaly, Membership inference attacks against machine learning models, *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, (IEEE, 2017), pp. 3–18.
38. Skatchkovsky, Nicolas and Jang, Hyeryung and Simeone, Osvaldo, Federated neuromorphic learning of spiking neural networks for low-power edge intelligence, *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE2020, pp. 8524–8528.
39. Song, Tao and Pan, Linqiang, Spiking neural P systems with rules on synapses working in maximum spiking strategy, *IEEE Transactions on Nanobioscience* **14**(4) (2015) 465–477.
40. Song, Tao and Pan, Linqiang and Wu, Tingfang and Zheng, Pan and Wong, ML Dennis and Rodríguez-Patón, Alfonso, Spiking neural P systems with learn-

ing functions, *IEEE Transactions on Nanobioscience* **18**(2) (2019) 176–190.

41. Song, Tao and Pang, Shanchen and Hao, Shaohua and Rodríguez-Patón, Alfonso and Zheng, Pan, A parallel image skeletonizing method using spiking neural P systems with weights, *Neural Processing Letters* **50** (2019) 1485–1502.
42. Tang, Guangzhi and Shah, Arpit and Michmizos, Konstantinos P, Spiking neural network on neuromorphic hardware for energy-efficient unidimensional slam, *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, (IEEE, 2019), pp. 4176–4181.
43. Tiwari, Anjani Kumar and Adhikari, Anupriya and Mishra, Lokesh Chandra and Srivastava, Abhishek, Current Status of Our Understanding for Brain Integrated Functions and its Energetics, *Neurochemical Research* **47**(9) (2022) 2499–2512.
44. Venkatesha, Yeshwanth and Kim, Youngeun and Tassiulas, Leandros and Panda, Priyadarshini, Federated learning with spiking neural networks, *IEEE Transactions on Signal Processing* **69** (2021) 6183–6194.
45. Wang, Jun and Shi, Peng and Peng, Hong and Pérez-Jiménez, Mario J and Wang, Tao, Weighted fuzzy spiking neural P systems, *IEEE Transactions on Fuzzy Systems* **21**(2) (2012) 209–220.
46. Wang, Zi-Qin and Manry, Michael T and Schiano, Jeffrey L, LMS learning algorithms: misconceptions and new results on converence, *IEEE Transactions on Neural Networks* **11**(1) (2000) 47–56.
47. Wolberg, William and Mangasarian, Olvi and Street, Nick and Street, W., Breast Cancer Wisconsin (Diagnostic) UCI Machine Learning Repository, (1995), DOI: https://doi.org/10.24432/C5DW2B.
48. Zhang, Gexiang and Zhang, Xihai and Rong, Haina and Paul, Prithwineel and Zhu, Ming and Neri, Ferrante and Ong, Yew-Soon, A layered spiking neural system for classification problems, *International Journal of Neural Systems* **32**(08) (2022) p. 2250023.
49. Zhang, Yujie and Yang, Qian and Liu, Zhicai and Peng, Hong and Wang, Jun, A Prediction Model Based on Gated Nonlinear Spiking Neural Systems, *International Journal of Neural Systems* **33**(06) (2023) p. 2350029.