Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks

Ivan Peralta[1 [0009−0004−8030−2320]], Nanci Odetti[1 [0009−0006−8547−3967]], and Hugo L. Rufiner[1,2 [0000−0002−1083−5891]]

[1] Laboratorio de Cibernética, Facultad de Ingeniería UNER, Oro Verde, Argentina
[2] Instituto Señales, Sistemas e Inteligencia Computacional, sinc(i) UNL-CONICET, Santa Fe, Argentina, http://www.sinc.unl.edu.ar

Abstract. The automatic recognition of speaker-independent digits requires high accuracy and robustness to noise and variability. Spiking neural networks (SNNs) are a promising model for this task, as they can mimic the temporal dynamics and energy efficiency of the human auditory system. However, SNNs are difficult to train and often require complex learning algorithms. Spoken digits provide a useful benchmark task to evaluate new SNN architectures. Performance in small vocabulary tasks is an important first step before scaling up to more complex recognition scenarios. In this paper, we propose to use an extreme learning layer (ELL) as a simple and effective way to improve the learning of SNNs for spoken digit recognition. The ELL is a randomly generated layer that maps the input features to the next layer without any further adjustment. The output layer is then trained by entropy minimization. We show that ELL can boost the performance of the SNN on the benchmark data set TIDIGITS. We also compare our approach with some state-of-the-art methods achieving competitive results with less computational cost and complexity. The proposed approach also shows good robustness to additive noise.

Keywords: Spiking Neural Networks · Extreme Learning Machines · Automatic Speech Recognition

1 Introduction

The task of speech recognition has numerous applications in various fields such as education, health, security, and communication. However, despite significant advances, one of the main barriers to speech recognition is the difficulty for current systems to handle noise in the signal, while the human brain is almost immune to it [1]. Therefore, it is important to explore new strategies to address this problem, especially those inspired by the functioning of the auditory system and the human brain.

Deep neural networks (DNNs) are the models that currently achieve the best performance in most speech-related tasks. DNNs also have the ability to learn features directly from raw speech samples. However, despite their success, DNNs

suffer from some drawbacks such as high computational requirements and high energy consumption [5], which encourages the scientific community to continue looking for alternatives.

One strategy that has generated growing interest in the last years is the use of spiking neural networks (SNNs), a special type of artificial neural network that aims to closely mimic the functioning of biological neurons [10]. This enables event-driven and parallel implementations to improve efficiency [8,3,13]. One of the most significant properties of SNNs is their ability to communicate between neurons using pulses that mimic the action potentials of biological neurons. These potentials can be efficiently modelled using binary codes, which facilitates the implementation of SNNs in digital devices [8]. In addition, SNNs mimic the internal potential of the membrane by following a temporal dynamic similar to biological neurons [3], making them suitable for processing temporal signals such as speech [13].

Extreme Learning Machines (ELM) is a learning technique proposed by Huang et al. [7] that allows for fast and efficient training of single hidden layer neural networks. The basic idea behind ELM is to randomly assign the weights of the connections between the input layer and the hidden layer and then calculate the weights of the connections between the hidden layer and the output layer using a closed-form analytical formula. ELMs have several advantages over other neural network training methods. Firstly, it is very fast, as it does not require iterations to adjust the connection weights. Secondly, ELM is very easy to implement, as it does not require the calculation of gradients or the adjustment of hyperparameters such as the learning rate or momentum. Third, ELM has been shown to perform well on various classification tasks, including speech recognition [6].

In this work, we propose a new supervised neural network (DELSNN, Digital Extreme Learning Spiking Neural Network) that is tested on a spoken digit recognition task. The DELSNN was inspired by the classical work presented by Unnikrishnan and Hopfield [14], but we also added an extreme learning layer. Unnikrishnan proposed an analogue network for the recognition of isolated digits using a binary temporal encoding of speech spectral features.

The main contributions are a novel SNN architecture incorporating random projections for feature expansion combined with direct output layer training by entropy minimization, alongside evaluations demonstrating noise robustness for isolated spoken digit recognition. Detailed analysis of the DELSNN training and results is presented for the first time. In a previous work [12], an FPGA implementation and some preliminary results showing real-time capacity were presented.

The rest of the article is organized as follows. Section 2 presents the design and implementation of the proposed DELSNN, as well as the training algorithm and the spike encoding method. Section 3 shows the experimental results obtained in a database of spoken digits in English. Finally, Section 4 presents the conclusions and possible lines of future work.

2 Methods

2.1 Proposed DELSNN Model

The SNN proposed in this work is called DELSNN (Digital Extreme Learning Spiking Neural Network), and its goal is to recognize spoken digits in clean or additive noise conditions. The SNN has a feedforward architecture with one input layer and two layers of neurons: the extreme learning layer (ELL) **I** and the output layer **J**. The input layer is not composed of neurons but of a vector **V** that receives a binary encoding of the speech signal as a stimulus, obtained by spectral analysis and mapping to spikes. The neurons of the ELL layer are connected with multiline connections on one side to each coefficient of the input vector, and on the other side to each neuron in the output layer. The output layer has as many neurons as digit classes to recognize (11 in this case), and each one emits a spike when it detects the presence of a specific digit.

The main feature of DELSNN is that it uses temporal delays and random weights in the connections of the neurons for the first layer, allowing for extreme learning without the need to adjust these weights using complex algorithms. Temporal delays introduce temporal diversity in neuron inputs, facilitating class separation. Random weights introduce spatial diversity in neuron inputs, promoting generalization capacity. The only component that requires training are the output layer weights, which are adjusted using a straightforward algorithm based on entropy minimization.

The neuronal model used for SNN neurons is the *spike response model* (SRM), which is one of the models with a good trade-off between realistic and efficient hardware implementation.

A simplified diagram of the DELSNN architecture is shown in Figure 1, illustrating the flow of data from input spikes to output spikes.

2.2 Speech Signal Encoding

Speech signal encoding is the process of transforming the speech signal into a binary sequence that can be used as a stimulus for SNN. Speech signal encoding consists of two main steps: feature extraction and spike mapping.

In this work feature extraction is based on spectral analysis performed using a Mel scale spectrogram (MSS). This is a widely used and effective psychoacoustic-motivated energy distribution of the speech signal over time and frequency.

The process of spike mapping involves converting the MSS into a binary representation by assigning each value a 0 (no spike) or a 1 (spike). This is accomplished by applying a thresholding function to each frequency band of the spectrogram. If the energy of the band surpasses a certain threshold, a spike is generated. In this study, the thresholding function used for spike mapping differs from [14] as it does not use the lateral inhibition strategy. Instead, an adjustable parameter is used to determine the threshold.

The result of the speech signal encoding is a matrix of spikes, where each row corresponds to a frequency band, and each column to a time window. This

sinc(i) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

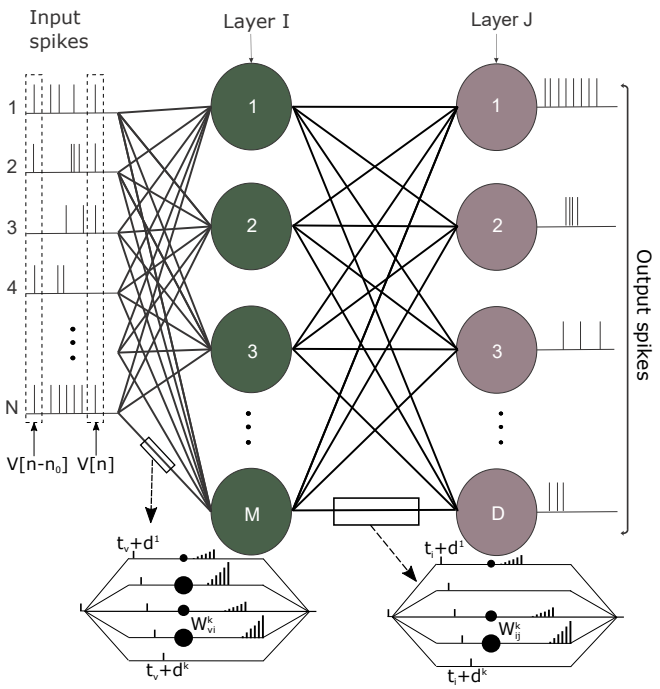

Fig. 1. Simplified diagram of the proposed DELSNN structure.

matrix is used as input for the SNN, where each frequency band corresponds to a coefficient of the vector $\mathbf{V}$.

2.3 Neuron Model

The spiking neuron model used for this work is the Spike Response Model (SRM). The SNN operates in a time-driven manner, where the simulation time increments by a constant value, and at each step, the presence of spikes in the input vector $\mathbf{V}$ is analyzed to update the state of all neurons in the SNN. Below, we describe the operation of a neuron i belonging to the layer $\mathbf{I}$. However, this explanation can be applied to any neuron in the SNN.

The state of neuron i is described by the state variable u_i. This neuron fires if u_i reaches the threshold ϑ. At the instant when the threshold is crossed, the discrete firing time $n_i^{(f)}$ is defined, where f indicates the number of firings or spikes emitted by the neuron i. The set of all firing instants of neuron i is defined as:

$$\mathcal{F}_i = \left\{ n_i^{(f)} \right\} = \{ n / u_i[n] > \vartheta \} . \tag{1}$$

The sequence of spikes emitted by a neuron in layer $\mathbf{I}$ can be written as a sequence of discrete Dirac deltas:

sinc(*i*) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

$$S_i[n] = \sum_{n_i^{(f)} \in \mathcal{F}_i} \delta[n - n_i^{(f)}], \quad with \quad \mathcal{F}_i = \left\{ n_i^{(1)}, ..., n_i^{(s)} \right\}. \tag{2}$$

The excitations to the neuron i are given by the spikes coming from each component of the input vector $\mathbf{V}$. Moreover, since the input vector is updated at each simulation instant, the temporal evolution of each component can be described as $V_v[n]$, where n is the simulation time instant, and v (with $1 \leq v \leq N$) indicates the v-th component of the vector $\mathbf{V}$. A spike that belongs to an element of the input vector is distributed among each of the propagation delays d^k of each connection leaving that component. Once the spikes have been delayed by the propagation delays, they reach the neurons as presynaptic pulses triggering a change in the neuronal state. Each presynaptic spike that excites neuron i increments (or decrements) its variable u_i by an amount $W_{v,i}^k, h[n - d^k]$, where h is the impulse response (spike) of the neuron. The increment or decrease in the variable depends on whether the weight is positive or negative, respectively. The state u_i of neuron i at time n is given by the linear superposition of the contributions from all propagation delays of all components of the input vector. This is described in Eq. 3[3]:

$$u_i[n] = \sum_{\nu=1}^{N} \sum_{k=1}^{K} \sum_{\tau=0}^{\infty} V_\nu[\tau]\, W_{\nu,i}^k\, h[n - \tau - d^k]. \tag{3}$$

If we consider a neuron in the output layer $\mathbf{J}$, the update equation for the state variable u_j of neuron j at time n is given by the linear superposition of the contributions of all spikes coming from neurons in layer $\mathbf{I}$. This is described in Eq. 4:

$$u_j[n] = \sum_{i=1}^{M} \sum_{k=1}^{K} \sum_{\tau=0}^{\infty} S_i[n]\, W_{i,j}^k\, h[n - \tau - d^k]. \tag{4}$$

Similar to Eq. 2, we can represent a sequence of spikes emitted by a neuron in layer $\mathbf{J}$ as a sequence of discrete Dirac deltas:

$$S_j[n] = \sum_{n_j^{(f)} \in \mathcal{F}_j} \delta[n - n_j^{(f)}], \qquad \mathcal{F}_j = \left\{ n_j^{(1)}, ..., n_j^{(s)} \right\}, \tag{5}$$

where $n_j^{(f)}$ is the instant of the f-th firing of neuron $j \in \mathbf{J}$.

Considering Eqs. 3 and 4, the internal operation of a neuron, that is, the variation of its state u, can be interpreted as the discrete linear convolution of incoming spike sequences with an impulse response function $h[n]$. This implies that when simulating the operation of the SNN, one can choose an "arbitrary" $h[n]$.

[3] The internal summation with the upper limit ∞ indicates that the summation is performed until the entire spike encoding of the digit is completed.

sinc(*i*) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

2.4 Output Layer Training through Entropy Minimization

During each iteration of the process, the SNN is presented with all the training pronunciations encoded in spikes. Projecting these spikes by the fixed random layer ELL enables weight updates of the next layer to be performed through an iterative process. When a pronunciation is presented, it produces a sequence of spikes ($S_i[n]$) at the outputs of neurons in layer **I**. These sequences of spikes ($S_i[n]$) are then distributed in the propagation delays to enter the neurons in the output layer **J**. Equation 4 can be rephrased in the following way:

$$u_j[n] = \sum_{i=1}^{M} \sum_{k=1}^{K} W_{i,j}^{k} \sum_{\tau=0}^{\infty} S_i[n] h[n - \tau - d^k], \tag{6}$$

$$u_j[n] = \sum_{i=1}^{M} \sum_{k=1}^{K} W_{i,j}^{k} R_{ik}[n], \tag{7}$$

where $R_{ik}[n] = \sum_{\tau=0}^{\infty} S_i[n] h[n-\tau-d^k]$ is the unweighted response of any neuron belonging to the output layer to the sequence of spikes emitted by the i-th neuron in layer I with the k-th delay. That is, for each digit w in the training word set $\mathbb{W}$, the input data to the second layer consists of $M \times K$ responses $R_{ik}^{w}[n]$, which will be used to describe the weight update function. In turn, the endpoint n^* is defined as the moment when digit w generated the last spike in layer I. That is:

$$n^* = \max \{ n \mid S_i[n] = 1 \}. \tag{8}$$

Considering the endpoint n^*, we can define $R_{ik}^{w}[n^*] = R_{ik}^{w}$.

Next, we will describe the training considering only the endpoints for each pronunciation of digit w. That is, for each digit, only the $D = 11$ values of the state variables u_j of the output neurons $j = 1, 2, ..., D$ will be used at the endpoint ($u_j[n^*]$). Later, the algorithm will be extended to the rest of the values of n.

The goal is to have the activity of each output neuron represent the membership to a specific digit class. To achieve this, the probability of belonging to a class can be approximated by applying a function to the state variable $u_j[n]$. For this, it is necessary to ensure that the range of this function varies between zero and one. The probability of digit w belonging to the j-th class at instant n^* is defined as $V_j^w[n^*] = V_j^w$. This value is determined by the SNN and is obtained by evaluating $u_j[n^*]$ with a sigmoid function whose range varies between zero and one. Eqs. 9 and 10 show two possible options for the sigmoid function:

$$V_j^w = \frac{1}{2} \left[1 + \tanh \left(\beta u_j[n^*] \right) \right], \tag{9}$$

$$V_j^w = \frac{1}{1 + e^{-a u_j[n^*]}}. \tag{10}$$

Associated with each digit w in the training set, there are the probabilities $P_{+,j}^{w}$ that the digit is an instance of category j, and $P_{-,j}^{w}$ that the digit is not an

sinc(*i*) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

instance of that category, where $P^w_{-,j} = 1 - P^w_{+,j}$. These probabilities can only take unambiguous values of "1" or "0", given that the class to which each training word belongs is known. The proposed algorithm uses the Kullback-Leibler divergence measure between two probability distributions. In other words, it minimizes the relative entropy between the distribution $P^w_{+,-,j}$ – which represents the desired output values of the SNN – and the distribution $V^w_{+,-,j}$, which is the actual output obtained by the SNN. This is expressed as:

$$\mathcal{D}_{KL}(P||V) = \sum_{\mathbb{W}} \sum_{j} \sum_{+,-} P^w_{+,-,j} \ln \left(\frac{P^w_{-,+,j}}{V^w_{-,+,j}} \right), \tag{11}$$

where $V^w_{+,j} = V^w_j = 1 - V^w_{-,j}$. The inclusion of non-membership cases (negative probabilities) is equivalent to what is done when training a multilayer perceptron (MLP). During the training of an MLP, the desired output value of the neuron representing the class of the input pattern is usually set to one, and the outputs of the rest of the neurons in the network are set to zero. This is done because, for a particular word, we only want the SNN output neuron representing the pronounced digit to be active.

To perform the weight adaptation in the last layer, the well-known Newton's method equation can be used:

$$\Delta W^k_{i,j} = -\epsilon \, \frac{\mathrm{d}\mathcal{D}_{KL}(P||V)}{\mathrm{d}W^k_{i,j}}. \tag{12}$$

$$\mathcal{D}_{KL}(P||V) = \sum_{\mathbb{W}} \sum_{j} \sum_{+,-} P^w_{+,-,j} \left[\ln \left(P^w_{+,-,j} \right) - \ln \left(V^w_{+,-,j} \right) \right] \tag{13}$$

Then, the negative derivative of the divergence is determined as follows:

$$-\frac{dD_{KL}(P||V)}{dW^k_{i,j}} = \sum_{\mathbb{W}} P^w_{+,j} \frac{V^{w'}_{+,j}}{V^w_{+,j}} - P^w_{-,j} \frac{V^{w'}_{+,j}}{1 - V^w_{+,j}} \tag{14}$$

If we use the sigmoid function described in Eq. 9, we can write:

$$V^{w'}_j = V^w_j \beta R^w_{i,k} \left[1 - \tanh \left(\beta u_j[n^*] \right) \right], \tag{15}$$

$$1 - V^w_{+,j} = \frac{1}{2} \left[1 - \tanh \left(\beta u_j[n^*] \right) \right]. \tag{16}$$

Then, the equation for the weight update becomes:

$$\Delta W^k_{i,j} = \epsilon \sum_{\mathbb{W}} [P^w_{+,j} - V^w_{+,j}] R^w_{i,k}, \tag{17}$$

where the constant ϵ determines the speed at which the algorithm attempts to reach the optimal weight values. If its value is very small, the algorithm will take many iterations to reach the goal. On the other hand, if ϵ is too large, it will not reach an optimal solution. In the experiments conducted in this work, $\epsilon = 0.008$

sinc(i) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

was used. It can be shown that using the sigmoid function from Eq. 10 leads to the same result. If we calculate the second derivative of the divergence, we get:

$$\frac{d^2 D_{KL}(P||V)}{dW_{i,j}^k} = \sum_{\mathbb{W}} \beta (R_{i,k}^w)^2 V_{+,j}^w [1 - \tanh(\beta u_j[n^*])] \geq 0. \tag{18}$$

Since Eq. 18 is always greater than or equal to zero, the use of Eq. 17 will always lead to the global minimum of the divergence $\mathcal{D}_{KL}(P||V)$.

While the algorithm was described for the endpoint n^*, it can be extended to any temporal point $n \in \mathbb{P}$ of the pronounced word. Simply include the evaluation points described by the set $\mathbb{P}$ in the summation and establish the desired value ($P_{+,j}^w$) for those points. In the experiments carried out in this work, all points of each pronunciation are used for adaptation. The weight update rule can be expressed by the following equation:

$$\Delta W_{i,j}^k = \epsilon \sum_{\mathbb{W}} \sum_{\mathbb{P}} \left[P_{+,j}^w - V_{+,j}^w[n] \right] R_{i,k}^{w[n]}, \tag{19}$$

where the desired values are determined by the following rules:

- If the pronounced word belongs to the class of the output neuron
  - $P_{+,j}^w = 1$ if the training point is the endpoint.
  - If the training point does not match the endpoint, $P_{+,j}^w$ is unspecified, and no adaptation is performed.
- If the output neuron does not match the class of the pronounced word, $P_{+,j}^w = 0$ at the endpoint and the other evaluation points.

The choice of the above rules allows training the output neurons to emit spikes in a period of time close to the endpoint of the pronunciation. The algorithm is initialized with all weights set to zero.

3 Results and Discussion

3.1 Database and Experiments

To train and evaluate DELSNN, subsets of the TI-DIGITS database [9] were used. This database comprises isolated English digit pronunciations and digit sequences spoken by various speakers from 21 different dialectal regions in the United States. For this work, only isolated-digit pronunciations were used. The considered digits range from 0 to 9, including both versions of the zero digit commonly used in English speech (zero and oh). The database consists of 326 speakers, including 111 males, 114 females, 50 boys, and 51 girls.

Each speaker produced two pronunciations for each digit, resulting in 22 isolated digit pronunciations per speaker. For the experiments carried out in this work, these speakers were divided into three distinct sets: train, test, and validation.

To train the weights and delay lengths of the final layer, various experiments were performed. The study considered 11 classes (zero, oh, one, two,..., nine), with 138 pronunciations per class used for the training dataset, 226 pronunciations per class for the evaluation dataset, and 50 pronunciations per class for the validation dataset. As each speaker contributed two pronunciations for each digit, the total number of speakers amounted to 207, with 113 for evaluation (57 females, 56 males), 69 for training (42 females, 27 males), and 25 for validation (15 females, 10 males). These datasets are mutually exclusive, meaning that speakers present in one dataset are not included in the others. Thus, the speech recognition process is independent of the speaker. The partitioning used adheres to the proposals provided in the TIDIGITS database to facilitate comparison with other research.

3.2 Design Decisions and Hyperparameter Tuning

The reproducibility of results is crucial in scientific research. To reproduce an experiment, all relevant parameters must be thoroughly documented. In machine learning, this includes specifying the hyperparameter configuration used to train models, as small variations can greatly impact final performance. A reduced subset of the training partition was used to find suitable values for relevant hyperparameters, such as:

- Number of frequency bands: Values between 16 and 64 were tested. 32 bands were chosen as a good trade-off between resolution and dimensionality.
- Number of neurons M for the ELL layer: Values between 16 and 64 were tested.
- Threshold for spike generation Ψ: Values between 0.25 and 0.75 were evaluated. 0.37 was selected as it provided the best results.
- Sigmoid parameter a (Ec. 10): Values between 0.0001 and 0.1 were tested. $a = 0.001$ provided the fastest convergence.
- Learning rate ϵ: Values between 0.001 and 0.1 were tried. 0.008 resulted in stable learning.

The number of output layer neurons was set to 11, to account for the 1-9 digit classes plus two extra neurons for distinguishing the two “0” versions. The number of training iterations was fixed at 100 based on convergence analysis.

The neuron model parameters were configured as follows: threshold $\theta = 15$ for ELL neurons and $\theta = 1$ for output layer neurons, and impulse response $h[n]$ matching a linearly increasing function. shape. Refractory periods were not taken into account. The weights of the ELL were randomly assigned with a mean $\mu = 0$ and variance $\sigma = 80$. Twenty delays per connection were used, and the duration of the delays starts at zero and increases by a constant value $T_i = 60$ milliseconds. The value T_i also coincides with the duration of the Spike-Response Model (SRM), which only mimics the morphology of a biological response, not its temporal duration. The SNN was configured so that all weights are of integer type to facilitate the transfer to digital design.

Table 1 resumes the main final hyperparameters of the proposed DELSNN model.

sinc(i) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

Table 1. DELSNN final hyperparameters. See Fig. 1 for notation references.

Stage	Hyperparameter	Value
Spectral Analysis	Sampling frequency (f_m)	8 kHz
	Highest frequency	3785 Hz
	Lowest frequency	227 Hz
	Frequency bands	32
	Window type	Hamming
	Window samples	80
	Overlap	50 %
Spike Encoding	Threshold Ψ	0.37
Neuron model	Firing threshold θ	15 and 1
	Refractory period	0 ms
	Inter-layer delays	0-1200 ms
	SRM T_i	60 ms
	$h[n]$	linearly increasing
SNN Layer **I**	Input neurons N	32
	ELL neurons M	32
	Delays per connection	20
	Weight mean μ	0
	Weight variance σ	80
SNN Layer **J**	Output neurons D	11
	Sigmoid a	0.001
	Learning rate ϵ	0.008

3.3 Performance Metrics

To evaluate the performance of the neural network, various criteria were established to determine the winning neuron when multiple neurons are active in the output layer. The following criteria were used:

- **CMATS (Class with Maximum Accumulated Time of Spikes)**: This criterion declares the neuron with the maximum duration of the spike train in its output as the winner.
- **CS (Class with Maximum Number of Spikes)**: In this criterion, the neuron that produces the maximum number of spikes in its output is declared the winner, regardless of the time period it took to emit those spikes.
- **DP (First to Discharge Class)**: The winner in this criterion is the neuron that fires first and emits a spike.
- **RAND (Random Class)**: In this case, the winning neuron is randomly chosen among all the neurons that emitted spikes.
- **HARD (Hard Criterion)**: This is the most demanding criterion, as if two neurons emit spikes, the network considers it a failure to determine the class of the input example.

The results obtained with these criteria showed no significant difference in recognition performance, except for the HARD and RAND criteria, which exhibited inferior performance. Consequently, the CMATS criterion was used to

measure recognition stability during training. Once stability was achieved, the training process was terminated, and the weights obtained at the last epoch were used to evaluate the network's performance with the evaluation data.

3.4 Effect of including ELL

The purpose of random weights between the input vector and the first layer is to project signals into a higher-dimensional space, thereby discriminating certain characteristics that could not be separated in the initial representation. The number of neurons in the first layer, which is a key parameter was set to 32, equal to the dimension of the input vector. Although increasing the number of neurons can potentially improve recognition rates, it also significantly increases computational costs. Despite the number of hidden layer neurons equalling the number of dimensions of the input vector, the transformation still increases the dimensionality of internal network representation. This is due to connection delays between the first layer and the input vector. This "expansion" effect in the temporal dimension is analogous to that observed during a convolution with an impulse due to the memory property of linear systems. An example of the response of the first layer of neurons to the presentation of a spike pattern corresponding to a digit at the input of the SNN can be seen in Fig. 2. The above-mentioned expansion of the representation in the time domain can be clearly observed.

The proposed DELSNN was evaluated on clean digits from the TIDIGITS dataset. The training, validation and test partitions described in Section 3.1 were used. To assess the benefits of incorporating the extreme learning layer, comparative experiments were conducted training the network with and without the ELL. Without the ELL, the SNN was trained using direct entropy minimization on the output weights. In contrast, the ELL configuration included an additional 20480 randomly initialized input-projection connections.

Figure 3 illustrates the test set accuracy over training iterations for both settings. It can be in observed in Fig. 3 (B) that accuracy rapidly increases during the first 10-20 iterations, reaching around 80% (CMATS). After that, more gradual improvements occur until converging at approximately 88% accuracy. After 100 iterations, the ELL network achieves 88.18% accuracy compared to 64.36% for the network without ELL (Fig. 3 (A)). So the ELL version converges faster and to a higher level of performance.

The incorporation of randomly projected features provides two key benefits. Firstly, it increases the separation between classes, facilitating the training process. Second, it improves generalisation by improving the variability of internal network representations.

Figure 4 shows the accuracy per class after training with and without ELL. As it can be seen, accuracy levels are higher for most digits when using the ELL, especially for challenging cases like 5 and 9. The ELL's ability to better distinguish acoustically similar digits highlights its advantages.

The high deviations in epochs 0-50 may be due to the randomness of the ELL weights and delays, which affect the initial state of the network . Some

sinc(i) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

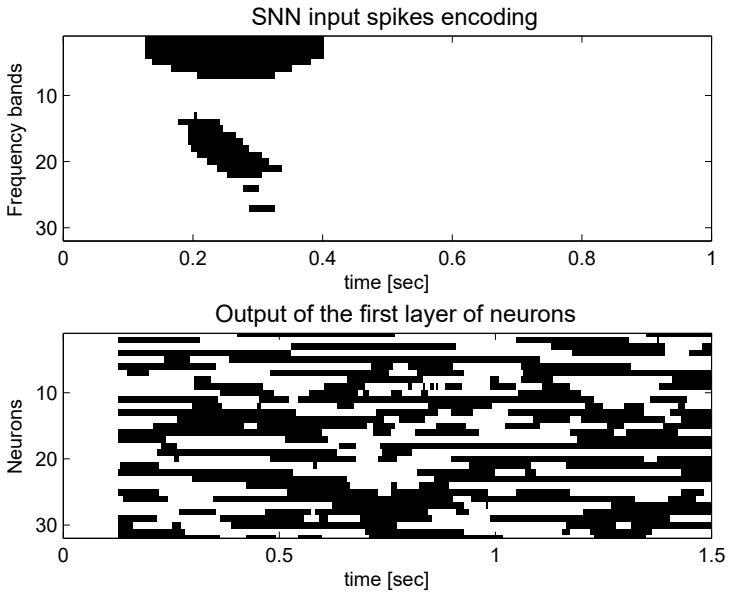

Fig. 2. Response of the first layer of neurons to the presentation of the digit "three". Top: input pattern to the SNN after coding in frequency bands. Bottom: response of the first layer of neurons to the presentation of this spectro-temporal pattern (hidden layer).

instabilities in the computation of KL divergence can also contribute to these deviations, as ELL may amplify them. As the training progresses, the output layer weights are adjusted to minimize the entropy, and these deviations decrease.

These results validate the benefits of the proposed extreme learning layer for boosting the training and accuracy of SNN models in speech recognition applications. ELL provides a simple and effective technique to incorporate randomness and improve learning in SNN. This also indicates successful training of the output layer weights through the proposed entropy minimisation algorithm.

3.5 Noise Robustness Evaluation

To evaluate the noise robustness of the proposed DELSNN, experiments were also performed by adding white noise from the NOISEX database [15] to the clean TIDIGITS digits. White noise was chosen to evaluate the noise robustness of the proposed neural network because it contains uniform power at all frequencies of the audible spectrum. This results in a broad spectral mask that obscures a wide variety of critical acoustic cues for discriminating between phonetically similar digits.

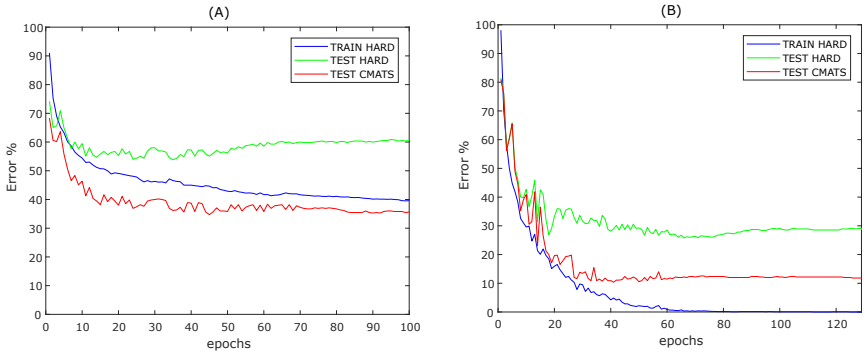

Fig. 3. Overall performance train/test accuracy comparison during training without ELL (A) and with ELL (B) (for clean speech signals).

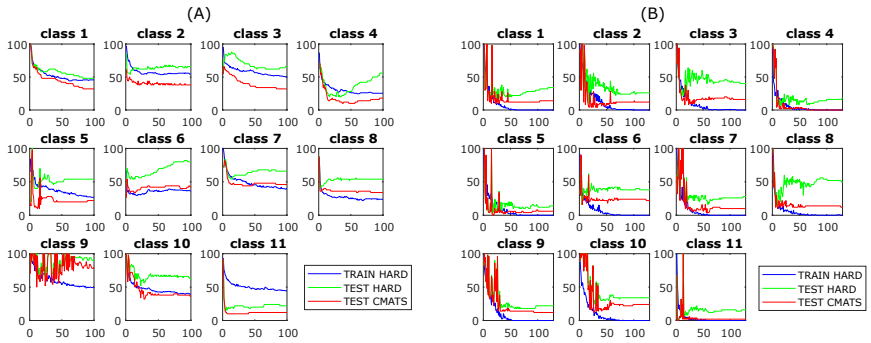

Fig. 4. Per-class performance train/test accuracy comparison during training without ELL (A) and with ELL (B) (for clean speech signals).

The best results obtained for clean and different signal-to-noise ratio (SNR) levels for different threshold Ψ are summarized in Table 2. For $\Psi = 0.370$ it can be observed that accuracy remains relatively stable down to 10 dB SNR, indicating the noise tolerance provided by the SNN encoding and architecture. However, performance degrades more sharply below 10 dB SNR, as noise completely masks discriminative speech characteristics. Nonetheless, the DELSNN can operate at SNR levels where conventional MFCC and HMM-based systems suffer from almost 70% error rates. For larger values of Ψ, stability of accuracy is maintained for higher levels of noise, but speech recognition in clean speech slightly deteriorates.

The increased noise tolerance of the proposed DELSNN compared to conventional speech recognition systems highlights the potential of SNN-based approaches. Further improvements may be achieved by exploring alternative noise-robust spike-coding strategies, such as phase coding [4] or Biologically plausible Auditory Encoding (BAE) [11]. A comparison with deep architectures like CNNs

sinc(i) Research Institute for Signals, Systems and Computational Intelligence (sinc.unl.edu.ar)
I. R. Peralta, N. Odetti & H. L. Rufiner; "Extreme Learning Layer: A Boost for Spoken Digit Recognition with Spiking Neural Networks"
Speech and Computer, pp. 3-17, 2023.

Table 2. Comparison of different speech recognizers for this task under various levels of white noise. Missing values correspond to values not reported in the bibliography.

Strategy	Threshold Ψ	Clean	20 dB	10 dB	5 dB	0 dB
Lyon ear + LSM [16]	-	97.50%	84.00%	79.50%	-	-
LAM + HMM [2]	-	**98.80%**	**95.75%**	72.79%	-	-
MFCC + HMM [2]	-	**98.80%**	27.50%	12.20%	-	-
BAE + SNN [11]	W/masking	97.40%	91.90%	87.50%	-	**78.20**
MSE+DELSNN	0.785	79.65%	79.44%	79.81%	**80.37%**	58.57%
	0.576	82.58%	82.54%	83.15%	78.76%	18.02%
	0.370	89.14%	89.10%	**88.2%**	43.08%	1.81%

that also use MSE as input feature would be interesting. However, we could not find any published results on the TIDIGITS dataset corrupted with additive noise to include in the comparison.

4 Conclusions and Future Work

In this paper, a novel DELSNN incorporating an extreme learning layer for robust spoken digit recognition was proposed and evaluated. The presented results provide initial evidence of the advantages of the EL approach to improving SNN training. Noise robustness experiments also showcase the potential of SNN architectures for speech processing under challenging conditions. Furthermore, as already demonstrated in our previous work, this model can be efficiently implemented on an FPGA platform, providing potential for real-time applications.

For future work, some research directions are identified, like the evaluation of larger vocabulary tasks to assess scalability and the exploration of alternative spike-coding strategies to further improve noise robustness. DELSNN architecture could also be extended to handle connected word recognition by incorporating some mechanisms to deal with coarticulation effects.

Acknowledgements

We would like to express our gratitude to the National University of Entre Ríos UNER for their support and resources provided within the framework of the research and development project PID6187, and to the National University of Litoral with project CAID 50620190100151LI, enabling us to conduct this investigation.

References

1. Bhangale, K.B., Kothandaraman, M.: Survey of deep learning paradigms for speech processing. Wireless Personal Communications **125**(2), 1913–1949 (2022)

2. Deng, Y., Chakrabartty, S., Cauwenberghs, G.: Analog auditory perception model for robust speech recognition. In: Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on. vol. 3, pp. 1705–1709. IEEE (2004)
3. Gerstner, W., Kistler, W.M.: Spiking neuron models: Single neurons, populations, plasticity. Cambridge university press (2002)
4. Guo, W., Fouda, M.E., Eltawil, A.M., Salama, K.N.: Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems. Frontiers in Neuroscience **15**, 638474 (2021)
5. Gupta, S., Agrawal, A., Pathak, A.: Energy-efficient deep learning: A review. Sustainable Computing: Informatics and Systems **25**, 100370 (2020). https://doi.org/10.1016/j.suscom.2020.100370
6. Huang, G., Huang, G.B., Song, S., You, K.: Trends in extreme learning machines: A review. Neural Networks **61**, 32–48 (2015)
7. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: theory and applications. Neurocomputing **70**(1), 489–501 (2006)
8. Izhikevich, E.M.: Which model to use for cortical spiking neurons? IEEE Transactions on Neural Networks **15**(5), 1063–1070 (2004)
9. Leonard, R.G., Doddington, G.: TIDIGITS. Linguistic Data Consortium, Philadelphia (1993)
10. Maass, W.: Networks of spiking neurons: the third generation of neural network models. Neural networks **10**(9), 1659–1671 (1997)
11. Pan, Z., Chua, Y., Wu, J., Zhang, M., Li, H., Ambikairajah, E.: An efficient and perceptually motivated auditory neural encoding and decoding algorithm for spiking neural networks. Frontiers in Neuroscience **13** (2020), https://www.frontiersin.org/articles/10.3389/fnins.2019.01420
12. Peralta, I., Odetti, N., Filomena, E., Rufiner, J., Ricart, N., Rufiner, H.L.: A new spiking neural network with extreme learning for FPGA implementation. In: Proceedings of the 10th Southern Programmable Logic Conference. pp. 49–54 (2019), http://sinc.unl.edu.ar/sinc-publications/2019/POFRRR19
13. Schrauwen, B., Van Campenhout, J.: Parallel hardware implementation of a broad class of spiking neurons using serial arithmetic. In: Proceedings of the 14th European Symposium on Artificial Neural Networks. pp. 623–628. D-Side Publications (2006)
14. Unnikrishnan, K., Hopfield, J.J., Tank, D.W.: Speaker-independent digit recognition using a neural network with time-delayed connections. Neural Computation **4**(1), 108–119 (1992)
15. Varga, A., Steeneken, H.J.: Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. Speech communication **12**(3), 247–251 (1993)
16. Verstraeten, D., Schrauwen, B., Stroobandt, D., Van Campenhout, J.: Isolated word recognition with the liquid state machine: a case study. Information Processing Letters **95**(6), 521–528 (2005)